Zero to Cloud-Native with IBM Cloud

Part 5: IBM Cloud Databases Provisioning and Configuration

Kevin Collins
Technical Sales Leader
IBM Cloud Enterprise Containers – Americas

IBM Cloud Databases

Introduction

Developing cloud native applications vs traditional applications is very different from a data perspective. In a traditional monolith application, it is very common to have one large SQL based database. Cloud native architecture on the other hand encourages multiple purpose built databases. A best practice while developing cloud native application is isolating the data where each service manages its own data.

While you grow your application architecture and have a large number of indepdent databases, following good 12-factor app design, and also have a wide variety of databases, it's a lot to manage. There are a number of database types you may want to leverage from Redis as a caching database, to a high performance document database like Cloudant, a general purpose SQL database like PostgreSQL, or maybe a high performance key value store database like etcd. This is on on the great parts of moving to cloud native in being able to select the best database type for your service, you are not stuck to just one like we typically see with a monolith. But how are you going to learn how to support, scale and secure each database.

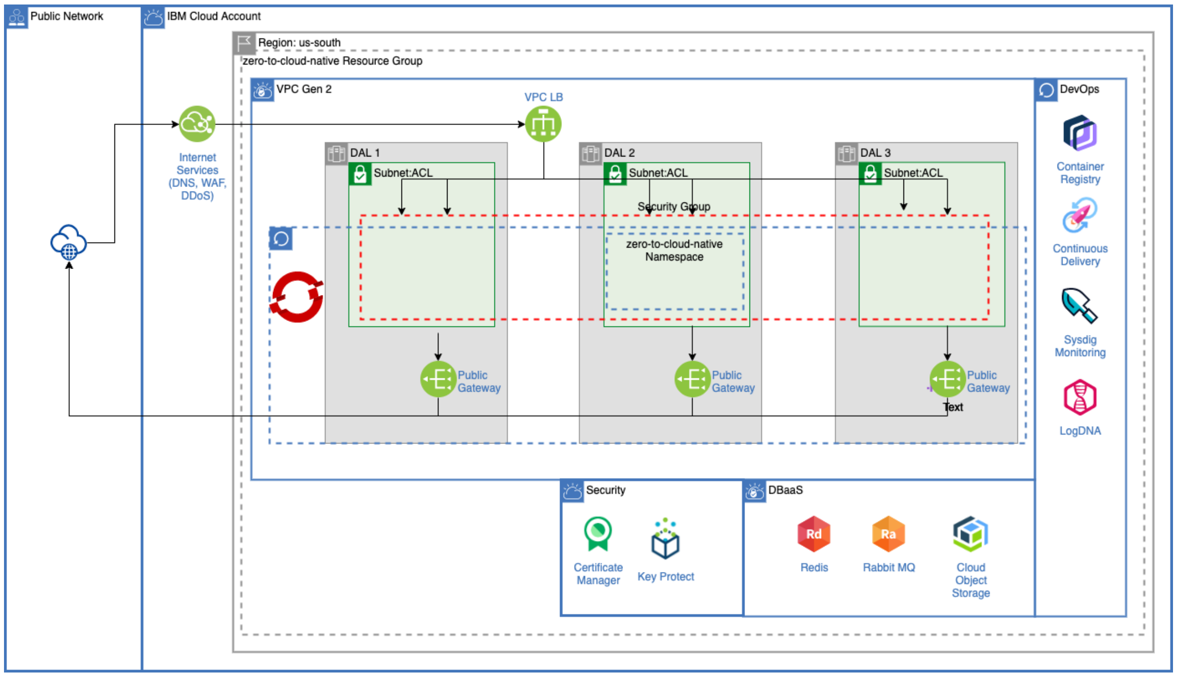
Enter IBM Cloud Databases. IBM Cloud Databases supports a wide range of databases that are managed for you.

