DELL EMC VXRAIL AND VXRACK SYSTEM SDDC™ CONTAINERS OVERVIEW

RAYONSTORAGE.COM | GREYBEARDSONSTORAGE.COM
SILVERTONCONSULTING.COM | TWITTER.COM/RAYLUCCHESI

Introduction

Containers are a powerful technology, primarily used to run cloud native services but can also be used to run highly scalable, on-premises applications. Containers support modern application development and operations models that enable rapid implementation, automated deployment and easy scalability.

VMware vSphere® can now run containers under any of the following:

- **Standalone Docker Container Engine** –container developers essentially do it all using command lines and APIs;
- **vSphere Integrated Containers** – container developers have a self-service GUI, integrated with a private, secure repository and vCenter operations management, visibility and control over container execution;
- **Pivotal Container Service** (PKS) – developers and operations staff can make use of a combination of command line and vCenter services to manage and scale container applications; and
- **Pivotal Cloud Foundry Application Service** - developers and operations staff can take advantage of sophistication and automation to step away from the technical details and let the system run the application by itself.

VMware vSphere presently offers the largest selection of methods to run containers in the enterprise. Dell EMC's extensive engineering to integrate and validate vSphere on Dell hardware together with their single support model make Dell EMC® VxRail™ and VxRack™ System SDDC (software-defined data center) hyper-converged engineered systems the fastest and simplest infrastructure to run vSphere today. If your IT team is interested in using containers in their data center, one of the best ways to do so is on VxRail and VxRack SDDC.

Containers

Although container applications have been around since the early 2000s, they really gained popularity when the open source Docker™ Project came out with its container image format and runtime. Docker images are a standardized, lightweight, standalone, executable package containing everything needed to run a function such as compiled code, runtime libraries, environment variables, and configuration files.

Containers are mainly used to implement cloud native web services and other highly scalable applications that distribute functionality across many individually executable units. Typically, dozens if not hundreds of containers are used by an

Silverton Consulting
Strategy, Storage & Systems

application, each of which implements a small amount of functionality, called a **microservice**. To scale application performance, multiple containers can execute the same microservice. Containers can also be used to implement non-microservice applications, but they're typically used for highly scalable, microservice applications.

A container executes functionality using kernel resource isolation features to allow multiple independent containers to execute under a single OS instance. Using these services, containers can avoid the overhead of running a full-blown virtual machine (VM) under a hypervisor and still provide a lightweight virtualized environment.

Containers are designed to run completely isolated from the host, only accessing files and ports if configured to do so. Containers also execute in a stateless fashion. That is, any container state information is lost when an instance is terminated. However, containers can make use of external services such as backing stores or databases to save application state if needed and recent research is moving to offer options for more stateful containers without need for external services.

Nowadays, container management systems (CMS) exist almost everywhere. Amazon Web Services™ (AWS™), Microsoft Azure™, Google Cloud Platform™ (GCP), IBM® SoftLayer, Pivotal® Web Services (PWS), Virtustream® and many others offer container support.

Development teams are using containers for a number of reasons, such as:

- **Faster development** – container applications can easily be built from existing or new microservices but can also be used to re-package/implement existing monolithic applications,
- **High resilience** – container applications typically run multiple containers, any of which can fail and be restarted very quickly,
- **Portability** – container applications can run just about anywhere (e.g., on local workstations, in virtualized infrastructure and in private or public cloud environments).
- **Scalability** – containers require less OS functionality to run, so they can start up or shut down rapidly to improve application performance.

IT organizations benefit by moving application engineering to a more modern development paradigm that is designed to run both on-premises and in the cloud. Moreover, when new functionality or changes are implemented using containers, applications typically evolve in small increments rather than through major releases, which allows for quicker rollout of new features and cloud native like, development lifecycles through the use of DevOps.

Containers exist as images that reside in registries. Docker supports its own publicly hosted registry service called the Docker Hub™,[1] which holds 100,000+ container

applications. For example, Redis™, MongoDB™ and MySQL™ app instances are all freely available under Docker Hub.

Container applications execute under a container host or engine, which supplies a mini-virtualization environment for containers. Container hosts also supply local service routing, container (work) scheduling, container spin up and spin down services and local (within host) dependency mapping services.

