PostgreSQL Backup and Restore on Microsoft Azure using Kasten K10

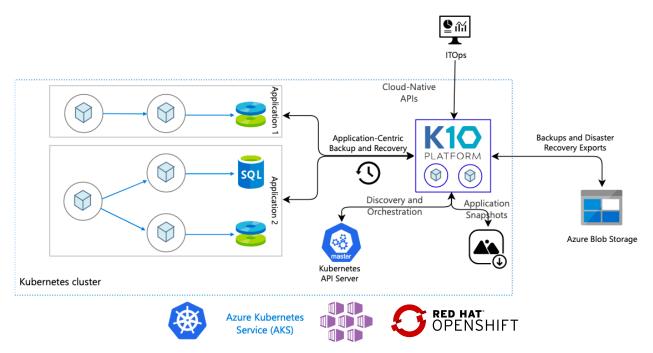


In this post, we will walk through how to use Kasten K10 to backup and restore PostgreSQL databases operating in a Kubernetes environment on Microsoft Azure. There are two primary options for running Kubernetes on Azure: <u>Azure Kubernetes Service (AKS)</u>, and <u>Azure Red Hat OpenShift</u>. This post is based on AKS.

The fully managed AKS makes deploying and managing containerized applications easy and offers serverless Kubernetes, an integrated continuous integration and continuous delivery (CI/CD) experience, and enterprise-grade security and governance.

PostgreSQL (often referred to as Postgres), is an Open Source relational database, popular in the cloud-native community.

Kasten's K10 data management platform, it is a secure software-only product that has been purpose-built for Kubernetes and provides operations teams an easy-to-use, scalable, and secure system for backup/restore, disaster recovery, and mobility of Kubernetes applications.



Top level reference diagram of Kasten K10 on Azure

This assumes that you already have an AKS cluster set up. If not, you can follow instructions <u>here to deploy an AKS cluster using the Azure portal</u>. The instructions in this post are organized in three sections:

- 1. Installing Kasten K10 on your AKS cluster
- 2. Installing PostgreSQL
- 3. Backup and restore workflow using Kasten K10

1. Installing Kasten K10 on Your AKS Cluster

Detailed instructions for installing K10 are available in the K10 documentation. In this example a "happy path" install is used for demo purposes. Before proceeding with the install, the install prerequisites (Helm package manager and Kasten Helm charts repository) need to be satisfied. The Helm commands use Helm v3, but using Helm v2 is also straightforward.

Add the Kasten Helm charts repository and create the namespace where K10 will be installed using the commands below.

```
$ helm repo add kasten https://charts.kasten.io/
$ kubectl create namespace kasten-io
```

Use the command below to install K10. You will need to specify your Azure tenant, service principal client ID, and service principal client secret.

\$ helm install k10 kasten/k10 --namespace=kasten-io \
 --set secrets.azureTenantId=<tenantID> \
 --set secrets.azureClientId=<azureclient_id> \
 --set secrets.azureClientSecret=<azureclientsecret>

To validate K10 install, use the command below in K10's namespace (kasten-io, by default) to confirm that all K10 pods display a status of Running within a couple of minutes.

\$ kubectl get pods --namespace kasten-io --watch

You can now access the K10 dashboard at http://127.0.0.1:8080/k10/#/ after running the command below.

\$ kubectl --namespace kasten.io port-forward service/gateway 8080:8000

2. Installing PostgreSQL

Use the commands below to create a namespace called postgresql and install PostgreSQL into your AKS cluster.

- \$ helm repo add bitnami https://charts.bitnami.com/bitnami
- \$ helm repo update
- \$ kubectl create namespace postgresql

\$ helm install my-release bitnami/postgresql

To validate the PostgreSQL install, use the command below in the postgresql namespace to confirm that all PostgreSQL pods display a status of Running within a couple of minutes.

\$ kubectl get pods --namespace postgresql

K10 automatically discovers the instance of PostgreSQL. Following the successful install of PostgreSQL, *click on the Applications card on the K10 dashboard to see the discovered PostgreSQL instance.*

< Dashbo	ard		
Appl	ications		
View details	or perform actions on applica	itions.	
የቆየ Filter	by status Y Filter by	name 2 applic.	ations 🔀 🗄
	default Not Protected by Policies		postgresql Not Protected by Policies
	🗲 Create a Policy 🗦		✓ Create a Policy >
	30 GB 🗐 3 ≪∃ 1 💿 3 🔘 4		8GB 🗐 1 ∘Ę 1 🞯 2 🔘 3
	© € snapshot restore export	i≣ details	©] snapshot restore export details

3. Backup and Restore Workflow using Kasten K10

In this example, we will use K10's default backup mechanism which relies on taking volume snapshots. *Click on the Applications card in the K10 dashboard and either create a backup policy, or for experimentation, simply create a restore point to do a full manual backup.*

	postgresql Not Protected by Policies	
	🗲 Create a Policy 🔿	
Manually	create a restore point for this application	
	© snapshot restore export	i≣ details

Check the progress of the backup action in the main K10 dashboard.

