Databases run better with Percona
Kubernetes Operators

Expanding Automation in Containerized Applications
Edith Puclla (Edi)

- Technology Evangelist at Percona
- The UK Global Talent Visa
- CNCF Ambassador
  - KCD, Lima Peru
- Docker Captain
  - Meetups in Ayacucho
- Open Source Contributor:
  - Apache Airflow, Kubernetes Website

LinkedIn: Edith Puclla
Matrix: @edithpuclla
Agenda

- Kubernetes
  - Deploying an Application
  - Default Resources
  - Limitations
- Kubernetes Operators
  - Components
    - CRD, CR, OLM, Controllers
  - Operator framework, Operator Hub, Capability Models
Kubernetes
k8s
Contenedores

OPERATING SYSTEM

PROCESS → Container

Container A
- Python app
- Python runtime, libraries, dependencies

Container B
- Java app
- Java runtime, libraries, dependencies

Container C
- C++ app
- C++ runtime, libraries, dependencies

Docker
Host Operating System
Infrastructure
Challenges with containers at scale

- Orchestration
- Security
- Monitoring and registration
- Scalability
- Data Storage and Persistence
Kubernetes Advantages

- Deployment automation
- Scaling based on demand
- Application portability
- Self-healing
- Good option for microservices
- Active community and wide adoption
Kubernetes Terminology

Pods
○ Containers
  ■ Ready storage

Deployments
○ Application Deployment
○ Desired state
○ Replicas

Services
○ Pod Access
Example: Voting Application

voting-app

Cats vs Dogs!

CATS

(Tip: you can change your vote)

DOGSS

result-app

CATS 100.0%

DOGS 0.0%

KodeKloud: www.youtube.com/watch?v=XuSQU5GrvIg
Containers

voting-app

result-app

redis

DB

worker
PODS

voting-app

result-app

redis

DB

worker

POD

POD

POD

POD

6379

5432

80

80

©2024 Percona
File: voting-app-deploy.yaml

apiVersion: apps/v1
kind: Deployment
metadata:
  name: voting-app-deploy
  labels:
    name: voting-app-deploy
    app: demo-voting-app
spec:
  replicas: 1
  selector:
    matchLabels:
      name: voting-app-pod
      app: demo-voting-app
  template:
    metadata:
      name: voting-app-pod
      labels:
        name: voting-app-pod
        app: demo-voting-app
    spec:
      containers:
        - name: voting-app
          image: kodekloud/examplevotingapp_vote:v1
          ports:
            - containerPort: 80
Kubernetes architecture

User Interface

UI

CLI

kubectl

Control Plane

API Server
Scheduler
Controller-Manager
etcd

Worker node 1

Pod 1
Container 1
Container 2
Container 3

Pod 2
Container 1
Container 1

Pod 3
Container 1
Container 2

Worker node 1

Pod 1
Container 1
Container 2
Container 3

Pod 2
Container 1

Pod 3
Container 1
Container 2

Docker
kubelet
kube-proxy

Docker
kubelet
kube-proxy

©2024 Percona
Kubernetes Operators
Stateless Application Scaling: Easy
kubectl scale deploy/staticweb --replicas=4
What about applications that store data?
“Deploy” a database: easy
Running a database *over time* is the hardest
Kubernetes Application Lifecycle

Day 0
Planning and Development

Day 1
Operations and escalation

Day 2
Management and continuous optimization
Kubernetes Operators:

Extend the Kubernetes API
How the API is Extended:

- Custom Resource Definitions (CRD)
- Custom controllers for specific applications
Custom Resource Definitions (CRD)

```yaml
apiVersion: apiextensions.k8s.io/v1
kind: CustomResourceDefinition
metadata:
  # name must match the spec fields below, and be in the form: <plural>.<group>
name: crontabs.stable.example.com
spec:
  # group name to use for REST API: /apis/<group>/<version>
  group: stable.example.com
  # list of versions supported by this CustomResourceDefinition
  versions:
    - name: v1
      # Each version can be enabled/disabled by Served flag.
      served: true
      # One and only one version must be marked as the storage version.
      storage: true
      schema:
        openAPIV3Schema:
          type: object
          properties:
            spec:
              type: object
              properties:
                crontab:
                  type: string
                  image:
                    type: string
                  replicas:
                    type: integer
          # either Namespaced or Cluster
          scope: Namespaced
          names:
            # plural name to be used in the URL: /apis/<group>/<version>/<plural>
            plural: crontabs
            # singular name to be used as an alias on the CLI and for display
            singular: crontab
            # Kind is normally the CamelCased singular type. Your resource manifests use this.
            kind: CronTab
            # Additional shorter string to match your resource on the CLI
            shortNames:
              - ct
```

