International Journal of Differential Equations and Applications
Volume **22**, No. 1 (2023), pages: 165-174
ISSN (Print): 1311-2872; ISSN (Online): 1314-6084;
url: https://www.ijdea.eu

SYSTEM ARCHITECTURE FOR AUTOMATED BACKUP AND RECOVERY OF DISK VOLUMES IN CONTAINERS

Rosen Hristev[1], Magdalena Veselinova[2], Kristiyan Kolev[3]

[1,2,3]Department of Mathematics and Informatics
University of Plovdiv Paisii Hilendarskiy
236, Bulgaria Blvd., 4000 Plovdiv, BULGARIA

ABSTRACT: In the field of IT, virtualization and containerization can be classified as solutions for deploying applications and effectively segmenting hardware resources. The two technologies operate in fundamentally different ways. In traditional monolithic application environments or even in virtual machines (VMs), the task of data backup and recovery is relatively straightforward. The methods used for full, incremental, or differential backup focus on preserving the state of the entire system or virtual machine at a given moment. Containerization is a technology that transforms the way software is developed, deployed, and managed. The fundamental idea behind containerization is encapsulating an application and its dependencies into a self-contained unit called a container. This encapsulation allows the application to run consistently and sequentially across different computing environments. Traditional backup methods are not aligned with containerized applications. They operate at the system level and lack the ability to understand the context or importance of the data they backup. However, in a containerized environment where multiple microservices interact and share data, understanding the context of the application becomes crucial for their effective recovery. The research explores the available methods for backup and recovering in containerization and proposes a system architecture to automate the backup process in containerized environments.

Key Words: Automated Backup, Containers, Docker, Data Recovery, System Architecture, Conteinerization Backup System

Received: October 9, 2023 **Revised:** December 10, 2023
Published: December 12, 2023 **doi:** 10.12732/ijdea.v22i1.12
Academic Publications, Ltd. https://acadpubl.eu

1. INTRODUCTION

Containerization is a technology that transforms the way software is developed, deployed, and managed. The fundamental idea behind containerization is encapsulating an application and its dependencies into a self-contained unit called a container. This encapsulation allows the application to run consistently and sequentially across different computing environments [1]. Docker is at the forefront of this technology, providing an open-source platform for automating the deployment, scaling, and management of applications in containers [2].

In a containerized system, the processes of data backup and recovery undergo significant transformation. As applications are divided into microservices and packaged into containers, traditional backup strategies that focus on backup the entire physical or virtual machine are not applicable [3]. This shift necessitates a closer examination of existing approaches and methods used for backup and recovery in containerized systems.

One fundamental aspect of containerization that impacts backup and recovery is the lifecycle of containers. By design, containers are stateless, meaning they can be created, destroyed, or replaced without affecting the state or data of the application. All relevant data is stored outside the container in data volumes. This feature allows seamless scaling and redeployment of containers but also complicates the backup process. To address this, container platforms like Docker provide built-in mechanisms for data integrity. Docker volumes, for instance, can be used to store data generated by and used by Docker containers. This data is separated from the container's lifecycle, ensuring data persistence beyond the lifespan of a container [4]. Backup these volumes is a common approach to ensuring data recovery in case of container failure or other disasters.

The traditional method of backup volumes involves launching a dedicated container with access to the volume that needs backup [5]. This archival container can then read data from the volume and write it to an external storage location. Tools like docker *cp* or more advanced ones like *rsync* can be used for this purpose. While this method provides an initial level of data protection, it requires manual intervention and doesn't cover other aspects such as incremental backup, data deduplication, or orchestration for data recovery.

Advanced settings allow for the creation of snapshots of the entire file system at regular intervals. These snapshots act as a momentary image of the file system, encompassing all data within all containers. Technologies such as ZFS or Btrfs are commonly

employed in these situations [6]. However, they come with their own sets of complexities and may not be compatible with all types of environments.

