redislabs
HOME OF REDIS

DATA SHEET

Redis Enterprise as a Stateful Service on Kubernetes

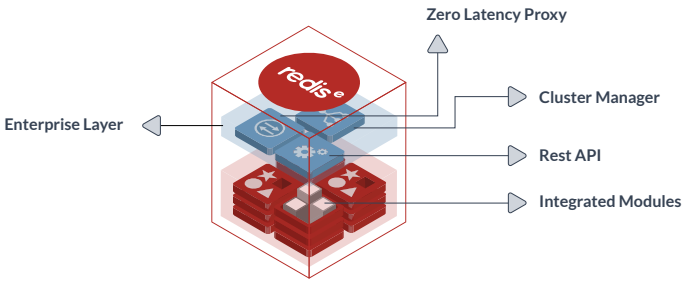
Embracing a stateful database for your microservices

Container technology is transforming application delivery and is rapidly becoming the basic unit of deployment for modern applications. Kubernetes is a container orchestration platform that helps organizations embrace containers and microservices and handles their deployment and management at scale. To orchestrate stateful applications, Kubernetes introduced StatefulSets primitive that allows managing a collection of Kubernetes pods with stateful properties.

Redis Enterprise takes advantage of the Kubernetes StatefulSet primitive, along with other primitives, to deploy and orchestrate Redis Enterprise pods as a stateful service seamlessly in a Kubernetes cluster.

Why Redis Enterprise on Kubernetes?

Running and orchestrating Redis Enterprise as a stateful service in a container is efficiently managed with Kubernetes. Stateful service enables data to be retained even after a container has been shut down or migrated. This reduces deployment complexity, enhances operational readiness and accelerates application development and delivery.



- **Persist data and cluster state:** Leverage persistent volumes for storage and outlive the lifecycle of a container for data persistence
- **Improve availability of data services with HA and instant failover:** Maintain uptime with pod readiness checks and automatic pod recovery. Ensure instant failover and recovery within single digit seconds
- **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
- **Deliver tenant isolation, agility and economy of scale:** Dynamically scale out the Redis Enterprise cluster across multiple pods; maximize resource utilization and minimize costs by serving a multi-tenant database model serving multiple applications

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.

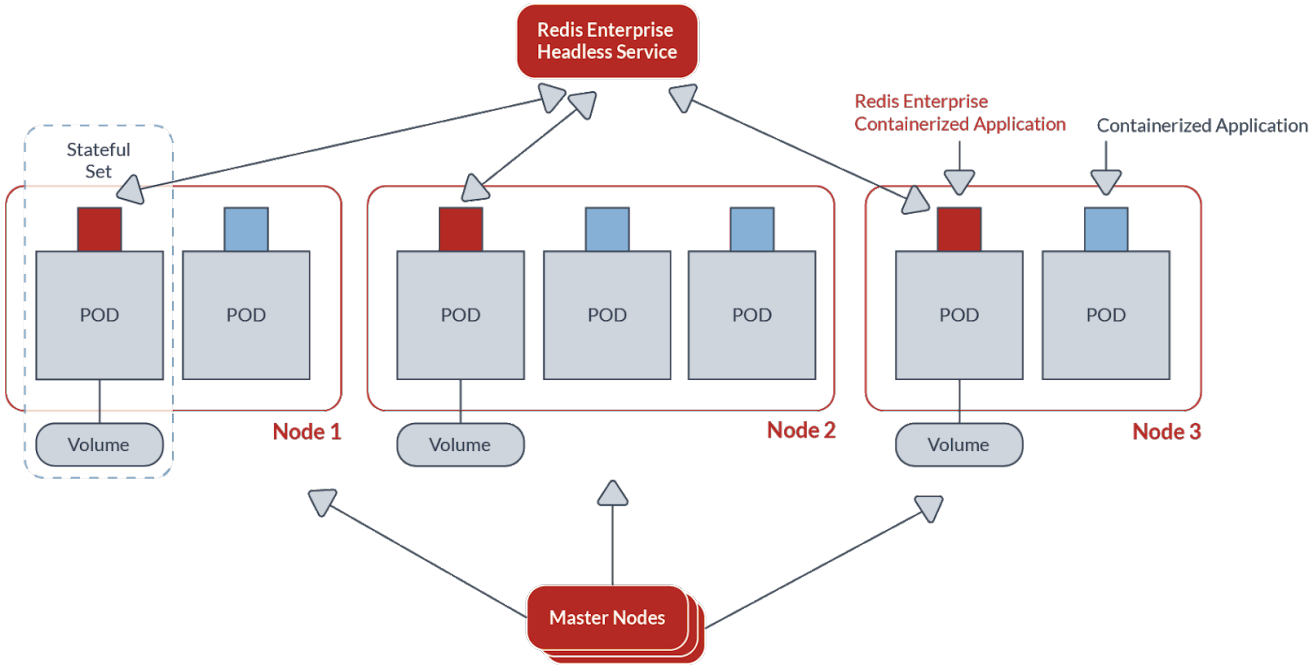
1. Deliver persistent storage by attaching the same persistent disk to a pod even when it gets rescheduled to new node
2. Auto bootstrap the redis-enterprise cluster pods in a secure manner. This enables on-demand scaling of pods using native Kubernetes primitives and reduces operational overhead
3. Perform rolling upgrades with zero downtime and apply updates across the entire cluster by incrementally updating pods 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), Pivotal Container Services (PKS) and Red Hat OpenShift

Quick Facts:

Redis Enterprise - Multi-model database for modern applications

- Most loved database - StackOverflow
- #1 database downloaded on Docker Hub
- Exceptionally fast 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
- Data persistence and durability at memory speed
- Runs on any cloud and hybrid deployment architectures
- Future proof technology for application growth

How it works



1. Redis Enterprise has built-in support for Helm charts. This helps you avoid error prone manual configuration and accelerates deployment using a single command to a Kubernetes namespace of your choice
2. Leveraging the Kubernetes StatefulSets primitive allows you to deploy Redis Enterprise as a persistent service in a consistent and reproducible way
3. The use of PersistentVolumes and PersistentVolumeClaims delivers flexibility to choose any storage to attach to the cluster for a persistent Redis Enterprise database
4. The Redis Enterprise headless service helps you automatically handle the DNS resolution of pods in the deployment. The headless service enables you to connect to any of the Redis Enterprise Kubernetes pods
5. The custom Redis Enterprise service controller publishes all new and deleted endpoints of the database in the Kubernetes service catalog
6. The Secrets primitive secures your Redis Enterprise deployment in Kubernetes and protects your sensitive configuration data like password and license information from accidental exposure
7. Using the LoadBalancer Kubernetes primitive, the Redis Enterprise web interface and REST API are exposed for a consistent operational workflow from outside the Kubernetes cluster

Get started with Redis Enterprise on Kubernetes today!

Download the Redis Enterprise Helm Charts:
https://github.com/RedisLabs/redislabs-helm
Download the Docker container image:
https://hub.docker.com/r/redislabs/redis/
Talk to a Redis Enterprise expert:
Contact expert@redislabs.com.

redislabs
HOME OF REDIS

700 E El Camino Real Suite 250, Mountain View, CA 94040 (415) 930 9666 redislabs.com