REFERENCE ARCHITECTURE

Protecting Kubernetes Apps

Increase applications protection using Pure Storage® FlashBlade® and Portworx® Enterprise for Kubernetes environments.

Contents

Introduction

Enterprises are quickly moving Kubernetes into production. How do you protect applications and data within your Kubernetes cluster? It's an inevitable question. Backup and recovery, business continuity, and disaster-recovery procedures are essential for any mission-critical applications running on Kubernetes infrastructures in production. Notably, Kubernetes falls short in the enterprise when it comes to disaster recovery and other business continuity techniques such as backup and restore. It is a common misconception that applications on Kubernetes don't need backup and restore because applications are "stateless and transient" or "distributed" and have "built-in failover." This misconception puts your most valuable resources, data, and ultimately your businesses at risk.

There are three reasons that traditional backup and recovery solutions don't work with Kubernetes. Traditional solutions:

- Focus on machines—but Kubernetes is all about applications
- Assume that the infrastructure from which you back up an application and where you recover it is the same—but Kubernetes apps are often backed up and restored across clouds, data centers, and platforms
- Are centrally administered—but Kubernetes is all about self-service

Resolving the issues of traditional approaches is where the Portworx® Storage Platform for Kubernetes comes in. It's a platform specifically designed to solve the significant operational challenges of running Kubernetes applications in production. PX-Backup, together with PX-DR, completes the application data-management and data-protection experience that you need when you run mission-critical applications on Kubernetes.

How to Use This Reference Architecture

This guide will show you how to use Portworx Enterprise (container-granular storage), PX-Backup (backup and restore), and PX-DR (disaster recovery) for Kubernetes. This guide uses Pure Storage® FlashArray™ as backing storage for Portworx Enterprise and uses Pure FlashBlade® and Azure Blob Storage as on-premises and cloud backup locations, respectively.

This reference architecture is for IT, storage, and Kubernetes administrators using Portworx by Pure Storage who want to build data protection and disaster recovery for both on-premises and cloud applications deployed on Kubernetes.

Pure Storage FlashBlade

Pure FlashBlade is the industry's most advanced all-flash storage solution for consolidating fast file and object data. Modernize your storage with a unified unstructured storage platform that delivers a Modern Data Experience™. FlashBlade delivers unprecedented performance in a small form factor. It is tuned to deliver multi-dimensional performance for any data size, structure, or access. And it delivers significant savings in power, space, and cooling compared to legacy solutions.

Single, Next-Gen Platform for Rapid Restore to Modern Analytics

Eliminate sprawling silos of complexity. The ideal solution is a single, modern data hub engineered to be fast and big for a wide range of workloads. From fast backup and recovery to instant access in test/dev to modern analytics and AI, a single, powerful platform should run all your applications. Pure Storage FlashBlade is one of the industry's first next-gen data platforms architected to consolidate all these applications. Now you can:

- Deliver instant restore for production and test/dev workloads.
- Consolidate silos, like backup appliances, tape, and data lakes, into a single platform.
- Leverage industry-leading solutions certified to deliver performance with Pure Storage.

Tuned-for-Everything Architecture

Next-gen storage must be architected for any type of data to accelerate a wide range of workloads. FlashBlade is uniquely engineered for unstructured data, delivering unprecedented performance for any workload from backup and restore to AI and analytics. FlashBlade's scale-out metadata architecture can handle tens of billions of files and objects with maximum performance and rich data services. Purity//FB supports cloud mobility with object replication and disaster recovery with file replication.

Deploying, updating, and managing FlashBlade is hassle-free thanks to automated APIs. Get high-performance native NFS, SMB, and S3 protocol support for all your modern file and object workloads. FlashBlade benefits from multiprotocol support with the addition of native Server Message Block (SMB) support to FlashBlade speeds backups and improves efficiency, delivering high performance for Windows applications and greater coverage of business needs and use cases.

Fast

- Elastic performance that grows with data, up to 17GB/s
- Always fast, from small to large files
- Massively parallel architecture from software to flash

Big

- Petabytes of capacity
- Elastic concurrency, up to tens of thousands of clients
- Tens of billions of objects and files

Simple

- An Evergreen™ subscription means you don't have to re-buy TBs you already own
- "Tuned for Everything" design doesn't require manual optimizations
- Instantly scale-out everything by simply adding blades

Portworx Enterprise Data Services Platform

Enterprise applications running on Kubernetes have non-negotiable business requirements like high availability, data security, backup and disaster recovery, strict performance SLAs and hybrid/multi-cloud operations. Portworx Enterprise was designed for precisely these applications. It includes the most scalable and robust capabilities (Figure 1) of the Portworx Storage Platform for Kubernetes with the ability to add-on PX-DR and PX-Backup for even greater levels of data protection. Scale up to 1000 nodes and 1 million volumes per cluster. For this architecture, let's look at PX-Store, PX-Backup, and PX-DR.

Figure 1: The Portworx portfolio is designed for enterprise applications.

PX-Store

PX-Store is the foundation of The Portworx Enterprise Data Services Platform for Kubernetes. Built from the ground up for containers, PX-Store provides cloud-native storage for applications running in the cloud, on-prem, and hybrid/multi-cloud environments. PX-Store provides:

- The reliability, performance, and data protection you'd expect from an enterprise storage company, but delivered as a container and managed 100% via Kubernetes and other leading container platforms.
- The ability to take your underlying hardware, even an existing SAN or NAS, and turn it into a cluster-wide storage pool for all your applications.
- Built-in high availability for all stateful applications and allows failed pods to recover in seconds.

PX-Backup

Built from the ground up for Kubernetes, PX-Backup delivers enterprise-grade application and data protection with fast recovery at the click of a button. Kubernetes creates a shift from machine-based infrastructure and operations to application-focused operations (Figure 2). This shift increases agility and reduces friction, but it creates problems for enterprises dependent on traditional backup tools for data protection. These solutions, built to back up machines, don't understand key Kubernetes concepts like container-granularity, namespaces, Kubernetes configurations, backing up distributed databases, and multicloud operations. PX-Backup allows you to:

- Back up entire Kubernetes applications, including data, app configuration, and Kubernetes objects across clouds.
- Restore any Kubernetes application to any Kubernetes cluster in the cloud or on-prem.
- Move a single Kubernetes resource, application, or an entire namespace between clusters in a single data center or between environments.

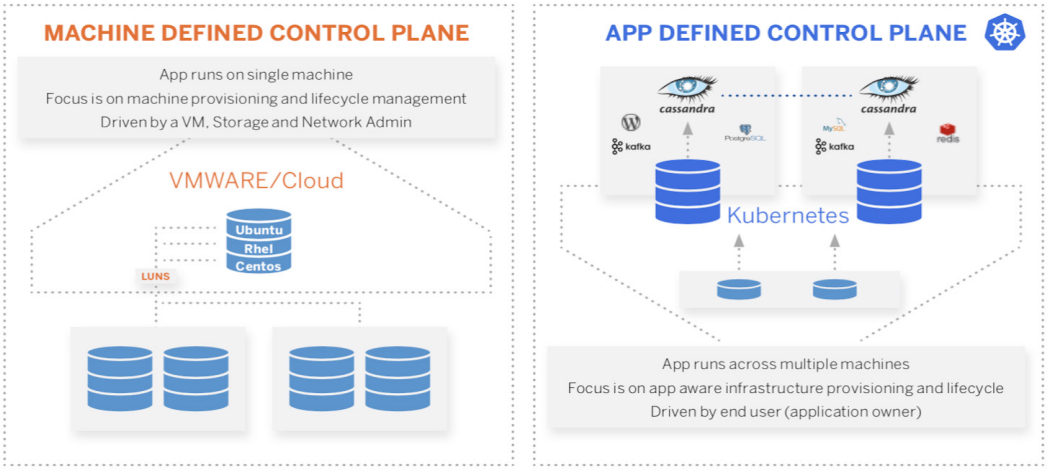

Figure 2. Shifting from machine-based infrastructure and operations to application-focused operations.

