Troubleshooting Databases on Kubernetes

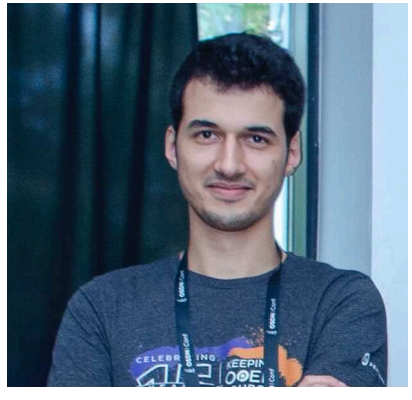
Cloud Native Databases - 2026-02-20

Ege Güneş

Senior Software Engineer

PERCONA

About me



- Living in Istanbul, Turkey
- Working on Percona Operators for ~5 years

Premeditatio Malorum

"*What is quite unlooked for is more crushing in its effect, and unexpectedness adds to the weight of a disaster. This is a reason for ensuring that nothing ever takes us by surprise. We should project our thoughts ahead of us at every turn and have in mind every possible eventuality instead of only the usual course of events…*

Rehearse them in your mind: exile, torture, war, shipwreck. All the terms of our human lot should be before our eyes." — Seneca

Why troubleshooting is harder on Kubernetes?

Images are (usually) rootless and containers are immutable.

Tools we use for debugging are not usable out of the box.

Reconciliation loops make tweaking stuff harder.

Basic troubleshooting

Check logs (duh).

Pro-tip: Use a tool like stern to tail logs from multiple pods.

```
$ stern instance1 -c database --tail 1
cluster1-instance1-9fqb-0 database 2026-02-09 09:30:59,964 INFO:
cluster1-instance1-4c5g-0 database 2026-02-09 09:30:59,974 INFO:
cluster1-instance1-52h8-0 database 2026-02-09 09:30:59,974 INFO:
```

Check events

Keep an eye on:

- CrashLoopBackOff
- OOMKilled
- FailedScheduling
- FailedAttachVolume
- FailedMount
- NodeNotReady
- DiskPressure
- MemoryPressure

Events are persisted only for **1 hour**!

Check statuses

Conditions usually contain useful information.

```
- lastTransitionTime: "2026-02-09T07:15:10Z"
  message: pgBackRest replica create repo is not ready for back
  observedGeneration: 1
  reason: StanzaNotCreated
  status: "False"
  type: PGBackRestReplicaRepoReady
```

Conditions usually contain useful information.

```
- lastTransitionTime: "2026-02-09T07:15:10Z"
  message: pgBackRest replica create repo is not ready for back
  observedGeneration: 1
  reason: StanzaNotCreated
  status: "False"
  type: PGBackRestReplicaRepoReady
```

Pro-tip: Use kubectl status plugin.

```
Pod/cluster1-rs0-0 -n psmdb-21113, created 4h ago by StatefulSet/cluster1-rs0, gen:1, started after 0s Running Burstable
  Current: Pod is Ready
  Managed by percona-server-mongodb-operator with cluster1 instance by percona-server-mongodb application within component mongod
  PodScheduled -> Initialized -> ContainersReady -> Ready for 3h
  InitContainers:
    mongo-init (docker.io/percona/percona-server-mongodb-operator:1.21.2) Started 3h ago and Completed after 0s
  Containers:
    backup-agent (docker.io/percona/percona-backup-mongodb:2.11.0) Running for 3h and Ready
      usage cpu usage:0.003/[no-req, no-lim], mem usage:15MB/[no-req, no-lim]
    logrotate (docker.io/percona/fluentbit:4.0.1) Running for 3h and Ready
      usage cpu usage:0.001/[0% of req:0.2, no-lim], mem usage:1.1MB/[1% of req:100MB, no-lim]
    logs (docker.io/percona/fluentbit:4.0.1) Running for 3h and Ready
      usage cpu usage:0.002/[1% of req:0.2, no-lim], mem usage:4.6MB/[5% of req:100MB, no-lim]
    mongod (docker.io/percona/percona-server-mongodb:8.0.17-6) Running for 3h and Ready
      usage cpu usage:0.05/[16% of req:0.3, 8% of lim:0.6], mem usage:168.5MB/[16% of req:1GB, 16% of lim:1GB]
  Known/recorded manage events:
    4h ago Updated by kube-controller-manager (metadata, spec)
    3h ago Updated by kubelet (status)
  Ephemeral storage 57.4MB/27.2GB(0%) used, 16.3GB free, 94/1.6M(0%) inodes used, 1.5M free.
  EmptyDirs:
    bin 39.4MB/27.2GB(0%) used, 16.3GB free, 15/1.6M(0%) inodes used, 1.5M free.
    mongosh 4kB/27.2GB(0%) used, 16.3GB free, 1/1.6M(0%) inodes used, 1.5M free.
  PVCs:
    Volume: mongod-data, 287.7MB/3GB(9%) used, 2.7GB free, 121/196.6k(0%) inodes used, 196.4k free.
    PersistentVolumeClaim/mongod-data-cluster1-rs0-0 -n psmdb-21113, created 4h ago Bound
      Current: PVC is Bound
      Managed by percona-server-mongodb-operator with cluster1 instance by percona-server-mongodb application within component mongod
      PVC uses PersistentVolume/pvc-8a2531e5-bd8e-45a5-8174-20fbf99b9ea4, with Filesystem mode, asks for 3Gi, provisioned by pd.csi.s
9725-default-pool-2646b695-mjv8
      Known/recorded manage events:
        4h ago Updated by kube-scheduler (metadata)
        3h ago Updated by kube-controller-manager (metadata, spec)
        3h ago Updated by kube-controller-manager (status)
      Binds:
        PersistentVolume/pvc-8a2531e5-bd8e-45a5-8174-20fbf99b9ea4, created 3h ago Bound
          Current: Resource is current
          PV is Bound managed by StorageClass/standard-rwo provisioned by pd.csi.storage.gke.io with ReadWriteOnce mode
          Created for PersistentVolumeClaim/mongod-data-cluster1-rs0-0 -n psmdb-21113
          Known/recorded manage events:
            3h ago Updated by csi-attacher (metadata)
            3h ago Updated by csi-provisioner (metadata, spec)
            3h ago Updated by kube-controller-manager (status)
```