IBM Db2 Cloudant MongoDB Redis PostgreSQL MySQL Elasticsearch ScyllaDB JanusGraph RabbitMQ etcd

This means that IBM takes care of all the hard parts of provision, supporting and maintaining different databases. IBM takes care of monitoring, backing up, making the database highly available, providing all the hardware and networking, and on top of that secures your databases. Another major benefit is these databases are provision in minutes. Providing all of these database services allows you to try out different databases to see which one best meets your application architecture. It's better to fail fast by developing say a quick MVP using MongoDB and being able to switch quickly to say a Cloudant database with minimal investment.

Tutorial – IBM Cloud Databases

Referring back to our application architecture of our tutorial application, we will be deploying two database / messaging services (Redis and RabbitMQ) that are part of IBM Cloud Databases.



Tutorial – IBM Cloud Databases for Redis

In the zero-to-cloud-native application we are going to develop and deploy, we are going leverage caching to return data very quickly. One of the most common use cases for using caching is to decrease data access latency. This is very important for frontend applications that may need to pull data to display in a user interface.

For this tutorial, we are going to simulate a long running task where we will replace the long running task with a cache using IBM Cloud Databases for Redis. We are going to do this through calling an API provided by IBM Cloud that retrieves a list of managed OpenShift versions. You can read and try the API here: https://containers.cloud.ibm.com/global/swagger-global-api/#/beta/v2GetVersions

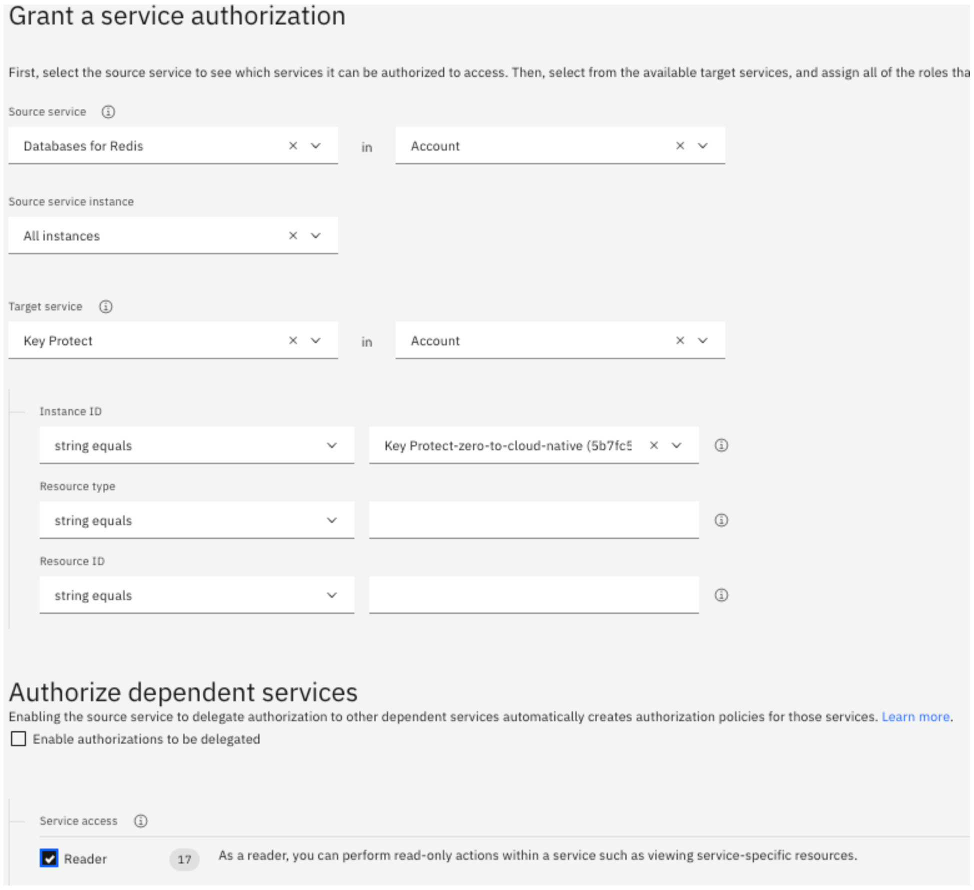
While this API does not take that long to run you will be able to understand how to implement caching for those long running APIs or database calls.