PX-DR

Portworx PX-DR delivers protection for all your mission-critical applications running on Kubernetes. Built exclusively for containerized applications, Portworx PX-DR protects your applications—data, application configuration, and Kubernetes objects—with a single command at the Kubernetes Pod, Namespace, or Cluster level. Enabling application-aware zero data loss and fast recovery for even complex distributed applications, PX-DR delivers true multicloud availability.

Portworx PX-DR extends Portworx Enterprise, the leading storage platform for Kubernetes, with a rich set of functionalities to deliver true multicloud application and data protection. PX-DR gives you:

- Zero data loss (RPO Zero) DR for data centers in a metropolitan area
- Continuous backup and protection across wide area networks (WAN)
- Protection for groups of Pods, a Namespace, or an entire cluster
- Single-command protection and restoration for applications
- Application-aware protection for complex distributed applications

Disaster Recovery, Backup, and Restore for Kubernetes Environments

FlashBlade is tuned to deliver multi-dimensional performance for any data size and structure, perfect for the dynamic workloads that run in modern cloud-native Kubernetes environments. Kubernetes objects comprise many objects, such as StatefulSets, ConfigMaps, Services, Secrets, and Persistent Volume Claims. These objects are considered metadata about the construction of an application but some hold actual data such as ConfigMaps, Secrets, and Persistent Volumes. These stateful applications in Kubernetes and can own anywhere from a few megabytes of data to multi-terabytes. A backup solution should be able to capture all these data types and offer flexible and efficient backup and restore operations for the enterprise. PX-Backup and FlashBlade does just that.

Solution Architecture

Figure 3 shows the logical architecture for the Data Protection solution for Kubernetes with FlashBlade. The data source represents any object or data within Kubernetes, which could be a deployment or a full application consisting of many Kubernetes objects and data within persistent volumes. PX-Backup is configured with FlashBlade as a backup location. PX-Backup then sends backup objects and volume snapshots to this location, staging it for restore operations.

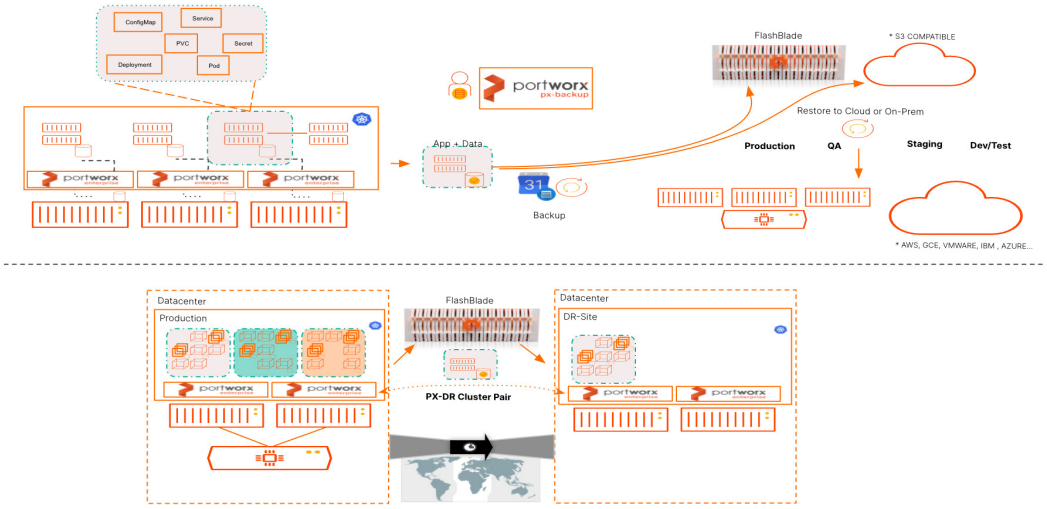

Figure 3. Logical architecture for the Data Protection solution for Kubernetes with FlashBlade.

Testing Environment Overview

PX-Backup v1.2.2 was deployed on top of Kubernetes 1.19.4 in a virtualized environment. PX-Backup requires some level of persistent storage and in this environment, we deployed on top of Portworx Enterprise running on Kubernetes to support the installation. This instance of PX-Backup serves as a centralized control plane and single pane of glass to manage backup and restore operations for all clusters in this environment.

We used three Kubernetes environments during testing to help show the agility of the solution to work across Kubernetes platforms and versions.

- Rancher Kubernetes 1.18.6
- Azure Kubernetes Service v1.18.14
- Kubespray deployed upstream v1.19.7.

The upstream Kubernetes cluster ran on top of a virtualized VMWare environment, and the Rancher cluster is a bare-metal cluster. The Rancher cluster is using the local disk, and the upstream cluster is using FlashArray attached LUNs.

PX-Backup backup locations configured for this environment are FlashBlade v3.1.0 and Azure Blob Storage. FlashBlade is used to backup Kubernetes applications on-prem. In contrast, Azure is used to showcase PX-Backup's capabilities to backup offsite directly to the cloud to enable restore in cloud Kubernetes services for Test and Dev. Figure 4 illustrates the Portworx and FlashBlade testing environment used in the development of this solution.

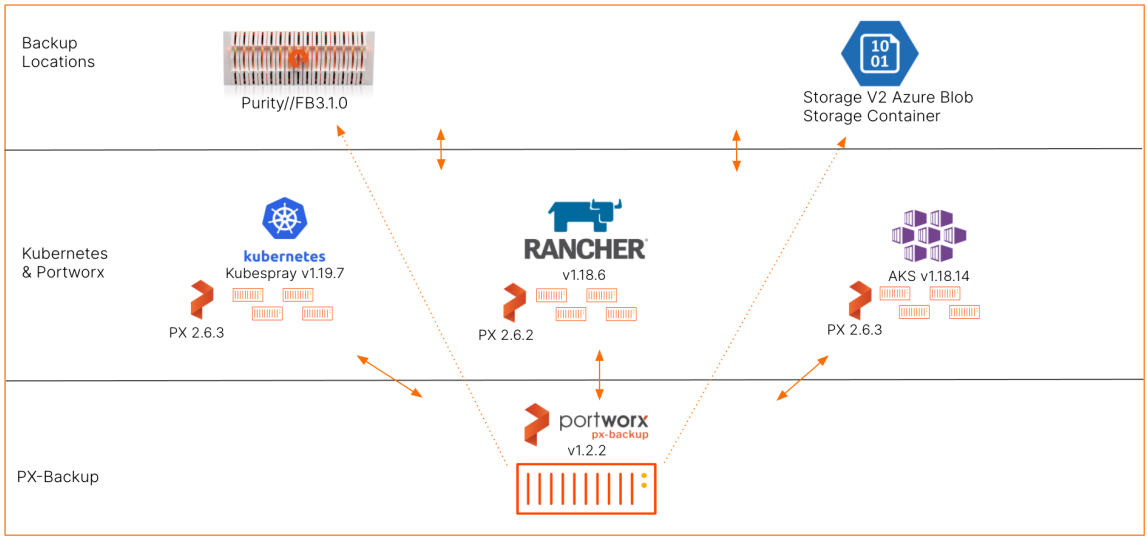

Figure 4. Portworx and FlashBlade testing environment.

