White Paper

Delivering Persistent Storage for Linux Containers

Durable, Enterprise-grade Persistent Storage for Applications Deployed in Containers

By Scott Sinclair, ESG Senior Analyst

November 2016

This ESG White Paper was commissioned by Red Hat and is distributed under license from ESG.

Contents

Introduction

Staying competitive in the digital age requires the ability to quickly analyze new information and adapt the business to evolving market demands. Achieving this level of business agility creates new and extensive requirements for IT organizations. Without the right tools, the management complexity of accelerating the development and deployment of new workloads alone is unsustainable for many organizations given their existing infrastructure.

The rise of server virtualization increased the utilization of infrastructure, such as server, storage, and networking components, as well as the agility of application deployment. Yet, the pervasiveness of server virtualization places the technology at or near the market saturation point. Application developers and their supporting IT organizations seek tools to further the speed and the efficiency level at which applications can be developed and deployed. As a result, a rising percentage of IT organizations are turning to container technology. Unlike server virtualization, containers package an application with all the necessary libraries in a virtual container rather than also including a guest operating system. The result is a more efficient development and deployment model. We're talking seconds and minutes rather than hours and days; hundreds, even thousands of containers, rather than dozens of virtual machines.

The recent surge in interest in containers may suggest that it is a fairly new technology. However, Linux was working with container technology 15 years ago. Early efforts, according to pioneers of the Linux-VServer solution, sought to "separate the user-space environment into distinct units in such a way that each Virtual Private Server looks and feels like a real server to the processes contained within."[1] The open source Docker project has played a significant role in helping to fuel the rise in container adoption by introducing standards and simplifying developer tools, which aid in packaging and deploying new applications.

In a recent ESG research study, one-third of the 300+ IT managers surveyed indicated that their organization is currently using containers. Add to that the number currently testing containers for use (32%) and those planning to use containers (16%), and this surge in interest becomes apparent.[2] In this era often defined by dramatic IT and business transformation, organizations are turning to container technology. Helping to enable that transition, Red Hat, an industry leader in Linux and open source software, is developing an ecosystem to assist enterprises harness the full potential of container technology. Red Hat's expertise in both enterprise and open source application development helps make it a trusted advisor for organizations investigating or deploying containers. The IT agility enabled by containers offers prominent business value, and that means greasing the skids for developers' vital work of bringing key applications to market faster than ever.

DevOps: A History of Conflicting Goals

Developers need speed and choice to support their businesses' transformation into highly agile entities. Containers offer a sort of isolated "sandbox" for application testing and running. The result delivers the necessary speed for optimal development time and the choice of infrastructure to accomplish the task.

Before DevOps became an everyday term, application development involved a delicate balancing act between developers and IT operations, often resulting in tension between the two. In essence, developers would ideally like a greater degree of control over tools, processes, application platforms, and environments to undertake development and testing. IT operations, on the other hand, have vast responsibilities to manage those tools, processes, application platforms, and environments consistently across the entire enterprise, with as few exceptions as possible.

Today DevOps defines a practice or modus operandi that drives toward collaboration and better communications between developers and IT operations. The goal today is to streamline the processes whereby the development, testing, and

[1] Source: Red Hat Enterprise Linux Blog, *The History of Containers*, August 2015.
[2] Source: ESG Research Report, *The Cloud Computing Spectrum, from Private to Hybrid*, March 2016.

ultimate release of new applications happens as speedily as possible and with almost rhythmic regularity. This harmony is often elusive at best, given the conflicts in goals mentioned.

Containers Bridging the DevOps Divide

Containers are a clear pathway to this DevOps harmony—the conflicting goals notwithstanding. Containers have the capability to offer autonomy for applications and the developers that create them by packaging applications with the libraries and certain binaries on which the applications depend. They combine relatively lightweight application isolation with an image-based deployment scheme. Without an operating system kernel, containers are meant to be lighter and more reliable than virtual machine environments.

For developers, containers deliver enhanced application isolation as well as better application stability. For IT operations, containers can result in higher quality software releases from the development side of the house. IT operations also gain more efficient virtual machine replacement in production, and generally easier application management.

Abstracting the storage software from the hardware with container solutions propels faster deployments in addition to more efficient software designs. This capability can go a long way to reduce the costs of tailoring and tuning the infrastructure to the needs of individual applications.

Put more simply, containers can help resolve the inherent conflict that has plagued developers and operations for years—a conflict that is much to the detriment of overall development and, therefore, the business.

