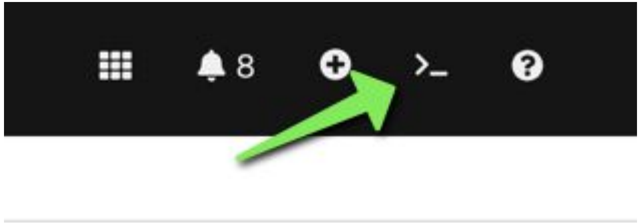
Prep hands-on (OpenShift environment)

OpenShift Access

Users	Password	Console URL
user1… user35	`VMFI2JU7tWcONacE`	https://console-openshift-console.apps.cluster-gp5q7.gp5q7.sandbox1549.opentlc.com

Connect to the OpenShift Console

- Connect to the OpenShift console:
  - Run in the browser https://console-openshift-console.apps.cluster-gp5q7.gp5q7.sandbox1549.opentlc.com
  - Connect to the console with your user and password
- Open the web terminal:
  - In the top right corner click on **>_** .



  - The web terminal session will be started

Connect to the OpenShift Console

- In the web terminal execute following steps::
    - Download the workshop scripts, run:

      ```
      git clone https://github.com/kappertj/edb-workshop.git
      ```

    - Change the directory to edb-workshop and set the file permissions:

      ```
      cd /home/user/edb-workshop
      chmod -R 755 *
      ```

    - Setup the local environment, run:

      ```
      ./00_setup.sh
      ```

    - Set the default namespace, run:

      ```
      oc project <user name>,
      ```

EDB

Postgres® for the AI Generation

Postgres on Kubernetes Workshop - Hands-on

Hands-on
(vm 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)