Activity	ns 🔵 – running 🌒	- completed - failed				
total jobs 1	completed jobs	failed jobs O	skipped jobs O	avg job duration 41 sec	live artifacts 17	retired artifacts O
Jobs		能 Filter				
Backup	O Snapshotting Applic O Snapshotting Applic O Snapshotting Workle		APP HAMISPACE postgresql	ORIGINATING POLICIES INDIAE	ARTHACTS 1 ③ snapshot 8 GB 12 ◎ spec	start Today, 6:30pm Duration 41 secs

Completion of the backup action will result in the creation of a restore point (a set of configuration and data artifacts) which can be used to restore from. To restore from the restore point, *go to the Applications card and click on* `*restore*` *button for the* `*postgresql*` *application.* Here you should see all the available restore points.

	postgresql Not Protected by Policies
	 ✓ Create a Policy > 8 GB
View res	store points for this application. Restore the same or different namespace.
	O O C Image: State of the

Click on the restore point. This will open the Restore panel where you can view and modify the restore parameters. *Click on the Restore button* to restore the associated data and specs.

Contractions Contractions

Restore application *postgresql*

Restore an application to a previous state. Restore points are shown and ordered based on scheduled execution time which may be different from the actual creation time. During a restore, the existing application is deleted and then recreated with the data artifacts restored from backups.

Select a restore point for details.

Past day

Today, 6:30pm 術 Manual Protect

Restore Point	×
SCHEDULED TIME CREATION TIME Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:00 Image: Mar 11, 2020 6:30 pm -07:0	
originating Policy	
KUBECTL COMMAND \$ kubectl getraw /apis/apps.kio.kasten.io/vialphai/restorepointcontents/postgre copy	
Application Name	
An existing application with the same name will be replaced with the restored application. Restore as "postgresql" Restore using a different name	
Optional Restore Settings Data-Only Restore Restore only the volume data and exclude other artifacts such as config files.	
⑦ Restore Cancel	

Check the progress of the restore action in the main K10 dashboard.

96 sec 72 sec - 60 sec - 24 sec - 0 sec							
total j		ns — running — - completed jobs 2	- completed - Laled failed jobs	skipped jobs O	avg job duration 63 sec	live artifacts 19	retired artifacts
Jobs			†於 Filter				
	store gresqlggsvk	Restoring Application	1 Components	TARGET NAMESPACE postgresql	omanutina rollaits none	леталстя 1 📋 volume 8 68	start Today, 9:40pm putation 1 min, 27 secs
	ckup usibackup-odwál	O Snapshotting Applica O Snapshotting Applica O Snapshotting Worklo	tion configuration	APP NAMESPACE postgresql	CRIGINATING POLICIES	ARTIFACTS 1 🖾 snapshot 8 68 12 🐵 spec	staat Today, 6:30pm ousamon 41 secs

The data and application configuration have been successfully restored.

Using Backup Policies

Note that the workflow demonstrated above used a manual backup. You can also create policies to execute backups on a scheduled basis. Policies are extremely configurable. You can set the backup schedule and snapshot retention schedule independently for fine-grained control over how often backups are performed and how much total storage they consume.

To try this out *click on Create New Policy* on the application card on the dashboard.

			Ne	w Policy				(
Action The action	that should be	e taken when thi	s policy	is executed				
	0	Snapshot			0	Import		
Action Fr	equency							
	Hourly	🔿 Daily	0	Weekly	O Mo	nthly	🔘 Yearly	
	r more times p	ies (optional) per hour. Retainir	ng 24 h	ourly snapsho	ts at 15 min	intervals,	would retain 6	
Run one o hours of si	r more times p napshots.		ng 24 h	ourly snapsho	ts at 15 min	intervals,	would retain 6	
Run one o	r more times p napshots.		ng 24 h	ourly snapsho	ts at 15 min	intervals,	would retain 6	-
Run one o hours of si Once a Snapsho	r more times p napshots. n hour t Retention	ber hour. Retainir			ts at 15 min	intervals,	would retain 6	•
Run one o hours of si Once a Snapsho	r more times p napshots. n hour t Retention	per hour. Retainin	ule if ne			intervals,	would retain 6	,

When a policy that applies to an application successfully executes a backup, the application's compliance with the policy is reported in the application card. In the screenshot below, we can see that our postgresql application is now compliant with all policies.



Advanced Use Cases: Disaster Recovery (DR) and Mobility

The workflow in this blog covers snapshot, backup, and restore in a single AKS cluster. K10 can be used to export the entire application stack and its data from production clusters and restore them to a geographically separate DR cluster. You can also mask data, store it in an object store, and then read it from your local development cluster. Such use cases are described in the K10 documentation.

Conclusion

This post has shared steps for backing up and restoring PostgreSQL running on Microsoft AKS using snapshots as the backup mechanism. You can also explore the more advanced backup and restore approaches (based on logical dumps and database quiescing) discussed here: <u>PostgreSQL Disaster</u> <u>Recovery and Data Mobility on Microsoft Azure using Kasten K10</u>.

In addition to backups and restores for PostgreSQL, K10 also supports backups and restores for a range of other relational databases (e.g., MySQL) and NoSQL systems (e.g., MongoDB or Elastic).