How to Think About Containers

Stripped down to bare essentials, containers feature a core design that packages an application with all its environmental dependencies. It runs on a single OS. It should run reliably when moved from one compute environment to another, say from a laptop to the test environment. It represents an entire run-time environment.

The shipping industry provides a fitting analogy to the job capabilities of containers. Shipping containers—those ubiquitous cargo-laden vaults that dominate shipping ports—are usually packaged at the point where goods are created, namely factories around the world. Then, they are delivered to the docks (not packed there). After that, they are fitted onto trucks, railroad cars, planes, and of course, ships. And the entire time, there is no need for repackaging as they move from one venue to another.

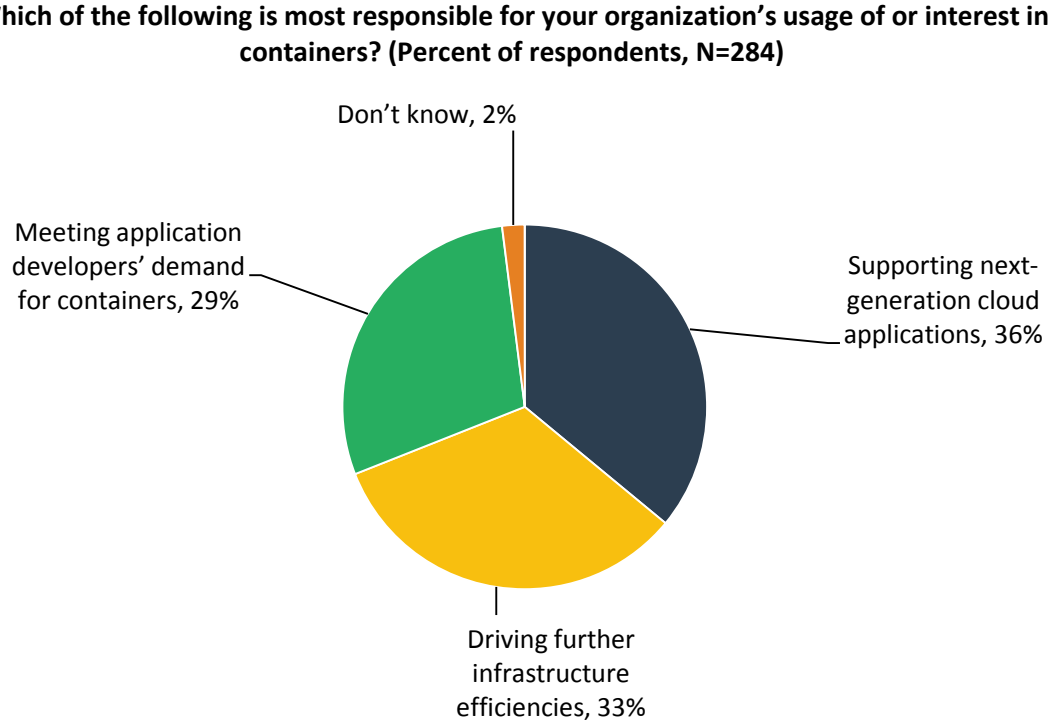
All shipping companies can easily exchange shipping containers, regardless of how they are transported, because the containers comply with international sizing standards. Similarly, there is a need for similar standards for software containers and their associated applications.

Drivers of Interest in Containers

As Figure 1 shows, the perceived benefits of containers align pretty closely with the overall demands of the application development environment. The two biggest drivers of interest in containers are support for next-generation cloud applications (36%) and driving greater efficiencies in infrastructure (33%). Closely following as the third driver is meeting application developers' demand for containers (29%).[3]

[3] Ibid

FIGURE 1. Factors Responsible for Interest in Containers



Source: Enterprise Strategy Group, 2016

This data suggests that containers are destined to become a key component of future enterprise workload deployment. The truth is that the IT world is moving decisively away from and beyond self-contained, isolated, and monolithic applications. Emerging workloads will be part of a connected application fabric, very flexibly stitched together to serve particular business needs, yet easily unbraided to be later recomposed.

While a natural affinity exists between containers and DevOps activities, the potential of container technology extends well beyond application development to a variety of enterprise workloads. Businesses are investigating containers for on-premises, cloud-native applications, web, and e-commerce, as well as to support more decentralized systems of engagement workloads, along with mobile or social applications. Not only do containers ease the transition between on-premises and public cloud, they also make it much easier to transition applications between cloud providers, helping to alleviate lock-in with public cloud providers. It is also worth noting here that to further support the rise of container technology, other infrastructure foundational blocks will need to adapt. These include compute and networking in addition to storage.

