Redis Enterprise on Kubernetes

A powerful data platform for modern applications

Container technology has transformed how applications are delivered and have become the basic unit of deployment and operations for modern applications.

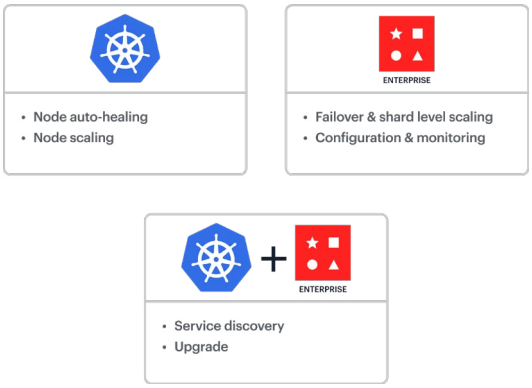
Kubernetes is a container orchestration platform that helps organizations embrace containers and microservices, handling their deployment and management at scale. Designed originally to support stateless applications, Kubernetes added support for stateful applications more than five years ago through the StatefulSets primitive that allows data to persist beyond the lifecycle of a container.

A Kubernetes primitive or Kubernetes object is a building block with specific features and functions that manage resources within the Kubernetes cluster.

Redis Enterprise takes advantage of the Kubernetes StatefulSet primitive and other primitives to deploy and orchestrate Redis Enterprise pods as a stateful service.

Why Redis Enterprise on Kubernetes?

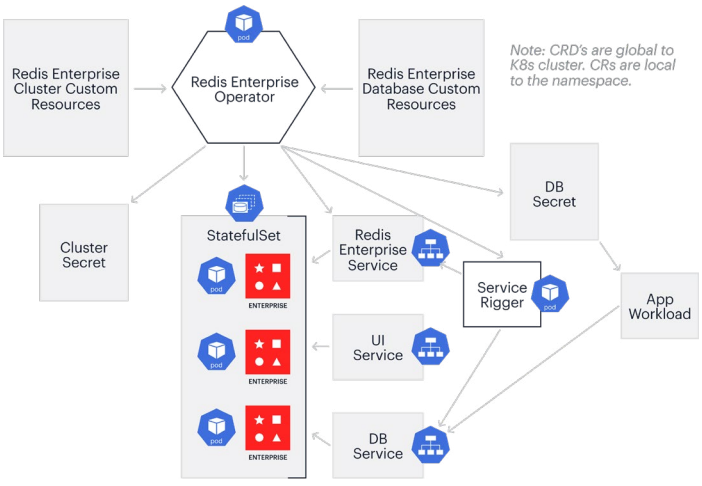
Kubernetes is a very efficient and effective tool to manage and orchestrate a Redis Enterprise cluster running as a stateful service in containers. Data is retained even after a container is shut down or migrated, enhancing operational readiness and accelerating app development and delivery.



Manage Kubernetes clusters at scale

The Redis Enterprise Operator for Kubernetes is a tool that's used to to control the provisioning, scaling, and availability of the Redis Enterprise database as well as managing the containers' lifecycle in any infrastructure.

The Operator delivers cloud portability, including on-prem and hybrid infrastructure, automating cluster operations, relieving organizations from complex infrastructure administration, thereby providing great flexibility to adopt agile practices.



Benefits of Redis Enterprise on Kubernetes

With Redis Enterprise as a service on Kubernetes, application developers are better empowered to rapidly spin up database instances on demand, breaking the development/operations barrier with benefits such as:

1. Deliver persistent storage by attaching the same persistent disk to a pod even when rescheduling to a new node.
2. Auto bootstrap the Redis Enterprise Cluster pods securely to enable on-demand scaling of pods using native Kubernetes primitives and reducing operational overhead.
3. Perform rolling upgrades with zero downtime and apply updates across the entire cluster by incrementally updating pod instances with zero downtime.
4. Enable automatic service discovery with the Redis Enterprise custom controller to automatically publish the new or deleted database endpoints to the Kubernetes service catalog.
5. Gain platform independence with flexible deployment options and ensure seamless portability across any cloud-native platforms, including Amazon Elastic Container Service for Kubernetes (EKS), Google Kubernetes Engine (GKE), Microsoft Azure Kubernetes Service (AKS), Red Hat OpenShift, VMware Tanzu, Rancher Kubernetes Engine (RKE), and Community Kubernetes (kOps).

How it works

1. The Redis Enterprise Operator for Kubernetes reads and validates the cluster resource definition (CRD) file for a consistent Kubernetes cluster specification.
2. By leveraging the Kubernetes StatefulSets primitive, the operator deploys Redis Enterprise as a persistent service.
3. The Operator uses the Redis Enterprise database custom resources (CR) file to validate the definition of the Redis Enterprise database.
4. The Operator creates the database using the Redis Enterprise headless service of the cluster. The database access credentials are stored in a Secret primitive to protect sensitive cluster information, such as passwords.
5. The service rigger discovers the new database and configures the Kubernetes service.
6. The LoadBalancer primitive exposes the REST API and the web interface of Redis Enterprise for a consistent operational workflow from outside the Kubernetes cluster.

Key features

- **Data Persistence.** Leverage persistent volumes for storage and outlive the lifecycle of a container for data persistence.
- **Improve availability.** Maintain uptime with pod readiness checks and automatic pod recovery. Ensure instant failover and recovery within single-digit seconds for a true HA solution.
- **Operate at scale.** Scale seamlessly across multiple Kubernetes pods with a declarative blueprint of the desired configuration and state. Linearly scale out your Redis Enterprise database with stable and predictable performance.
- **Tenant isolation.** Maximize resource utilization and minimize costs by serving a multi-tenant database model capable of serving multiple applications and to dynamically scale across multiple pods.

Quick facts

- Most loved NoSQL database - Stack Overflow
- #1 database downloaded on Docker Hub
- Easily manage Redis Enterprise cluster on Kubernetes
- Reduce operational complexity with built-in automation
- Run on any cloud and hybrid architectures
- Containerized database with sub-millisecond performance
- Infinite, seamless linear scale
- True high-availability and instant auto-failover
- Lower TCO with built-in multi-tenancy and tenant-level tunability
- Future proof technology for application growth

Get started:

Contact us

Learn more from our tutorials

Learn more about our Kubernetes Operator