Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2021, 8(11):123-127**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Persistent Storage for Containers: Deployed and managed persistent storage solutions like Portworx for containerized applications

Gowtham Mulpuri

Silicon Labs, TX, USA
Email: gowtham.mulpuri@silabs.com

Abstract This paper explores the critical role of persistent storage in containerized applications, with a focus on deploying and managing persistent storage solutions like Portworx. Drawing from over a decade of experience as a senior DevOps engineer, it delves into the practical aspects of ensuring data durability, availability, and performance in containerized environments. The paper aims to provide insights into the experiences gained from working with Portworx and other similar solutions, highlighting the challenges and advantages of implementing persistent storage in containers. The advent of containerization technologies has significantly streamlined the deployment and scaling of applications across various computing environments. Despite the benefits, managing stateful applications that require persistent data storage presents unique challenges in these ephemeral environments. This paper delves into the concept of persistent storage within containerized infrastructures, focusing on solutions like Portworx that facilitate the deployment and management of persistent storage across containerized applications. We explore the architecture, capabilities, and real-world application scenarios of such solutions, aiming to underscore their importance in modern cloud-native ecosystems.

Keywords Persistent Storage, Portworx, Containerized Applications, Data Durability, Availability, Performance, DevOps, Deployment, Management **Introduction**

1. Introduction

In the era of cloud-native and containerized applications, persistent storage has become a critical component of application architecture. Containers, by their nature, are ephemeral, meaning that data stored within them is lost when the container is destroyed. This poses significant challenges for applications that require persistent data, such as databases, file systems, and caches. Persistent storage solutions like Portworx are designed to address these challenges by providing durable, scalable, and high-performance storage for containerized applications. This paper aims to share insights and experiences gained from working with Portworx and similar solutions, focusing on the deployment and management of persistent storage in containers.

Persistent Storage for Containers: Deployed and Managed Solutions Portworx is a cloud-native storage solution designed for containerized applications. It provides a unified, multi-cloud data management platform that offers features such as data replication, snapshots, and clones.

- **Deployment:** Portworx can be deployed on-premises or in the cloud, making it highly flexible. It supports various container orchestration platforms, including Kubernetes, Docker Swarm, and Mesos.
- **Management:** Portworx provides a user-friendly interface for managing storage, including creating volumes, setting policies, and monitoring performance. It also offers automated data management features, such as data replication and snapshots.

- **Scalability:** Portworx is designed to scale with the application, ensuring that storage capacity can be easily increased or decreased as needed.
- **Performance:** By leveraging the underlying hardware and network infrastructure, Portworx provides high-performance storage for containerized applications.

Real-Time Use Cases and Advantages Real-Time Use Cases

- **Database Applications:** Portworx provides durable storage for databases, ensuring that data is not lost in the event of a container failure
- **File Systems:** For applications that require persistent file systems, Portworx offers scalable and high-performance storage solutions.
- **Caching Systems:** Portworx can be used to implement caching systems in containerized environments, improving application performance.
- **Data Durability:** Ensures that data is not lost when containers are destroyed, providing a reliable storage solution for containerized applications.
- **Scalability:** Allows for easy scaling of storage capacity, accommodating the growth of containerized applications.
- **Performance:** Offers high-performance storage, ensuring that applications can access data quickly and efficiently.
- **Multi-Cloud Compatibility:** Provides a unified storage solution that can be deployed across multiple cloud environments, offering flexibility and ease of management.
- **Portworx Deployment Architecture:** A diagram showing how Portworx can be deployed in various environments, including on-premises and in the cloud.

Figure 1: GCP Cloud

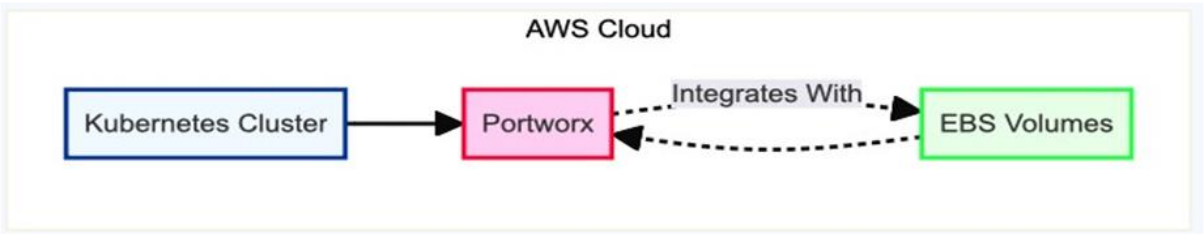

Figure 2: AWS Cloud

Figure 3: On-Premises Environment

The ***Figure 1*** GCP Cloud shows Portworx integration with GCP Persistent Disks in the context of a Kubernetes cluster running in Google Cloud.