System Configuration Details

Component	On-Prem Kubespray Kubernetes Cluster	On-Prem Rancher Kubernetes Cluster	Cloud AKS Kubernetes Cluster
OS	Centos 7 (Core)	Centos 7.6	OS: AKS Ubuntu 18.04
K8s	1.19.7	1.18.6	K8s: 1.18.14
CPU	8 vCPU	48 CPU	CPU: 2 vCPU
Mem	128GB	504GB	Mem: 7GB
Portworx	2.6.3	2.6.2	Portworx: 2.6.3
Portworx Backing Disk	FlashArray attached iSCSI	Local Disk	Local Disk
Node Size			Standard_DS2_v2

Table 1. Hyper-Converged Application Clusters.

FlashBlade Configuration

FlashBlade Backup Location

FlashBlade is used as a PX-Backup backup location for application backup and restores. To configure PX-Backup, you first need to create an object storage endpoint within your FlashBlade deployment. The steps below outline how to do this.

1. Navigate to **Storage** → **Object Storage**. First, create an account name and add a user for PX-Backup to use. Supply the account name, username and select to add the Access Key which will be used when you configure PX-Backup to use FlashBlade.

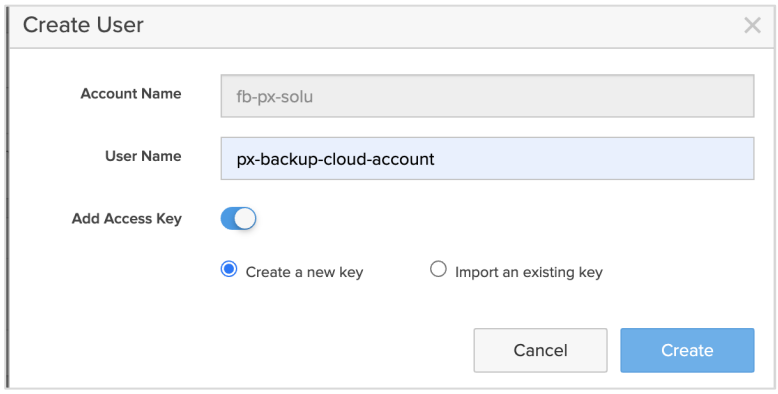

Figure 5. Creating an account name.

2. Once you have created the account and user, create a bucket within FlashBlade for Portworx to use. If you do not perform this step, PX-Backup will create one for you. However, it is a best practice to name your PX-Backup bucket something meaningful.

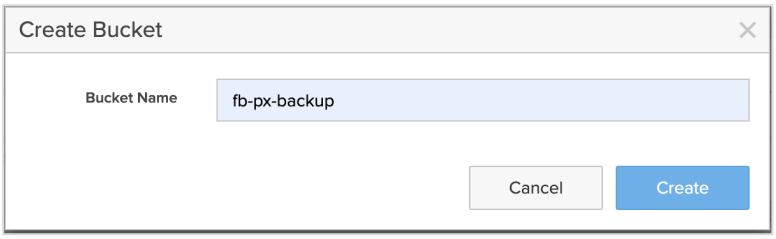

Figure 6. Creating a bucket within FlashBlade for Portworx.

3. After you create these resources in FlashBlade, you should have something that resembles the following information.

   Note: Your Account, Username, Bucket Name, Access Key ID will be different than the below.

   - Account: fb-px-solu
   - Username: fb-px-solu/px-backup-cloud-account
   - Bucket: fb-px-backup
   - Access Key ID: AAAAAAAAAAABBBBBBBBBBBBBBBCCCCCCCCDDDDDDD
   - Secret Access Key: ••

PX-Backup Configuration

With PX-Backup installed, navigate to **Settings** → **Cloud Settings** and configure the Cloud Credential with the information from FlashBlade.

1. Use the Access Key and Secret Access Key from the FlashBlade configuration and name the account within PX-Backup.

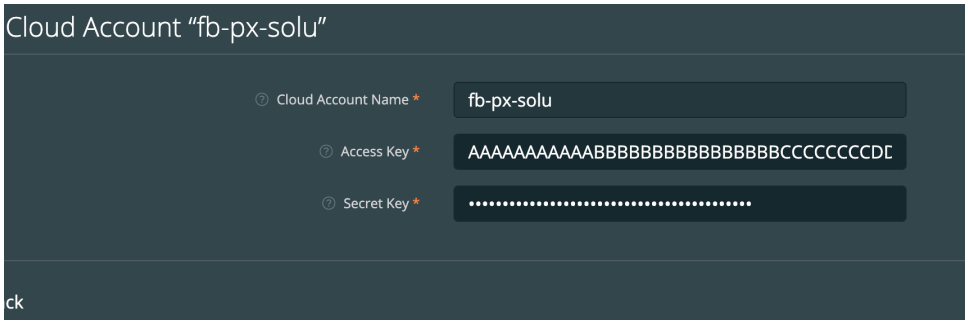

Figure 7. Editing the cloud account.

2. To configure the FlashBlade backup location, navigate to **Settings** → **Cloud Settings**. Use the FlashBlade data VIP endpoint information from the Network section of the FlashBlade settings to find the IP or qualified name to input as the endpoint in the PX-Backup configuration.

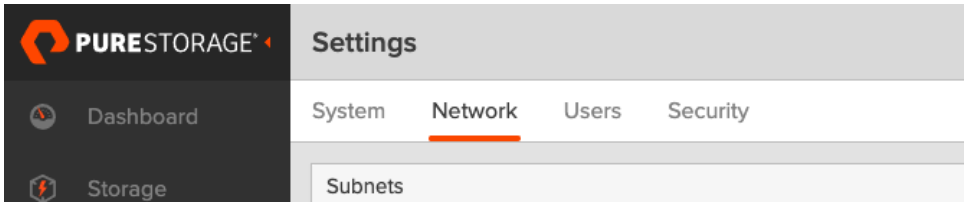

Figure 8. Network section in FlashBlade settings.

3. Give the backup location a name, select the FlashBlade cloud account, and entering the region and endpoint information.

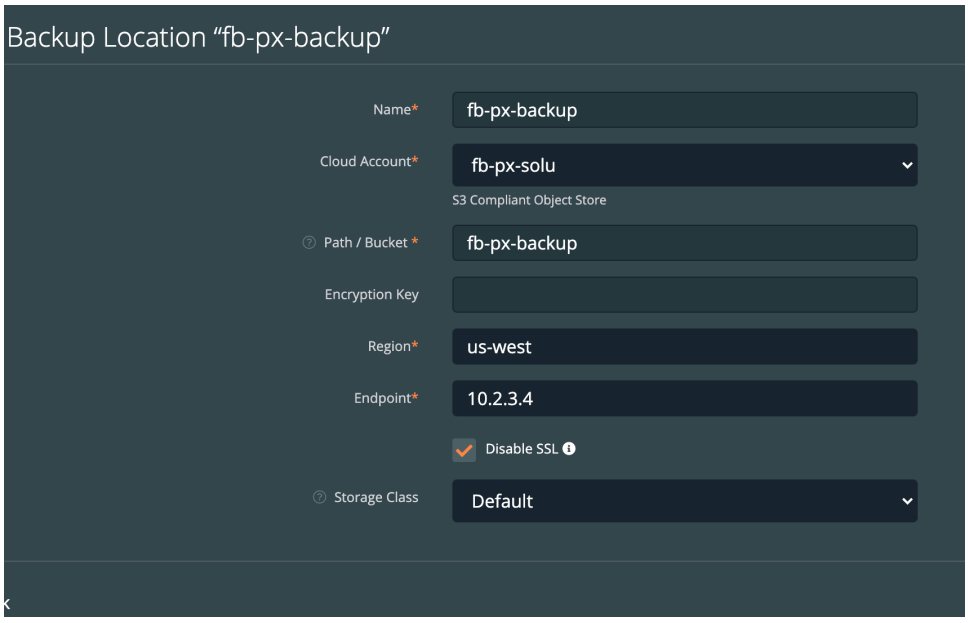

Figure 9. Backup location settings.