As businesses scale and evolve over time, the need for more secure and automated backup and recovery strategies becomes imperative. For containerized applications operating in orchestrated environments like Kubernetes, there are backup tools specifically designed for such platforms. Tools like Velero not only provide data backup but also capture the entire state of the application, including configuration and metadata [7]. Additionally, there is a trend towards using cloud storage services for backup purposes. The cloud provide scalable and often more cost-effective storage solutions, with some services providing built-in data backup and durability features. However, data security and compliance considerations play a crucial role in deciding whether to adopt cloud-based backup strategies [8].

Current approaches to backup and recovery in containerized systems are as diverse as the systems themselves. While some techniques and tools may work well in one context, they may be inadequate in another.

2. CHALLENGES WITH TRADITIONAL BACKUP AND RECOVERY METHODS.

Backup and recovery are key aspects for any IT infrastructure, and container systems are no exception. However, traditional backup and recovery methods can fail in this relatively new environment for several reasons:

Ephemeral Nature of Containers: The first challenge arises from the nature of containers themselves. Containers are designed to be ephemeral and stateless - they can be started or stopped at any time, with all necessary data stored outside the container. This design is advantageous for scalability and flexibility but poses a significant obstacle to traditional backup methods that rely on persistent environments like physical servers or virtual machines.

Application Complexity: As businesses adopt microservices and DevOps practices, application complexity is increasing. With potentially hundreds or thousands of containers running in an environment, each containing a different microservice, orchestrating a comprehensive backup strategy can be a complex task [9].

Persistent Data Management: Even though the containers themselves are ephemeral, the data they generate and use is not. Data generated by containers is typically stored in Docker volumes or persistent volumes in Kubernetes. While these storage solutions provide a certain level of data resilience, they introduce their own

challenges regarding backup and recovery, requiring specific strategies to ensure that the data is adequately protected.

Integration of Orchestration System: Container orchestration systems, such as Kubernetes, provide a framework for managing and scaling containerized applications. However, integrating traditional backup solutions with these orchestrators is not always straightforward. These solutions need to be able to understand the metadata, configurations, and state of the orchestrator to perform comprehensive backup.

Impact on Performance: As with any backup solution, the process of backing up data from containerized systems can impact system performance. This impact can be magnified in a containerized environment due to the large number of containers that can run on a system [10]. Any slowdown or drop in performance can have implications in production environments, especially for organizations running time-sensitive applications.

Inadequate Recovery Mechanisms: Traditional backup and recovery methods may not fully account for data recovery in a container environment. Recovering data in a specific container or orchestrating a complete system recovery after a disaster requires a deep understanding of the container orchestration platform and dependencies between different microservices.

Scalability: Container environments are often dynamic and highly scalable [11]. As such, backup and recovery solutions need to scale seamlessly with the environment. Traditional backup and recovery methods may struggle to cope with the rapid scalability commonly associated with container deployments.

These challenges demonstrate that traditional backup and recovery methods are often not equipped to handle the complexity and demands of containerized systems.

3. NEED FOR AN AUTOMATED BACKUP AND RECOVERY SYSTEM IN CONTAINER ENVIRONMENTS

The challenges associated with traditional backup and recovery methods in containerized systems make the need for automated solutions apparent. This point will explore the reasons for this need, outlining how automated backup and recovery solutions can address the issues presented by containerization.

Scalability: Container systems often exist in highly scalable environments. Traditional backup solutions may struggle to adapt to rapid scaling, resulting in incomplete backups or the need for more extended backup time windows that can impact system performance. On the other hand, automated solutions are inherently designed to han-

dle scaling. They can detect changes in the environment, such as adding or removing containers, and adjust backup processes accordingly. This dynamic adaptation capability ensures that even when the container environment is scaled up or down, all data remains protected.

Efficiency: In traditional systems, backups often need to be scheduled in time ranges when the system is not under load to minimize the impact on system performance. However, the ephemeral nature of containers can make this approach inefficient. An automated backup solution can perform continuous, incremental backups, capturing changes as they occur. This not only reduces the impact on performance, but also ensures that even the most recent data changes are backed up.

Recovery Speed: In the event of a disaster, time is of the essence. The faster a system can be restored, the less impact on business operations. Automated recovery solutions can decrease recovery time by eliminating the need for manual intervention. By utilizing metadata stored with each backup, these solutions can restore the entire containerized environment, from individual containers to orchestration configurations, ensuring a swift return to operational status.

