POSTGRES ON OPENSHIFT WORKSHOP

David Martini - Senior Solution Architect, Red Hat
Lucie Zeng - Sales Engineer, EDB
Raphaël Chir - Senior Sales Engineer, EDB
Eric Pillon - Strategic Account Executive, EDB

24th February, 2026

Agenda

Commencer	Fin	Session
12:30	13:30	Accueil Cocktail Lunch
13:30	13:45	Partenariat entre Red Hat OpenShift et EDB (Red Hat - David Martini)
13:45	14:00	Présentation du marché Postgres et d'EDB (EDB - Eric Pillon)
14:00	14:30	Architecture de référence de l'opérateur CNPG (EDB - Raphaël Chir)
14:30	16:30	Atelier pratique (EDB - Lucie Zeng & Raphaël Chir)
16:30	17:00	Pause
17:00	17:30	Conclusion et perspectives

Red Hat OpenShift with EDB

David Martini

Senior Solution Architect

Red Hat

The world's leading provider of open source enterprise IT solutions

More than **90%** of the Fortune **500** use **Red Hat** products and solutions[1]

~22,000 employees

105+ offices

The first **$3 billion** open source company in the world[2]

Sources: 1-Red Hat client data and Fortune 500 list, September 2024.
2 - Red Hat SEC filings prior to the acquisition by IBM. Note: Currency in U.S. dollars.

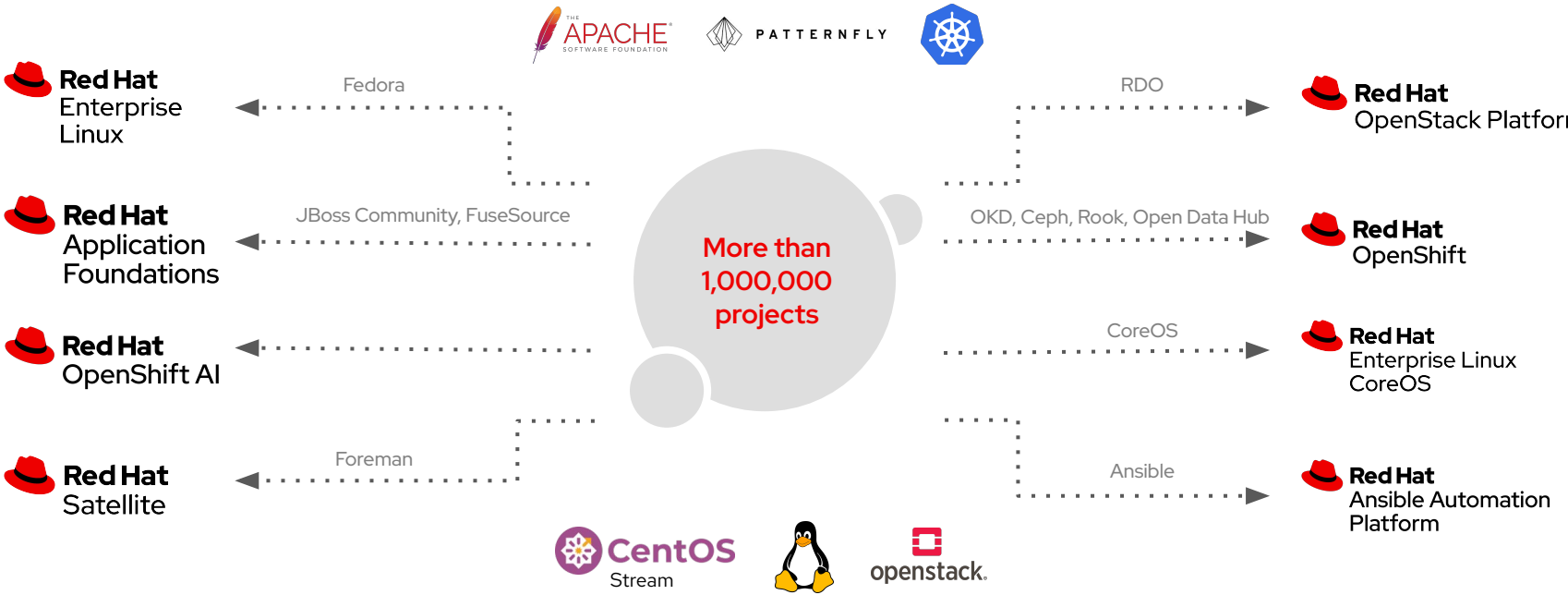
Red Hat

Open source

Open source is about more than developing software.
It's how we built our company.

And it's why we have been so successful.

From communities to enterprise



Red Hat

Red Hat | Linux contributions

- Platinum members
- Red Hat has been working on Linux since 1993
- Linux is the foundation of all Red Hat products
- The first Patent promise was issued by Red Hat in 2002

Red Hat | OpenInfra contributions

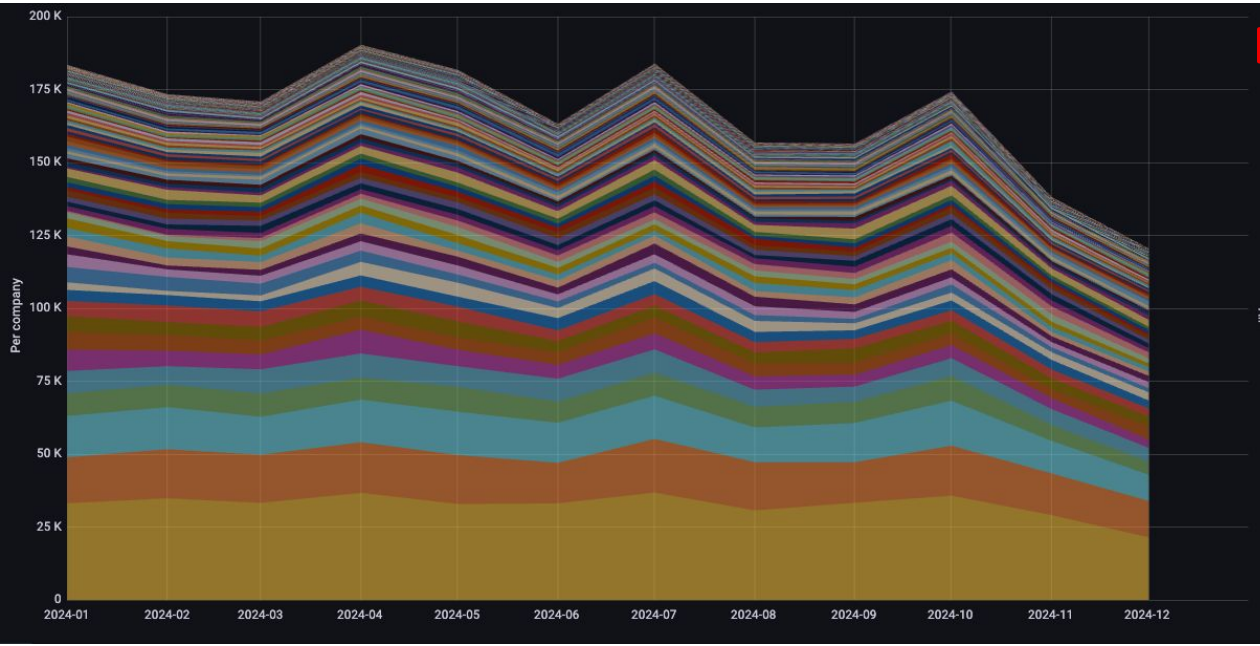
- Historical member
- First Red Hat OpenStack release in 2013
- Most represented member on the board (3/21)
- Red Hat & IBM = more than 3 times contributions as the second contributor

Red Hat | Cloud-native contributions

- Co-founder in 2015
- Platinum members
- Second Kubernetes contributor after Google
- First CNCF contributor in 2022, 2023, 2024, etc...

Red Hat

CNCF 2024 Global contribution



	min	max	avg	current	total
Red Hat Inc.	21.49 K	36.82 K	32.59 K	21.49 K	391.11 K
Microsoft Corporation	12.51 K	18.41 K	15.83 K	12.51 K	189.98 K
Google LLC	8.94 K	15.43 K	13.37 K	8.94 K	160.43 K
Independent	4.51 K	8.49 K	7.29 K	4.51 K	87.50 K
Isovalent	4.74 K	8.30 K	6.74 K	4.74 K	80.87 K
VMware Inc.	2.82 K	8.13 K	5.13 K	2.82 K	61.59 K
SUSE LLC	2.95 K	6.06 K	4.54 K	4.54 K	54.52 K
Splunk Inc.	3.40 K	5.53 K	4.50 K	3.40 K	54.02 K
International Business Machines Corporation	2.79 K	5.65 K	4.12 K	2.79 K	49.47 K
Amazon	2.69 K	4.52 K	3.56 K	2.69 K	42.78 K
Solo.io	1.41 K	4.79 K	3.12 K	2.80 K	37.45 K
Dynatrace LLC	1.43 K	5.26 K	2.88 K	1.43 K	34.51 K
DaoCloud Network Technology Co. Ltd.	1.81 K	4.37 K	2.75 K	2.11 K	32.97 K
Huawei Technologies Co. Ltd	1.33 K	3.67 K	2.53 K	2.08 K	30.40 K
Kong Inc.	1.48 K	3.49 K	2.51 K	1.50 K	30.10 K
Spotify AB	1.53 K	3.58 K	2.40 K	1.53 K	28.82 K
CNCF	1.31 K	3.40 K	2.28 K	1.31 K	27.41 K
Ericsson	1.67 K	2.78 K	2.10 K	1.73 K	25.19 K
Elasticsearch Inc.	498.00	3.10 K	1.90 K	2.03 K	22.79 K
Grafana Labs	1.42 K	2.30 K	1.89 K	1.42 K	22.66 K
Tetrate.io	1.15 K	2.45 K	1.88 K	1.59 K	22.57 K
Cisco	1.29 K	2.25 K	1.82 K	1.29 K	21.87 K

Red Hat is the #1 company for contributions to CNCF in 2024.

https://all.devstats.cncf.io/d/4/company-statistics-by-repository-group?orgId=1&var-period=m&var-metric=contributions&var-repogroup_name=All&var-companies=All&from=1704063600000&to=1735685999000

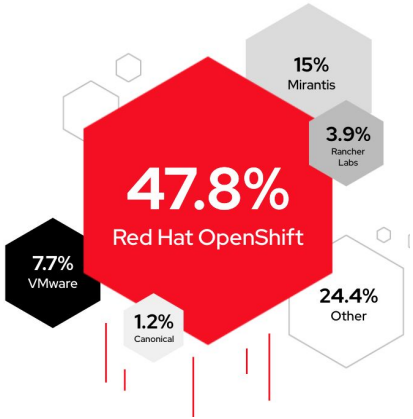
Red Hat

Red Hat portfolio

Red Hat's open source solutions work on bare metal and the public cloud ...

... and everything in between.

A multi-faceted platform for strategic goals



RED HAT OPENSHIFT

Container platform market share leader

OpenShift fleet management tools

Inventory ⁝ Policy ⁝ Compliance ⁝ Security ⁝ Configuration ⁝ Workloads

OpenShift platform

CaaS/PaaS

Ecosystem Partners

EDB POSTGRES AI

Multi Architectures

CPU

x86 arm ppc

ACCELERATORS

NVIDIA AMD intel Google aws IBM

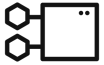
Physical Virtual Private cloud Sovereign cloud Public cloud Edge

Red Hat OpenShift

Red Hat

Red Hat OpenShift Application Platform

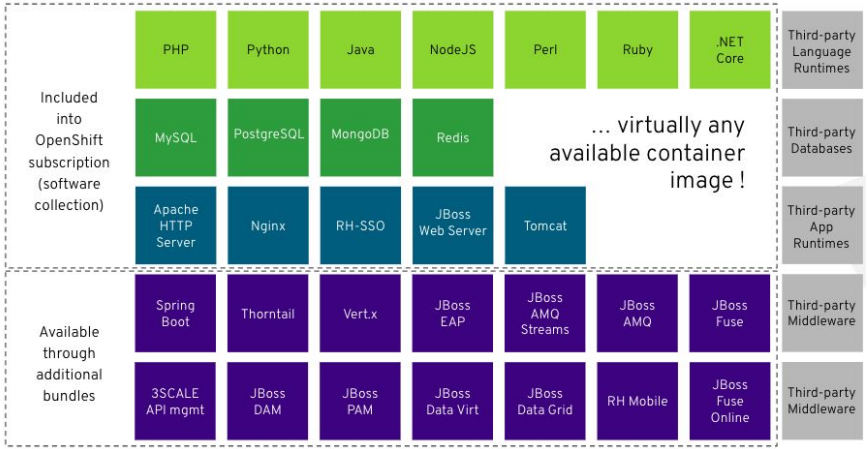
Trusted, comprehensive and consistent across hybrid cloud

Application Build	 Developer Tools	 Runtimes & languages	 Serverless & functions	 Cloud Services APIs	 Migration tools
Application Delivery	 CI/CD & GitOps	 Container registry	 Observability	 Logging	 Virtual machines
Application Security	 Service mesh	 Vulnerability scanning	 Security policies	 Cert & secret tooling	 Auth and SSO services

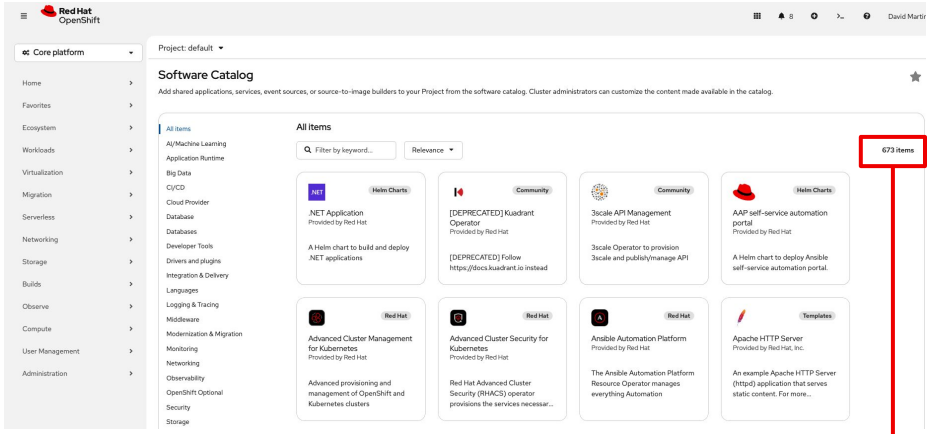
Red Hat

OpenShift is also content !

