Best practices for OpenShift high-availability deployment field experience

Traditional disaster recovery means maintaining underutilized datacenters, infrastructure, and software. During outages, your organization endures downtime while backup datacenters and applications come back online. However, success in today’s always-on, always-available world cannot tolerate interruption of services. Yesterday’s disaster recovery strategies and technologies are inadequate for today’s business requirements.

With an active-active, elastic SQL database designed for containers, hybrid cloud, and modern applications, you don’t need to force-fit traditional technologies into today’s world.

In this session, you will learn:

- Best practices for OpenShift high-availability (HA) deployment.
- The challenges of traditional disaster recovery strategies for today's requirements.
- How elastic SQL databases like NuoDB innately provide active-active capabilities.
- Examples of disaster recovery using an active-active database and Red Hat technologies.

- Date: Wednesday, May 9
- Time: 11:45 AM - 12:30 PM
OUTLINE

- Introduction to speakers (all) - 5min
- The challenges of traditional high availability strategies for today's requirements. - 10 min
  - Discuss the considerations throughout the stack - from developer pipeline to application (orgad) to database (NuoDB) to storage (Orgad/NuoDB)
  - Discussion traditional database DR approaches (NuoDB)
- Introducing OpenShift, red hat products, and how they address HA (Orgad) - 5min
- Introducing NuoDB and how it's an active-active database (NuoDB) - 5 min
- Best practices in the field - 15 min
  - OpenShift high-availability (HA) deployment. (Orgad)
  - NuoDB high availability deployments (NuoDB)
- Additional considerations for customers when building HA solutions with OpenShift (both??) - 5 min
BEST PRACTICES FOR HIGH AVAILABILITY IN OPENSSHIFT

Wednesday, May 9th, 2018.
11:45 - 12:30 pm. Moscone Center West, Room 2004.

Orgad Kimchi
Senior Cloud Architect
Red Hat

Ariff Kassam
VP, Products
NuoDB

Christina Wong
Director of Partnerships and Product Marketing
NuoDB
AGENDA

● Introduction to speakers
● Today’s challenges for high availability and disaster recovery
● Best practices in the field
● Parting thoughts
INTRODUCTIONS
WELCOME!

INTRODUCTION TO SPEAKERS

Orgad Kimchi
- Team Leader _Senior Cloud Architect
- Electric Guitar Player

Ariff Kassam, VP of Products, NuoDB
- Responsible for product strategy and roadmap
- A Canadian

Christina Wong, Director of Partnerships and Product Marketing, NuoDB
- Former Red Hatter, fascinated by microservices, databases, DevOps, and all things cloudy
- Jalopy race car mechanic and driver
TODAY’S CHALLENGES FOR HIGH AVAILABILITY (HA) AND DISASTER RECOVERY (DR)
TODAY, CIOs REQUIRE MORE

REQUIREMENTS
● No outages
● React quickly to demand
● Provide service 24 x 7 x 365

IMPLEMENTATION
● DevOps, CI/CD
● Rolling upgrades
● Redundancy
● HA architectures
● HA operational processes

CONTINUOUS SERVICE DELIVERY
YESTERDAY’S TECHNOLOGIES NO LONGER DELIVER

APPLICATION DEVELOPMENT & OPERATIONS
- Monolithic applications and databases
- Tightly coupled system integration
- Slow to respond to changes
- Manual processes

DATACENTERS & INFRASTRUCTURE
- Datacenter
- Resources
- Migration -> downtime
- Scale up
- High availability = added complexity
TRADITIONAL DATABASE APPROACHES TO HA & DR

- Multiple add-ons to support HA and DR capabilities
- Additional complexity in a container environment

**Maximum Availability Architecture (MAA)**
Low-Cost, Integrated, Fully Active, High ROI

**Production Site**
- RAC
  - Scalability
  - Server HA
- Flashback
  - Human error correction
- Online Redefinition, Edition-based Redefinition, Data Guard, GoldenGate
  - Minimal downtime maintenance, upgrades, and migrations

**Active Replica**
- Active Data Guard
  - Data Protection, DR
  - Query Offload
- GoldenGate
  - Active-active
  - Heterogeneous
- Oracle Secure Backup
  - Backup to tape / cloud

Deployed at numerous mission-critical sites worldwide!!
TRADITIONAL DR APPROACHES

COLD STANDBY
- Applications and data unavailable until standby is brought online
- Data loss highly likely