Wanted: A Storage Platform for Containers

Containers hold the potential to revolutionize DevOps, owing to the peerless way they encapsulate application logic. But there is a rub: Containers on their own do not offer an acceptable or viable way to store application data across the entire lifecycle of the container.

So-called ephemeral or local storage is just not enough because stateful container-based applications demand that data be available beyond the life of the container. What's more, application data stored on local disk is often lost if containers fail. In addition, application requirements are frequently not served by cloud storage APIs, which can be narrow in focus. This factor can force inappropriate semantics onto the applications, creating complexity.

All this is in direct conflict with overarching infrastructure efficiencies that organizations require for cloud-based applications. Data for containerized applications must remain stable and available beyond the life of the container if containers are to become true enterprise-grade solutions.

Owing to the growing interest in containers, established IT vendors have readily embraced them. Many vendors now offer storage available to applications running in containers. But that is not enough. What's really needed is an underlying, comprehensive storage platform that matches the container host platform itself when it comes to agility and flexibility. This implies truly distributed storage that can scale effortlessly and leverage industry-standard hardware. This is not just because commodity hardware is often less expensive; moving away from proprietary hardware solutions enables the tailoring of different hardware options to ideally match the workload of the given job. Containers need software-defined storage, and software-defined storage lends itself very nicely to fulfilling the vision of containers.

A Perfect Fit for Software-defined Storage

As it turns out, software-defined storage (SDS) is a near-perfect fit for mitigating the various challenges to development of an enterprise-grade container strategy. It is not an understatement to say that SDS is a new evolution in storage that fundamentally changes how storage will be managed and deployed over time.

Though there is no uniform definition of SDS, it is helpful to think of it as a solution that offers storage functionality as software, abstracting the capability from the underlying hardware infrastructure. SDS-based offerings can combine with industry standard, or server, hardware to act as an enterprise storage repository offering some combination of block, file, or object storage. SDS technology also may be integrated into a hyperconverged schema where both storage and applications run on the same servers, side by side. The potential benefit of SDS-based technology is derived from its deployment flexibility and adaptability. The same SDS technology can be deployed across a variety of hardware environments that can differ in optimization, e.g., for performance, capacity, or cost. Additionally, infrastructure can adapt as workloads demands evolve while the storage intelligence and application interface remains constant. Ultimately, it is this abstraction or decoupling of the storage intelligence from the hardware that explains why SDS offers the flexibility and agility to meet the needs of container-based applications.

The Benefits of Software-defined Storage

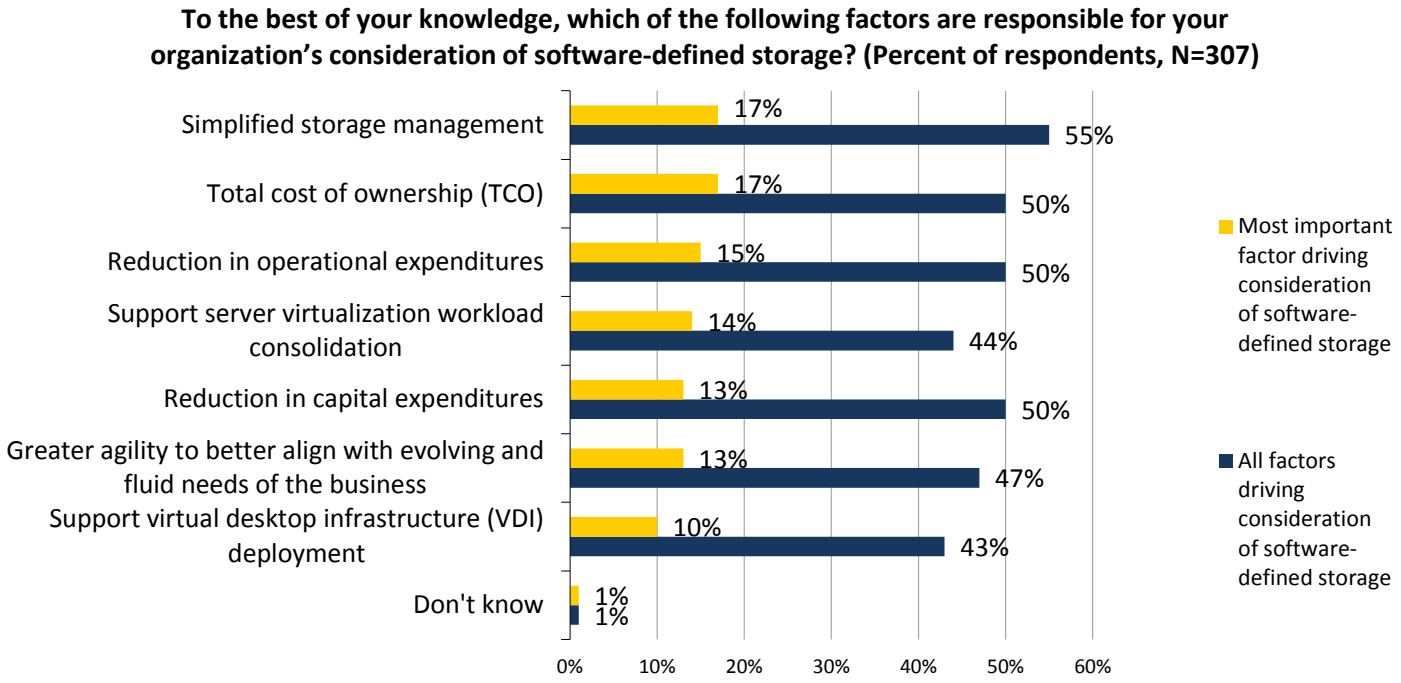
SDS is naturally hardware independent, which is a plus for effectively pairing the hardware with specific, unique tasks. SDS is also capable of managing heterogeneous and evolving hardware elements under a single control plane, which traditional storage schemes can struggle to do.