Application runtimes



Marketplace



+600 items

Red Hat

A large ecosystem

AI / ML

GIGASPACES, CognitiveScale, ProphetStor, H2O, SELDON, PerceptiLabs, UBIX

Customer Code

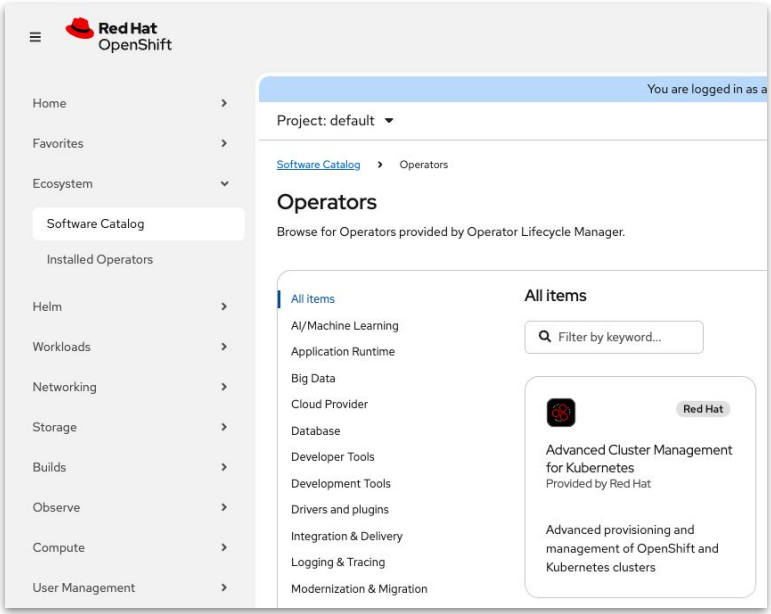
{ | }

DevOps Tools

JFrog, sonatype, spot, Kong, Solution-Soft, ESCALATE, IBM, Red Hat, VACAVA, psMx, here, KubeMQ

Monitoring & Logging

BROADCOM, sysdig, DATADOG, ZABBIX, INSTANA, dynatrace, IBM, Red Hat, turbonomic, sematext, APPDYNAMICS part of Cisco, New Relic, splunk>



Application Runtimes

GTSoftware, IBM, Joget, Red Hat, Lightbend

Databases & Big Data

memSQL, hazelcast, Starburst, EDB, redislabs HOME OF REDIS, crunchy data, ArangoDB, NUODB, PERCONA, PingCAP, IBM, Couchbase NoEQUAL, mongoDB., Cockroach LABS

Security

NeuVector, JETSTACK, SYNOPSYS, portshift, Twistlock, CYBER ARMOR, Zettaset, IBM, aqua, TREMOLO SECURITY, anchore, tufin Orca

Networking

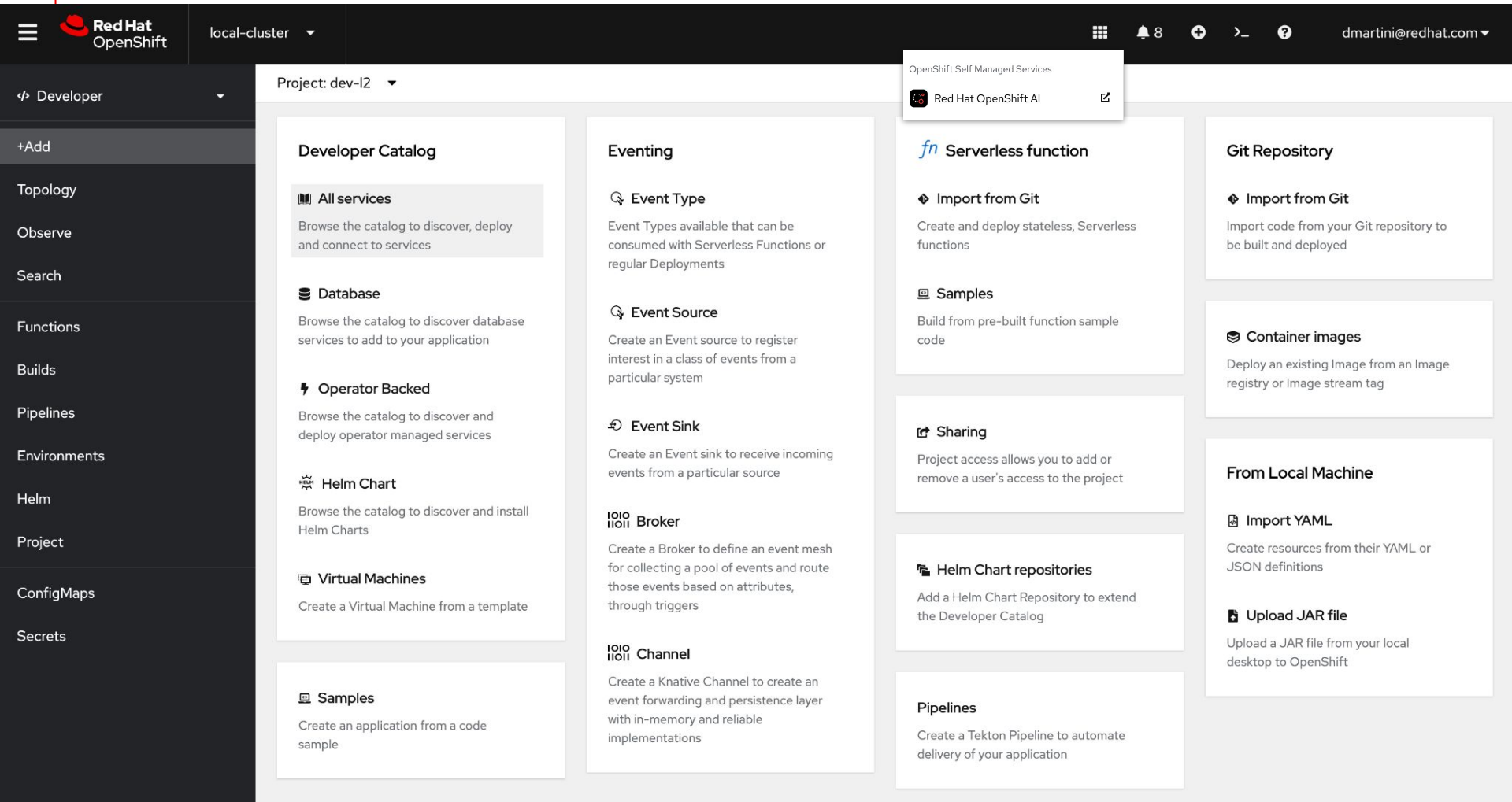
TIGERA, CONTAINOUS, f5, CITRIX

Storage

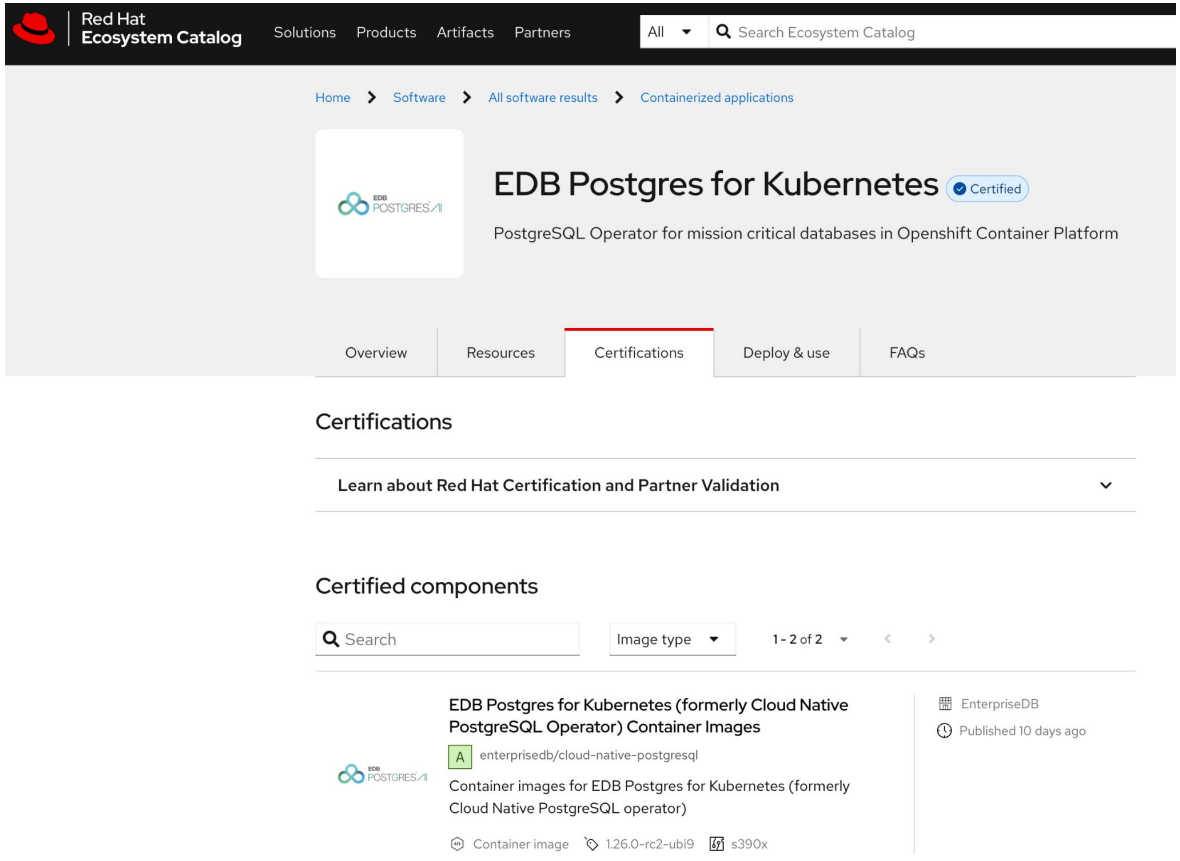
TRILIO, INFINIDAT, ROBIN Storage, Red Hat, portworx, Hewlett Packard Enterprise, kasten by Veeam, NUTANIX, STORAGEOS, IBM

Red Hat

aaS platform for users



Why Red Hat OpenShift for EDB: Operator Certification



EDB Postgres for Kubernetes is a certified Level 5 Operator for Red Hat OpenShift

- This is designed to streamline Day 2 operations of PostgreSQL databases
- Enhanced Database Management
- Supports point-in-time recovery (PITR)
- Ensures robust data protection and recovery options
- Integration with business continuity solutions such as Red Hat OpenShift API for Data Protection (OADP) and Veeam Kasten, Trilio, Portworx Backup, IBM Fusion, and others

Red Hat

Proven Solution at Scale

EDB Postgres on OpenShift use cases

- Cloud-Native Database Deployment
- Database as a Service (DBaaS)
- High Availability and Disaster Recovery (HA & DR)
- DevOps and Continuous Integration/Continuous Deployment (CI/CD)
- Microservices and Application Modernization
- Move from VMWare to OpenShift
- Data Security and Compliance (using TDE and Advanced Security provided by EPAS)
- Hybrid and Multi-Cloud Deployments
- Multi-Tenant Applications (isolation)
- AI/ML Sovereign Platform

Euro Information

Company profile

Euro-Information is the fintech company of the Crédit Mutuel group. Euro-Information manages the IT systems of 16 federations of Crédit Mutuel as well as those of CIC and of all the financial, insurance, property, consumer credit, private banking, financing, telephony and technological subsidiaries.

- Red Hat OpenShift
- EDB Postgres for Kubernetes
- PostgreSQL
- EPAS

- EDB considerably reduces IT costs associated with database maintenance.
- 280 cores: Enterprise Plan + Production Support

Summary

Use Case	On prem DBaaS (in Production)
Workload	Transactional
Application Name	All internal Postgres applications
EDB Tools of Interest	PostgreSQL and EDB Postgres for Kubernetes

Problem

- Fast database deployment
- Adopt a supported and secure Open Source platform
- Onprem DBaaS
- Align to in-house RDBMS standardization

Solution

- Use Postgres capabilities to build and maintain local applications
- Use Red Hat OpenShift platform to accelerate the provisioning of databases and applications

Results

- Applications running with PostgreSQL databases in a centralized environment
- Massive reduction of TCO of database service operations

Airbus

Company profile

Airbus SE is a European aerospace corporation. The company's primary business is the design and manufacturing of commercial aircraft but it also has separate defence and space and helicopter divisions.

- Red Hat OpenShift
- EDB Postgres for Kubernetes
- EPAS
- TDE

- Improve database deployment speed
- Reduce DB support
- Cost reduction

Summary

Use Case	On prem DBaaS (Production)
Workload	Transactional
Application Name	All VMWare PostgreSQL databases
EDB Tools of Interest	EDB Postgres Advanced Server with Oracle and TDE (optional)

Problem

- Flexibility
- Cost reduction
- New managed service in OpenShift

Solution

- EDB Postgres for Kubernetes with EPAS. Depending of the applications needs, EPAS and/or TDE will be activated

Results

- POC done
- Decision is taken
- Number of cores not yet communicated

Banque de France

Company profile

The Banque de France is France's central bank. A two-hundred-year-old institution, privately-owned when it was founded on January 18, 1800 under the Consulate by General Bonaparte, it became state-owned on January 1, 1946 when it was nationalized by General de Gaulle.

- Red Hat OpenShift
- CloudNativePG
- PostgreSQL

- 100 cores
- Subscription plan:
  - Community360 plan + Production Support

Summary

Use Case	OnPrem DBaaS (In production)
Workload	Transactional
Application Name	Multiple applications
EDB Tools of Interest	PostgreSQL, CloudNativePG

Problem

- Fast database deployment
- Provide containerized Postgres DBaaS

Solution

- Use OpenShift to provide this service with the operator
- Fast deployment and with Open Source database

Results

- OpenShift based PostgreSQL cluster deployments expand the internal offering alongside traditional VM based database cluster deployments

La Poste

Company profile

La Poste is a postal service company in France, operating in Metropolitan France, the five French overseas departments and regions and the overseas collectivity of Saint Pierre and Miquelon. Under bilateral agreements, La Poste also has responsibility for mail services in Monaco through La Poste Monaco and in Andorra alongside the Spanish company Correos.

- Red Hat OpenShift
- EDB Postgres for Kubernetes
- PostgreSQL

- EDB considerably reduces IT costs associated with database maintenance.
- 12 Cores: Standard Plan + Premium Support

Summary

Use Case	On prem DBaaS with HA and DR (In Production)
Workload	Transactional
Application Name	Portail XaaS
EDB Tools of Interest	PostgreSQL and EDB Postgres for Kubernetes

Problem

- Provide a database HA solution for Ansible Automation Platform (AAP)
- Database must be in HA and DR

Solution

- Use EDB Postgres for Kubernetes to provide a HA and DR solution for PostgreSQL databases
- Deploy in 2 OpenShift clusters our operator

Results

- La Poste developer can use their internal ‘La Post Service Portal’ to provision more than 64 backends.
- Reduce risk deploying EDB solutions.