Before we can deploy our code, which as you will see leverages Redis, we need to deploy an instance that will store versions of IBM Cloud Managed OpenShift.

Before provisioning IBM Cloud Databases for Redis, we will need to authorize the service to access out Key Protect instance that we create in part 3. You will need provide this authorization before provisioning the IBM Cloud Databases for Redis instance.

We will follow that same steps that we did in part 3, in authorizing Cloud Internet Services to access Certificate Manager.

1. Navigate to **IBM Cloud > Manage > Access (IAM) > Authorizations**.
2. Click **Create** and assign a source and target service. The source service is granted access to the target service based on the roles that you set in the next step.
- Source: IBM Cloud Databases for Redis
- Target: Key Protect
3. Specify a service instance for both the source and the target.
4. Assign the **Reader** role to allow IBM Cloud Databases for Redis to access the Key Protect. Then, click **Authorize**.


Grant a service authorization

First, select the source service to see which services it can be authorized to access. Then, select from the available target services, and assign all of the roles tha

Source service: Databases for Redis in Account

Source service instance: All instances

Target service: Key Protect in Account

Instance ID: string equals Key Protect-zero-to-cloud-native (5b7fc5

Resource type: string equals

Resource ID: string equals

Authorize dependent services

Enabling the source service to delegate authorization to other dependent services automatically creates authorization policies for those services. Learn more.

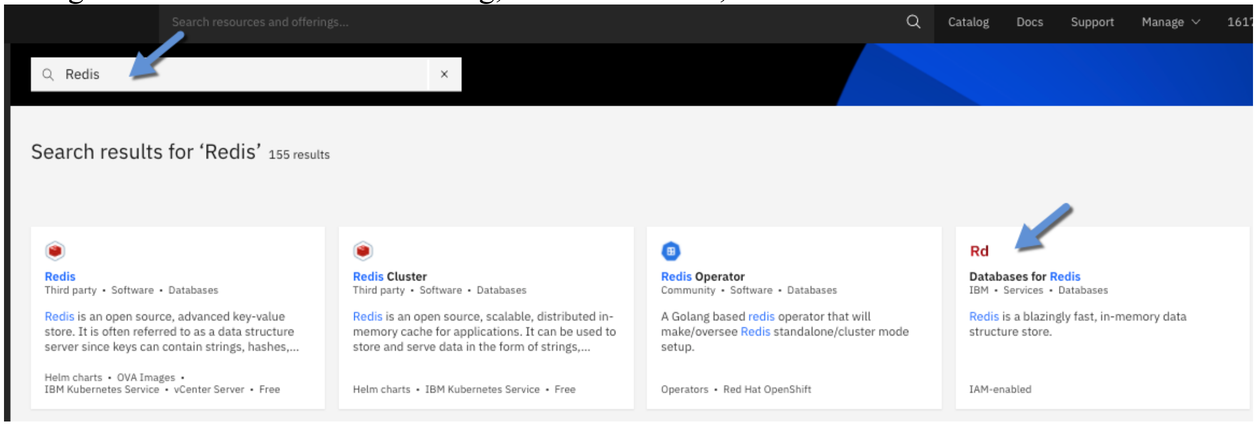
☐ Enable authorizations to be delegated

Service access

☑ Reader 17 As a reader, you can perform read-only actions within a service such as viewing service-specific resources.

You will notice that I selected all instances in my account for Database for Redis to a specific instance of Key Protect. The reason for this is because I may want to create multiple Redis databases that I want to use Key Protect to encrypt the data, and don't want to have to go in and authorize provide this authorization. If you do want to be more granular with your authorizations then you certain can.

Going back to the IBM Cloud Catalog, search for Redis, and click the Databases for Redis tile.