Accuracy: Manual backup and restore processes are susceptible to human error. Errors such as missing a critical container in a backup or misconfiguring a restore process can result in data loss or extended downtime. An automated solution mitigates this risk by managing all aspects of the backup and restore process, ensuring nothing is overlooked.

Integration with CI/CD Pipelines: Containerized environments often use CI/CD pipelines for fast and reliable software delivery. Automated backup and recovery solutions can be integrated into these containers. This means that backups can be triggered by events in the pipeline, such as the deployment of a new service, ensuring that the most up-to-date system state is always protected.

Cluster-Level Incremental Backups: In a large-scale, distributed environment, maintaining consistent backups across multiple clusters can be challenging. Automated backup and recovery tools can communicate with container orchestration platforms to gain a holistic view of the environment. This allows them to create consistent backups across all clusters, ensuring complete data protection.

Cost Efficiency: Although it may seem counterintuitive, automated solutions can often be more cost-effective than traditional ones. They can reduce the need for intensive manual labor, reduce the risk of downtime (and thus lost revenue), and improve resource utilization by performing efficient, incremental backups.

4. SYSTEM ARCHITECTURE FOR AUTOMATED DISK VOLUME CREATION AND RECOVERY IN CONTAINERIZATION

An automated backup and restore system for container disk volumes can be designed as a distributed system architecture, with the ability to scale horizontally depending on the size and complexity of the environment. At the core of this architecture are three main components (Figure 1): the backup agent, the management server, and the storage server. Each of these components has a critical role to play must be designed to work harmoniously in the environment.

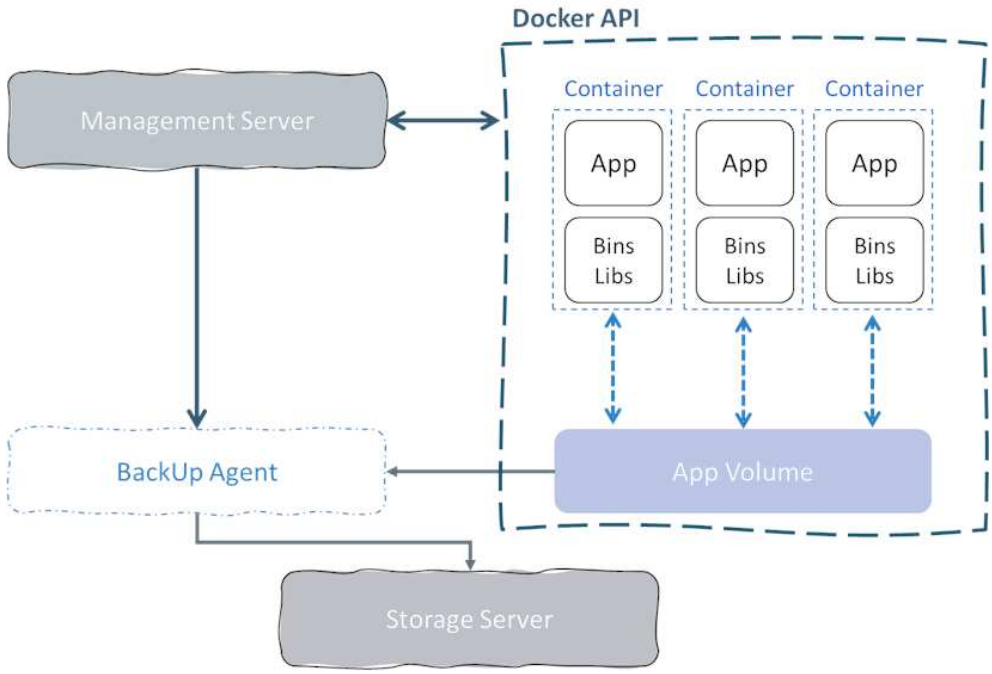

Figure 1: Architecture of the Proposed System

BackUp Agent

The backup agent runs at the node level in the container cluster. It is a container microservice that is deployed on every node that runs application containers. The key role of the Backup Agent is to continuously interact with the Docker Engine API and keep track of all running containers on its node and, more importantly, the volumes associated with those containers, as volumes are the persistent repository where application state is stored.