Container engines are typically managed by a CMS, such as Docker Swarm™, Google Kubernetes® (often called K8S), Mesosphere® DC/OS, and Cloud Foundry™ Container Service. A CMS deploys and manages the lifecycle of containers and container engines.

How to run containers on vSphere

Options for running container apps using vSphere include standalone Docker Container Engine, vSphere Integrated Containers Engine, standalone Kubernetes, and using Pivotal Cloud Foundry with its Pivotal Container Service (PKS) and Pivotal Application Service. We review some of the more popular ones below.

Standalone Docker Container Engine

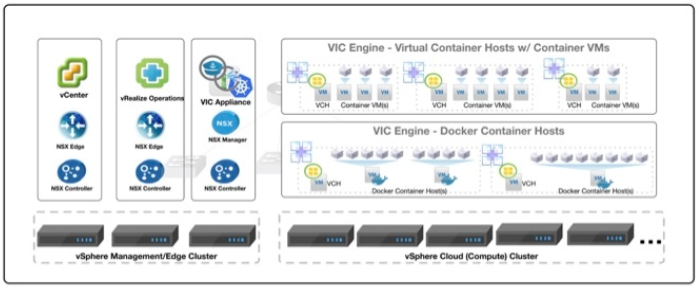
A Docker Container Engine (DCE) or host can be run under a Linux OS VM to execute containers under vSphere running on VxRail or VxRack SDDC. That way, Linux OS, the DCE and its containers run within a single VM. This requires the most development expertise and responsibility because it's essentially only a container runtime without a CMS.

Such a DCE VM can make use of public or private container registries. VMware admins can provision the VM with any resources it needs, and developers can use the Docker application programming interface/command line interface (API/CLI) to deploy and run containers within that DCE VM.

Note that using a standalone DCE doesn't provide cluster management or high availability app execution, it's simply a standardized container runtime. However, organizations could run one of the Docker CMSs discussed above. Such an approach wouldn't be able to take advantage of many vSphere services or to use knowledge of the vSphere cluster environment and they would be under developer responsibility to control.

For developers, use of a standalone DCE will require a high level of skill in Docker API/CLI. Further, operators will only have the single (DCE) VM to manage and will lack any visibility into containers running in that VM.

vSphere Integrated Containers



vSphere Integrated Containers (VIC) is another method for executing container apps under VxRail or VxRack SDDC. Containers under VIC can run in one of two modes:

- Using a **Virtual Container Host (VCH)** resource pool – the VCH and all its containers execute as separate, lightweight VMs in vSphere.
- Using a **Docker Container Host (DCH)** VM – similar to the standalone DCE above, Linux OS, Docker Engine and all its containers execute within a single VM.

VCH operates as a bridge between vSphere and the application's containers, which uses PhotonOS a lightweight Linux kernel that interfaces well with vSphere and is quick to boot, deploy and run.

In addition to the Docker Hub registry discussed above, VIC container apps can use Harbor™,[2] VMware's open source project, which implements a secure, enterprise-class container registry that can be hosted on-premises. Under Harbor, container application repositories are implemented as projects, and users can have different privileges on different projects. Harbor also has built-in vulnerability scanning to insure images are secure.

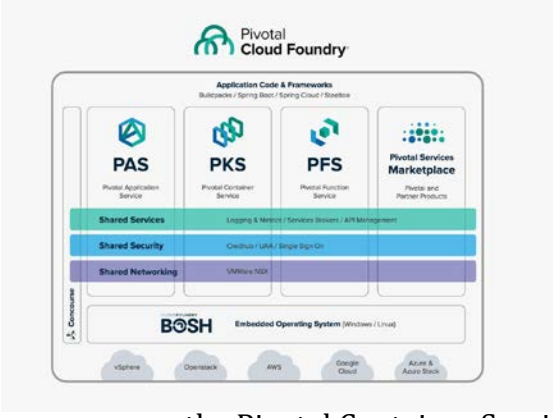
Furthermore, under VIC containers, developers can use a self-service web portal called Admiral™,[3] VMware's open source project for container lifecycle management. Admiral makes use of the Docker API to manage containers and container hosts. Admiral provides developers a GUI container management console and automated policy management for deploying and running containers in VIC. However, developers can still use a CLI to interact with the environment. Operators also benefit from its increased security and governance control.