Furthermore, solutions that deliver tools to integrate storage and application development areas will add incremental value—the result of developers gaining greater control over the way they provision and manage storage. For that reason, SDS is a great fit for container technology, given the benefits for both developers and IT operations.

According to research from ESG, the most commonly cited benefits expected from SDS (see Figure 2) include benefits to management simplification and agility, along with cost savings.[4]

[4] Source: ESG Research Report, *2015 Data Storage Market Trends*, October 2015.

Figure 2. Factors Responsible for Organizations' Consideration of Software-defined Storage



Source: Enterprise Strategy Group, 2016

These data points suggest that by abstracting the storage software from the underlying hardware infrastructure, IT organizations are able to deliver a simpler and more agile storage environment. SDS solutions allow the storage interface and management to persist as the underlying hardware ecosystem evolves. This consistency in the wake of hardware innovation simplifies storage management and offers agility in adjusting the hardware to meet the needs of the workloads as well as speed up the integration of new hardware technologies. This flexibility in hardware choice leads to a reduction in operational and capital expenditures. By deploying storage as software, new capacity can be deployed and provisioned almost immediately, eliminating the multi-month procurement and deployment process for more traditional storage systems. The result is an efficient and agile architecture that can keep pace with the speed of container-based development.

As part of its larger enterprise strategy, Red Hat offers an extensive portfolio of open source SDS-based solutions. With its Ceph and Gluster storage software technologies, Red Hat is able to deliver block, file, and object storage technology across a diverse set of hardware options. Developers can quickly download and deploy Red Hat's open source SDS technology on existing industry standard hardware to accelerate the development, testing, and validation of new applications. Then the same SDS technology can be deployed across an enterprise-level hardware infrastructure to serve as the production environment. For DevOps environments, the deployment flexibility offered by SDS provides a level of consistency necessary to ease and speed up the transition from the development environment to production environment.

Red Hat, SDS, and Containers

Red Hat is one technology leader investing heavily in this emerging technology by building a storage ecosystem around containers. Through this investment, Red Hat has designed its storage infrastructure to offer the flexibility and agility for a new generation of container-enabled IT environments. Moreover, Red Hat backs its SDS Linux container portfolio with a single point of support for container platforms and container storage.

Businesses can choose to cobble together their own container environments by pressing their IT operations staff into this service and hiring independent consultants to support it. But Red Hat's approach is to relieve businesses of this tricky and time-consuming task by instead offering a full technology stack approach that provides an end-to-end Linux container ecosystem.

Containerized applications can gain access to highly available, persistent storage regardless of form. Red Hat's SDS approach also allows storage to be shared between containers or isolated to an individual container—developers' choice.

Requirements for Success

For container technology to reach its full potential and enable a new generation of application development, the underlying storage must be distributed, persistent, and secure. Persistent storage generally refers to any data storage device that retains data after power to that device is shut off. It is also sometimes referred to as non-volatile storage. Serving enterprise workloads, however, requires more than just persistence.

The success of containers may be ultimately determined by their acceptance by IT operations and infrastructure leaders. Enterprise operations and storage leaders demand the high levels of resilience, seamless scalability, and manageability that they find in existing enterprise storage solutions. They also look for persistent storage between development, test, and production environments so as not to hinder application deployment with inconsistencies.