Thank you

Red Hat is the world's leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make
Red Hat a trusted adviser to the Fortune 500.


linkedin.com/company/red-hat


youtube.com/user/RedHatVideos

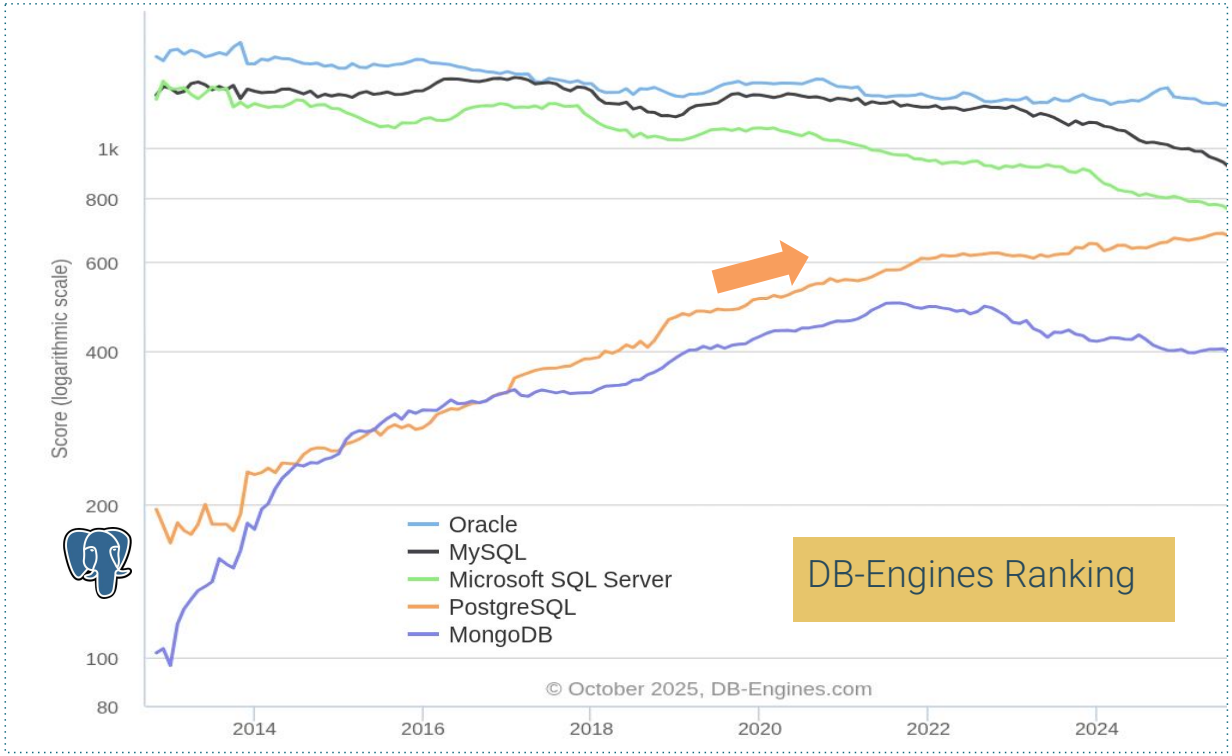

facebook.com/redhatinc


twitter.com/RedHat

Red Hat

Introduction to Postgres and EDB

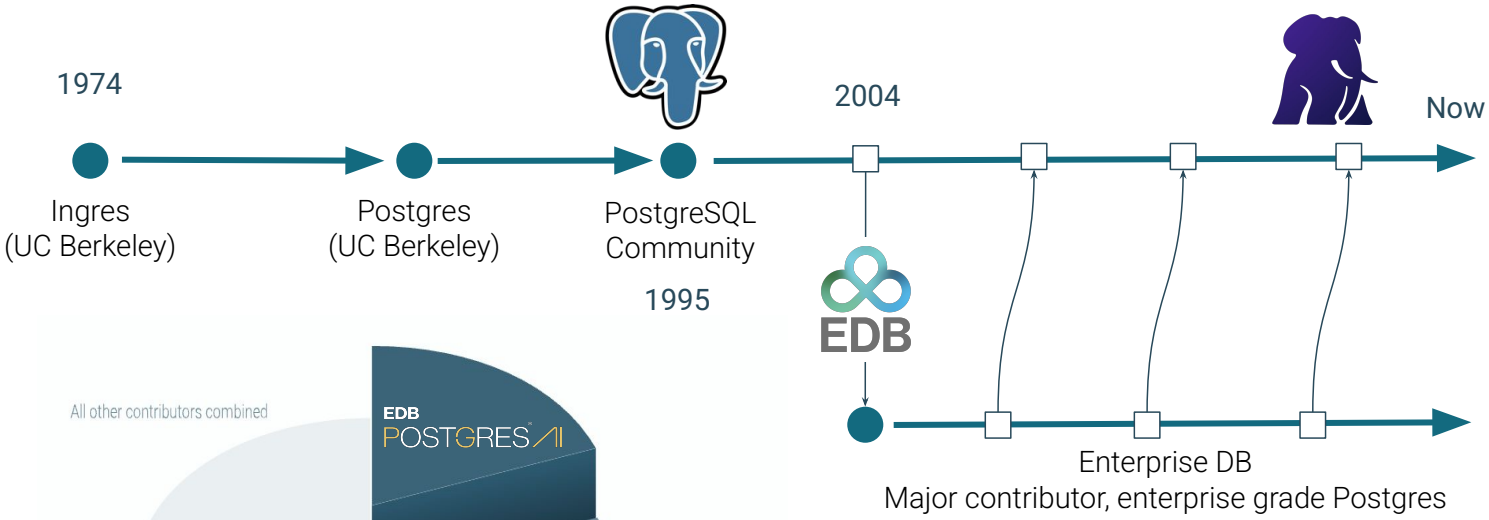
PostgreSQL is Booming !

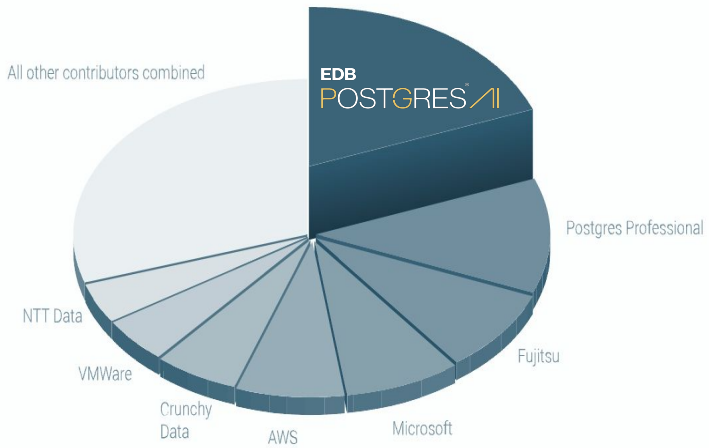


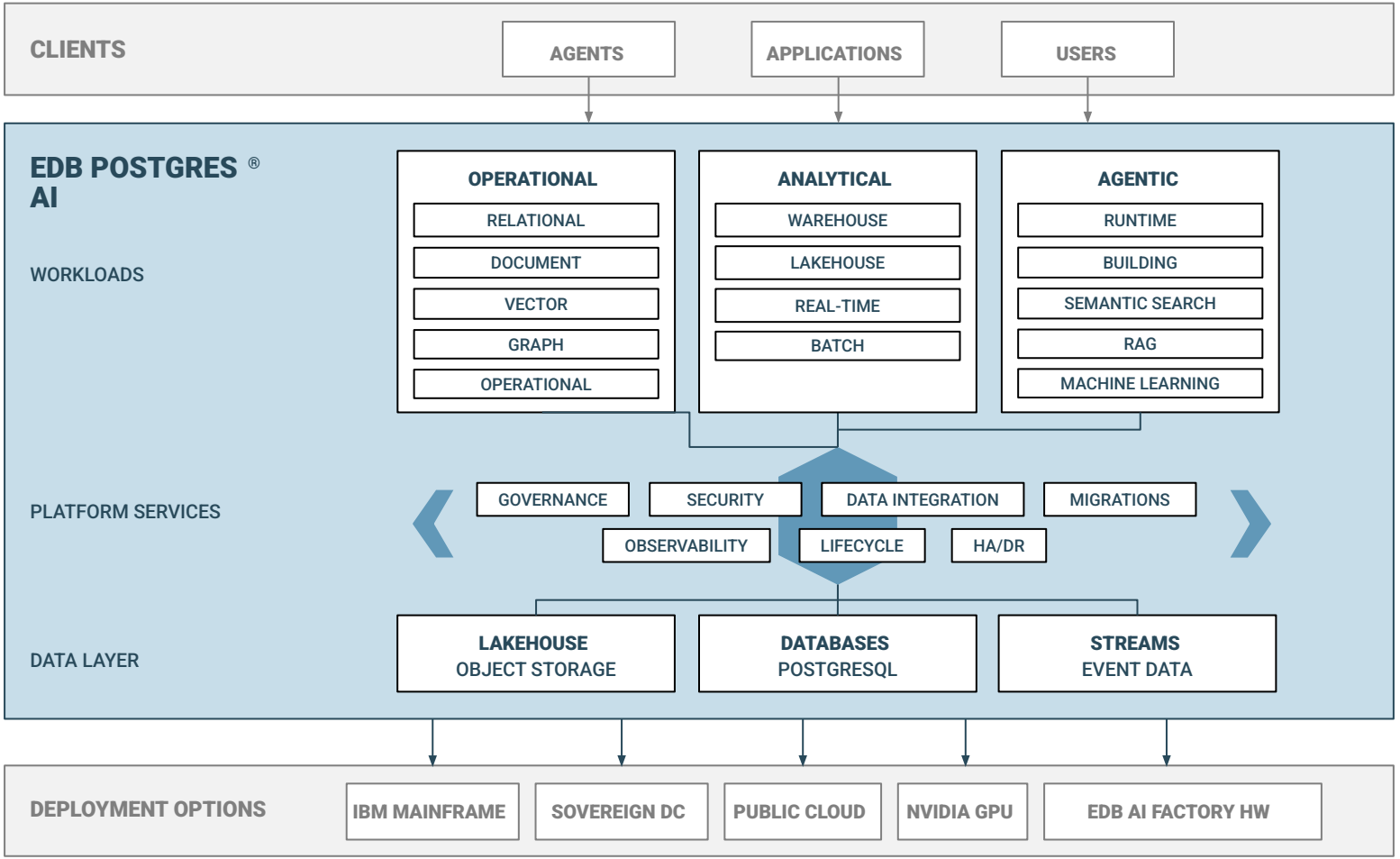
35%

Thirty-five percent of Enterprises will consider Postgres for their next project

EnterpriseDB Biggest PostgreSQL Contributor







CNPG Operator: Reference Architecture and functionalities

Kubernetes timeline

- 2014, June: Google open sources Kubernetes
- 2015, July: Version 1.0 is released
- 2015, July: Google and Linux Foundation start the CNCF
- 2016, November: The operator pattern is introduced in a blog post
- 2018, August: The Community takes the lead
- 2019, April: Version 1.14 introduces Local Persistent Volumes
- 2019, August: EDB team starts the Kubernetes initiative
- 2020, June: we publish this blog about benchmarking local PVs on bare metal
- 2020, June: Data on Kubernetes Community founded
- 2021, February: EDB Cloud Native Postgres (CNP) 1.0 released
- 2022, May: EDB donates CNP and open sources it under CloudNativePG
- 2025, January: CloudNativePG was recognized as an official #CNCF project

A kubernetes operator for Postgres

Kubernetes adoption is rising and it is already the de facto standard orchestration tool

PostgreSQL clusters “management the kubernetes way” enables many cloud native usage patterns, e.g. spinning up, disposable clusters during tests, one cluster per microservice and one database per cluster

CNPG tries to encode years of experience managing PostgreSQL clusters into an Operator which should automate all the known tasks a user could be willing to do

Our PostgreSQL operator must simulate a part of the work of a DBA

Autopilot

It automates the steps that a human operator would do to deploy and to manage a Postgres database inside Kubernetes, including automated failover.

Security

CloudNativePG is secured by default.

It doesn’t rely on statefulsets and uses its own way to manage persistent volume claims where the PGDATA is stored.

Data persistence

Designed for Kubernetes

It’s entirely declarative, and directly integrates with the Kubernetes API server to update the state of the cluster — for this reason, it does not require an external failover management tool.

Features

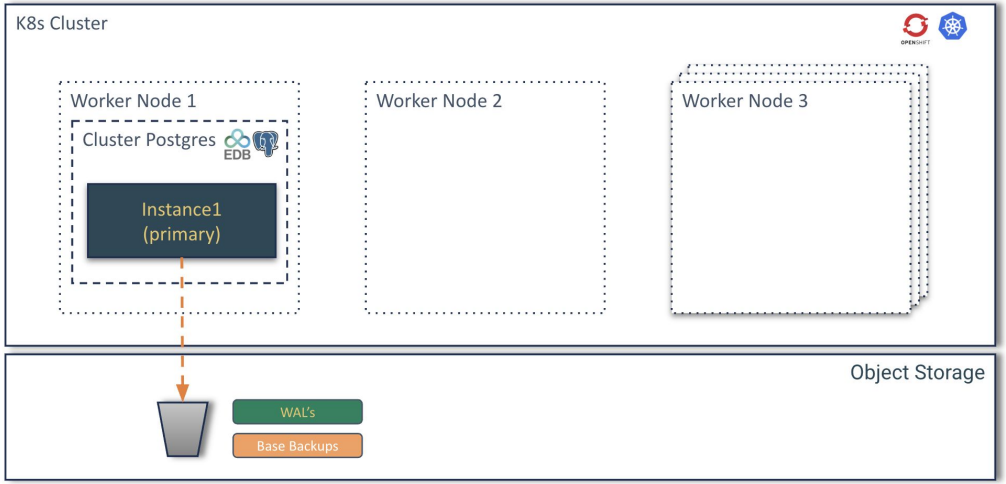
Deployment	Administration	Backup & Recovery	Monitoring	Security	High Availability
Kubernetes operator	Single node	Backup	Prometheus	TDE	Switchover
Kubernetes plugin	Cluster (Multi node)	Recovery	Grafana dashboards	Certificates	Failover
EDB Postgres (EPAS)	PostgreSQL configuration	PITR	Postgres Enterprise Manager	Data redaction	Scale out / scale down
PostGIS	Pooling	Volume Snapshots	Logging	Password management	Minor / Major updates

Use Cases

Use case 1 architecture

A single database is the simplest setup, involving one instance of a database server.

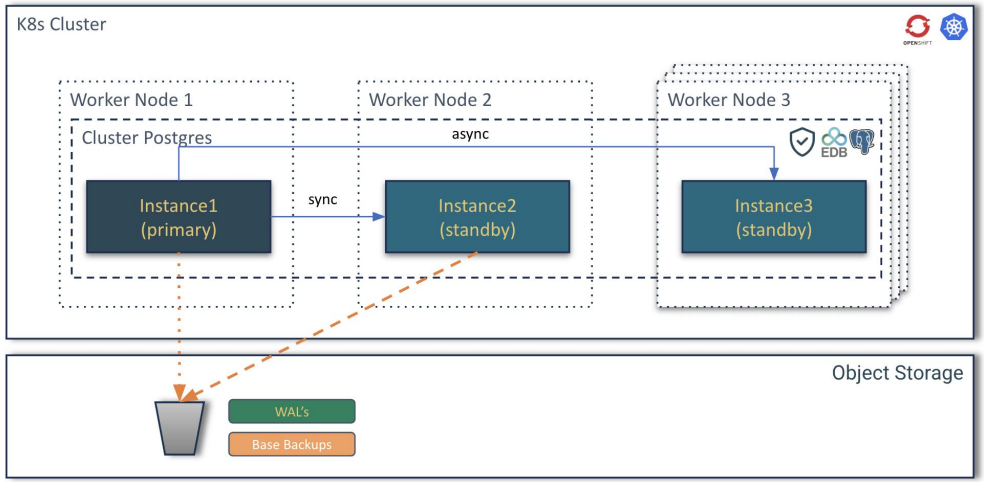
- Development and testing environments
- Small applications with low traffic
- Non-critical data analysis
- Applications with high tolerance for downtime
- Cost-sensitive projects



Use case 2 architecture

An HA database setup aims to minimize downtime by having redundant components. If one component fails, another takes over automatically or with minimal intervention. This usually involves techniques like clustering, replication, or mirroring within the same data center or availability zone.