PERCONA

Sometimes you need to take control!

Bypass the StatefulSet lock.

```
NAME                                READY   STATUS    RESTARTS   AGE
cluster1-rs0-0                      4/4     Running   0          111m
cluster1-rs0-1                      3/4     Running   0          110m
```

```
$ kubectl exec cluster1-rs0-1 -- touch /data/db/sleep-forever
```

MySQL: /var/lib/mysql/sleep-forever, PostgreSQL: /pgdata/sleep-forever

```
NAME                                READY   STATUS    RESTARTS   AGE
cluster1-rs0-0                      4/4     Running   0          113m
cluster1-rs0-1                      4/4     Running   0          112m
cluster1-rs0-2                      4/4     Running   0          110m
```

Stop the reconciliation.

```
$ kubectl patch pg cluster1 \
    --type=merge \
    -p '{ "spec": { "unmanaged": true } }'
```

or if the operator doesn't have this kind of feature:

```
$ kubectl scale deployment percona-server-mongodb-operator \
    --replicas=0
```

Use your toolbox.

Ephemeral containers to the rescue.

Since Kubernetes v1.25

```
$ kubectl debug pod/cluster1-pxc-1 -it \
    --target=pxc \
    --share-processes=true \
    --image=egegunes/troubleshoot:ubi9 \
    --profile=sysadmin
```

Ephemeral containers to the rescue.

Since Kubernetes v1.25

```
$ kubectl debug pod/cluster1-pxc-1 -it \
    --target=pxc \
    --share-processes=true \
    --image=egegunes/troubleshoot:ubi9 \
    --profile=sysadmin
```

Ephemeral containers to the rescue.

Since Kubernetes v1.25

```
$ kubectl debug pod/cluster1-pxc-1 -it \
    --target=pxc \
    --share-processes=true \
    --image=egegunes/troubleshoot:ubi9 \
    --profile=sysadmin
```

You can debug a node as well.

```
$ kubectl debug node/worker-node-0 -it \
    --image=egegunes/troubleshoot:ubi9 \
    --profile=sysadmin
```

The root filesystem of the Node will be mounted at /host.

With debug containers you can run:

- `percona-toolkit` collection,
- `strace`,
- `tcpdump`,
- `iostat` and more.

You can go crazier.

```
$ kubectl exec -it cluster1-pxc-0 -- bash
bash-5.1$ vim
bash: vim: command not found
```

No vim?? Let's install it!

```
$ kubectl debug -it cluster1-pxc-0 \
    --target=pxc \
    --share-processes=true \
    --image=registry.access.redhat.com/ubi9/ubi \
    --profile=sysadmin

$ yum install --downloadonly --downloaddir=/tmp/rpm vim
$ cp /tmp/rpm/* /proc/1/root/tmp/
$ nsenter --mount --net --pid --target 1 bash
$ cd /tmp/
$ rpm -Uvh --nodeps *.rpm
```

Voila!

```
$ kubectl exec -it cluster1-pxc-0 -- which vim
/usr/bin/vim
```

Epilogue

Experimenting with these beforehand would help you be prepared when things break.

Things will eventually break.

Percona Live is back!

PERCONA Live 26
BAY AREA

May 27-29, 2026
Computer History Museum,
Mountain View, California

perconalive.com

PERCONA

Reach out to us!

- Create topics on Percona Community Forum,
- Creates issues on GitHub,
- Contribute code!

Thank you for listening!