Deployment

High Availability

Administration

Monitoring

Backup and Recovery

Last CloudNativePG tested version is 1.25

This demo is in

https://github.com/sergioenterprisedb/kubecon2022-demo

Hand-on documentation

https://github.com/borysneselovskyi/cnpg_workshop_documentation

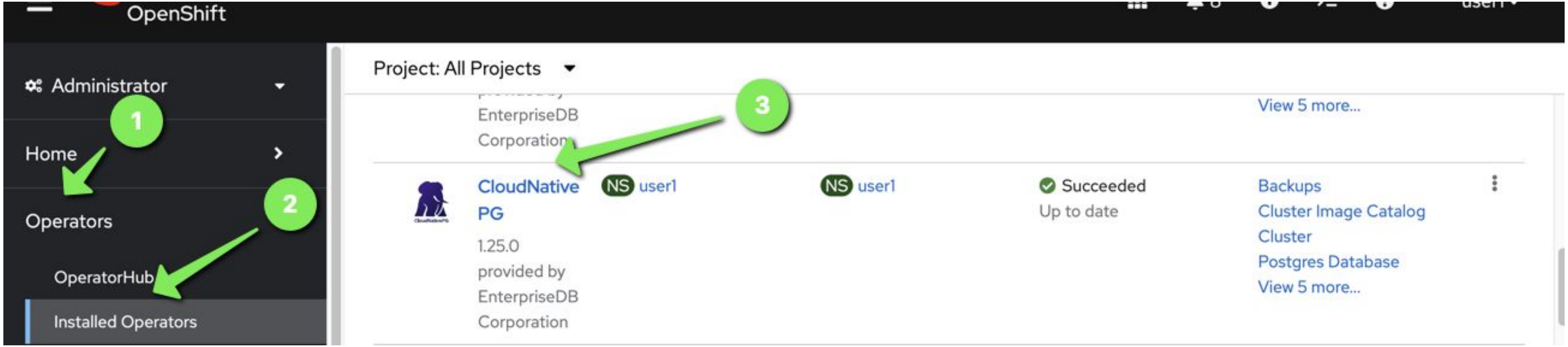
Download the Hand-on Document

CNPG_ Workshop_OpenShift_Hands-on.pdf

Operator Installation

Step 1: Check the installed CNPG Operator in the console

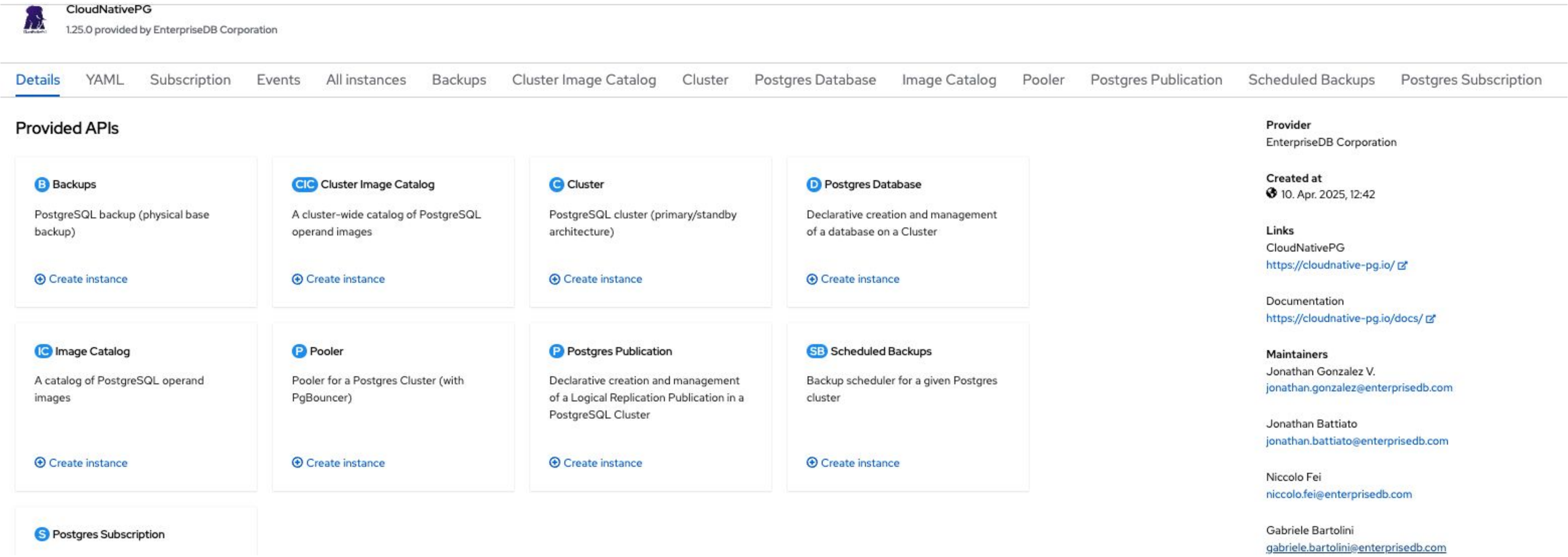
- In the OpenShift console navigate to:
  - -> Operators
  - -> Installed Operators
  - -> Klick on the Operator installed in your namespace, for example: user1:



EnterpriseDB Corporation
View 5 more...

CloudNative PG
1.25.0
provided by EnterpriseDB Corporation
NS user1
NS user1
Succeeded
Up to date
Backups
Cluster Image Catalog
Cluster
Postgres Database
View 5 more...

Step 2: Discover the features of the Operator in the OpenShift environment



CloudNativePG
1.25.0 provided by EnterpriseDB Corporation

Provided APIs

B Backups
PostgreSQL backup (physical base backup)
⊕ Create instance

CIC Cluster Image Catalog
A cluster-wide catalog of PostgreSQL operand images
⊕ Create instance

C Cluster
PostgreSQL cluster (primary/standby architecture)
⊕ Create instance

D Postgres Database
Declarative creation and management of a database on a Cluster
⊕ Create instance

IC Image Catalog
A catalog of PostgreSQL operand images
⊕ Create instance

P Pooler
Pooler for a Postgres Cluster (with PgBouncer)
⊕ Create instance

P Postgres Publication
Declarative creation and management of a Logical Replication Publication in a PostgreSQL Cluster
⊕ Create instance

SB Scheduled Backups
Backup scheduler for a given Postgres cluster
⊕ Create instance

S Postgres Subscription

Provider
EnterpriseDB Corporation

Created at
10. Apr. 2025, 12:42

Links
CloudNativePG
https://cloudnative-pg.io/

Documentation
https://cloudnative-pg.io/docs/

Maintainers
Jonathan Gonzalez V.
jonathan.gonzalez@enterprisedb.com

Jonathan Battiato
jonathan.battiato@enterprisedb.com

Niccolo Fei
niccolo.fei@enterprisedb.com

Gabriele Bartolini
gabriele.bartolini@enterprisedb.com

Step 3: Check the installed CNPG Operator in the web terminal

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

```
cd /home/user/edb-workshop
./03_check_operator_installed.sh
```

Step 4: Install CNPG plugin

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

```
cd /home/user/edb-workshop
./01_install_plugin.sh
```

- Set PATH to the plugin kubectl-cnpg:

```
source /home/user/.bashrc
```

- Call the help for the CNPG Plugin, run:

```
kubectl-cnpg help
```

Create the postgres cluster

Step 5: Configure and Install the Postgres cluster

- Open the new tab in the web terminal (2):
  - Go to the directory /home/user/edb-workshop:

    ```
    cd /home/user/edb-workshop
    ```

  - Set the default namespace and PATH variable, run:

    ```
    oc project <user name>
    source /home/user/.bashrc
    ```

  - Check the Postgres cluster status, run the script:

    ```
    ./06_show_status.sh
    ```

- In the web terminal (1) create the postgres cluster::
  - Create the yaml file, run:

    ```
    ./04_get_cluster_config_file.sh
    ```

  - Create the postgres cluster:

    ```
    ./05_install_cluster.sh
    ```

Step 6: Create table test with 1000 rows

- In the terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./07_insert_data.sh
```

Upgrade the postgres cluster

Step 7: Check the cluster status

- In the web terminal 1:
  - Run the command

    ```
    kubectl-cnpg status cluster-example
    ```

  - Review the output:
    - check Postgres version: “PostgreSQL Image: ghcr.io/cloudnative-pg/postgresql:**16.1**”
    - check “Continuous Backup status”: “**Not configured**”
  - Check the updated cluster configuration - file cluster-example-upgrade.yaml

    ```
    cd /home/user/edb-workshop
    less cluster-example-upgrade-template.yaml
    ```

    - Check Postgres version: “imageName: ghcr.io/cloudnative-pg/postgresql:**16.4**”
    - Check the Backup section

Step 8: Run the script 09_upgrade.sh

- With this step we will:
  - Run the postgres minor update from the version 16.1 to 16.4
  - We will configure the WAL files backup to the S3 storage
- In the web terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./09_upgrade.sh
```

- In the web terminal **2**:
  - Check the upgrade status:

```
./06_show_status.sh
```

Backup & Restore

Step 9: Create the full backup

- With this step we will:
  - Create the full backup of the postgres cluster in the MinIO storage:
- In the web terminal 1:
  - Run the script:

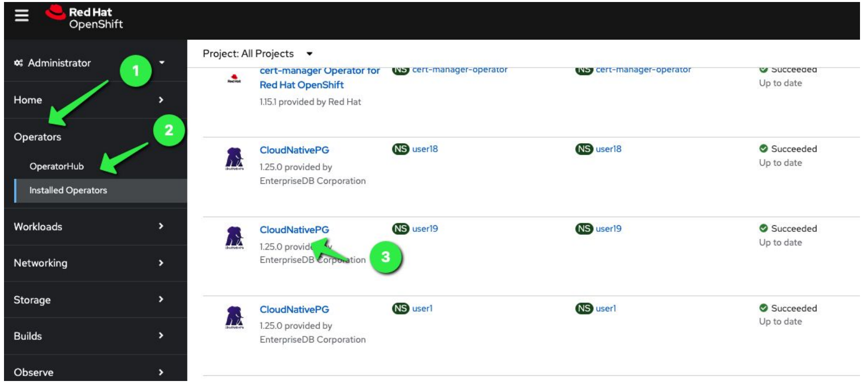
```
cd /home/user/edb-workshop
./10_backup_cluster.sh
```

  - Check the backup status:

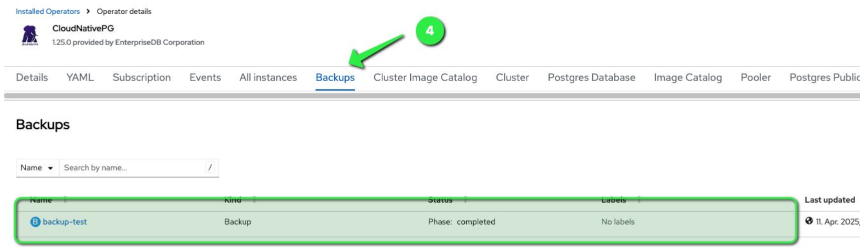
```
cd /home/user/edb-workshop
./11_backup_describe.sh
```

Step 10: Check Backup in the Openshift Console

- Navigate to:
  - -> Operators
  - -> Installed Operators
  - -> Press on the Operator installed in your namespace, for example: user1:
  - -> Go to the Backup section and show the created backup:



cert-manager Operator for Red Hat OpenShift 1.15.1 provided by Red Hat	NS cert-manager-operator	NS cert-manager-operator	Succeeded Up to date
CloudNativePG 1.25.0 provided by EnterpriseDB Corporation	NS user18	NS user18	Succeeded Up to date
CloudNativePG 1.25.0 provided by EnterpriseDB Corporation	NS user19	NS user19	Succeeded Up to date
CloudNativePG 1.25.0 provided by EnterpriseDB Corporation	NS user1	NS user1	Succeeded Up to date