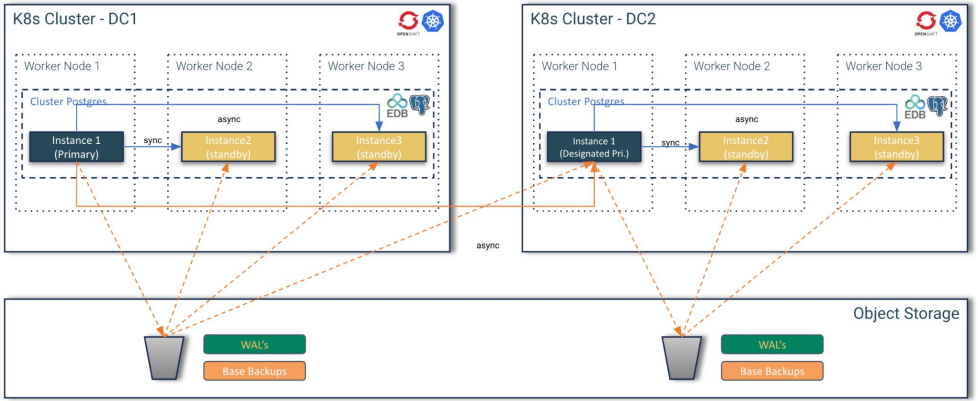
- Business critical Applications
- Applications with stringent SLAs
- Real-time systems
- Improving user experience
- Minimizing planned downtime



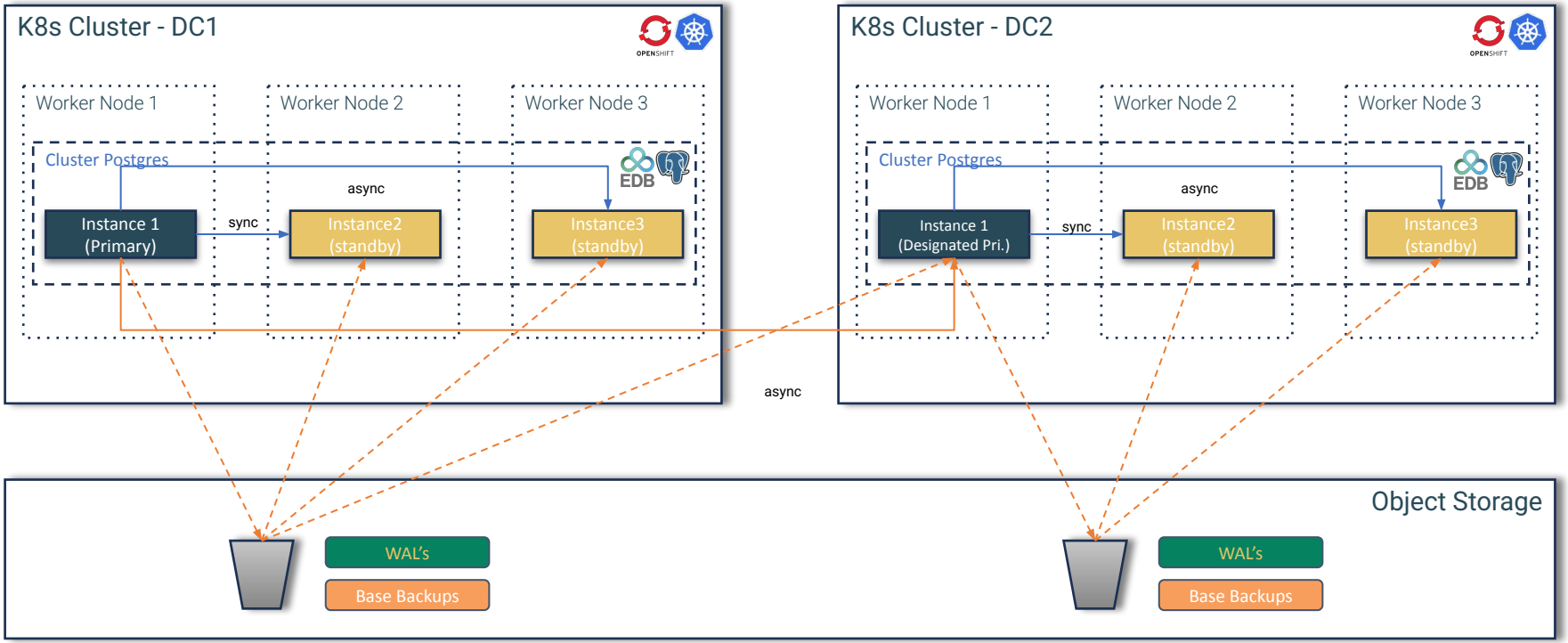
Use case 3 architecture

A DR database setup focuses on protecting data and ensuring business continuity in the event of a large-scale disaster affecting an entire data center or region (e.g., natural disasters, power outages, cyberattacks). This typically involves replicating data to a geographically separate location.

- Regulatory compliance
- Protecting against catastrophic data loss
- Ensuring business continuity for mission-critical systems



Use case 3 architecture



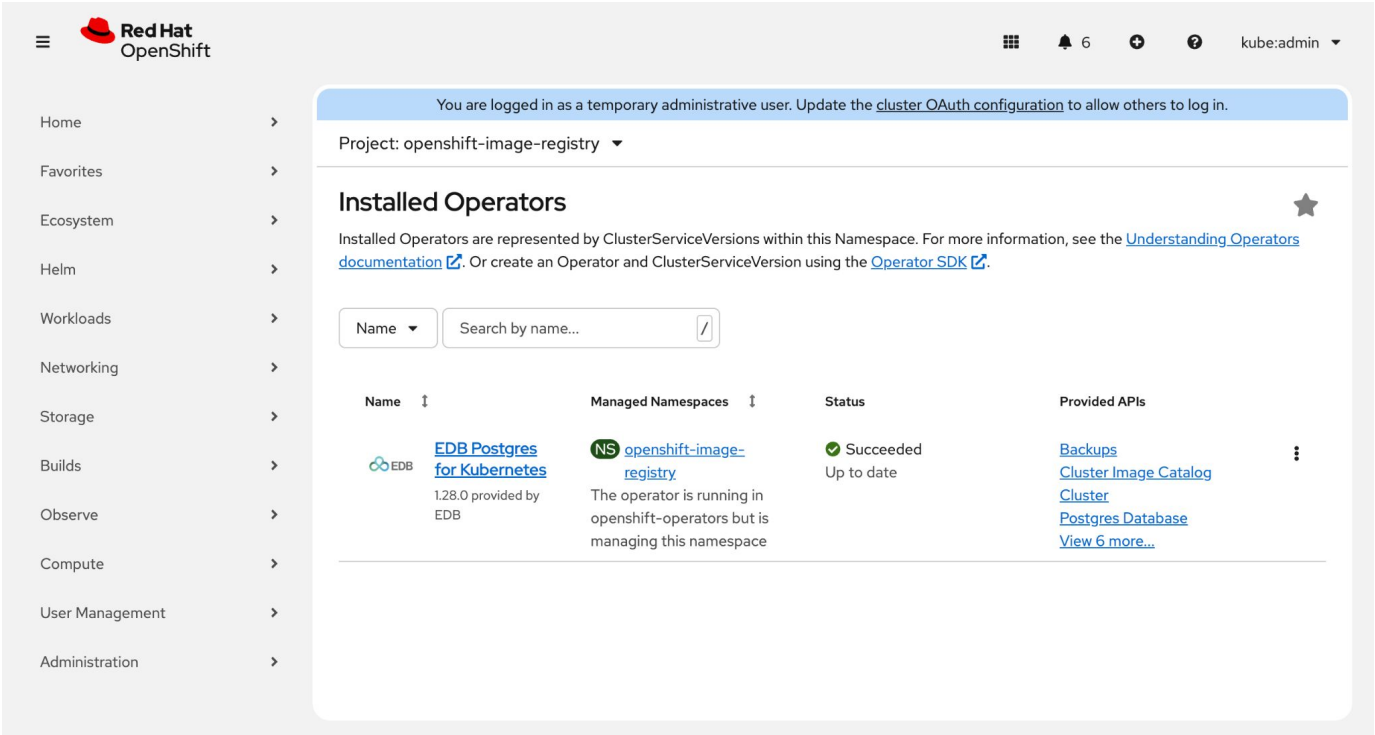
Interactive session
It’s time to go hands-on!

Environment

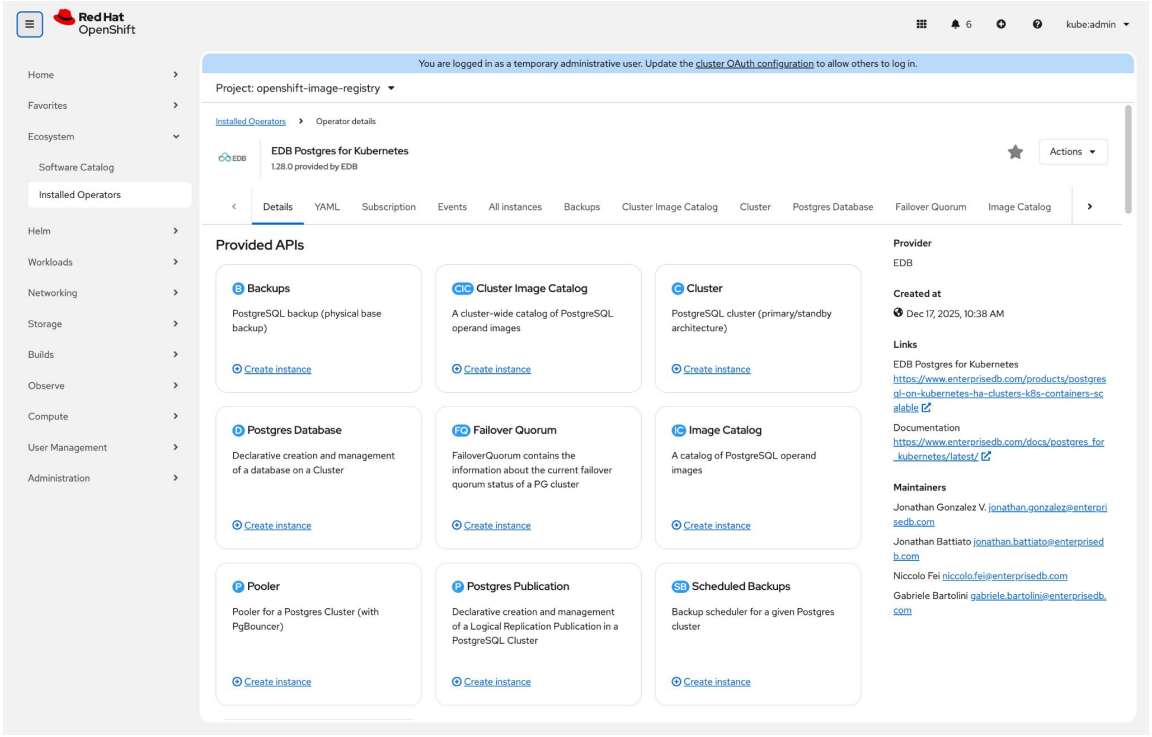
OpenShift Console	https://console-openshift-console.apps.cluster-ldqnq.ldqnq.sandbox2860.opentlc.com
OpenShift User name	user1..user30
OpenShift Password	edbredhat
Minio UI	https://minio-ui-minio.apps.cluster-ldqnq.ldqnq.sandbox2860.opentlc.com/
Minio internal service	http://minio-service.minio.svc.cluster.local:9000
Minio API	https://minio-api-minio.apps.cluster-ldqnq.ldqnq.sandbox2860.opentlc.com/
Minio User	minio
Minio password	edb-workshop

Red Hat OpenShift Login

Red Hat OpenShift environment



Red Hat OpenShift operator



Use case
The environment

Features shown during the demo

- Kubernetes plugin install
- Check the CloudNativePG operator status
- Postgres cluster install
- Insert data in the cluster
- Failover
- Backup
- Recovery
- Scale out/down
- Fencing
- Hibernation
- Monitoring
- Rolling updates (minor and major)

Last CloudNativePG tested version is 1.25

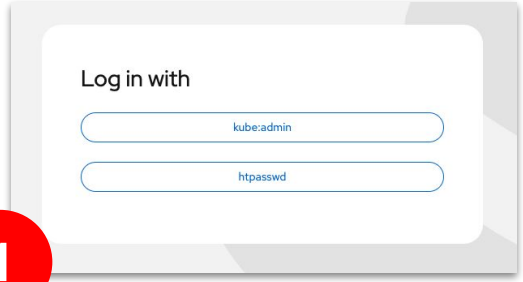
This demo is in

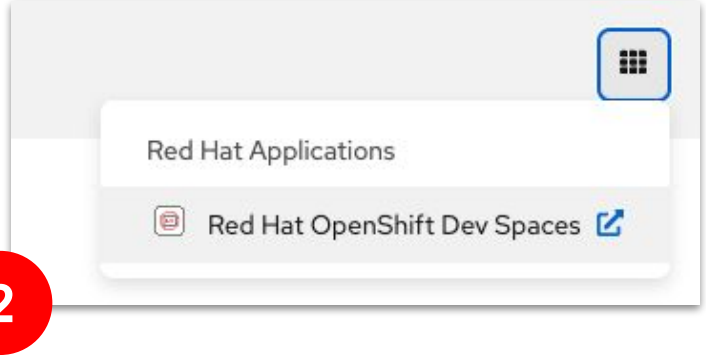
https://github.com/raphael-chir/edb-postgres-for-kubernetes-in-openshift

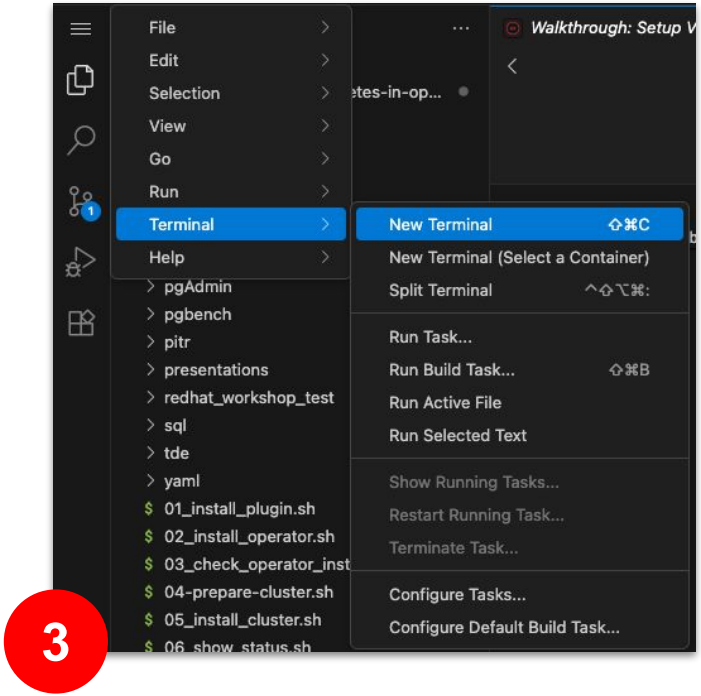
Hands-on

OpenShift and DevSpaces access

https://github.com/raphael-chir/edb-postgres-for-kubernetes-in-openshift







Plugging and OpenShift login

Scripts

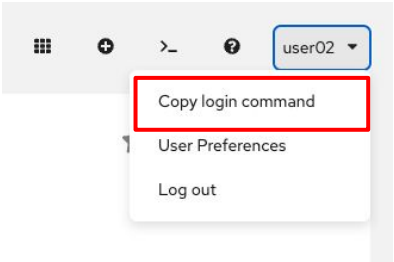
`01_install_plugin.sh`

Install plugin

CloudNativePG provides a plugin for kubectl to manage a Postgres cluster in Kubernetes. You can manage status, promote, certificates, restart, reload, maintenance, report, logs, destroy, hibernation, backups, …

Doc link

OpenShift Login and namespace creatop,



Code

```
# Install plugin

curl -sSfL \
  https://github.com/EnterpriseDB/kubectl-cnp/raw/main/install.sh
| sh -s -- -b .


# Login to the OpenShift Cluster

oc login ...

# Create your namespace

oc new-project userXX
```

Checking operator installation

Scripts

`03_check_operator_installed.sh`

Checking operator installation

In Kubernetes, the operator is by default installed in the postgresql-operator-system namespace as a Kubernetes Deployment. The name of this deployment depends on the installation method. When installed through the manifest or the cnp plugin, by default, it is called postgresql-operator-controller-manager. When installed via Helm, by default, the deployment name is derived from the helm release name, appended with the suffix -edb-postgres-for-kubernetes (e.g., <name>-edb-postgres-for-kubernetes).