WARM STANDBY
- Periodic backups
- Idle and Offline

Active Data Center
Full Read/Write

Standby Data Center
Full Read/Write

Batch or continuous replication
Idle and Offline
TRADITIONAL DR APPROACHES

- Standby available for immediate use in case of active data center failure
- Zero or read-only data processing utilization in standby data center
TRADITIONAL DR APPROACHES

- Additional technology
- Configuration
- Coordination
- Complicated & difficult
BEST PRACTICES FROM THE FIELD
INTRODUCTION TO OPENSSHIFT HIGH AVAILABILITY (HA)

- Single site application (Traditional)
  - No disaster recovery

- DR application (Traditional)
  - Failover to a secondary site

- Multi site application (Cloud-native)
  - Deploy across multiple sites
  - Disaster recovery is built-in. Simply re-scale.
  - Recommend three sites versus two
OPENS SHIFT HA INFRASTRUCTURE CONSIDERATIONS
DEPLOYING HA APPLICATIONS ON OPENShift

BUILD FOR HA PERFORMANCE - SCALE AND AUTOSCALE
(Reference: https://docs.openshift.org/latest/dev_guide/pod_autoscaling.html)

- Build application components to scale!
  - At least 2 pods per component
  - e.g. web framework, messaging tier, datastore
- By default, OpenShift deploys pods on separate nodes
- Multiple pods are essential for upgrade and maintenance
  - Upgrade 1 node at a time without downtime
- Scale to minimum throughput, plus 1 pod
  - Removing 1 pod will still leave enough compute power to satisfy demand
- Use horizontal pod autoscaling to ensure you always have enough capacity
DEPLOYING HA APPLICATIONS ON OPENSHIFT
RUNNING STATEFUL APPS AT SCALE - STATEFULSETS

- **Description:**
  - Provides an identity to each pod of the set that corresponds to that pod's persistent volume(s)
  - If a StatefulSet pod is lost, a new pod with the same virtual identity is reinstated and the associated storage is reattached

- **Benefits**
  - Alleviate complex, state-related problems
  - Automation of manual process
  - Easy to run stateful applications at scale
DEPLOYING HA APPLICATIONS ON OPENSHIFT

MANAGING APPLICATION HEALTH - LIVENESS AND READINESS PROBES
(Reference: https://docs.openshift.org/latest/dev_guide/application_health.html)

- Detects when a pod is in a bad state and should be taken out of (or not added into) rotation
- Implementation
  - HTTP calls
  - Container executions
  - TCP socket calls
  - Custom logic to validate an application component is healthy

Example: Readiness HTTP check

```yaml
...
readinessProbe:
  httpGet:
    path: /healthz
    port: 8080
    initialDelaySeconds: 15
    timeoutSeconds: 1
  ...
```
DEPLOYING HA APPLICATIONS ON OPENSSHIFT

TUNING

(Reference: https://docs.openshift.org/latest/dev_guide/compute_resources.html#dev-compute-resources)

- Tune based on memory and CPU consumption
- OpenShift-provided images
  - Auto-tune themselves based on allocated memory
  - Simply allocate enough for the application to function properly
- Custom images
  - Auto-tune the application runtime based on the resource limits set.

Allocate memory and CPU to meet Quality of Service (QoS) requirements
HIGH AVAILABILITY NODE DEPLOYMENT

CONTROLLING POD PLACEMENT - NODE LABELS AND NODE SELECTORS
(Reference: https://docs.openshift.org/latest/admin_guide/managing_projects.html#using-node-selectors)

● Deploy application across data centers or racks
  ○ Define nodes based on region or rack
  ○ Ensures application is not deployed on a single data center or single rack

● Basic placement-related concept
  ○ Generic node labeling system to group nodes
  ○ Every node will at least have its node name as a label

● Node labels
  ○ Target for deployments using node selectors
  ○ Set at either:
    ■ Project level (can use to restrict project access to certain nodes)
    ■ Pod level

● Example groupings:
  ○ Availability and latency - Region/Zone/Rack
  ○ Segregate environments - Production/Stage/Test - e.g. o segregate
Several layers requiring replication for disaster recovery

- Project/namespace
- Images
- Persistent volumes
OPENSHIFT DISASTER RECOVERY STRATEGIES

PROJECTS

- **Objects**
  - Everything in OpenShift is an object - e.g. deployment configs, secrets, build configs
  - Easily export/import via YAML or JSON