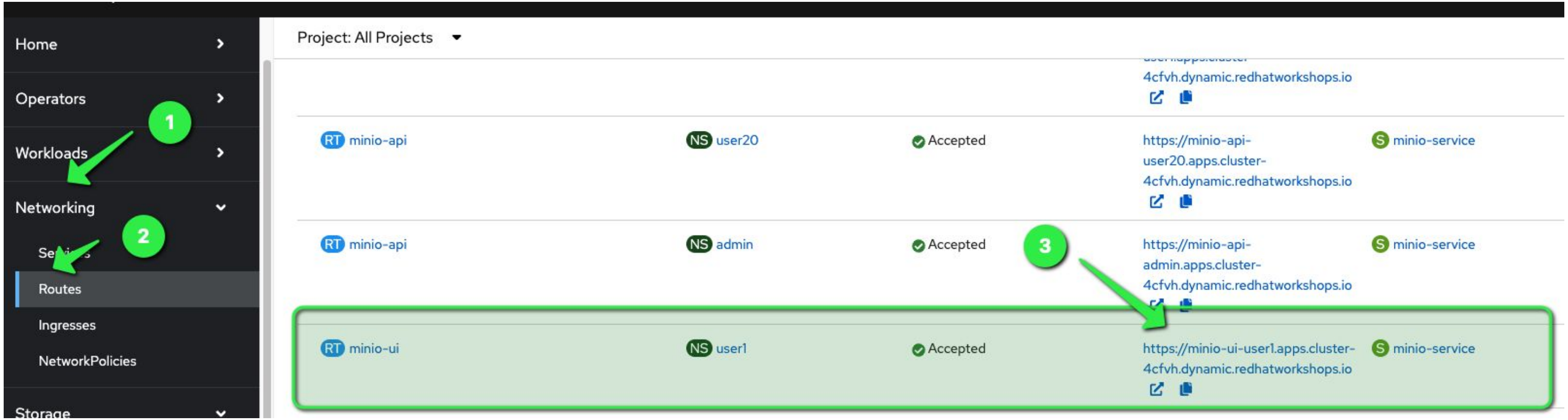
CloudNativePG
1.25.0 provided by EnterpriseDB Corporation

Backups

Name	Kind	Status	Labels	Last updated
B backup-test	Backup	Phase: completed	No labels	11. Apr. 2025,

Step 11: Check Backup in MinIO UI

- In the Openshift console navigate to Networking -> Routes
- Search for route minio-ui for your user and press the button with url:

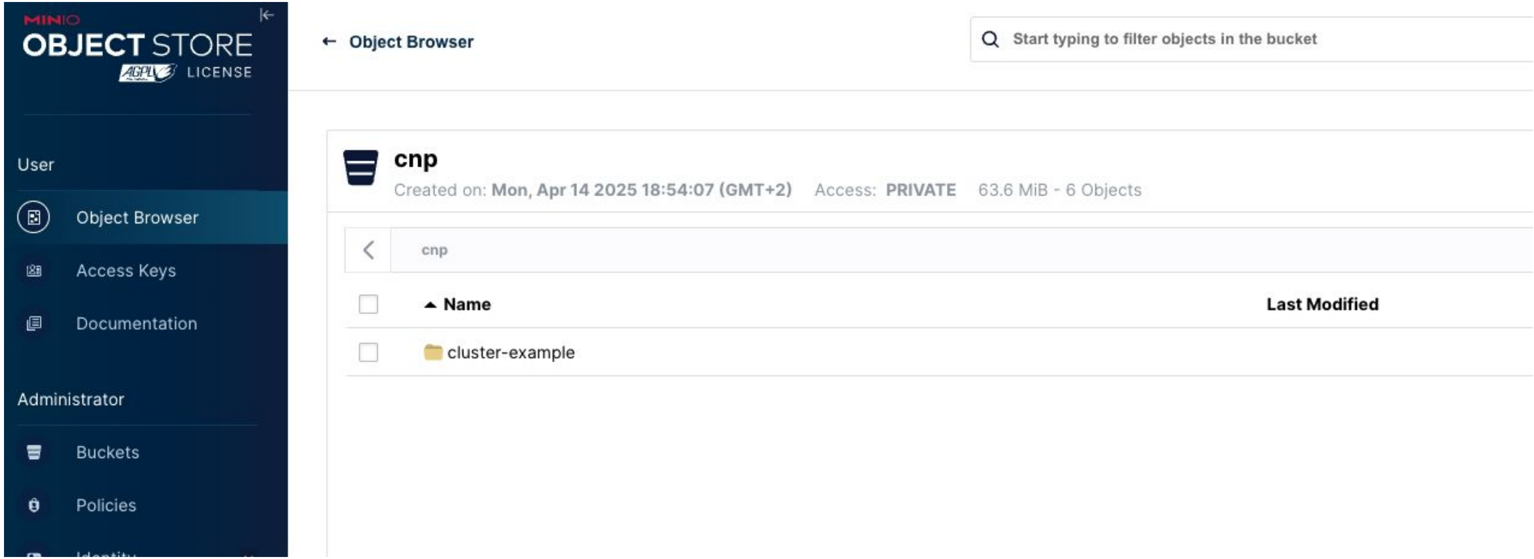


Project: All Projects

			4cfvh.dynamic.redhatworkshops.io	
RT minio-api	NS user20	Accepted	https://minio-api-user20.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S minio-service
RT minio-api	NS admin	Accepted	https://minio-api-admin.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S minio-service
RT minio-ui	NS user1	Accepted	https://minio-ui-user1.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S minio-service

Step 11: Check Backup in MinIO UI

- Connect as user **admin** with the password: **password**
- The page will appear:



cnp

Created on: Mon, Apr 14 2025 18:54:07 (GMT+2) Access: PRIVATE 63.6 MiB - 6 Objects

cnp

□	Name	Last Modified
□	cluster-example	

Step 12: Restore the database from the backup

- With this step we will:
  - Create the new cluster cluster-restore
  - Restore the full backup created in the previous step in the new cluster:
- In the terminal 1:
  - Run the restore:

    ```
    cd /home/user/edb-workshop
    ./12_restore_cluster.sh
    ```

  - Check the creation status:

    ```
    kubectl get pods -w  # after creation stop the execution with <ctrl>+c
    ```

  - Check the table test in the cluster-restore, run the script:

    ```
    cd /home/user/edb-workshop
    ./check_restore_table_test.sh
    ```

Failover

Step 13: Run failover test

- With this step we will:
  - Delete the primary database of the cluster cluster-example
  - Check the cluster status in the another terminal window
- In the web terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./13_failover.sh
```

- In the web terminal **2**:
  - Check the failover cluster status:

```
cd /home/user/edb-workshop
./06_show_status.sh
```

Scale-out and scale-down

Step 14: Scale-out the postgres cluster

- With this step we will:
  - Add the 1 standby to the cluster
- In the web terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./14_scale_out.sh
```

- In the web terminal **2**:
  - Check the cluster status:

```
cd /home/user/edb-workshop
./06_show_status.sh
```

Step 15: Scale-down the postgres cluster

- With this step we will:
  - Remove 2 standby pods from the cluster
- In the web terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./15_scale_down.sh
```

- In the web terminal **2**:
  - Check the cluster status:

```
cd /home/user/edb-workshop
./06_show_status.sh
```

Fencing

Step 16: Stop postgres process on the pod

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

    ```
    cd /home/user/edb-workshop
    ./30_fencing_on.sh
    ```

- In the web terminal **2**:
  - Check the cluster status:

    ```
    cd /home/user/edb-workshop
    ./06_show_status.sh
    ```

Step 17: Start the postgres process on the pod

- In the terminal 1:
  - Run the script:

    ```
    ./31_fencing_off.sh
    ```

- In the terminal **2**:
  - Check the cluster status:

    ```
    ./06_show_status.sh
    ```

Hibernation

Step 18: Stop the postgres cluster

- In the terminal 1:
  - Run the script:

```
cd /home/user/edb-workshop
./32_hibernation_on.sh
```

- In the terminal **2**:
  - Check the cluster status:

```
cd /home/user/edb-workshop
./06_show_status.sh
```

Step 19: Start the postgres cluster

- In the terminal 1:
  - Run the script:

    ```
    cd /home/user/edb-workshop
    ./33_hibernation_off.sh
    ```

- In the terminal **2**:
  - Check the cluster status:

    ```
    cd /home/user/edb-workshop
    ./06_show_status.sh
    ```

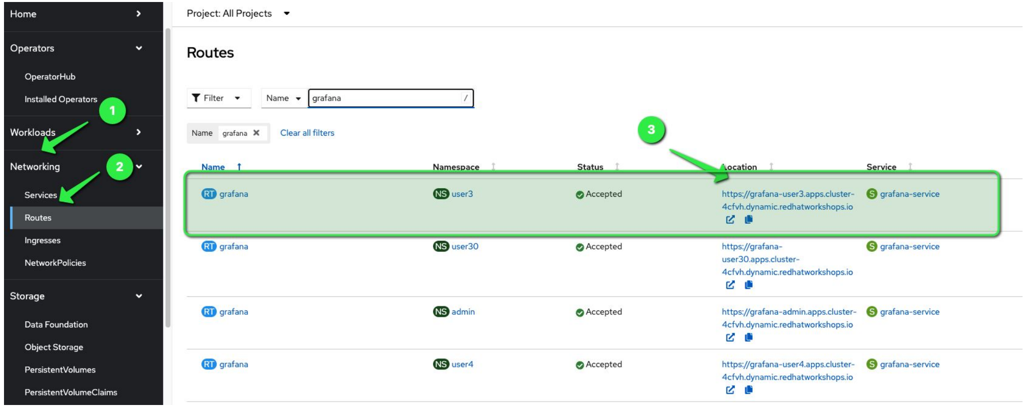
Monitoring

Step 20: Setup monitoring

- In the web terminal 1:
  - Configure Grafana:

    ```
    cd /home/user/edb-workshop
    ./monitoring.sh
    ```

  - In the Openshift console navigate to Networking, then to Routes
    - Search for the route "grafana" for your user, then click on location:



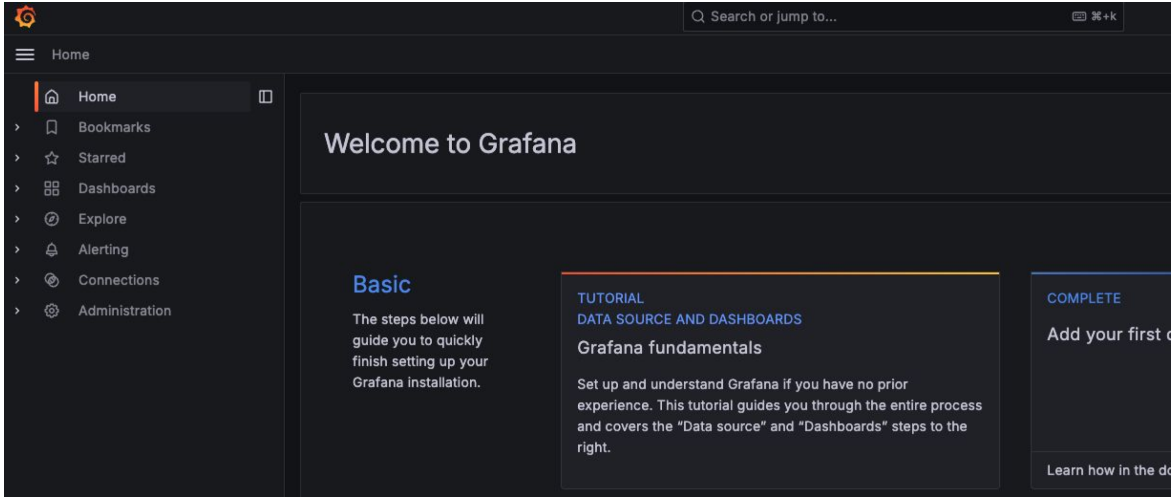
Routes

Name: grafana

Name	Namespace	Status	Location	Service
RT grafana	NS user3	Accepted	https://grafana-user3.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S grafana-service
RT grafana	NS user30	Accepted	https://grafana-user30.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S grafana-service
RT grafana	NS admin	Accepted	https://grafana-admin.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S grafana-service
RT grafana	NS user4	Accepted	https://grafana-user4.apps.cluster-4cfvh.dynamic.redhatworkshops.io	S grafana-service

Step 21: Access the Grafana page

- Grafana interface appears
- Connect as user **admin** with the password: **grafana**
- The grafana page will appear



Welcome to Grafana

Basic

The steps below will guide you to quickly finish setting up your Grafana installation.

TUTORIAL
DATA SOURCE AND DASHBOARDS

Grafana fundamentals

Set up and understand Grafana if you have no prior experience. This tutorial guides you through the entire process and covers the “Data source” and “Dashboards” steps to the right.

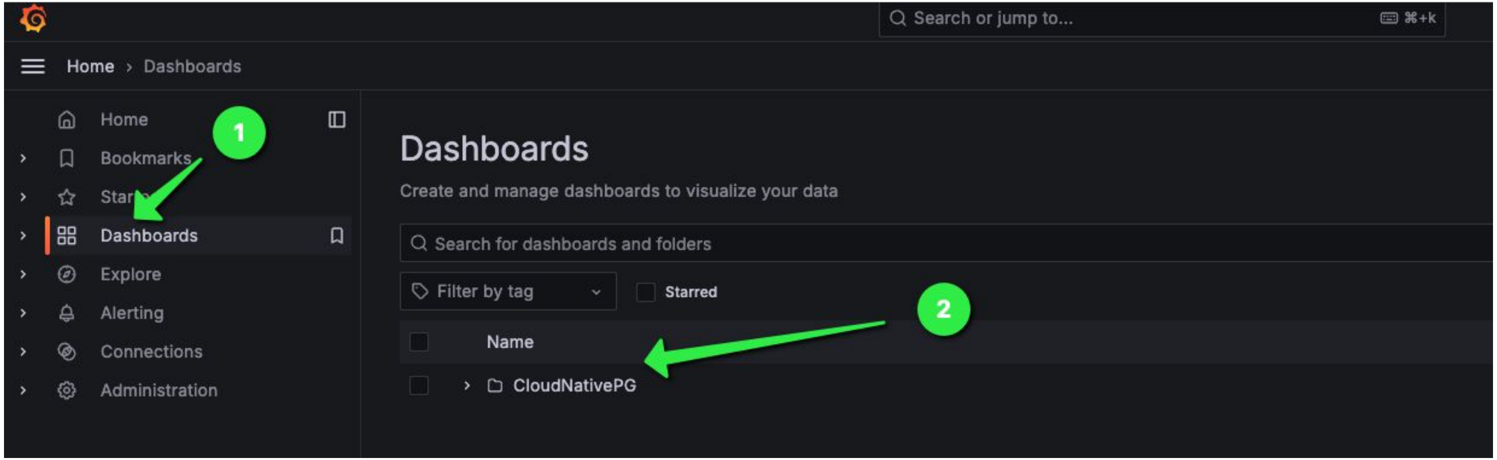