The ***Figure 2*** AWS Cloud demonstrates the integration of Portworx with AWS EBS (Elastic Block Store) volumes for dynamic provisioning of storage within an AWS Kubernetes cluster.

The ***Figure 3*** shows On-Premises Environment where Portworx integrates with existing storage hardware or server attached storage, managed by a Kubernetes cluster.

Please note, the diagram depicts a simplified overview and does not detail every component, such as networking or specific Kubernetes configurations, which are essential for a real-world deployment.

Portworx Storage Management: A diagram illustrating the management interface of Portworx, highlighting features such as volume creation and policy setting.

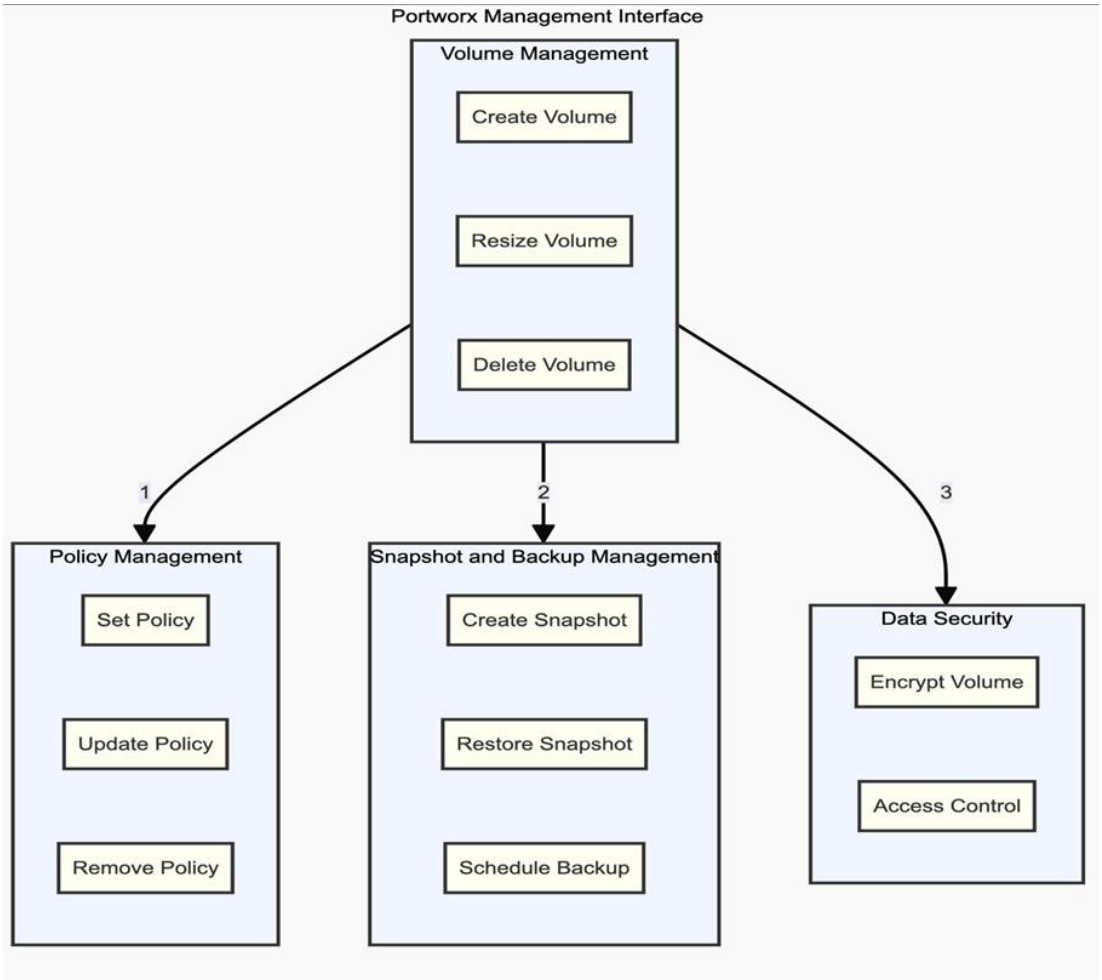

Figure 4: Portworx Management Interface

This ***Figure 4*** provides an overview of the Portworx management interface, categorized into four main sections:

Volume Management: Operations related to Portworx volumes, such as creating, resizing, and deleting volumes.

Policy Management: Setting, updating, or removing policies that govern the behavior of storage volumes, such as replication factors and encryption.

Snapshot and Backup Management: Managing data safety and recovery through snapshots and backups.

Data Security: Enhancing data protection with features like volume encryption and access control.