4. After you create the cloud account and backup location, you are all set to backup and restore Kubernetes resources from PX-Backup connected clusters.

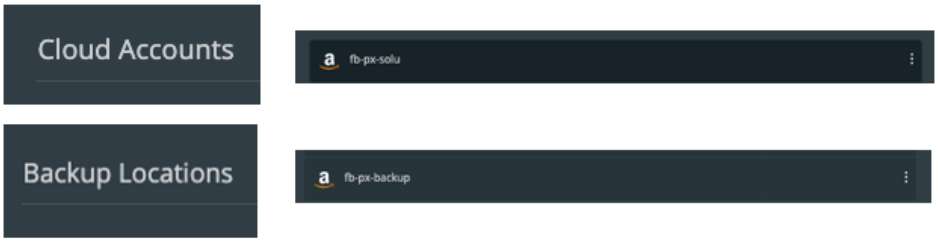

Figure 10. Cloud account and backup location.

Add Clusters to PX-Backup

To add your Kubernetes clusters to PX-Backup, first, navigate to the + **Add Cluster** button on the primary backup page.

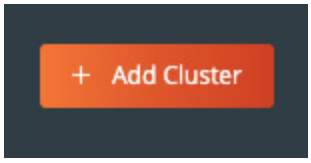

Figure 11. Adding a cluster.

Then, follow the prompts and inputs for the cluster name, Kubernetes Kubeconfig, and platform information (Figure 12).

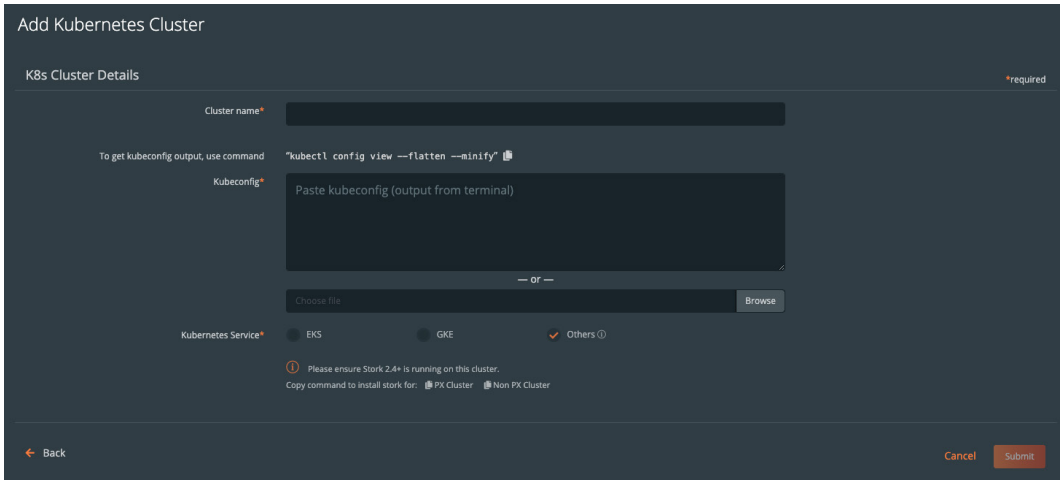

Figure 12. Adding a Kubernetes cluster.

Portworx Enterprise Configuration

To enable the complete data protection capabilities for PX-Backup with FlashBlade, Portworx Enterprise runs on each protected cluster. This enables all data and metadata to be portable across Kubernetes environments, whether in different data centers or clouds. We used the following Portworx versions:

- 2.6.3. Both the on-prem production and cloud Kubernetes clusters were running 2.6.3.
- 2.6.2. The on-prem, bare metal Rancher cluster was installed with 2.6.2.

You can find notable configurations for these clusters in the following sections. Note, however, that this document will not show you how to install each Portworx Enterprise cluster but rather focus on the main components. Please note that each cluster used the spec generator tool located at https://central.portworx.com to generate installation resources

On-Prem Production

The “production” cluster consists of Kubernetes worker nodes that have FlashArray LUNs attached to provide backing storage. Figure 13 shows the high-level architecture of this installation.

FlashArray is the world’s first 100% all-flash end-to-end NVMe and NVMe-oF array, ideal for the most demanding enterprise performance requirements. FlashArray provides customers with a modern data experience, delivering breakthroughs in speed, simplicity, flexibility, and consolidation. It’s ideal for departmental to large-scale enterprise deployments, high performance, and mission-critical applications.

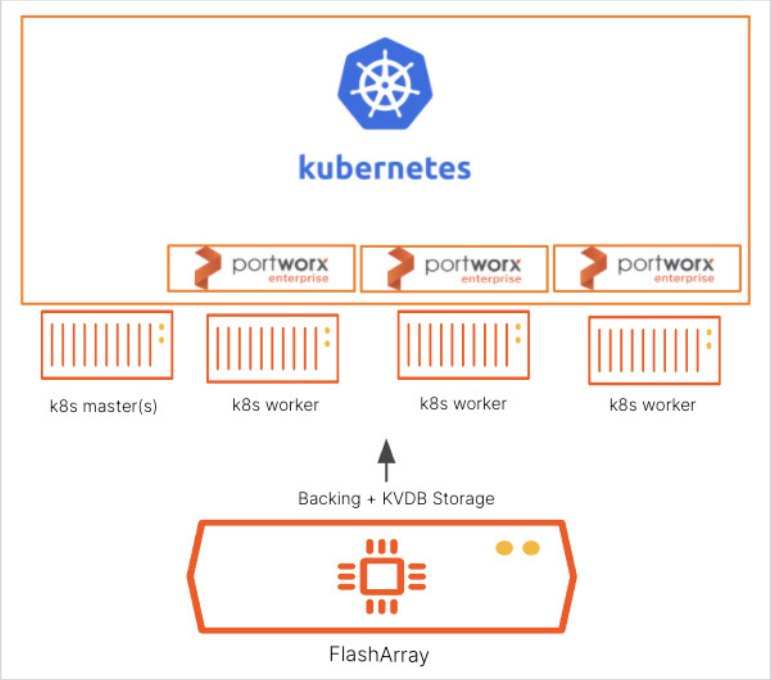

Figure 13. High-level architecture of Kubernetes worker nodes with attached FlashArray LUNs.

FlashArray was used for both Portworx Backing Storage (aggregated SDS layer) and the internal KVDB ETCD metadata layer. Figure 14 shows six volumes: Three are 128GB and used for backing storage;the others are 100GB and used for Portworx internal KVDB.