Search results for 'Redis' 155 results

Redis
Third party • Software • Databases
Redis is an open source, advanced key-value store. It is often referred to as a data structure server since keys can contain strings, hashes,...
Helm charts • OVA Images • IBM Kubernetes Service • vCenter Server • Free

Redis Cluster
Third party • Software • Databases
Redis is an open source, scalable, distributed in-memory cache for applications. It can be used to store and serve data in the form of strings,...
Helm charts • IBM Kubernetes Service • Free

Redis Operator
Community • Software • Databases
A Golang based redis operator that will make/oversee Redis standalone/cluster mode setup.
Operators • Red Hat OpenShift

Databases for Redis
IBM • Services • Databases
Redis is a blazingly fast, in-memory data structure store.
IAM-enabled

You will notice that there are several Redis options. If you want to manage, scale and secure Redis yourself you can install it directly on your cluster. For the reasons mentioned in the introduction, we will be using IBM Cloud Databases for Redis.

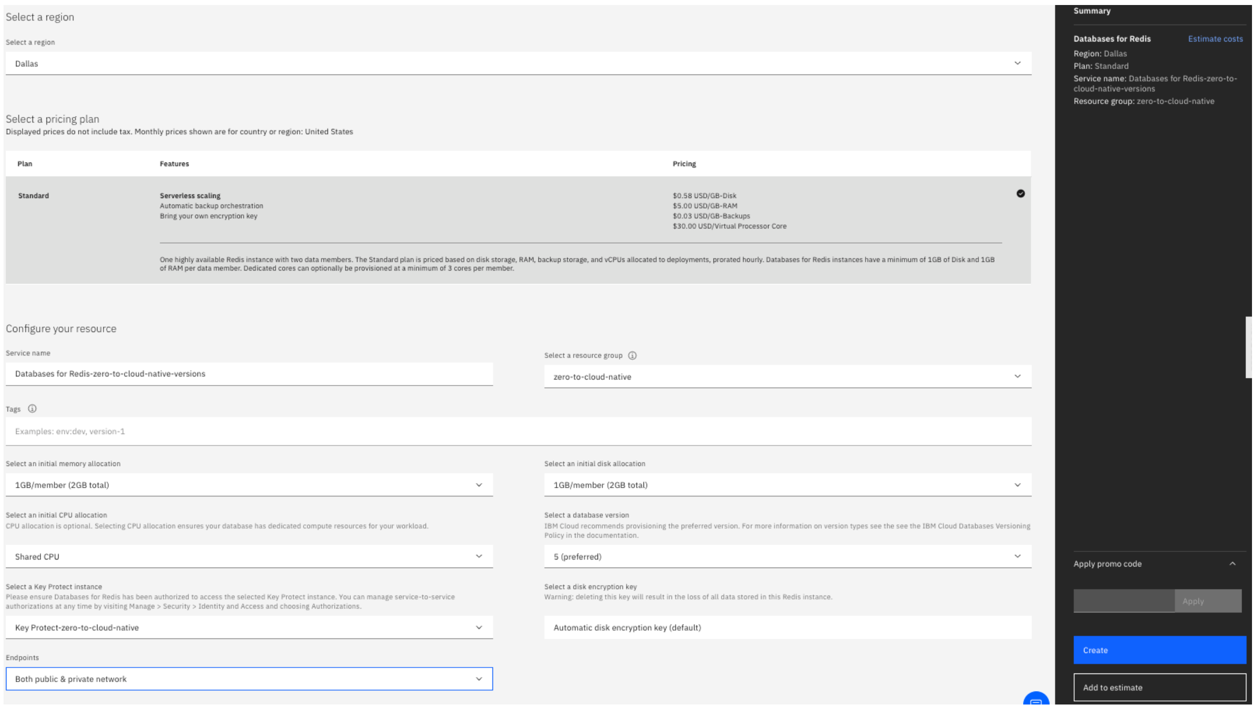
The first thing I'm going to select is Dallas for the region. I want the Redis instance to be in the same geographic area as my cluster for best performance.