Doc link

Code

```
# Check operator installation

oc get deploy -A | grep postgres
NAMESPACE:      openshift-operators
NAME:           postgresql-operator-controller-manager
READY:          1/1
UP-TO-DATE:     1
AVAILABLE:      1
AGE:            5d21h
```

Install Postgres cluster

Scripts

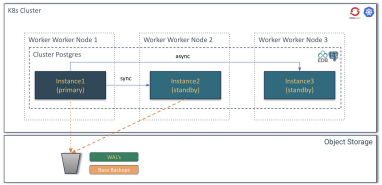
```
04-prepare-cluster.sh
05_install_cluster.sh
06_show_status.sh
07_insert_data.sh
```

Prepare Postgres cluster template

Setup credentials (AWS or Minio) and prepare context for your users

Deploy Postgres cluster in HA

Different types of architectures can be deployed. In our demo, 3 instances are deployed with HA available.



Insert Data

Insert data in `test` table

Doc link

Code

```
export cluster_name="cluster-userXX"
export TMP=$TMPDIR

# Get template
envsubst < templates/${cluster_name}-template.yaml >
$TMP/${cluster_name}.yaml

# Deploy Postgres cluster
${kubectl_cmd} apply -f $TMP/${cluster_name}.yaml
---------------------------------------------------------------------------
ApiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: cluster-example
spec:
 instances: 3
 imageName: "docker.enterprisedb.com/k8s/edb-postgres-advanced:17.6"
 enableSuperuserAccess: true

 replicationSlots:
   highAvailability:
     enabled: true
```

Monitoring

Show status

```
Cluster Summary
Name                     default/cluster-sergio-1
System ID:               7566239054312464401
PostgreSQL Image:        docker.enterprisedb.com/k8s/edb-postgres-advanced:17.6
Primary instance:        cluster-sergio-1
Primary promotion time:  2025-10-28 11:52:15 +0000 UTC (4m30s)
Status:                  Cluster in healthy state
Instances:               3
Ready instances:         3
Size:                    127M
Current Write LSN:       0/6032300 (Timeline: 1 - WAL File: 000000010000000000000006)

Continuous Backup status (Barman Cloud Plugin)
ObjectStore / Server name:      object-store/cluster-sergio
First Point of Recoverability:  -
Last Successful Backup:         -
Last Failed Backup:             -
Working WAL archiving:          OK
WALs waiting to be archived:    0
Last Archived WAL:              000000010000000000000005.00000028.backup   @   2025-10-28T11:54:11.908195Z
Last Failed WAL:                -

Streaming Replication status
Replication Slots Enabled
Name              Sent LSN   Write LSN  Flush LSN  Replay LSN  Write Lag  Flush Lag  Replay Lag  State      Sync State  Sync Priority  Replication Slot
----              --------   ---------  ---------  ----------  ---------  ---------  ----------  -----      ----------  -------------  ----------------
cluster-sergio-2  0/9000000  0/9000000  0/9000000  0/9000000   00:00:00   00:00:00   00:00:00    streaming  quorum      1              active
cluster-sergio-3  0/9000000  0/9000000  0/9000000  0/9000000   00:00:00   00:00:00   00:00:00    streaming  quorum      1              active

Instances status
Name              Current LSN  Replication role  Status  QoS         Manager Version  Node
----              -----------  ----------------  ------  ---         ---------------  ----
cluster-sergio-1  0/9000000    Primary           OK      Guaranteed  1.28.0           ocp4-multiarch-worker1-ppc64
cluster-sergio-2  0/9000000    Standby (sync)    OK      Guaranteed  1.28.0           ocp4-multiarch-w5gg4-master-2
cluster-sergio-3  0/9000000    Standby (sync)    OK      Guaranteed  1.28.0           ocp4-multiarch-w5gg4-master-1
```

Promote and Failover

Scripts

08_promote.sh

Promote

The meaning of this command is to promote a pod in the cluster to primary, so you can start with maintenance work or test a switch-over situation in your cluster:

Doc link

Code

```
# Promote

. ./config.sh

${kubectl_cnp} promote ${cluster_name} ${cluster-name}-2
```

Minor upgrade

Scripts

09_upgrade.sh

Minor upgrade

To upgrade to a newer minor version, simply update the PostgreSQL container image reference in your cluster definition, either directly or via image catalogs. CloudNativePG will trigger a rolling update of the cluster, replacing each instance one by one, starting with the replicas. Once all replicas have been updated, it will perform either a switchover or a restart of the primary to complete the process.

Doc link

Code

```
# Minor upgrade

# From
apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: cluster-sergio1
spec:
 instances: 3
 imageName:
"docker.enterprisedb.com/k8s/edb-postgres-advanced:17.6"
...

# To
apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: cluster-sergio1
spec:
 instances: 3
 imageName:
"docker.enterprisedb.com/k8s/edb-postgres-advanced::17.7"
...
```

Backups

Scripts

```
10_backup_cluster.sh
11_backup_describe.sh
```

Backup

Backups are used using Barman cloud with the new Barman Plugin. In case the plugin is not compatible (like in Red Hat OpenShift), we will continue using Barman Cloud (without the plugin).

PostgreSQL natively provides first class backup and recovery capabilities based on file system level (physical) copy. These have been successfully used for more than 15 years in mission critical production databases, helping organizations all over the world achieve their disaster recovery goals with Postgres.

Doc link

Code

```
cat $TMP/backup.yaml

apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Backup
metadata:
 name: cluster-example-backup-test
spec:
 cluster:
   name: cluster-example

${kubectl_cmd} apply -f ./yaml/backup.yaml
```

Recovery

Scripts

12_restore_cluster.sh

Recovery

In PostgreSQL, recovery refers to the process of starting an instance from an existing physical backup. PostgreSQL's recovery system is robust and feature-rich, supporting Point-In-Time Recovery (PITR)—the ability to restore a cluster to any specific moment, from the earliest available backup to the latest archived WAL file.

Doc link

Code

```
cat $TMP/restore.yaml

apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: cluster-restore
spec:
 instances: 1
 imageName: quay.io/enterprisedb/postgresql:16.4
 imagePullPolicy: IfNotPresent
 bootstrap:
   recovery:
     source: source
 plugins:
 - name: barman-cloud.cloudnative-pg.io
   isWALArchiver: true
   parameters:
     barmanObjectName: object-store
 externalClusters:
 - name: source
   plugin:
     name: barman-cloud.cloudnative-pg.io
     parameters:
       barmanObjectName: object-store
       serverName: cluster-example

${kubectl_cmd} apply -f $TMP/restore.yaml
```

Promote and Failover

Scripts

`13_failover.sh`

Automated Failover

In the case of unexpected errors on the primary for longer than the `.spec.failoverDelay` (by default 0 seconds), the cluster will go into failover mode. This may happen, for example, when:

- The primary pod has a disk failure
- The primary pod is deleted
- The `postgres` container on the primary has any kind of sustained failure

In the failover scenario, the primary cannot be assumed to be working properly.
Doc link

Code

```
# Automatic Failover (simulate failover killing pods of primary)
${kubectl_cmd} delete pvc/${primary} \
                      pvc/${primary}-wal \
                      pod/${primary} \
                      --force
```

Scale out, scale down

Scripts

```
14_scale_out.sh
15_scale_down.sh
```

Scale out and down

The operator allows you to scale up and down the number of instances in a PostgreSQL cluster. New replicas are started up from the primary server and participate in the cluster's HA infrastructure.

Doc link

Code

```
# Scale out
kubectl scale cluster cluster-example --replicas=4


# Scale down
kubectl scale cluster cluster-example --replicas=2
```

Fencing and Hibernation

Scripts

```
30_fencing_on.sh
31_fencing_off.sh
```

Fencing

Fencing is the process of protecting the data in one, more, or even all instances of a PostgreSQL cluster when they appear to be malfunctioning. When an instance is fenced, the PostgreSQL server process is guaranteed to be shut down, while the pod is kept running. This ensures that, until the fence is lifted, data on the pod isn't modified by PostgreSQL and that you can investigate file system for debugging and troubleshooting purposes.

Doc link

Code

```
# Fencing on
${kubectl_cnp} fencing on ${cluster_name} ${replica}

# Fencing off
${kubectl_cnp} fencing off ${cluster_name} ${replica}
```

Fencing and Hibernation

Scripts

```
32_hibernation_on.sh
33_hibernation_off.sh
```

Hibernation

CloudNativePG supports hibernation of a running PostgreSQL cluster in a declarative manner, through the cnpg.io/hibernation annotation. Hibernation enables saving CPU power by removing the database pods while keeping the database PVCs. This feature simulates scaling to 0 instances.

Doc link

Code

```
# Hibernation on
${kubectl_cmd} annotate cluster ${cluster_name} \
            --overwrite k8s.enterprisedb.io/hibernation=on

# Hibernation off
${kubectl_cmd} annotate cluster ${cluster_name} \
            --overwrite k8s.enterprisedb.io/hibernation=off
```

Major upgrade (copying data)

Scripts

```
20_upgrade_major_version.sh
```

Major upgrade (with native logical replication)
Major PostgreSQL releases introduce changes to the internal data storage format, requiring a more structured upgrade process.

CloudNativePG supports three methods for performing major upgrades:

- Logical dump/restore – Blue/green deployment, offline.
- Native logical replication – Blue/green deployment, online.
- **Physical with pg_upgrade** – In-place upgrade, offline (covered in the "Offline In-Place Major Upgrades" section below).

Doc link

Code

```yaml
apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: ${cluster_major_upgrade}
spec:
 instances: 1
 imageName: ${postgres_major_upgrade_image}

 storage:
   size: 2Gi

 bootstrap:
   initdb:
     import:
       type: microservice
       databases:
         - app
       source:
         externalCluster: ${cluster_name}

 externalClusters:
   - name: ${cluster_name}
     connectionParameters:
       # Use the correct IP or host name for the source database
       host: ${cluster_name}-rw
       user: postgres
       dbname: postgres
     password:
       name: ${cluster_name}-superuser
       key: password
```

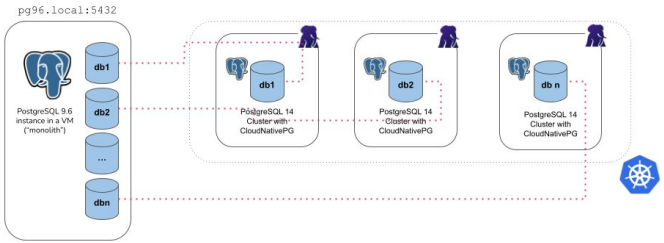
Major upgrade (copying data)

Scripts

`20_upgrade_major_version.sh`

Major upgrade (with native logical replication)

- In this step we will migrate the app database from our existing cluster to the new cluster
- We will create the yaml file with the setting "**import**" in the bootstrap section
- The operator uses internally postgres tools pg_dump and pg_restore
- This method can be used to **migrate** another database or to run **out-of-the-place upgrade**
- Possible settings:
  - Microservices
  - Monolith



Code

```yaml
apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: cluster-sergio-major
spec:
 instances: 1
 imageName: docker.enterprisedb.com/k8s/postgresql:18.1
 imagePullSecrets:
 - name: postgresql-operator-pull-secret
 enableSuperuserAccess: true

 bootstrap:
   initdb:
     import:
       type: microservice
       databases:
         - postgres
       source:
         externalCluster: cluster-sergio

 externalClusters:
   - name: cluster-sergio
     connectionParameters:
       host: cluster-sergio-rw
       user: postgres
       dbname: postgres
     password:
       name: cluster-sergio-superuser
       key: password
...
```

Nos prochains évènements

- **18 Mars :** Exclusive Virtual Wine Tasting
- **Mars/Avril :** Workshop/webinar en partenariat avec ACMI
- **23 Mars :** KubeCon + CloudNativeCon (Retex HSBC)
- **18 Juin :** Club Utilisateurs EDB (Retex Michelin)

Thank you

For any queries, contact:
sergio.romera@enterprisedb.com

Backup Slides

Architectures

When to choose Kubernetes over VMs?

01 | Cloud Native Applications that already run in Kubernetes

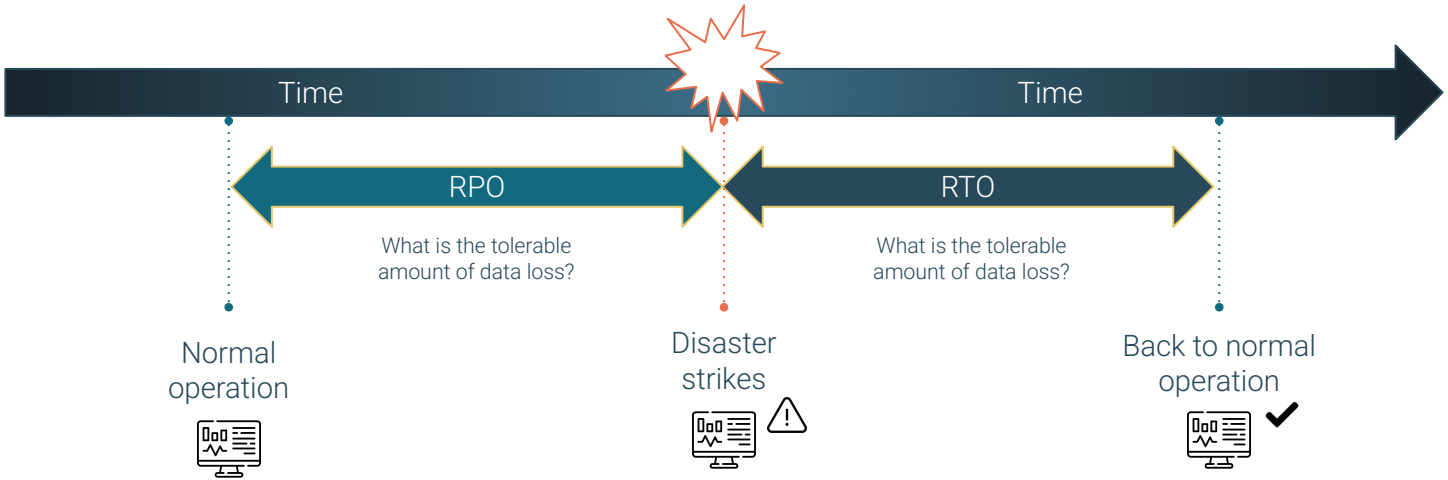
02 | Scalable, replicated databases

03 | Applications requiring automated failover and self-healing

04 | Teams skilled in Kubernetes who want a unified infrastructure

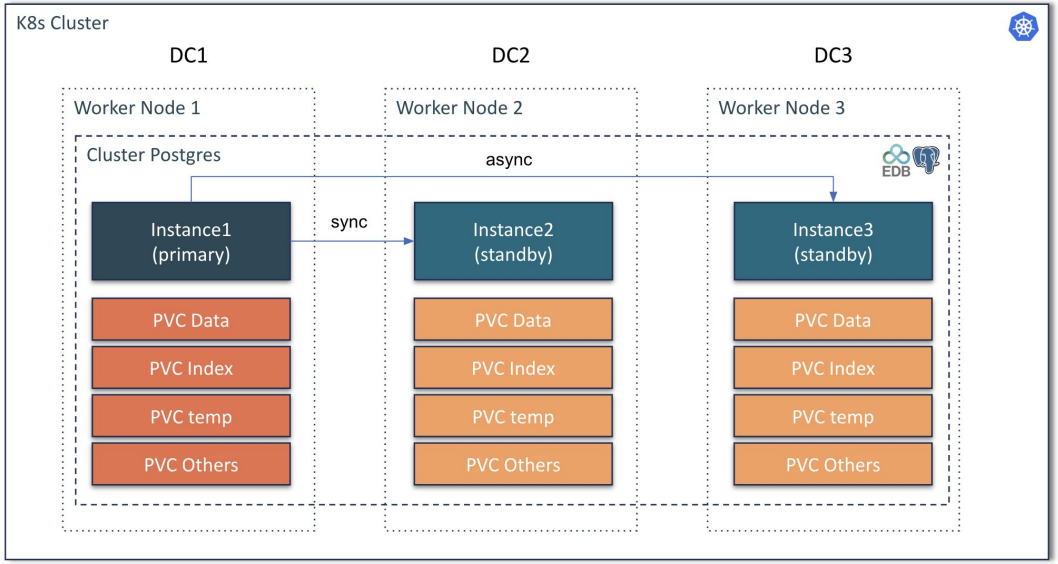
Concepts

- Recovery Point Objective (**RPO**) and Recovery Time Objective (**RTO**) are key concepts in disaster recovery and business continuity planning, particularly related to data loss and system downtime.