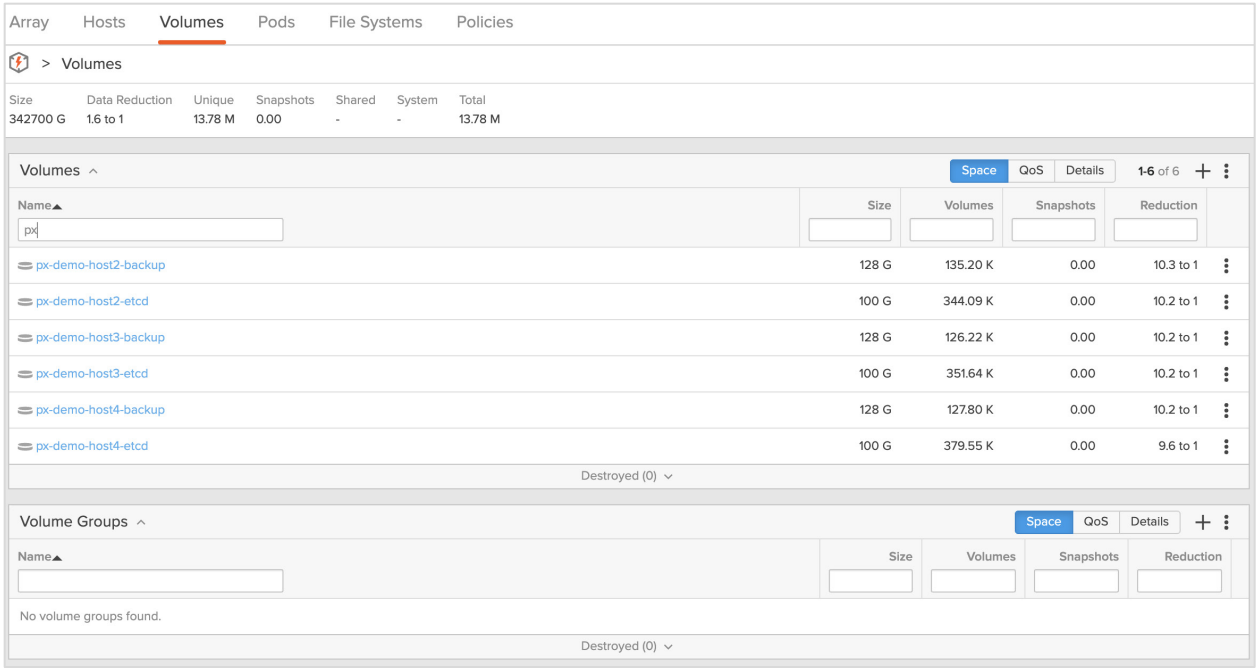

Figure 14. Six px volumes.

On Prem Rancher

The on-premises Rancher Kubernetes Engine cluster is using backing storage that is local to each bare metal server. The cluster is also set up to use an external ETCD endpoint.

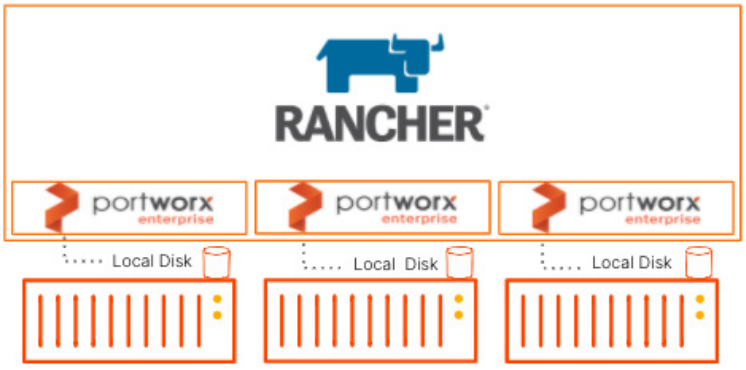

Figure 15. Rancher and Portworx architecture.

On Prem Rancher and Portworx

For the cloud-based Portworx cluster, an AKS cluster was created using the standard AKS CLI with three nodes using the following command:

```
az aks create --resource-group FBSoluResourceGroup --name FBSoluAKSCluster --node-count 3 --enable-addons monitoring --generate-ssh-keys
```

Then, within the Portworx Spec Generator, Portworx is configured to use 150GB Premium disks that Portworx provisions automatically.

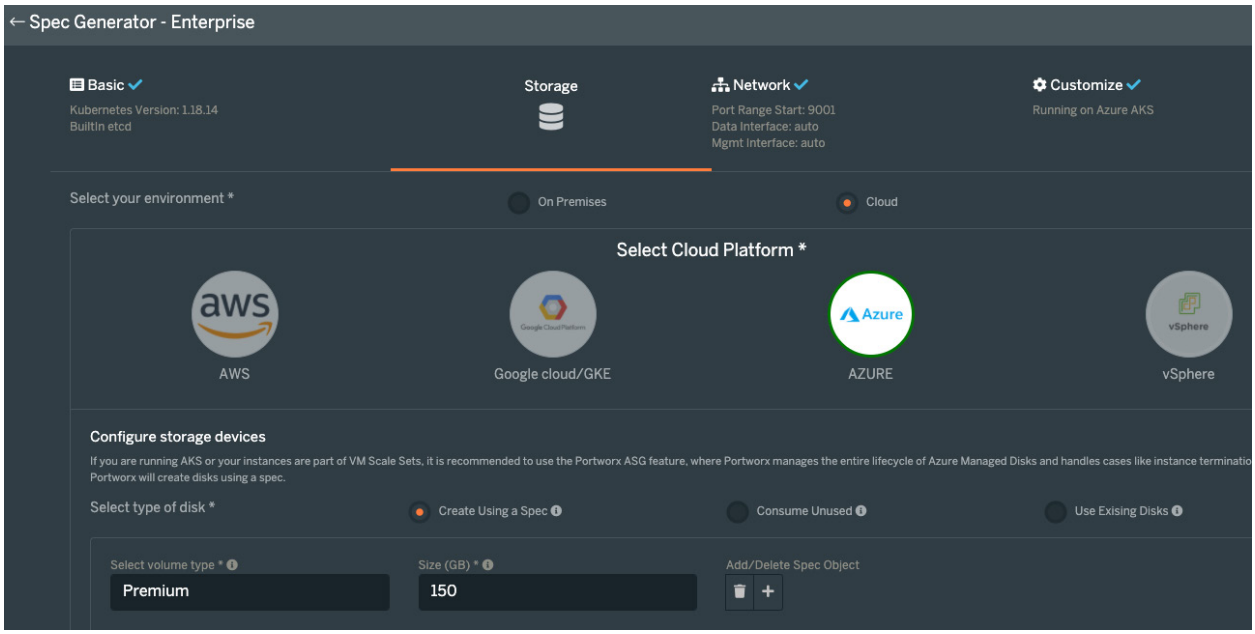

Figure 16. Configuring Portworx.

PX-DR Configuration

Setting Up and Configuring the Disaster Recovery ClusterPair

1. First, retrieve the destination UUID. This UUID will be used in the credential's creation. It will be used within the string "clusterPair_<UUID>`.

```
kubectl exec -it $PX_POD -n kube-system -c portworx -- /opt/pwx/bin/pxctl status | grep UUID | awk '{print $4}'
d25c8086-e93f-4d62-bcd4-dcd973e743e0
```

2. Create a bucket for DR within FlashBlade. Note that this step is optional as PX-DR will create one for you, however it is a best practice to name your PX-DR bucket something meaningful.