For VIC VCH container VMs, vSphere vCenter services can also be used by operations to manage container VMs in a 1 to 1 model (container to lightweight VM) vs. VIC DCH that runs multiple containers in a single VM in a many to 1 model. These include vSphere vMotion™, Distributed Resource Scheduler™ and High Availability™ services.

Also, by introducing standard vSphere services to the container execution environment, with VIC developers can take advantage of VMware storage to maintain state across container executions and other vSphere services such as HA, DRS and others. This means that traditional, monolithic enterprise applications can

be repackaged to take advantage of container deployment and operations models. Monolithic container apps wouldn't be standard microservices containers, but they could still be run as containerized applications under VIC and utilize Admiral and Harbor.

For VIC, developers new to containers can get by with less knowledge of Docker API/CLI as they can use the Admiral self-service container management console. Operators can also take advantage of Admiral and for VIC VCH, they can also make use of vCenter to manage the single container VMs and the VCH VM.



Pivotal Cloud Foundry

Pivotal Cloud Foundry (PCF) is a cloud native platform for deploying and operating modern applications. This modern platform allows for multiple layers of abstraction. Two of interest to us are the Pivotal Container Service™, a Containers-as-a-Service (CaaS) offering and Pivotal Application Service™, a Platform-as-a-Service (PaaS) offering, both are discussed below.

PCF also supports the Pivotal Function Service™ (PFS™), which provides server-less functionality. PFS is a future deliverable and will not be discussed any further in this document.

Pivotal Container Service

Pivotal Container Service (PKS™) runs under Pivotal Cloud Foundry as its CaaS service that combines Cloud Foundry BOSH™ infrastructure management with production grade Google Kubernetes CMS and engine as another way to run containers under VxRail. BOSH[4] and Kubernetes[5] are both open source projects, and combined are called the Pivotal Container Service.

Google originally developed Kubernetes to manage its production workloads across its data centers, and Kubernetes currently runs billions of containers a week. Kubernetes executes under GCP, Amazon EKS (Elastic Container Service), Microsoft AKS (Azure Container Service), IBM Cloud Container Service or standalone on vSphere, as well as in a bare metal environment such as Windows and Linux. For this section however, we review it in context with PKS.

BOSH is used for release engineering, deployment lifecycle management and distributed systems monitoring/restart. BOSH is used internally in GCP, AWS EC2 (Elastic Compute Cloud), Pivotal Cloud Foundry and OpenStack®. BOSH also supports availability zones (AZs) as its unit of high availability or failure domain.

Silverton Consulting
Strategy, Storage & Systems

PKS container apps can still take advantage of VMware's Harbor container registry but do not operate under VIC's Admiral container management tool.

Kubernetes can run almost anywhere – on bare metal servers, on virtualized infrastructure and in the private or public cloud. As a result, application engineers can develop and test container functionality using a bare metal Kubernetes cluster and, when ready, promote them to vSphere PKS or scale them up to the cloud.

With PKS, developers benefit from multi-container scheduling and deployment and need only supply the Docker container image with instructions on how to run it and the platform takes care of the rest. Operations get a production grade distribution with a consistent deployment and robust enterprise grade, platform experience.

Pivotal Application Service

Pivotal Application Service (PAS™) runs as part of the Pivotal Cloud Foundry as a complete PaaS solution. The Application service supplies the **Application Runtime** with container execution engines, the **Ops Manager**, a container app GUI deployment and management tool, all while using BOSH for release engineering, deployment and high availability. PCF PAS can operate bare metal, under vSphere as well as inside AWS, Microsoft Azure and GCP.

Container apps under PAS are deployed in one of two formats:

- **Droplet** container images created and packaged using PCF buildpacks; or
- **Docker** container images created and packaged using Dockerfiles.

PAS runs both Droplet and Docker images as container apps. Docker containers are just a special form of PCF Droplets. However, PCF Droplet containers can inherently make use of persistent storage and remove much of the developer burden for container creation, packaging, and management. Docker container images provide flexibility in PAS for those organizations that may have already standardized on the use of Docker images.