Next, we need to give the Redis instance a name. Best practice is to have a Redis instance for each service that will leverage it. I will name it Databases for Redis-zero-to-cloud-native-versions which I can tell which application and service of the application that this instance of Redis will be used for.

I'll change the resource group to the zero-to-cloud-native resource group I've been using for all my IBM Cloud services.

The last thing I'm going to change is I want the service to leverage the Key Protect instance that we created in part 3, to encrypt data in the Redis database.

After entering all these settings, click create.


Select a region

Select a region

Dallas

Select a pricing plan

Displayed prices do not include tax. Monthly prices shown are for country or region: United States

Plan	Features	Pricing
Standard	**Serverless scaling** Automatic backup orchestration Bring your own encryption key	$0.58 USD/GB-Disk $5.00 USD/GB-RAM $0.03 USD/GB-Backups $30.00 USD/Virtual Processor Core

One highly available Redis instance with two data members. The Standard plan is priced based on disk storage, RAM, backup storage, and vCPUs allocated to deployments, prorated hourly. Databases for Redis instances have a minimum of 1GB of Disk and 1GB of RAM per data member. Dedicated cores can optionally be provisioned at a minimum of 3 cores per member.

Configure your resource

Service name: Databases for Redis-zero-to-cloud-native-versions

Select a resource group: zero-to-cloud-native

Tags: Examples: env:dev, version-1

Select an initial memory allocation: 1GB/member (2GB total)

Select an initial disk allocation: 1GB/member (2GB total)

Select an initial CPU allocation
CPU allocation is optional. Selecting CPU allocation ensures your database has dedicated compute resources for your workload.
Shared CPU

Select a database version
IBM Cloud recommends provisioning the preferred version. For more information on version types see the see the IBM Cloud Databases Versioning Policy in the documentation.
5 (preferred)

Select a Key Protect instance
Please ensure Databases for Redis has been authorized to access the selected Key Protect instance. You can manage service-to-service authorizations at any time by visiting Manage > Security > Identity and Access and choosing Authorizations.
Key Protect-zero-to-cloud-native

Select a disk encryption key
Warning: deleting this key will result in the loss of all data stored in this Redis instance.
Automatic disk encryption key (default)

Endpoints: Both public & private network

Summary

Databases for Redis — Estimate costs

Region: Dallas
Plan: Standard
Service name: Databases for Redis-zero-to-cloud-native-versions
Resource group: zero-to-cloud-native

After clicking create, your database will be ready to use in a few minutes. This is why I strongly recommend using IBM Cloud Databases to manage and provision your cloud native databases, it doesn't get easier than this to setup and configure.

Tutorial – IBM Cloud Messages for RabbitMQ

We will be using RabbitMQ as the messaging layer in-between microservices. Just like with Redis, I don't want to have to worry about how I'm going to provision, scale and secure RabbitMQ. I want IBM Cloud to do all the hard work for me, so I'm going to leverage IBM Cloud Messages for RabbitMQ.