COMPLETE

Add your first d

Learn how in the dc

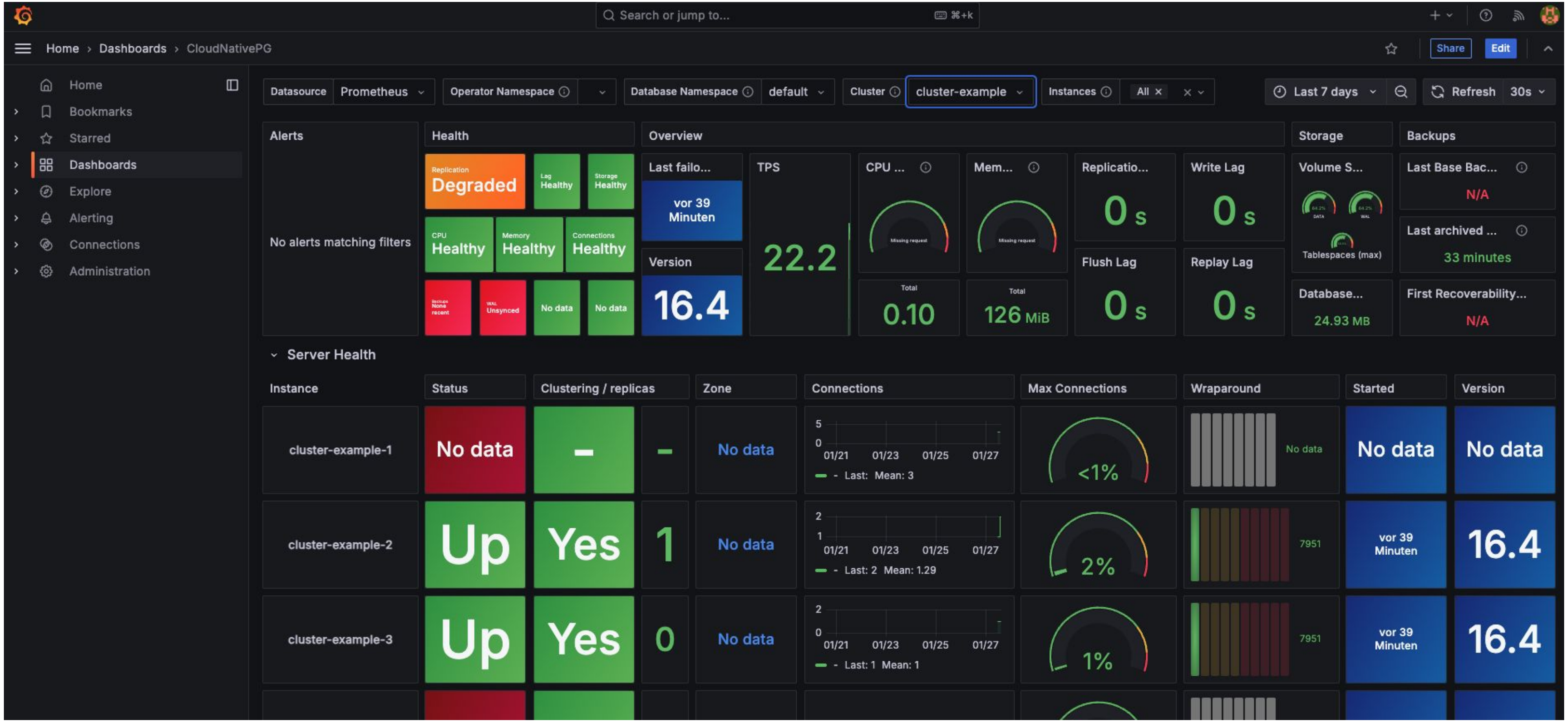
Step 22: Navigate to CloudNativePG Dashboard

- Go to Dashboards and click on CloudNativePG:



Step 23: Explore CNPG Dashboard

- Explore the CNPG Dashboard:



Datasource: Prometheus
Operator Namespace:
Database Namespace: default
Cluster: cluster-example
Instances: All
Last 7 days
Refresh 30s

Alerts: No alerts matching filters

Health: Replication Degraded; Lag Healthy; Storage Healthy; CPU Healthy; Memory Healthy; Connections Healthy; Backups None recent; WAL Unsynced; No data; No data

Overview: Last failo... vor 39 Minuten; Version 16.4; TPS 22.2; CPU ... Total 0.10; Mem... Total 126 MiB; Replicatio... 0 s; Write Lag 0 s; Flush Lag 0 s; Replay Lag 0 s

Storage: Volume S... DATA, WAL, Tablespaces (max); Database... 24.93 MB

Backups: Last Base Bac... N/A; Last archived ... 33 minutes; First Recoverability... N/A

Server Health

Instance	Status	Clustering / replicas		Zone	Connections	Max Connections	Wraparound	Started	Version
cluster-example-1	No data	–	–	No data	Last: Mean: 3	<1%	No data	No data	No data
cluster-example-2	Up	Yes	1	No data	Last: 2 Mean: 1.29	2%	7951	vor 39 Minuten	16.4
cluster-example-3	Up	Yes	0	No data	Last: 1 Mean: 1	1%	7951	vor 39 Minuten	16.4

Major Upgrade

Step 24: Delete all created clusters and backup

- In the web terminal 1:
  - Delete the cluster cluster-example

    ```
    kubectl delete cluster cluster-example
    ```

  - Delete the cluster cluster-restore:

    ```
    kubectl delete cluster cluster-restore
    ```

  - Delete the backup:

    ```
    cd /home/user/edb-workshop
    kubectl delete -f backup.yaml
    ```

Step 24: Create the Postgres 16 Cluster

- In the web terminal 1:
- Change directory to cd /home/user/edb-workshop/major_upgrade_demo/