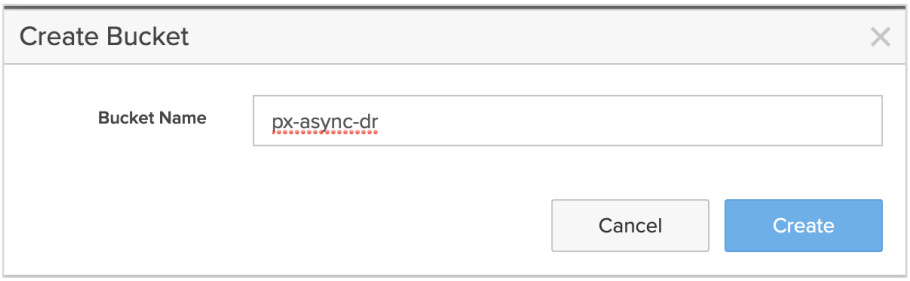

Figure 17. Creating a bucket name.

3. Next, gather the FlashBlade information and credentials to setup the PX-DR ClusterPair.

- FlashBlade Data Endpoint: 10.1.2.3

- Bucket: px-async-dr
- Access Key ID: AAAAAAAAAAABBBBBBBBBBBBBBBCCCCCCCCDDDDDDD
- Secret Access Key: ••

4. Finally, put this information together and create the credential for PX-DR to use on both the source and destination (DR) cluster. Note: the S3 endpoint will be your FlashBlade data VIP.

```
/opt/pwx/bin/pxctl credentials create \
--provider s3 \
--s3-disable-ssl \
--bucket px-async-dr \
--s3-access-key AAAAAAAAAAABBBBBBBBBBBBBBBCCCCCCCCDDDDDDD \
--s3-secret-key •••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••• \
--s3-region us-west \
--s3-endpoint 10.1.2.3 \
clusterPair_d25c8086-e93f-4d62-bcd4-dcd973e743e0
Credentials created successfully, UUID:0ea2d19e-d236-46ce-bbb7-136d93fc51bf
```

5. Next, generate the ClusterPair from the DR-Site Portworx cluster. This will output a YAML file that will be modified and applied at the source.

```
> storkctl generate clusterpair -n fbsolu-ns dest-cluster > clusterpair.yaml
```

You will also want to get the cluster token from the DR-Site. This will be used within the ClusterPair configuration file.

```
>  kubectl exec -it $PX_POD -n kube-system -c portworx -- /opt/pwx/bin/pxctl cluster token show
```

The token is:

```
9a21db35a3c0d8c2d41ea63d2eeda12732249341826423a1c97478ff277bfa34ada421c51665379a54d7c09d92
2d92851cbf05fa611c6f49661ac79f6f6d6edc
```

6. Next, you will want to copy the ClusterPair YAML file to a location on your source cluster that has admin kubectl access. Then edit the ClusterPair information with the following parameters. Figure 7 shows what the edited options look like.

options:

```
    ip: " 10.2.3.4"
    port: "9001"
    token:
"9a21db35a3c0d8c2d41ea63d2eeda12732249341826423a1c97478ff277bfa34ada421c51665379a54d7c09d922d92851cbf05fa6
11c6f49661ac79f6f6d6edc"
    mode: DisasterRecovery
```

7. Apply the ClusterPair at the source Kubernetes cluster:

```
[root@node1 ~]# kubectl create -f clusterpair.yaml clusterpair.stork.libopenstorage.org/dest-cluster
created
```

8. Get the status of the ClusterPair by using storkctl, the storage orchestrator for Kubernetes CLI provided by Portworx:

```
[root@node1 ~]# storkctl get clusterpair -n fbsolu-ns
NAME           STORAGE-STATUS   SCHEDULER-STATUS   CREATED
dest-cluster   Ready            Ready              09 Feb 21 16:50 EST
```

The above steps effectively pair the source and DR site together using FlashBlade. Both statuses should be in “Ready” state for DR to be used.

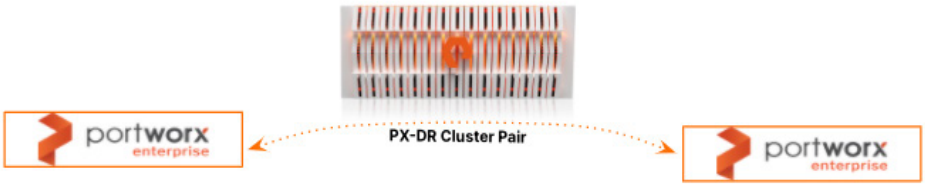

Figure 18. Disaster recovery site and FlashBlade.

Use Cases and Results

We identified three use cases to showcase the breadth of functionality and flexibility that PX-Backup with FlashBlade for Kubernetes can offer. They target single data center level protection, multi-datacenter disaster recovery and cloud-based offsite backups ready for test and dev. Details of the individual use cases and tests run within them are in the following sections.

On-Prem PX-Backup with FlashArray and FlashBlade

The first use case focuses on data protection for Kubernetes on-premises. FlashBlade acts as the backup location for PX-Backup and Kubernetes applications. Portworx Enterprise provides persistent storage and data management for applications. Applications along with these volumes are backed up to FlashBlade and can be restored to any other Kubernetes cluster with Portworx on prem.

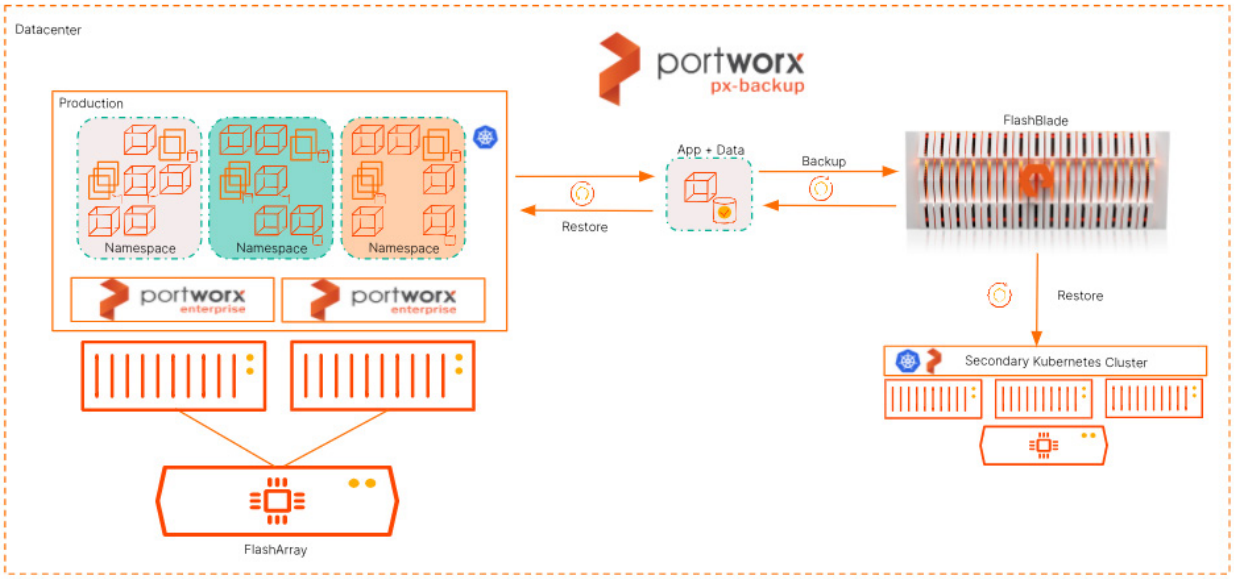

Figure 19. Data protection for Kubernetes on-premises.