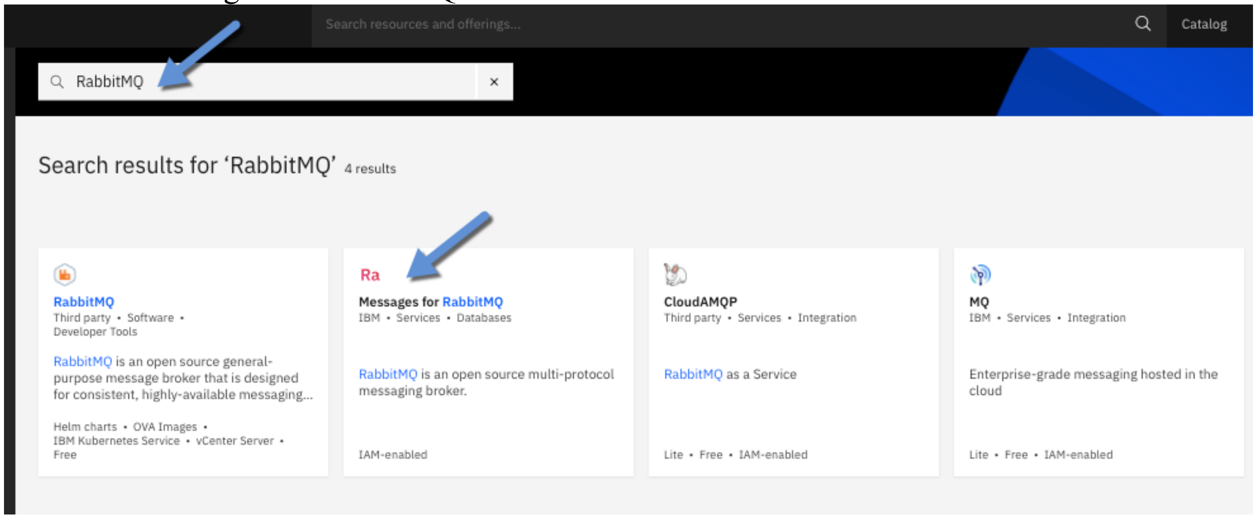
Just like we did the Redis, we will need before provisioning IBM Cloud Messages for RabbitMQ, we will need to authorize the service to access our Key Protect instance that we created in part 3. You will need provide this authorization before provisioning the IBM Cloud Messages for RabbitMQ instance.

We will follow that same steps that we did above with Redis.

5. Navigate to **IBM Cloud > Manage > Access (IAM) > Authorizations**.
6. Click **Create** and assign a source and target service. The source service is granted access to the target service based on the roles that you set in the next step.
- Source: Messages for RabbitMQ
- Target: Key Protect
7. Specify a service instance for both the source and the target.
8. Assign the **Reader** role to allow IBM Cloud Messages for RabbitMQ to access the Key Protect. Then, click **Authorize**.

This time, I'm going to give Messages for RabbitMQ authorization for every Key Protect service in the zero-to-cloud-native resource group so you can see different granularity options.

Now that we have authorized Messages for RabbitMQ with Key Protect we can provision tour RabbitMQ service. To get started, go to the IBM Cloud Catalog, search for RabbitMQ, and then click on the Messages for RabbitMQ tile. Just like you saw this Redis, IBM Cloud offers different RabbitMQ services from do it yourself to a managed service with IBM Cloud Databases Messages for RabbitMQ.



Search results for 'RabbitMQ' 4 results

RabbitMQ
Third party • Software • Developer Tools
RabbitMQ is an open source general-purpose message broker that is designed for consistent, highly-available messaging...
Helm charts • OVA Images • IBM Kubernetes Service • vCenter Server • Free

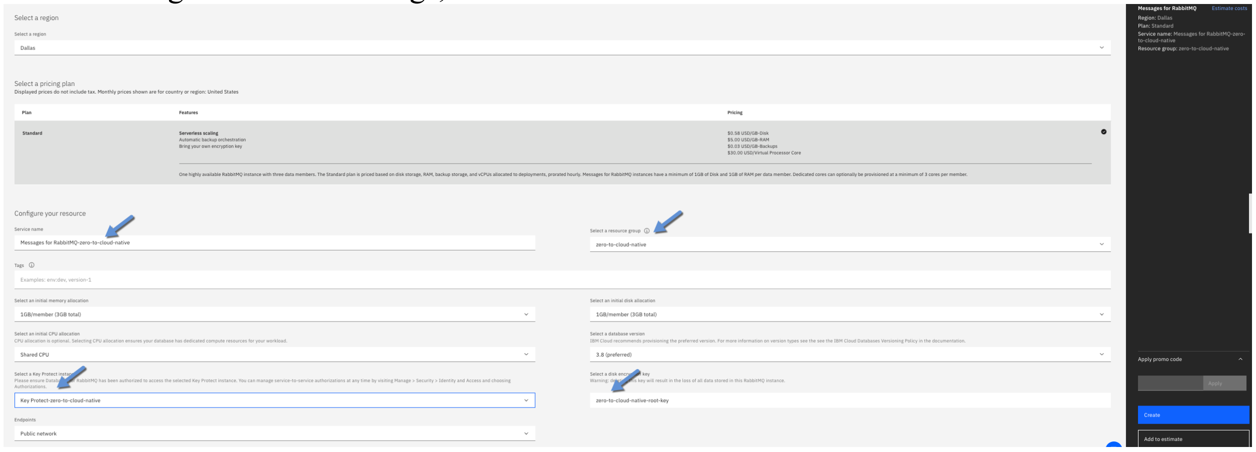
Messages for RabbitMQ
IBM • Services • Databases
RabbitMQ is an open source multi-protocol messaging broker.
IAM-enabled