The backup agent is also integrated with the Docker event stream, which provides a real-time feed of container lifecycle events such as container creation, launch, shutdown,

and destruction. This integration allows the Backup Agent to react to these events in real-time and dynamically adjust its backup schedules without the need for manual intervention.

The Backup Agent can be a Docker container deployed on any host running Docker containers on the system. It monitors the Docker event stream to identify containers and their associated volumes, trigger backups based on a predefined schedule, and communicate with the management server.

Management Server

The management server is the orchestrator and coordinator of the system. Its primary role is to control the backup agents and trigger backup operations when necessary, according to the schedule or other conditions specified in the backup policy, as it is shown on Figure 2.

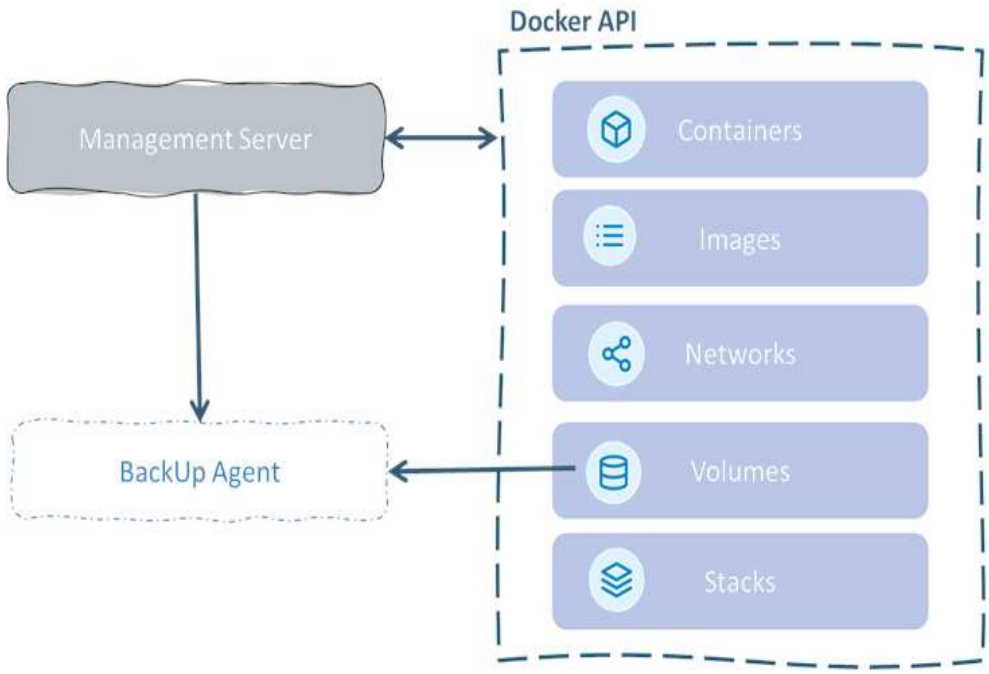

Figure 2: Architecture of the Proposed Management Server System

When a backup operation is triggered, the management server communicates with the appropriate backup agent and initiates the backup process. After a backup, it also collects metadata about the backup from the Backup Agent, which includes details such as backup time stamp, container and volume details, backup size, location where the backup was stored, and so on. This metadata is then stored in a database managed by the management server, which is then used to facilitate fast and efficient restore

operations.

In addition, the management server also interfaces with the storage server to store backup data and ensure that data is stored securely and efficiently and can be retrieved quickly when needed. This component serves as the control center of the system, managing the backup agents and interfacing with the storage server. It also maintains a database of backup metadata that is used during restore operations.

Storage Server

The storage server is responsible for storing the actual backup data that is created by the backup agents. The storage server design is flexible as it supports different types of basic storage technologies such as block storage, file storage or object storage (Figure 3).