**CRD example**
Custom Resource Definitions (CRD)

my-crontab.yaml

```yaml
apiVersion: "stable.example.com/v1"
kind: CronTab
metadata:
  name: my-new-cron-object
spec:
cronSpec: "* * * * */5"
image: my-awesome-cron-image
```
kubectl apply -f my-crontab.yaml

kubectl get crontab

<table>
<thead>
<tr>
<th>NAME</th>
<th>AGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>my-new-cron-object</td>
<td>6s</td>
</tr>
</tbody>
</table>
Custom Controller

Observes changes to resources defined in the CRD in the Kubernetes cluster, detects changes, and reacts to level them out.
Kubernetes without Operators
Kubernetes with Operators
How do I create the operators?

The Operator Framework is an open source toolkit to manage Kubernetes native applications, called Operators, in an effective, automated, and scalable way.

What's in the Framework?
The Operator Framework is a set of developer tools and Kubernetes components, that aid in Operator development and central management on a multi-tenant cluster.
Welcome to OperatorHub.io

OperatorHub.io is a new home for the Kubernetes community to share Operators. Find an existing Operator or list your own today.
Capability Model

Level I: Basic Install
- Automated application provisioning and configuration management

Level II: Seamless Upgrades
- Patch and minor version upgrades supported

Level III: Full Lifecycle
- App lifecycle, storage lifecycle (backup, failure recovery)

Level IV: Deep Insights
- Metrics, alerts, log processing and workload analysis

Level V: Auto Pilot
- Horizontal/vertical scaling, auto config tuning, abnormal detection, scheduling tuning
Percona Operator for MySQL based on Percona XtraDB Cluster

Percona Operator for MySQL based on Percona XtraDB Cluster manages the lifecycle of Percona XtraDB cluster instances.

Percona Operator for MySQL based on Percona XtraDB Cluster

Percona is Cloud Native

Percona Operator for MySQL based on Percona XtraDB Cluster is an open-source drop in replacement for MySQL Enterprise with synchronous replication running on Kubernetes. It automates the deployment and management of the members in your Percona XtraDB Cluster environment. It can be used to instantiate a new Percona XtraDB Cluster, or to scale an existing environment.

Consult the documentation on the Percona Operator for MySQL based on Percona XtraDB Cluster for complete details on capabilities and options.

Supported Features

- Scale Your Cluster change the size parameter to add or remove members of the cluster. Three is the minimum recommended size for a functioning cluster.
- Manage Your Users add, remove, or change the privileges of database users
- Automate Your Backups configure cluster backups to run on a scheduled basis. Backups can be stored on a persistent volume or S3-compatible storage. Leverage Point-in-time recovery to reduce RPO/RTO.
- Proxy integration choose HAProxy or ProxySQL as a proxy in front of the Percona XtraDB Cluster. Proxies are deployed and configured automatically with the operator to provide network services.

Capabilities Level

- Basic Install
- Seamless Upgrades
- Full Lifecycle
- Deep Insights
- Auto Insights
Percona Operators simplify setting up and maintaining robust, enterprise-grade MySQL, PostgreSQL, and MongoDB clusters on Kubernetes

Try Percona Operators:
→ Percona Operator for MySQL
→ Percona Operator for MongoDB
→ Percona Operator for PostgreSQL

Percona Everest open source, cloud-native database platform

→ docs.percona.com/everest

Ask questions and leave your feedback:
→ percona.community
→ forums.percona.com
→ github.com/percona
Building K level 1
Thank You!