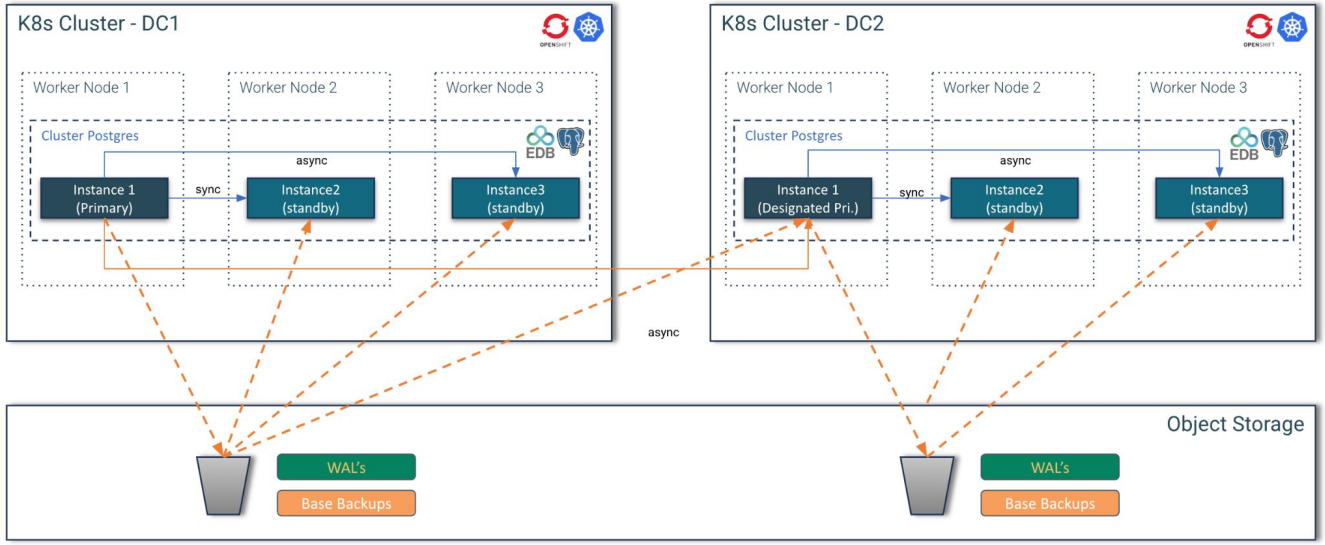
Red Hat Recommendation

Red Hat recommend stretched clusters ONLY when latencies don't exceed 5 milliseconds (ms) round-trip time (RTT) between the nodes in different locations, with a maximum RTT of 10 ms.

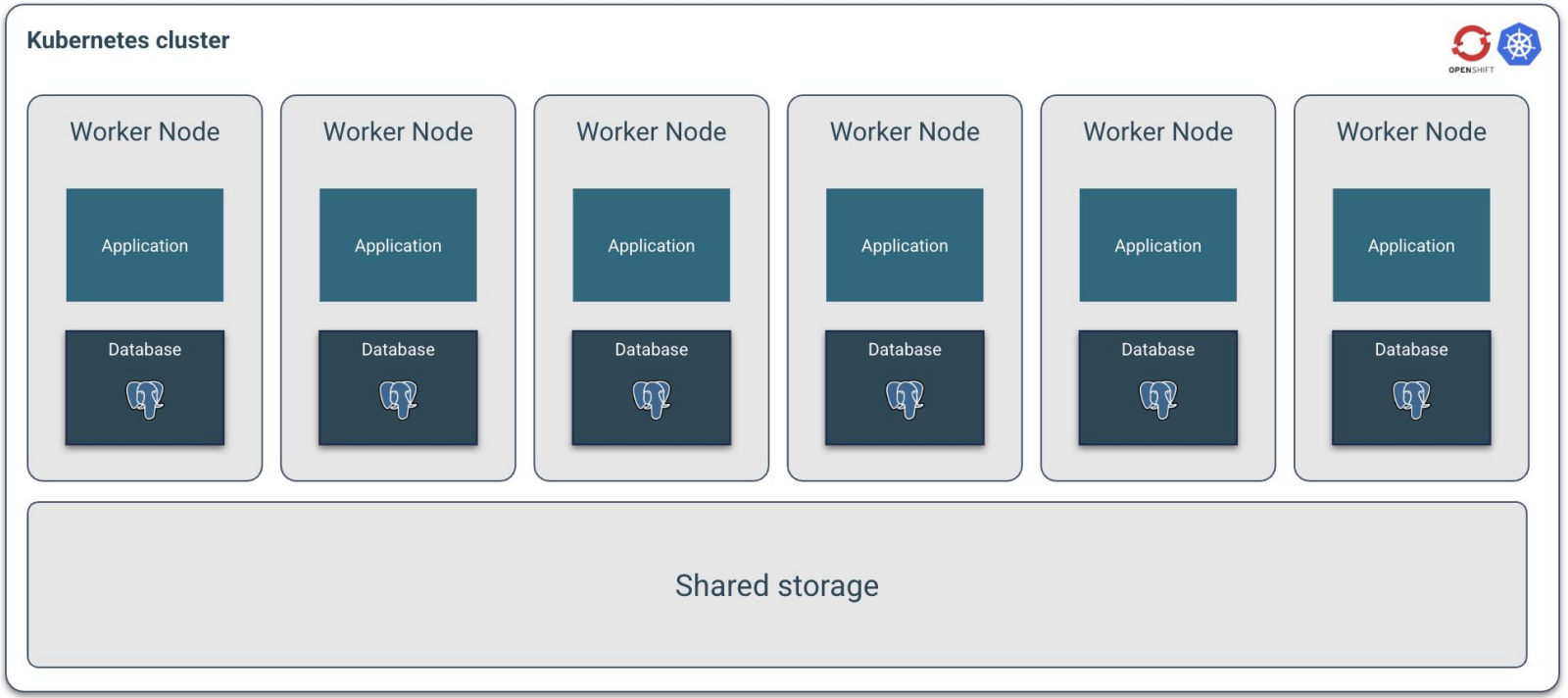


Two separate single data center Kubernetes clusters

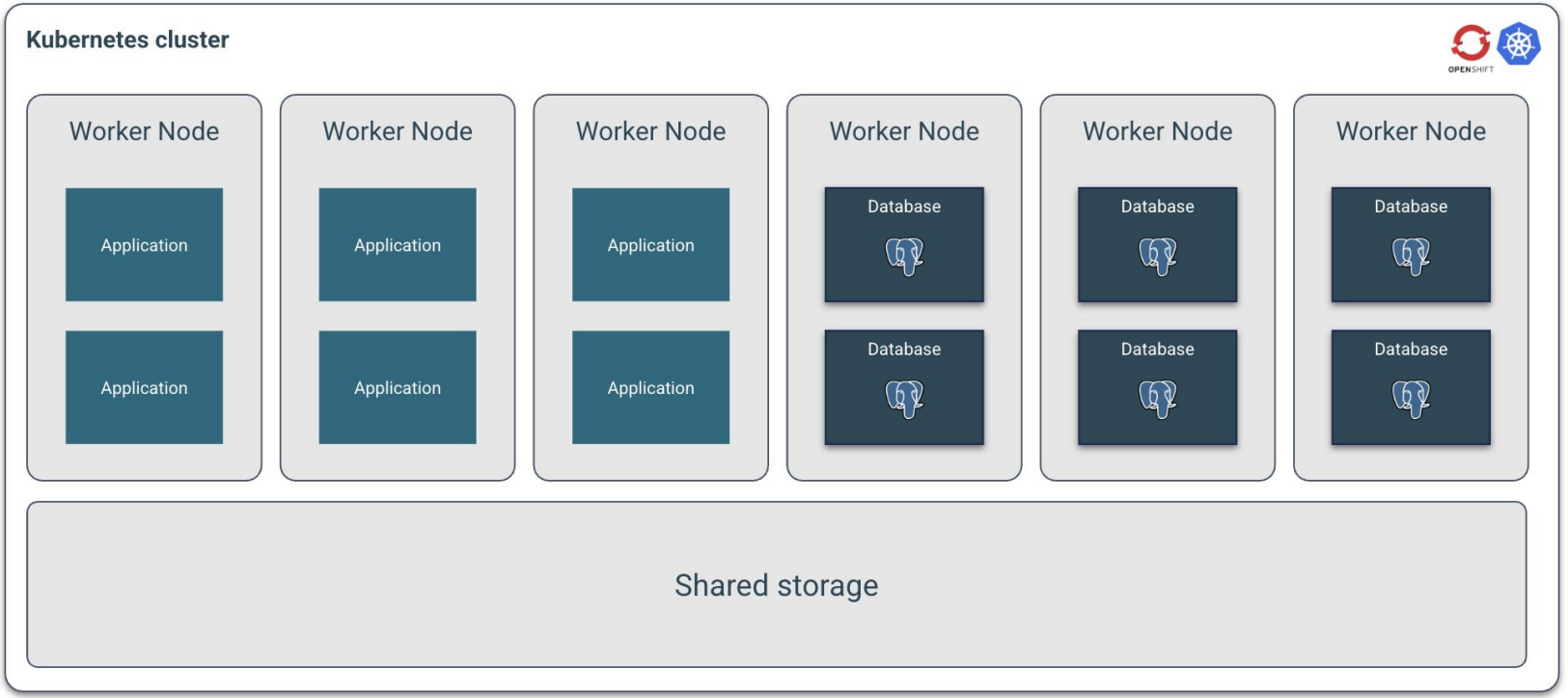
In case you cannot go beyond two data centers and you end up with two separate Kubernetes clusters, don't despair.