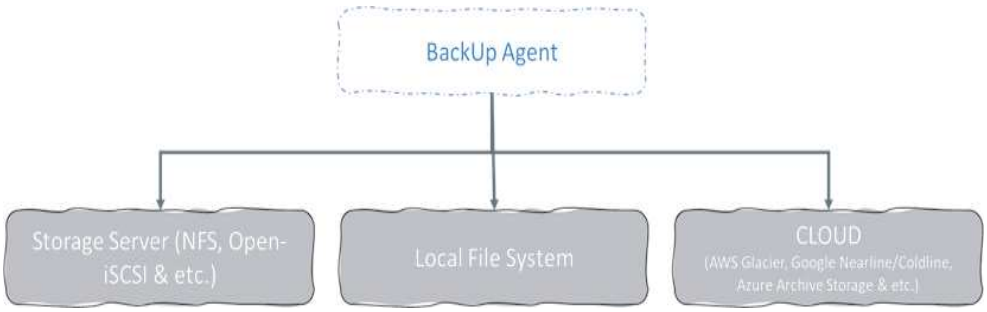

Figure 3: Backup Agent and Storage Options

This flexibility allows the system to be tailored to the specific storage requirements of a given environment, thus ensuring optimal performance and cost efficiency. The storage server interfaces with the management server to receive the backup data, store it securely and efficiently, and retrieve it when needed for a recovery operation. The storage server is responsible for storing backup data. It interfaces with the management server to receive backup data, store it efficiently, and retrieve it during recovery operations.

Together, these components work collaboratively to provide a robust, scalable, and efficient automated backup and recovery solution for containerized applications. The architecture is designed to handle the dynamics and scale of modern container environments while ensuring the reliability, performance and security required for enterprise-class backup and recovery solutions.

5. CONCLUSION

The proposed architecture can provide a stable, comprehensive, and seamless solution for automating the processes of backup and recovering containerized environments. Offering multiple advantages that enhance the reliability, efficiency, and manageability of backup and recovery operations. Some of the benefits that will be observed as a result of developing the three main components of the system – backup agent, management server, and storage server would include: Comprehensive data protection, system availability, scalability, secure data recovery in case of a system failure.

6. ACKNOWLEDGMENT

This research is funded by the Bulgarian National Science Fund under Project KP-06-NP62/1.

REFERENCES

[1] Bentaleb, O., Belloum, A.S.Z., Sebaa, A. et al. Containerization technologies: taxonomies, applications and challenges. *J Supercomput* **78**, (2022),1144–1181.

[2] Rad, B.B., Bhatti, H.J. and Ahmadi, M., 2017. An introduction to docker and analysis of its performance. *International Journal of Computer Science and Network Security (IJCSNS)*, **17(3)**, p.228.

[3] Turnbull, J., *The Docker Book: Containerization is the new virtualization.* James Turnbull, 2014.

[4] https://www.docker.com/, last visit: December 2023.

[5] Clark, C., Fraser, K., Hand, S., Hansen, J.G., Jul, E., Limpach, C., Pratt, I. and Warfield, A., 2005, May. Live migration of virtual machines. *In Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation*,**Volume 2**, 273-286.

[6] Tarasov, V., Rupprecht, L., Skourtis, D., Li, W., Rangaswami, R. and Zhao, M., 2019. Evaluating Docker storage performance: from workloads to graph drivers. *Cluster Computing*, **22**, 1159-1172.

[7] Kaur, K., Guillemin, F. and Sailhan, F., 2023. *Live migration of containerized microservices between remote Kubernetes Clusters.*

[8] Mell,P. , Grance,T. The NIST Definition of Cloud Computing. https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf, last visit: December 2023.

[9] Johannes Thönes. Microservices. *IEEE Software,* 2015. 113-116.

[10] M. T. Chung, N. Quang-Hung, M. -T. Nguyen and N. Thoai, Using Docker in high performance computing applications, *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE),* Ha-Long, Vietnam, 2016, 52-57.

[11] Dewi,L. P. , Noertjahyana, A., Palit N.H. and Yedutun,K. , Server Scalability Using Kubernetes,2019 4th Technology Innovation Management and Engineering Science International Conference (TIMES-iCON), Bangkok, Thailand, 2019, pp. 1-4,