```
cd /home/user/edb-workshop/major_upgrade_demo/
```

  - Create the cluster v16::

```
./04_create_cluster_v16.sh
```

  - Check the cluster status:

```
./05_show_status_v16.sh
```

  - Insert the data:

```
./06_insert_data_cluster_v16.sh
```

  - Verify the inserted data::

```
./07_verify_data_inserted.sh
```

Step 25: Create the Postgres 17 Cluster and import data from PG 16

- In the web terminal 1:
  - Change directory to cd /home/user/edb-workshop/major_upgrade_demo/

    ```
    cd /home/user/edb-workshop/major_upgrade_demo/
    ```

  - Check the file less cluster-example-upgrade-16-to-17.yaml:

    ```
    less cluster-example-upgrade-16-to-17.yaml
    ```

    -> check the section bootstrap
  - Create the cluster postgres v17 and import the data from the postgres v16:

    ```
    ./08_upgrade_v16_to_v17.sh
    ```

  - Check the cluster status:

    ```
    ./09_show_status_v17.sh
    ```

  - Verify the data in the postgres v17:

    ```
    ./10_verify_data_migrated_16_17.sh
    ```

EDB Postgres AI | HCS company | Red Hat

SURVEY

Share Your Opinion & Win a LEGO Star Wars TIE Inceptor!

Scan the QR Code & complete the survey to let us know your feedback!

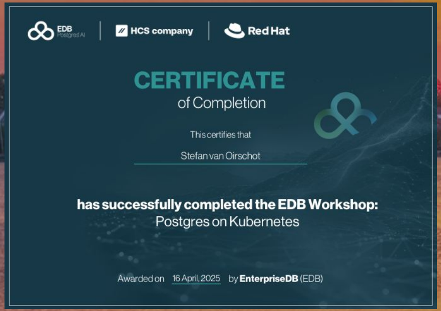
EDB Postgres AI | HCS company | Red Hat

https://wheelofnames.com/

EDB Postgres AI | HCS company | Red Hat

Thank you for participating in the Postgres on Kubernetes Workshop

Please pick up your certificate :-)



EDB Postgres AI | HCS company | Red Hat

CERTIFICATE
of Completion

This certifies that

Stefan van Oirschot

has successfully completed the EDB Workshop:
Postgres on Kubernetes

Awarded on 16 April, 2025 by **EnterpriseDB** (EDB)