CloudAMQP
Third party • Services • Integration
RabbitMQ as a Service
Lite • Free • IAM-enabled

MQ
IBM • Services • Integration
Enterprise-grade messaging hosted in the cloud
Lite • Free • IAM-enabled

To create your Messages for Rabbit MQ instance you need to select a region. Like we have with all our other services we want to choose Dallas so all of our services are in the same geographic region.

Next, give you service a meaningful name such as Messages for RabbitMQ-zero-to-cloud-native.
Select the zero-to-cloud-native resource group.
To leverage our Key Protect instance we create earlier, select the zero-to-cloud-native Key Protect instance along with the encryption key we created in step 3.
After entering all of these settings, click Create.



Select a region

Select a region: Dallas

Select a pricing plan
Displayed prices do not include tax. Monthly prices shown are for country or region: United States

Plan	Features	Pricing
Standard	**Serverless scaling** Automatic backup orchestration Bring your own encryption key	$0.58 USD/GB-Disk $5.00 USD/GB-RAM $0.03 USD/GB-Backups $30.00 USD/Virtual Processor Core

One highly available RabbitMQ instance with three data members. The Standard plan is priced based on disk storage, RAM, backup storage, and vCPUs allocated to deployments, prorated hourly. Messages for RabbitMQ instances have a minimum of 1GB of Disk and 1GB of RAM per data member. Dedicated cores can optionally be provisioned at a minimum of 3 cores per member.

Configure your resource

Service name: Messages for RabbitMQ-zero-to-cloud-native

Select a resource group: zero-to-cloud-native

Tags: Examples: env:dev, version-1

Select an initial memory allocation: 1GB/member (3GB total)

Select an initial disk allocation: 1GB/member (3GB total)

Select an initial CPU allocation
CPU allocation is optional. Selecting CPU allocation ensures your database has dedicated compute resources for your workload.
Shared CPU

Select a database version
IBM Cloud recommends provisioning the preferred version. For more information on version types see the see the IBM Cloud Databases Versioning Policy in the documentation.
3.8 (preferred)

Select a Key Protect instance
Please ensure Databases for RabbitMQ has been authorized to access the selected Key Protect instance. You can manage service-to-service authorizations at any time by visiting Manage > Security > Identity and Access and choosing Authorizations.
Key Protect-zero-to-cloud-native

Select a disk encryption key
Warning: deleting this key will result in the loss of all data stored in this RabbitMQ instance.
zero-to-cloud-native-root-key

Endpoints: Public network

Messages for RabbitMQ
Region: Dallas
Plan: Standard
Service name: Messages for RabbitMQ-zero-to-cloud-native
Resource group: zero-to-cloud-native

Just like the provisioning of Cloud Databases for Redis, that is all you have to do. Messages for RabbitMQ will be available in a couple of minutes.

Now that we have created all the various IBM Cloud services our application will be using, we can now start looking at the microservices code. However, before going through that you need to have a development environment setup including a command line environment with various CLIs installed and configured, a development IDE, and other various tools that will make your cloud-native development life easier.