Within this use case, the following tests were performed to show the operations and functionality between PX-Backup and FlashBlade when using them for data protection of Kubernetes applications on premises.

Test	Test Procedure	Test Results
Full Backup to FlashBlade	1. Create MongoDB in Mongo namespace. 2. Insert 26k records. 3. Trigger backup to FlashBlade backup location.	Success
Full Backups with App-Aware Rules	1. Create MongoDB in Mongo namespace. 2. Insert 26k records. 3. Create pre-backup rule and post backup rule to use fsync.Lock() / fsync.Unlock(). 4. Trigger backup to FB backup location.	Success
Incremental Backups Using a Schedule Policy	1. Create MongoDB in Mongo namespace. 2. Insert 26k records. 3. Create a schedule policy for daily and hourly incremental backups. 4. Trigger scheduled backup to FB backup location. 5. Confirm success over several days.	Success
Resource-level Backup	1. Create MongoDB in Mongo Namespace. 2. Insert 26k Records. 3. Trigger backup of PV and PVC resources only on FB backup location.	Success
Restore to Same Namespace	1. Create restore from existing MongoDB backup. 2. Select the same cluster and same namespace. 3. Select replace resources.	Success
Restore to Different Namespace	1. Create restore from existing MongoDB backup. 2. Select the same cluster and new namespace.	Success
Restore to Alternate On-Premises Cluster	1. Create restore from existing MongoDB backup. 2. Select the new cluster and new namespace.	Success

Table 2. Tests, procedures, and results.

Policy-based Backups to FlashBlade and Azure Cloud

The second use case uses PX-Backup to create backups based on a policy that stages backups both on-premises and in the cloud. This enables organizations to quickly restore on prem but also utilize cloud compute resources to quickly test and develop against real data.

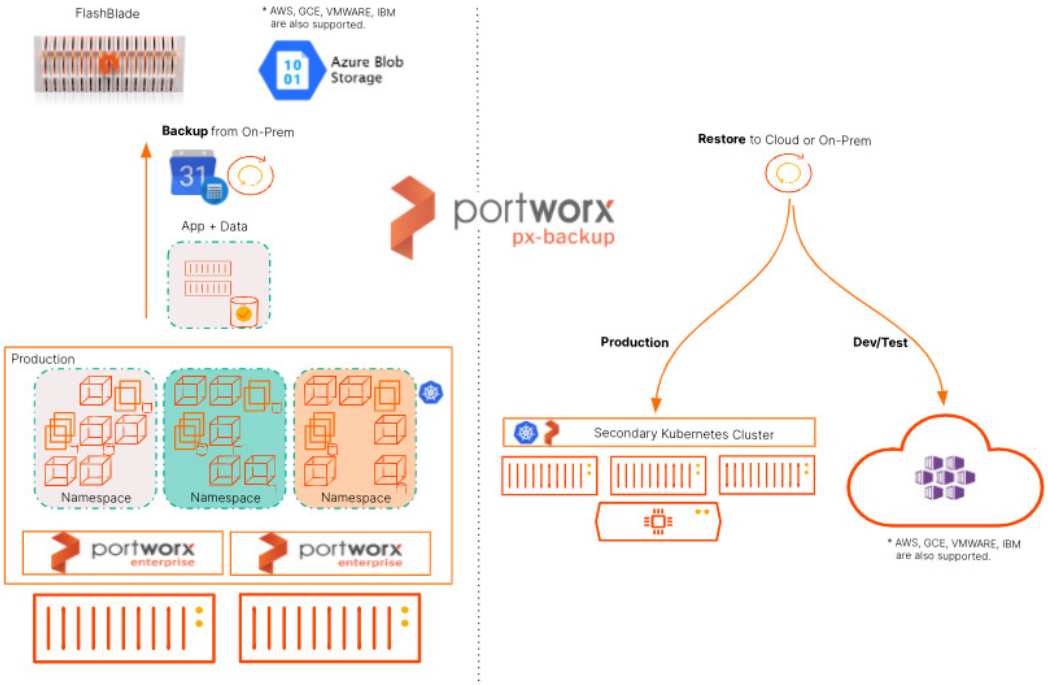

Figure 20. Using PX-Backup to create on-prem and clous backups.

Test	Test Procedure	Test Results
Full Backup from on prem to Cloud	1. Configure Azure Blob storage endpoint in PX-backup on-prem. 2. Take full backup of MongoDB to Azure backup location.	Success
Restore to On-Prem app to Cloud-Based Kubernetes Cluster	1. Create an AKS cluster and install Portworx. 2. Connect AKS cluster to PX-Backup. 3. Restore to AKS from full backup of MongoDB in Azure backup location.	Success
Backup with FB and Cloud based Policies (Incremental Backup)	1. Create schedule policy for hourly backups. 2. Create backup with hourly schedule to azure backup location. 3. Create backup with hourly schedule to FlashBlade. 4. Confirm backups are being sent to both on-prem FlashBlade and off-site Azure backup locations. 5. Restore from FlashBlade to on-prem, restore from Azure to AKS, confirm both restored applications and data match.	Success

Table 3. Tests, procedures, and results.

PX-DR (Asynchronous DR) On-prem to On-Prem

The third use case looks at disaster recovery of production applications in Kubernetes. A second datacenter acts as a warm disaster recovery site that Portworx continuously replicates applications and data snapshots to and stages them within the DR Site. This gives you the ability to activate the DR site in case of failure of the production environment with very low RPO and RTO values.

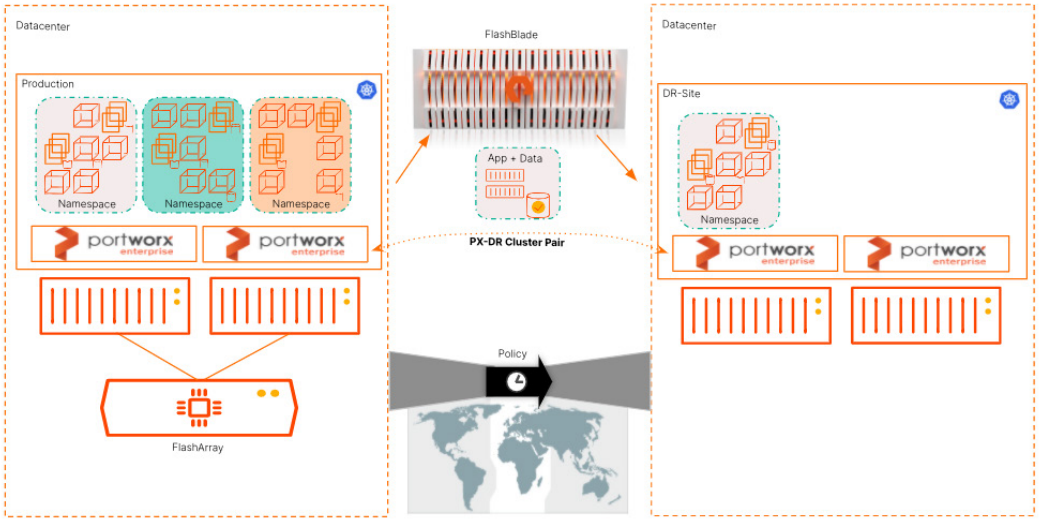

Figure 21. PX-DR (Asynchronous DR) On-prem to On-Prem.