Robust Storage Requirements

While these requirements sound reminiscent of those for typical server virtualization environments, the storage needs of container solutions may even exceed those of virtualization environments. Designed to serve developers in the DevOps world as well as cloud applications and, in the future, the delivery of micro services, storage for containers must be more agile than a typical server virtual environment. This fact reinforces the requirement for SDS-based solutions. Container storage solutions must be at least as flexible as the application development environments they support; therefore, requirements include:

- Support for very fast and agile deployments and flexible hardware support through SDS-based solutions that allow storage to be deployed as software.
- Underlying storage technologies that scale performance and capacity in a single pool as container demands soar (i.e., scale-out architecture).
- A service interface by which storage can be dynamically provisioned on-demand by the developer of the containerized application.

Traditional storage systems were built to support a different era essentially. SDS, by contrast, was built in the era of cloud, where scale and flexibility were available from day one, not bolted on after the fact.

Red Hat Gluster Storage holds data for stateful containers needing common file storage for data sharing. In such a case, the application requires a persistent file system for sharing data across containers spread among different hosts. Gluster is able to serve out file system storage to a cluster orchestration framework like Kubernetes, which has quickly emerged as a leading technology for container orchestration.

Tight integration with OpenShift Container Platform

Red Hat's OpenShift Container Platform (formerly OpenShift Enterprise) offers services as well as a choice of native storage platform. Red Hat OpenShift Container Platform provides developers with a powerful platform for provisioning, building, and deploying applications and their components in a self-service fashion.

With a focus on integration, Red Hat's Container Native Storage approach offers a differentiated option for container environments. As part of this architecture, Red Hat is able to deploy both applications and storage in a converged manner where the storage intelligence resides inside a container. This architecture reduces both capital and operations expenses by eliminating the requirement to deploy and manage separate storage infrastructure. The innate ability of SDS to support multiple deployment types, such as converged or as a separate storage layer, further supports the technology as a strong choice for container environments.

As part of this approach, Red Hat has undertaken considerable work to integrate storage into OpenShift Container Platform to boost the overall customer experience. Included in these efforts are:

- The ability to secure persistent volumes for containers with Access Control and SELinux.
- The ability to choose between the use of local disk, network storage, or hyperconverged storage for container persistence.
- The ability for developers to provision storage for their applications without involving storage administrators and the ability for administrators to provide quotas to control what storage the developers can provision.
  - Storage plug-ins for Gluster, as well as for NFS, iSCSI, Fibre Channel, Google Cloud Engine, Amazon EBS, OpenStack Cinder, Azure and many others.
  - Heketi, a RESTful service, which serves as volume management providing automated creation, deletion, and expansion of GlusterFS volumes across multiple GlusterFS clusters supporting cloud services such as OpenShift and OpenStack.
- A persistent volume selector that helps users differentiate between storage back-ends with similar access modes.
  - Integration with Docker Engine, Kubernetes, and other emerging industry-standard technologies.
  - Unified management and operation.

The bottom line is that traditional storage technologies were never optimized for containers, but rather for monolithic applications that are often self-contained and isolated. SDS, on the other hand, delivers the functionality that ultimately enables containers to work their magic in the DevOps environment and beyond, as micro services gain appeal.

The Bigger Truth

While containers adoption is arguably still in its early days, interest is very high, owing to the great potential benefits both to IT in terms of streamlining DevOps and to the business in terms of higher quality, speedier application development, and deployment. Broader adoption will require the kind of enterprise functionality and consistency that IT operations demands, and rightfully so. Persistent storage is a key underpinning of this functionality. And while traditional storage methodologies are too rigid and incapable of scaling seamlessly, SDS is an excellent fit for the container environment. The world of containers, however, is still evolving. It is a technology area expected to experience a wealth of innovations over

the coming years. Choosing a partner when a technology is still in its adolescence can be a daunting task. The key is to choose a trusted vendor with experience delivering in the enterprise. Additionally, review the velocity of innovations to date as a potential proxy for the pace of future innovation. While many venerable IT vendors have jumped into this burgeoning market for containers, Red Hat exists as a trusted and experienced partner offering solutions across the container stack and can tout a strong history of supporting the developer community.

Enterprise Strategy Group is an IT analyst, research, validation, and strategy firm that provides actionable insight and intelligence to the global IT community.

 www.esg-global.com

 contact@esg-global.com

 P. 508.482.0188