Simplifying Stateful Apps On Kubernetes: A Git Like Workflow

Jagadish Mukku, Robin.io
Dhanashankar Venkatesan, Robin.io
Agenda

- Stateful Application
- Background on Containers and Kubernetes
- Data Management of Stateful App
- Challenges and Solutions
- Q & A
Spectrum Of Applications

Stateless Applications

Web Apps

Stateful Applications

SQL Databases

NoSQL Databases

Big Data

Web Apps: WordPress, node.js, MySQL, PostgreSQL, Oracle Database

SQL Databases: MariaDB, MySQL, PostgreSQL

NoSQL Databases: MongoDB, Couchbase, Cassandra, Redis, InfluxDB, Prometheus

Big Data: Elasticsearch, Splunk, Cloudera, HortonWorks, Greenplum
Traditional Deployment

App

App

App

Operating System

Hardware
Virtualized Deployment

- Hypervisor
- Operating System
- Hardware

App
Bin/Library
Operating System
Virtual Machine

App
Bin/Library
Operating System
Virtual Machine
Containerized Deployment

- App
  - Bin/Library
  - Container

Container Runtime

Operating System

Hardware
Kubernetes Overview

- Portable, Extensible, Open-Source platform for managing containerized workloads and services
- Agile
  - Modularity and Scalability
- Life Cycle Management
  - Health Checks, Automated Rollouts, Canary Deployment, Load Balancing
- Reduces Costs
  - Containerize apps and consolidate resources
Simple Stateful App (mysql)

Traditional

Cloud Native
Distributed Stateful App (Cassandra)

What makes it complex?
- Multiple volumes
- Multiple nodes
- Large data
Distributed Stateful App (Cassandra)
Storage Allocation Challenges

Still resilient to disk failure

Software Defined Storage
Let us fix it …

Replica-1

Replica-2

Replica-3

Software Defined Storage
Storage Allocation Challenges

Software Defined Storage

POD

pvc

redo

data

2019 Storage Developer Conference. © Robin.io, All Rights Reserved.
Let us fix it..

Software Defined Storage
Challenges of running Stateful Apps on Kubernetes

- Servers/Disks fail and humans make errors → Apps become unavailable, Data is lost
- Data must be protected and secured → Expects users to become storage experts
- Unpredictable performance from sharing resources → Application SLAs not met, unhappy users
- Unable to share data between prod and dev/test → Low productivity and slower innovation
- Data gravity prevents app mobility across environments → Cloud or Infrastructure Lock-in
Distributed Stateful App (Cassandra)
Git Operations

$ git commit –m ”[descriptive message]”
$ git reset [commit]
$ git branch [branch-name]
$ git push [branch]
$ git clone [url]
Manage App As If Its In Git…

Database App

commit/snapshot

revert/rollback

clone/import

Snapshots

branch

push

Thin Clones

Cloud Storage
Robin Architecture Overview

1-click application Deploy, Snapshot, Clone, Scale, Upgrade, Backup
Application workflows configure Kubernetes, Storage & Networking

Application Workflow Manager

Robin’s built-in enterprise-grade storage stack
Snapshots, Clones, QoS, Replication, Backup, Data rebalancing, Tiering, Thin-provisioning, Encryption, Compression

Built-in flexible networking
OVS, Calico, VLAN, Overlay networking, Persistent IPs

Works any where
Logical Volume Allocation

Volume comprises of logical 1GiB slices
Physical Space provisioning

- Multiple volumes of app
- Affinity requirements
- Anti-affinity requirements
- Latency sensitive, Bandwidth intensive
- Node, Rack, Datacenter awareness
Log Structured Data Layout

- **Volume**: collection of slices
- **Slice**: collection of segments
- **Segment**: log structured data blocks
Robin IO path

COMPONENTS

› RIO – Robin IO Manager
› RDVM – Robin Distributed Volume Manager

FEATURES

› High performance IO stack
› IO tagging helps with end-to-end QoS enforcement
› Application-aware data placement
› Supports multi-node volume access (clustered volume manager)
› Block-level data management (Snapshots, Clones, Compression, Encryption, Tiering etc) enables a common workflow for any and all applications
Volume Time Travel

Volume

segment 1
segment 2
segment 3
segment 4
segment 5

segments

time

2019 Storage Developer Conference. © Robin.io, All Rights Reserved.
Maintain periodic checkpoints of your entire app with data

$ robin snapshot app1 snap1

Rollback entire app+data to healthy state to recover from corruptions or user errors

$ robin rollback snap1 app1

Backup entire app+data as into external backup targets

$ robin push snap1 target

Restore entire app+data to healthy state from catastrophic hardware and datacenter failures

$ robin pull target snap1

- ROBIN Backups are fully self-contained
- Entire app resources can be restored in the same or different data center or cloud even if the source is completely destroyed
Make DevOps Productive With Stateful Apps

Use Cases:
- Clone databases from prod to dev/test for running reports
- Validate upgrades before applying to production
- Enable git like push/pull for geo-dispersed teams to collaborate

**On-prem**

**Kubernetes Cluster 1**

**Google Cloud Platform**

**Kubernetes Cluster 2**

**AWS**

**Kubernetes Cluster 3**

**CLOUD OBJECT STORE**

(ClickFS, S3, GCS, AzureBlob)

---

**STEP1:** `robin snapshot mysql mysql-snap`

**STEP2:** `robin clone mysql-snap testdev-mysql`

**STEP3:** `robin push mysql-snap gcs://bucket`

**STEP4:** `robin pull gcs://bucket/mysql-snap mysql`

---

Use Cases:
- Clone databases from prod to dev/test for running reports
- Validate upgrades before applying to production
- Enable git like push/pull for geo-dispersed teams to collaborate
App Snapshot

$ git commit –m "[descriptive message]"

$ robin snapshot <snapname> <appname> –m "[descriptive message]"

- Application Consistent
- Quiesce/Unquiesce volumes
- Complete app state snapshotted
  - Kubernetes Objects (Metadata)
  - Volumes (Data)
App Rollback/Reset

$ git reset [commit]
$ robin app rollback <snapname> <appname>

- App config and data restored
- Handle scale-in/scale-out scenarios
- Revert on corruption, user errors, upgrade failures
Thin Clone

$ git branch [branch-name]

$ robin app create <appname> <snapname>

- Instant app deployment from snapshot
- No data copied
- Customize cloned app config
- Blue/Green deployment
- Test upgrades
- Run analytics
Push to Cloud

$ git push [branch]

$ robin app push <snapname> <reponame>

- Uses app snapshot
- Self sufficient copy of app
- Incremental transfer to repos
  - AWS S3, Google GCS, Azure Block blob, NFS share
- Encrypted and Compressed
Pull from Cloud

$ git clone <url>

$ robin app pull/create <snapname> <reponame>

- Repair app from repo copy
- Create new app from repo copy
- App mobility
  - Run app in the cloud
- Protect from site failures
  - Use cloud copy as backup
Q & A
Thank You

Please visit robin.io for more information
Download the software at get.robin.io and try it out