Within this use case, the following tests were performed to show the operations and functionality of providing disaster recovery for Kubernetes applications across data centers. FlashBlade is used as an intermediary location where snapshots and metadata are sent and then staged from in the DR site.

> Tech Note: The use case here is specifically Asynchronous Disaster Recovery. This means it uses snapshot replication for volumes between clusters and can achieve low RPO numbers within minutes. However, PX-DR is also capable of Synchronous DR when the network between sites is within 5ms RTT, and capable of RPO values with Zero data loss. Synchronous DR was not covered in this white paper but is absolutely achievable with Portworx and FlashBlade.

Test	Test Procedure	Test Results
Create schedule and migration objects	**1. MongoDB is deployed as the source production cluster.** **2. Create a schedule policy for RPO objectives.** **3. Create migration schedule that used schedule policy.** **4. Start migration schedule to start replication to DR-Site using FlashBlade.**	**Success**
Failover Application by turning on migration in the DR site.	**1. Simulate failure by turning off application and DR replication at the source.** **2. Activate application in DR site and confirm application is healthy and data is available.** **3. Observe RTOs. (RTOs were in the range of tens of seconds.)**	**Success**

Table 4. Tests, procedures, and results.

Best Practices

When planning data protection for application with PX-Backup, make sure the resources you need backed up can be part of the provided backup. PX-Backup can back up the following Kubernetes resources:

PV	**Service**	**RoleBinding**
PVC	**Secret**	**ClusterRole**
Deployment	**DaemonSet**	**ClusterRoleBinding**
StatefulSet	**ServiceAccount**	**Ingress**
ConfigMap	**Role**	

Using Labels

PX-Backup provides namespace and label selectors, allowing you to create granular backups of the application you want. You can broadly back up an entire namespace, or you can use label selectors to select only certain resources to backup. This selection method also helps preserve associated configuration and pod data, ensuring that your backups will work properly once restored. For example, PX-backup can back up a MySQL deployment containing pods, PVCs, and volumes tagged with an app = mysql label. Given this system, PX-Backup can back up stateful apps as easily as stateless ones.

Using Schedule Policies

You can schedule backups by creating an independent schedule policy that defines when backups run and how many rolling copies they keep, and you can associate this schedule policy with as many backups as you want.

Utilizing Backup Rules

Avoid manual prep-work and minimize interruptions to your cluster associated with backup tasks by creating rules that run before and after backups are taken. As with schedule policies, you can associate rules with multiple backups.

PX-Backup Installation Configuration

When planning your PX-Backup deployment, it's important to take note of the `persistentStorage.storageClassName` and `persistentStorage.enabled` parameters. They will provide PX-Backup itself with data storage so no backup references will be lost in the event the central PX-Backup server has issues.

Conclusion

Traditional backup and recovery solutions don't fit with modern Kubernetes applications. Kubernetes Data Protection for cloud-native environments with PX-Backup and FlashBlade gives you the ability to effortlessly backup and restore Kubernetes applications across any environment with the multi-dimensional performance and configurability of Pure FlashBlade.

Additional Resources

- Read more about PX-Backup Product Documentation.
- Learn more about how to keep your Containerized Applications with Kubernetes Backup safe.
- Try Portworx with a free trial.

Appendix: PX-Backup Components

To use PX-Backup, it's helpful to understand the components that make it up. You'll use these components to perform backup and restore operations.

Application view: You can interact with PX-Backup through a central application view. From here, you can see all the resources currently on your cluster, filter them by namespace and labels, and create a backup.

> Note: Users will only be able to access applications that can be controlled by the KUBECONFIG that was used to add the cluster to PX-Backup.

Backup locations: Backup locations are object stores you've added to PX-Backup. PX-Backup stores backups on any compatible object storage such as:

- AWS S3 or compatible object stores
- Azure Blob Storage
- Google Cloud Storage

Backups: Backups in PX-Backup contain backup images and configuration data. You can attach schedule policies to run them at designated times and keep a designated number of rolling backups. You can also attach rules to perform commands before or after a backup run.

Clusters: A cluster in PX-Backup is any Kubernetes cluster that PX-Backup makes backups from or restores backups to. PX-Backup supports pretty much any Kubernetes cluster that's network accessible. With PX-Backup, you can monitor, back up, and restore across all your Kubernetes clusters.

Restores: Restore your backups to the original cluster or different clusters, replace applications on the original cluster or restore to a new namespace. Perform partial restores to selected namespaces from the backup.

Rules: Use rules to create commands which run before or after a backup operation is performed.

Schedule policies: Create schedule policies, attach them to backups to run them at designated times, and keep a designated number of rolling backups.

Glossary

Application-Aware: The ability for Portworx to tap into the application runtime. Often to provide freezing, locking and quiescing.

ConfigMap: See https://kubernetes.io/docs/concepts/configuration/configmap/.

Deployment: See https://kubernetes.io/docs/concepts/workloads/controllers/deployment/.

HA: High Availability, when used in the context of Portworx, this describes the ability that Portworx provides for containers to maintain data availability in case of failure.

Hyperconverged: The co-location of Portworx Storage Nodes and Kubernetes Worker nodes. Applications will run on the same hosts as Portworx is running.

KUBECONFIG: A file used to configure access to clusters. This is a generic way of referring to configuration files. It does not mean that there is a file named kubeconfig.

PersistentVolume: See https://kubernetes.io/docs/concepts/storage/persistent-volumes/.

PersistentVolumeClaim: See https://kubernetes.io/docs/concepts/storage/volumes/#persistentvolumeclaim.

Stateful: An application that requires non-volatile storage for data, such as a database.

StatefulSet: See https://kubernetes.io/docs/concepts/workloads/controllers/statefulset/.

Stateless: An application that does not require non-volatile storage, such as a web server.

About the Author

Ryan Wallner is a technical marketing manager at Pure Storage. He is responsible for defining solutions around Portworx Enterprise, Backup, and Disaster Recovery for Kubernetes Applications. Ryan has worked in the data management field for 10 years as both a field practitioner in healthcare and as a vendor developing products for emerging technologies. Ryan joined Pure Storage in October 2020 with Pure's acquisition of Portworx Inc.

©2021 Pure Storage, the Pure P Logo, and the marks on the Pure Trademark List at https://www.purestorage.com/legal/productenduserinfo.html are trademarks of Pure Storage, Inc. Other names are trademarks of their respective owners. Use of Pure Storage Products and Programs are covered by End User Agreements, IP, and other terms, available at: https://www.purestorage.com/legal/productenduserinfo.html and https://www.purestorage.com/patents

The Pure Storage products and programs described in this documentation are distributed under a license agreement restricting the use, copying, distribution, and decompilation/reverse engineering of the products. No part of this documentation may be reproduced in any form by any means without prior written authorization from Pure Storage, Inc. and its licensors, if any. Pure Storage may make improvements and/or changes in the Pure Storage products and/or the programs described in this documentation at any time without notice.

THIS DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID. PURE STORAGE SHALL NOT BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS DOCUMENTATION. THE INFORMATION CONTAINED IN THIS DOCUMENTATION IS SUBJECT TO CHANGE WITHOUT NOTICE.

Pure Storage, Inc.
650 Castro Street, #400
Mountain View, CA 94041

PS2036-01 03/2021