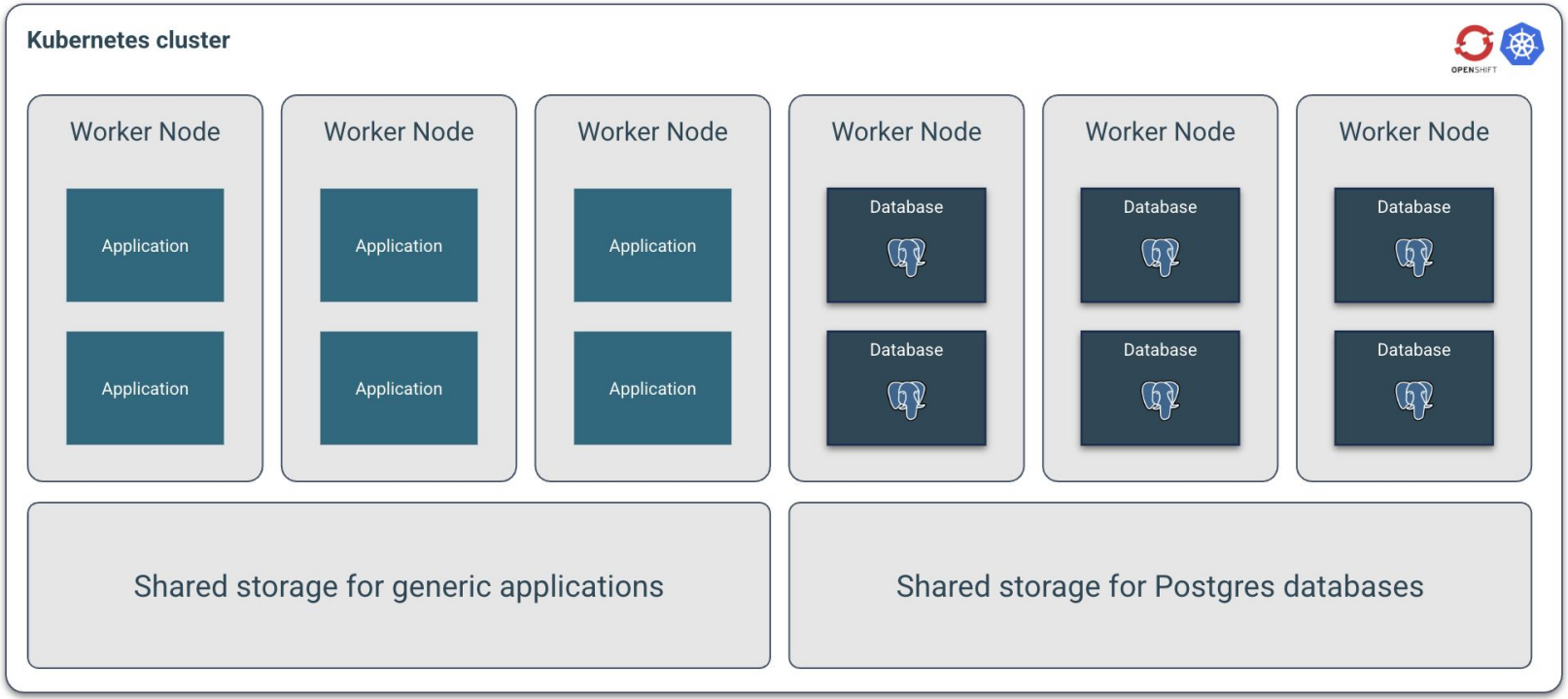
Shared workload, shared storage



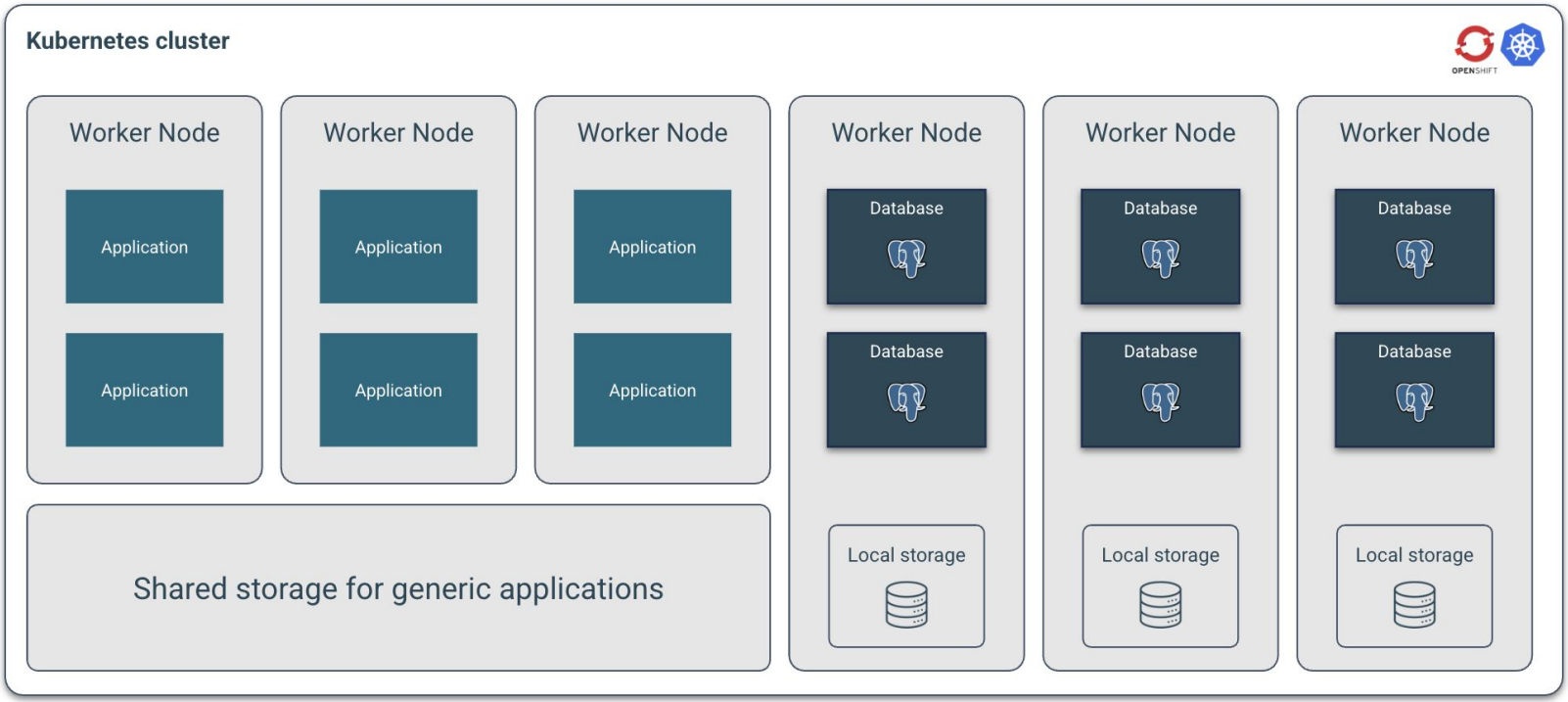
Shared workload, shared storage



Shared workload, shared storage

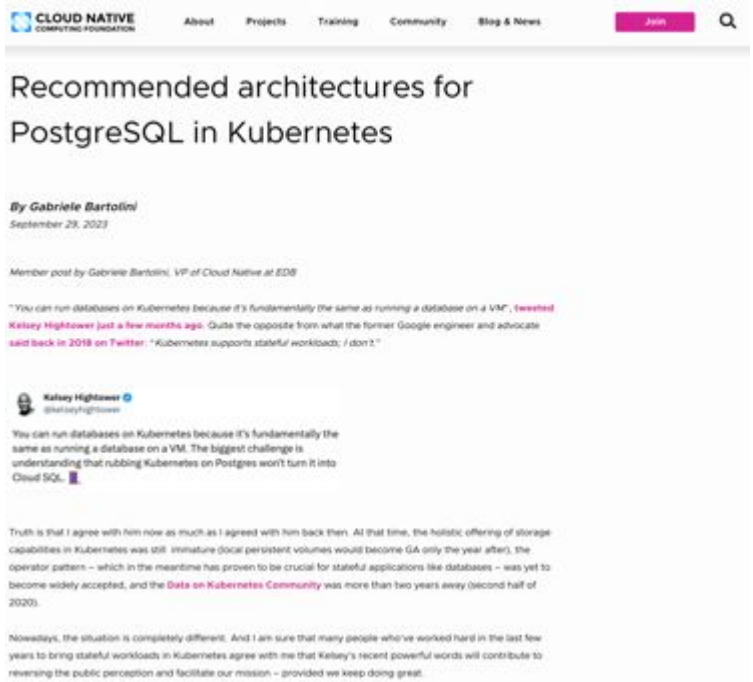


Shared workloads, local storage



Recommended architectures

https://www.cncf.io/blog/2023/09/29/recommended-architectures-for-postgresql-in-kubernetes/



CLOUD NATIVE COMPUTING FOUNDATION

About Projects Training Community Blog & News Join

Recommended architectures for PostgreSQL in Kubernetes

By Gabriele Bartolini
September 29, 2023

Member post by Gabriele Bartolini, VP of Cloud Native at EDB

"*You can run databases on Kubernetes because it's fundamentally the same as running a database on a VM*", tweeted Kelsey Hightower just a few months ago. Quite the opposite from what the former Google engineer and advocate said back in 2018 on Twitter: "*Kubernetes supports stateful workloads; I don't.*"

Kelsey Hightower
@kelseyhightower

You can run databases on Kubernetes because it's fundamentally the same as running a database on a VM. The biggest challenge is understanding that rubbing Kubernetes on Postgres won't turn it into Cloud SQL.

Truth is that I agree with him now as much as I agreed with him back then. At that time, the holistic offering of storage capabilities in Kubernetes was still immature (local persistent volumes would become GA only the year after), the operator pattern – which in the meantime has proven to be crucial for stateful applications like databases – was yet to become widely accepted, and the Data on Kubernetes Community was more than two years away (second half of 2020).

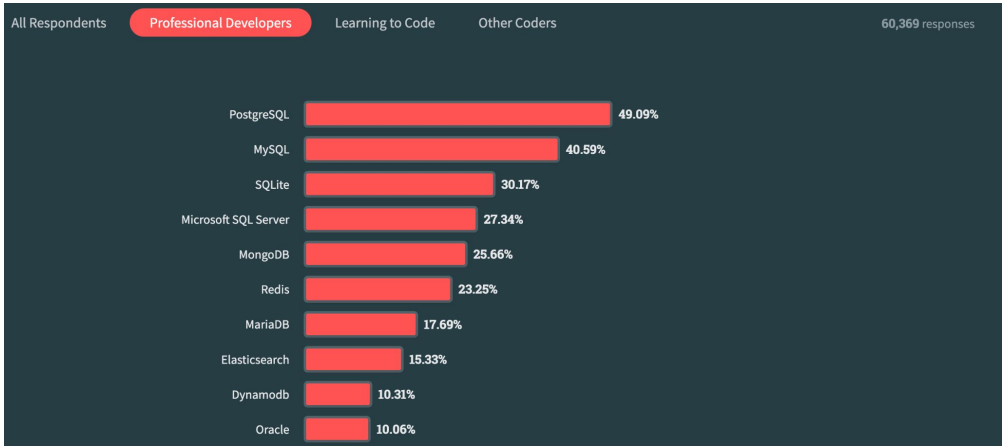
Nowadays, the situation is completely different. And I am sure that many people who've worked hard in the last few years to bring stateful workloads in Kubernetes agree with me that Kelsey's recent powerful words will contribute to reversing the public perception and facilitate our mission – provided we keep doing great.

PostgreSQL most loved database (2023)

Source: Stackoverflow

This year, PostgreSQL took over the first place spot from MySQL. Professional Developers are more likely than those learning to code to use PostgreSQL (50%) and those learning are more likely to use MySQL (54%).

MongoDB is used by a similar percentage of both Professional Developers and those learning to code and it's the second most popular database for those learning to code (behind MySQL).



Links:

Openshift Console:

https://console-openshift-console.apps.cluster-826x2.826x2.sandbox5464.opentlc.com/

Users:

name: user2..user32
Password: edb-workshop

Devspaces url:

https://devspaces.apps.cluster-826x2.826x2.sandbox5464.opentlc.com

Short url to Devspaces:

https://tinyurl.com/mrxs95pb

Minio:

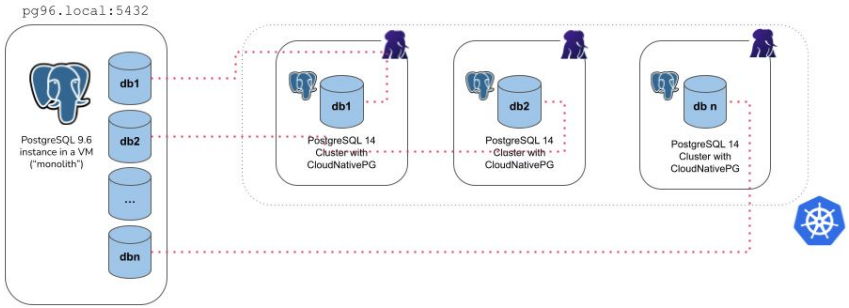
UI: https://minio-ui-minio.apps.cluster-826x2.826x2.sandbox5464.opentlc.com
API: https://minio-api-minio.apps.cluster-826x2.826x2.sandbox5464.opentlc.com

User: minio
Password: edb-workshop

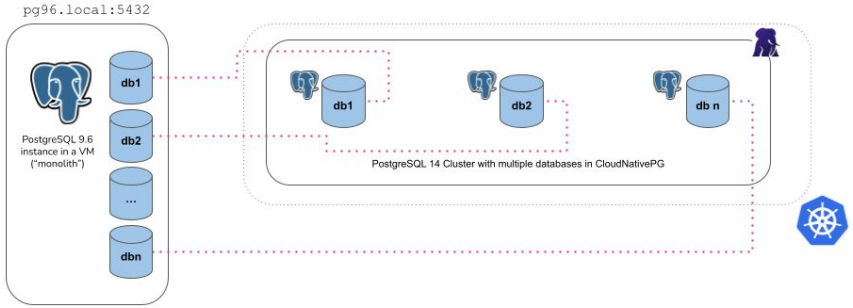
Database Migration

- In this step we will migrate the app database from our existing cluster to the new cluster
- We will create the yaml file with the setting "**import**" in the bootstrap section
- The operator uses internally postgres tools pg_dump and pg_restore
- This method can be used to **migrate** another database or to run **out-of-the-place upgrade**
- Possible settings:

Microservices:



Monolith:



Major upgrade (copying data)

Scripts

```
24_major_upgrade_in_place.sh
25_verify_major_upgrade_16_17.sh
```

Major upgrade (offline In-Place Major Upgrades)

CloudNativePG performs an offline in-place major upgrade when a new operand container image with a higher PostgreSQL major version is declaratively requested for a cluster.

Major upgrades are only supported between images based on the same operating system distribution. For example, if your previous version uses a bullseye image, you cannot upgrade to a bookworm image.

Doc link

Code

```
apiVersion: postgresql.k8s.enterprisedb.io/v1
kind: Cluster
metadata:
 name: ${cluster_major_upgrade}
spec:
 instances: 1
 imageName: ${postgres_major_upgrade_image}

 storage:
   size: 2Gi

 bootstrap:
   initdb:
     import:
       type: microservice
       databases:
         - app
...
```

Disaster recovery

Scripts

```
./dr/01-create-all.sh
./dr/02-insert-eu.sh
./dr/03-switchover-cluster.sh
```

Disaster recovery

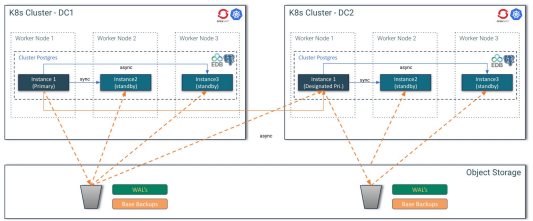
A replica cluster is a CloudNativePG Cluster resource designed to replicate data from another PostgreSQL instance, ideally also managed by CloudNativePG.

Typically, a replica cluster is deployed in a different Kubernetes cluster in another region. These clusters can be configured to perform cascading replication and can rely on object stores for data replication from the source, as detailed further down..

Two clusters:

- pg-eu
- pg-us

Doc link



Code

```
# EU
kubectl delete objectstores.barmancloud.cnpg.io object-storage-eu
envsubst < ../templates/dr/azure-object-storage-eu.yaml >
./tmp/object-storage-eu.yaml
kubectl apply -f ./tmp/object-storage-eu.yaml
sleep 2
rm -f ./tmp/pg-eu.yaml
envsubst < ../templates/dr/pg-eu.yaml > ./tmp/pg-eu.yaml
print_info "Creating pg-eu cluster...\n"
kubectl apply -f ./tmp/pg-eu.yaml
print_info "pg-eu cluster created.\n"

# US
kubectl delete objectstores.barmancloud.cnpg.io object-storage-us
envsubst < ../templates/dr/azure-object-storage-us.yaml >
./tmp/object-storage-us.yaml
kubectl apply -f ./tmp/object-storage-us.yaml
sleep 2
rm -f ./tmp/pg-us.yaml
envsubst < ../templates/dr/pg-us.yaml > ./tmp/pg-us.yaml
kubectl  wait --timeout=30m --for=condition=Ready cluster/pg-eu
print_info "Creating pg-us cluster...\n"
kubectl apply -f ./tmp/pg-us.yaml
./backup_cluster.sh pg-eu
print_info "pg-us cluster created.\n"
```

The “cnp” plugin for kubectl

- The official CLI for CloudNativePG
  - Available also as RPM or Deb package
- Extends the ‘kubectl’ command:
  - Customize the installation of the operator
  - Status of a cluster
  - Perform a manual switchover (promote a standby) or a restart of a node
  - Issue TLS certificates for client authentication
  - Declare start and stop of a Kubernetes node maintenance
  - Destroy a cluster and all its PVC
  - Fence a cluster or a set of the instances
  - Hibernate a cluster
  - Generate jobs for benchmarking via pgbench and fio
  - Issue a new backup
  - Start pgadmin

```
Name:               cluster-example
Namespace:          default
System ID:          7100921006673293335
PostgreSQL Image:   ghcr.io/cloudnative-pg/postgresql:14.3
Primary instance:   cluster-example-2
Status:             Cluster in healthy state
Instances:          3
Ready instances:    3
Current Write LSN:  0/C000060 (Timeline: 4 - WAL File: 00000004000000000000000C)

Certificates Status
Certificate Name             Expiration Date                Days Left Until Expiration
----------------             ---------------                --------------------------
cluster-example-replication  2022-08-21 13:15:00 +0000 UTC  89.95
cluster-example-server       2022-08-21 13:15:00 +0000 UTC  89.95
cluster-example-ca           2022-08-21 13:15:00 +0000 UTC  89.95

Continuous Backup status
First Point of Recoverability:  2022-05-23T13:37:08Z
Working WAL archiving:          OK
WALs waiting to be archived:    0
Last Archived WAL:              00000004000000000000000B   @   2022-05-23T13:42:09.37537Z
Last Failed WAL:                -

Streaming Replication status
Name               Sent LSN   Write LSN  Flush LSN  Replay LSN  Write Lag  Flush Lag  Replay Lag  State      Sync State  Sync Priority
----               --------   ---------  ---------  ----------  ---------  ---------  ----------  -----      ----------  -------------
cluster-example-3  0/C000060  0/C000060  0/C000060  0/C000060   00:00:00   00:00:00   00:00:00    streaming  async       0
cluster-example-1  0/C000060  0/C000060  0/C000060  0/C000060   00:00:00   00:00:00   00:00:00    streaming  async       0

Instances status
Name               Database Size  Current LSN  Replication role  Status  QoS         Manager Version
----               -------------  -----------  ----------------  ------  ---         ---------------
cluster-example-3  33 MB          0/C000060    Standby (async)   OK      BestEffort  1.15.0
cluster-example-2  33 MB          0/C000060    Primary           OK      BestEffort  1.15.0
cluster-example-1  33 MB          0/C000060    Standby (async)   OK      BestEffort  1.15.0
```

Install CNPG plugin

NOT NEEDED DURING WORKSHOP
For illustrative purposes.