- Use existing scripts or easily write your own

- **Note!!**
  - SELinux groups, UIDs and ranges must be exactly the same between the production and DR
OPENSHIFT DISASTER RECOVERY STRATEGIES
INTEGRATE HIGH AVAILABILITY AND DISASTER RECOVERY INTO CI/CD PIPELINE

- Each application update results in a new image
- Distribute images to DR projects by adding DR to the pipeline
- Images will be pushed between OpenShift clusters
STORAGE REPLICATION STRATEGIES

OVERVIEW OF PERSISTENT VOLUMES

- Containers themselves are immutable but can mount Persistent Volumes (PVs) via a Persistent Volume Claim (PVC)
- Persistent Volume - A mapping to a physical disk or file system (e.g. file based storage Gluster).
- Persistent Volume Claim - Binds a Persistent Volume to a particular pod where a single or multiple containers are running
- Handling storage when a pod move nodes
  - must be unmapped, remapped and mounted on the new node
STORAGE REPLICATION STRATEGIES

INFRASTRUCTURE-BASED ASYNCHRONOUS REPLICATION

https://blog.openshift.com/deploying-openshift-applications-multiple-datacenters/
STORAGE REPLICATION STRATEGIES

APPLICATION-BASED ASYNCHRONOUS REPLICATION

https://blog.openshift.com/deploying-openshift-applications-multiple-datacenters/
OPENSHIFT NETWORKING CONSIDERATIONS

- OpenShift depends on low-latency network across its control plane to synchronously replicate state.
- Need to have multiple separate OpenShift clusters.
- High-latency networks complicate data replication for applications like databases deployed as containers.
HA AND DR CHALLENGES

● Single DC HA
  ○ Replicated Gluster storage works well for Active-Standby database configuration
  ○ But requires an application outage when failing over from Active to Standby

● Multi-DC DR
  ○ Storage must be replicated either by the application or asynchronously (Master-Slave)
  ○ Storage replication must be setup manually (not part of the DevOps pipeline)
  ○ Network latencies can impact application behavior

● Operational challenges
  ○ Requires significant amount of manual setup
  ○ How to coordinate backups of the application and data

● Application developers should not have to worry about all these considerations
● Application developers should be able to leverage a container-native database that provides all of these services automatically
INTRODUCING NUODB
CONTAINER-NATIVE SQL DATABASE

- Inherently distributed database
  - Distributed multi-process
  - Scale out
  - Single logical database
- Built-in data replication
  - Automatic HA
  - Automatic DR
  - Fully Active-Active
- Automated deployment & upgrades
  - CI/CD pipelines
- Fully SQL & ACID compliant
FLEXIBLE DEPLOYMENT CONFIGURATIONS
HIGH AVAILABILITY FOR ANY TRANSACTIONAL WORKLOAD

- Independently scale-out either compute or storage tiers to address different workload use cases

Web/Mobile Application
+ Read dominated
+ Scale-out compute tier

OLTP Application
+ Mixed read/write
+ Scale-out compute & storage tiers

Logging Application
+ Insert dominated
+ Scale-out storage tier

HTAP Application
+ Mixed read/write
+ Dedicated analytical nodes
DEPLOY ACROSS MULTIPLE ENVIRONMENTS

AVAILABILITY ZONE 1

Application

AVAILABILITY ZONE 2

SQL Query Processing

AVAILABILITY ZONE 3

Data Durability

Cloud Native Storage

#redhat #rhsummit
AUTOMATE DATABASE UPGRADES
INTEGRATE DATABASE INTO CI/CD PIPELINE
BENEFITS OF NUODB IN OPENSHIFT

- Inherently a distributed system with database capabilities
- Elastic scale out and in
- Integrated in-memory data cache
- Automatic HA & DR and Active-Active
- Programatically deployable in existing CI/CD pipelines
PARTING THOUGHTS
KEY TAKEAWAYS
MODERN HIGH AVAILABILITY AND DISASTER RECOVERY WITH CONTAINER APPLICATIONS

- Traditional methods do not meet today’s high availability requirements
- OpenShift and containers provides options for high availability throughout the application stack
- For storage and data, reduce complexity with a container-native database like NuoDB
- NuoDB = High availability, modern disaster recovery, elastic scale out for SQL applications
ATTEND NuoDB’s DEMO SESSIONS

- Wed, May 9th at 5:45pm, Partner Pavilion
- Thurs, May 10th at 12:00 pm, OpenShift Theater
- Overview and demo by NuoDB DevOps Engineer, Ben Higgins

THANK YOU

LEARN MORE ABOUT NuoDB AT BOOTH #731

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