Each section represents a key aspect of the Portworx Storage Management capabilities, showcasing how it provides a comprehensive suite of tools for the lifecycle management of storage resources in containerized environments. The links attached to various operations guide users to documentation for more detailed information on each action.

Keep in mind that while this diagram offers a structured overview, Portworx's real-world usage might involve more complex interactions and integrations with container orchestration platforms like Kubernetes.

Deploying Persistent Storage with Portworx

Deploying Portworx in a containerized environment involves several key steps:

- **Cluster Preparation:** Before installing Portworx, ensure that your Kubernetes cluster meets the necessary prerequisites, including network configurations and resource allocations. It's crucial to have a compatible underlying infrastructure, whether on-premises or in the cloud.
- **Installation:** Portworx can be installed on a Kubernetes cluster using the Portworx Operator or Helm charts. The Operator provides a Kubernetesnative way to deploy and manage Portworx, automating many tasks related to its lifecycle management.
- **Storage Class Creation:** Once Portworx is installed, define storage classes in Kubernetes. Storage classes specify the type of storage to be used and allow for dynamic provisioning of persistent volumes as needed by applications.
- **Persistent Volume Claims (PVCs):** Applications can then request storage through Persistent Volume Claims. PVCs specify the size of the storage required and reference the storage class to determine how the volume should be provisioned.

Managing Persistent Storage with Portworx

Once deployed, Portworx offers a range of features to manage and scale persistent storage effectively:

- **High Availability:** Portworx ensures high availability of data by replicating volumes across multiple nodes in the cluster. In the event of a node failure, Portworx can automatically failover to a replica, minimizing downtime.
- **Data Security:** Portworx provides built-in encryption for data at rest and in transit, ensuring that sensitive data is securely stored and communicated across the cluster.
- **Backup and Recovery:** Portworx facilitates disaster recovery through its backup and restore capabilities. It allows for consistent snapshots of volumes, which can be used to restore data in case of accidental deletions or corruptions.
- **Multi-Cloud and Hybrid Deployments:** Portworx supports multi-cloud and hybrid cloud deployments, enabling data mobility across different cloud environments. This is particularly valuable for organizations adopting a multi-cloud strategy to avoid vendor lock-in or to meet specific regulatory requirements.

Use Cases for Persistent Storage in Containers

Persistent storage solutions like Portworx are essential for a wide range of applications, including:

- **Databases and Stateful Services:** Applications such as MySQL, PostgreSQL, and MongoDB require persistent storage to ensure data is not lost between container restarts.
- **CI/CD Pipelines:** Storing build artifacts, test results, and logs for continuous integration and deployment workflows.
- **Machine Learning and Big Data:** Storing datasets, models, and results for machine learning applications and big data processing.

Conclusion

As containerized applications continue to grow in complexity and scale, the need for robust persistent storage solutions becomes increasingly critical. Portworx provides a comprehensive, Kubernetes-native storage solution that addresses the challenges of data persistence, security, and mobility in containerized environments. By leveraging Portworx, organizations can ensure that their containerized applications are not only agile and scalable but also resilient and secure, with persistent data that supports the core business functions. My experience deploying and managing Portworx in diverse environments has reinforced its value in enabling seamless, efficient, and reliable storage solutions for the modern cloud-native landscape.

References

[1]. Public Cloud Kubernetes Storage Performance Analysis - Lubos Mercl, Jakub Pavlík (2019).

[2]. INCIDENT ANALYSIS AND FORENSICS IN DOCKER ENVIRONMENTS ERNW WHITE PAPER 64/ (02, 2018) - Ernw Enno, Rey Netzwerke GmbH (2018)
[3]. Studies on Different Substrates and Container Volumes in the Emergence and Ministumps Formation of Physalis peruviana L. - Jeniffer Ribeiro de Oliveira et al. (2021)
[4]. Forecasting the Daily Container Volumes Using Data Mining with CART Approach - Jun-su Ha et al. (2021)
[5]. The Daily Container Volumes Prediction of Storage Yard in Port with Long Short-Term Memory Recurrent Neural Network - Yinping Gao et al. (2019)
[6]. SUBSTRATES AND CONTAINER VOLUMES IN THE PRODUCTION OF MANGABEIRA SEEDLINGS (Hancornia speciosa Gomes) - Aline Borges Vilela Silva et al. (2020)
[7]. Different Substrates and Container Volumes in the Emergence and Ministumps Formation of Physalis peruviana L. - Jeniffer Ribeiro de Oliveira et al. (2020)
[8]. Alterations in the volume of thalamic nuclei in patients with schizophrenia and persistent auditory hallucinations - M. Pérez-Rando et al. (2022)
[9]. Shibboleth IdP for Single Sign-On with Kubernetes and Persistent Volume Longhorn - Ikhwan Alfath Nurul Fathony, M. A. Setiawan (2022)