- In the web terminal run the script 01_install_plugin.sh:

  ./01_install_plugin.sh

Call to action: Check CNPG plugin

- Call the help for the CNPG Plugin, run:

  kubectl-cnp help

Use case
Operator installation

Operator Installation demonstration

NOT NEEDED DURING WORKSHOP
For illustrative purposes.

- Install the operator
- Check the installed CNP Operator in the console
- Discover the features of the Operator in the OpenShift environment
- Check the installed CNP Operator in the web terminal

Install the CNPG Operator and check the it in the web terminal

NOT NEEDED DURING WORKSHOP
For illustrative purposes.

- In the web terminal install the operator:

  ./02_install_operator.sh

  -> will require admin privs on Openshift

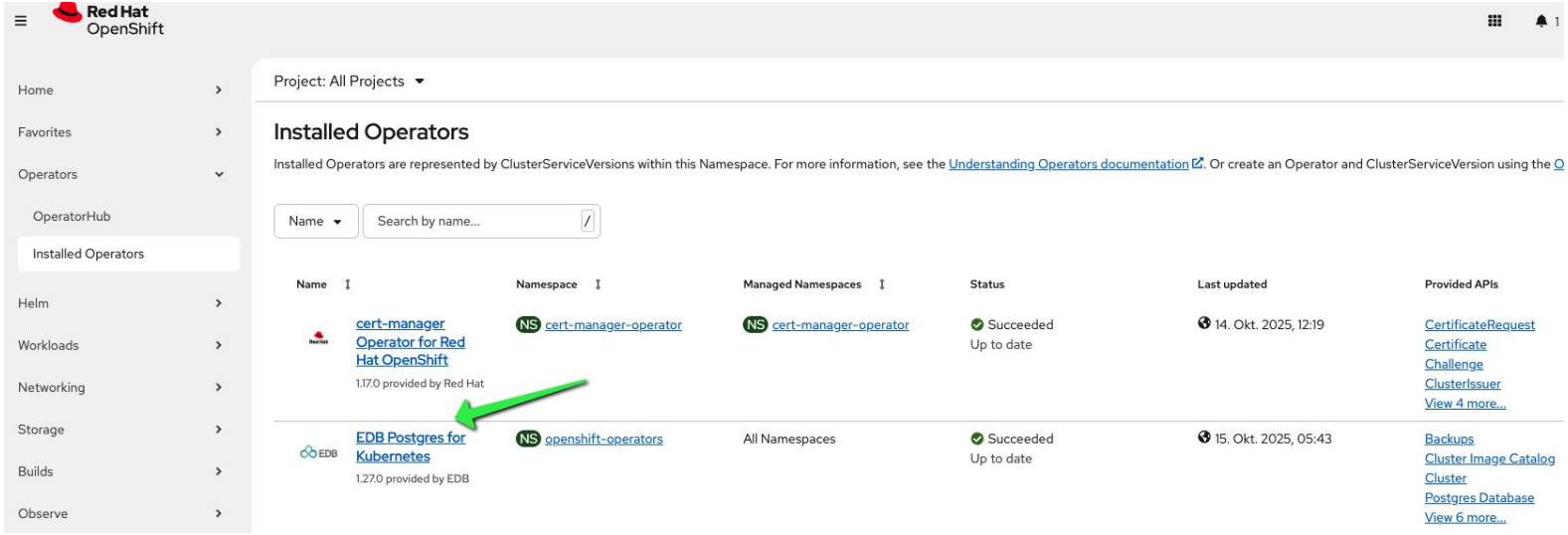
Call to action: Check the it in the web terminal

- In the web terminal check the installation of the operator:

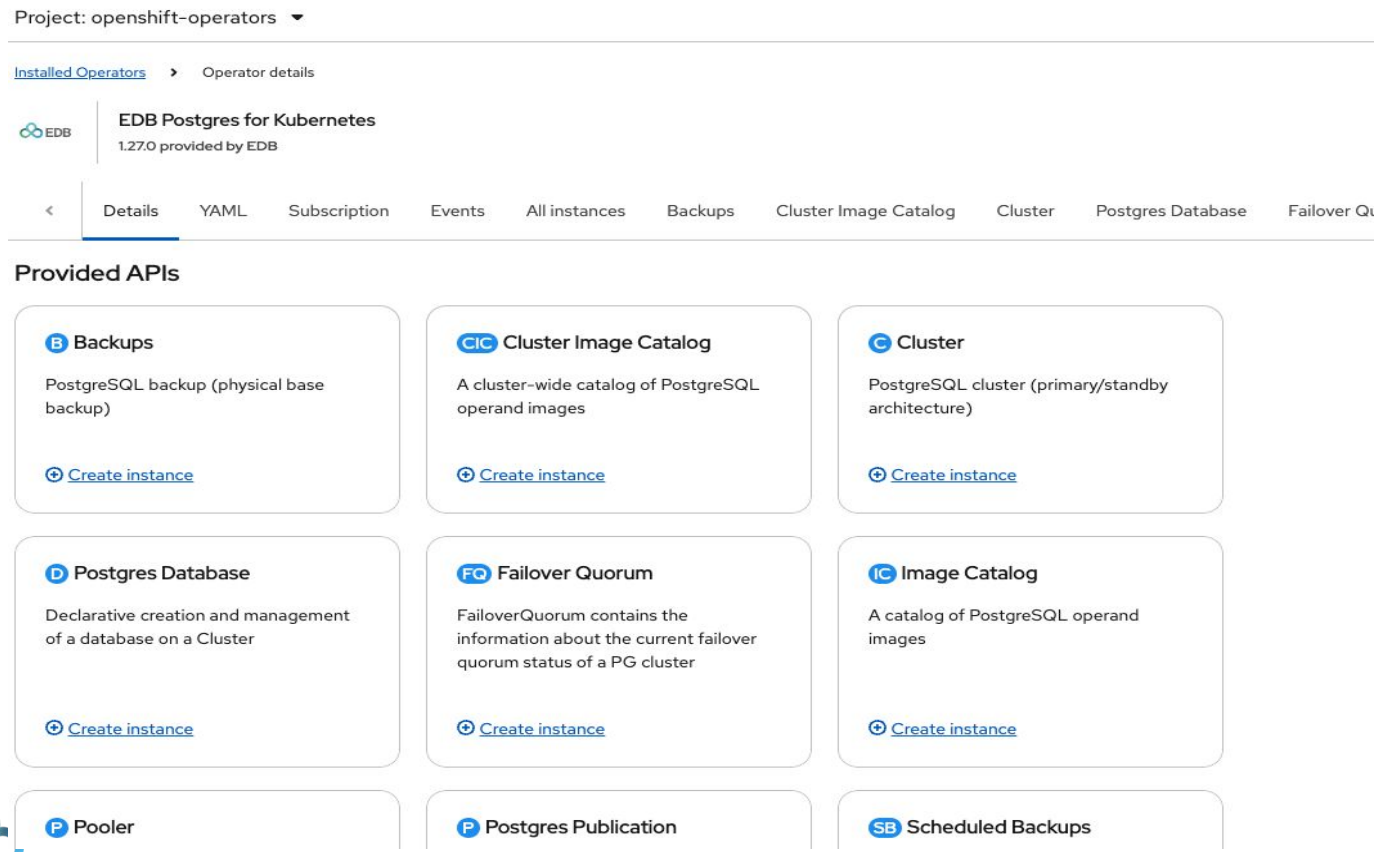
  ./03_check_operator_installed.sh

Call to action: Check the installed CNPG Operator in the Openshift console

- In the OpenShift console navigate to:
  - -> Operators
  - -> Installed Operators
  - -> Klick on the Operator installed:



Call to action: Discover the features of the Operator in the OpenShift environment

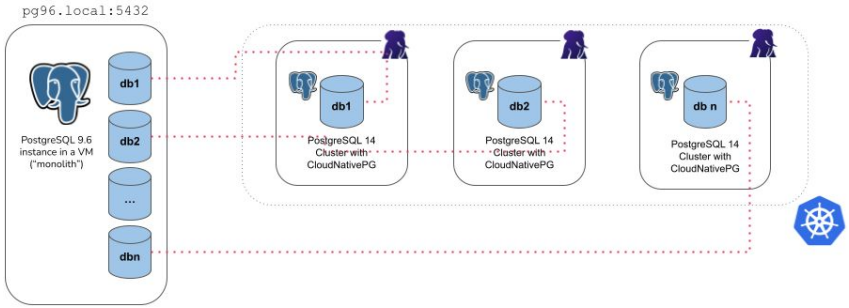


Use case
Database Migration / Major Version Upgrade

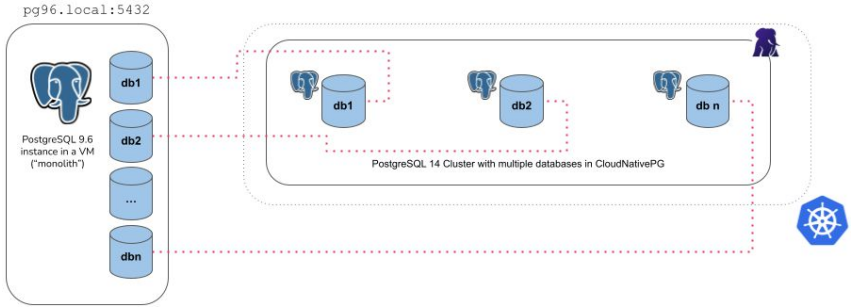
Database Migration

- In this step we will migrate the app database from our existing cluster to the new cluster
- We will create the yaml file with the setting "**import**" in the bootstrap section
- The operator uses internally postgres tools pg_dump and pg_restore
- This method can be used to **migrate** another database or to run **out-of-the-place upgrade**
- Possible settings:

Microservices:



Monolith:



Call to action: Run migration or out of the place upgrade

- In the web terminal 1:
  - Run the script:

    ./20_upgrade_major_version.sh
  - Check the cluster creation process:

    kubectl get pods -w
  - Check the table test in the cluster-user<X>-17, run the command:
    - Connect to the cluster:

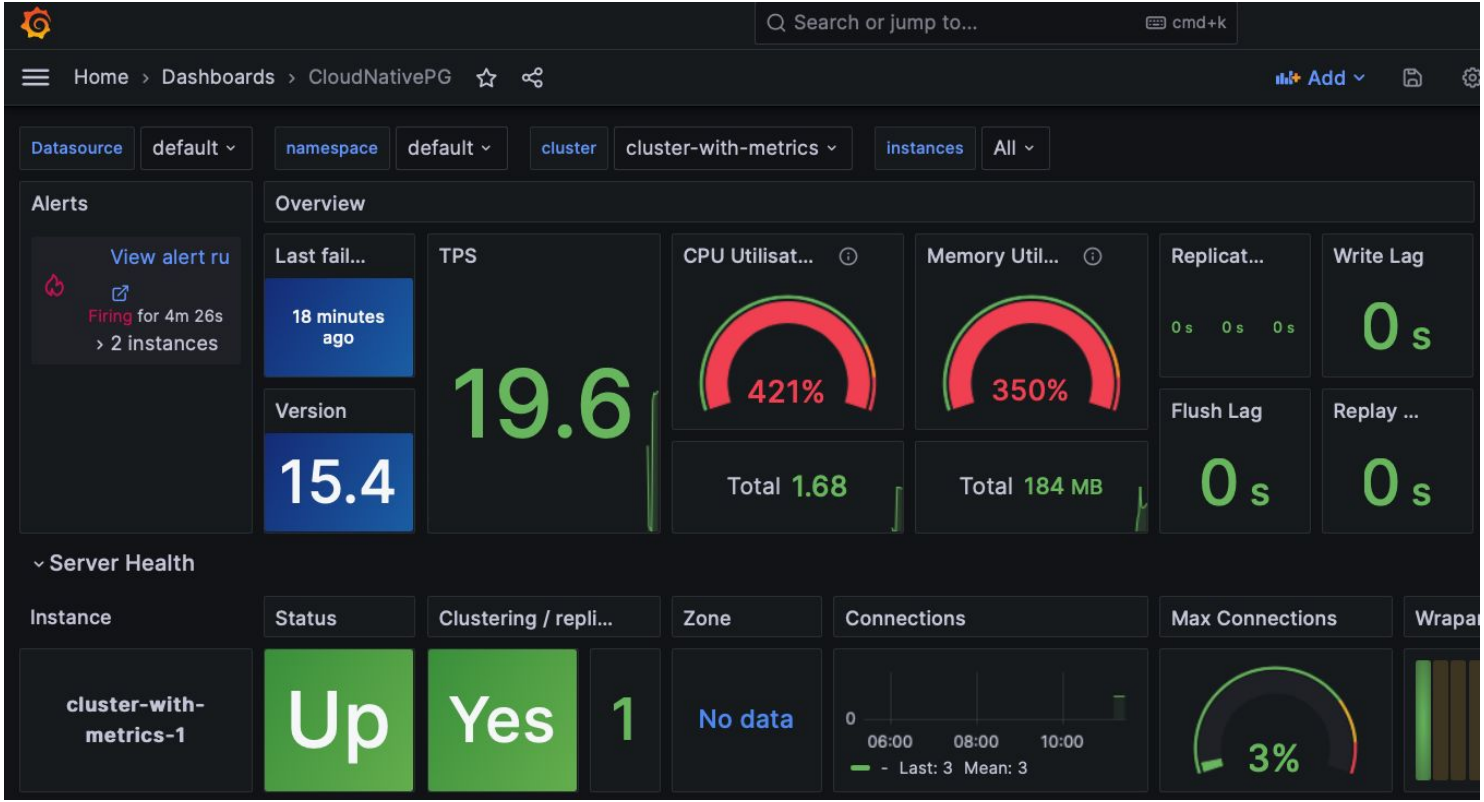
      oc cnp psql cluster-user<X>-17
    - Connect to the database app:

      \c app
    - Run sql commands:

      select version();

      select count(*) from test;

Grafana Dashboard



Agenda

Start	End	Session
12:30	13.30	Registration & Welcome Lunch
13:30	13:45	Camptocamp Vorstellung
13:45	14:00	Red Hat OpenShift & EDB Partnerschaft
14:00	14.15	Einführung in den Postgres-Marketplace
14:15	14:30	Coffee
14:30	15:00	CNPG Operator Reference Architecture
15:00	15:30	Functionalities for CNPG
15:30	17:15	Interactive session & demo
17:15	18:30	Drinks & Apero