PCF Droplet apps are generally cloud native apps that use a 12-factor app development approach,[6] which specifies a methodology for building inherently cloud native applications.

PAS container apps can make use of Harbor registries or other private and public registries to hold container images. However, PAS Ops Manager is the management console for PCF PAS container apps.

Developers can use PAS to create containers for them in a completely automated fashion and allows them to just focus on writing code rather than how applications are deployed. Moreover, operators can also make use of PAS to automatically

manage container app execution and take advantage of an enterprise grade platform operations experience.

When to use standalone DCE, VIC DCH, VIC VCH, PKS or PAS

	Standalone DCE	VIC DCH	VIC VCH	PKS	PAS
Initial container use	Yes	Yes	Yes	No	No
Containers in production	No	No	Yes	Yes	Yes
Containers as VMs	No	No	Yes	No	No
Harbor support	No	Yes	Yes	Yes	Yes
Admiral support	No	Yes	Yes	No	No
Multi-AZ support[7]	No	No	No	Yes	Yes
CMS cloud compatible	Yes	Yes	No	Yes	Yes
Developer effort needed	High	High	High	Med.	Low
Operations control	Low	Med.	High	High	High

Standalone DCE is best for customers who are getting started with containers for the first time. This is really just a standardized Docker container runtime for developers. Engineers could develop container apps on bare metal and then readily migrate those apps to a standalone DCE VM. However, standalone DCE apps won't run as fast as VIC apps. Multi container deployments and deployment management are the responsibility of the developer to maintain. This requires a high level of skill on the developer's part to create and maintain. Operations has limited control and visibility beyond managing the single DCE VM.

VIC DCH is a stepping stone to production use of containers on vSphere. The advantage of using VIC DCH is that engineers who lack the authority to spin up VMs can develop, deploy and run container apps using vSphere Harbor registries and the Admiral management console. Developers may still use the Docker API/CLI but can also use Admiral to create containers. Operations can use Harbor to provide a secure and controlled registry and Admiral to manage container execution but have limited control beyond managing the single DCH VM.

VIC VCH is a production environment for container apps under vSphere. Developers will use Admiral or the Docker CLI/API to create containers. Operators can also use Admiral and have complete vCenter visibility and control to manage the VCH and container VMs.

Silverton Consulting
Strategy, Storage & Systems

PKS is ideal for running stateful applications and applications that are already containerized. PKS is also for developers and operations staff who need more control over container setup and deployment options by using Kubernetes with BOSH. Development will use the Kubernetes API/CLI to create containers. Operations will use both the Kubernetes and BOSH APIs/CLIs to manage container app execution.

PAS is for development and operations teams that want the easiest way to run container and 12 factor apps using a full implementation of PaaS. Developers will use PAS to automate the creation of containers and 12 factor apps for them. Operations will use PAS to automate the management of container apps execution throughout the PAS cluster.

VxRail and VxRack SDDC container considerations

VxRack SDDC can run every one of the above options but PKS and VxRail can run all of them. Moreover, VxRail also offers the **Pivotal Ready System** solution which is a certified, validated, pre-engineered and supported offering from Dell EMC, specifically designed for enterprise customers wanting to use PCF PKS and PCF PAS.

Summary

Using vSphere on VxRail and VxRack SDDC solutions offer many alternatives for container development, deployment and execution. Container apps can execute in as simple an environment as a standalone DCE VM or as complex an environment as a full-blown PaaS using PCF PAS. Using VIC with both DCH and VCH offers a more native VMware environment experience for both development and operations to help manage and run container apps across virtualized infrastructure. In addition, PCF PKS together with VxRail can give organizations a fully functional cluster orchestration, along with a rich, scalable and highly available container execution environment, without having to make use of a full-blown PaaS.

In summary, IT staff who are interested in using containers to develop next-generation cloud native applications can't go wrong with VxRail or VxRack SDDC, as they both cover just about any container app usage scenario and can be used by inexperienced and experienced operations teams alike.

Silverton Consulting
Strategy, Storage & Systems

[1] Please see https://hub.docker.com/explore/?page=1 as of 06Dec2017.

[2] Please see https://github.com/vmware/harbor as of 06Dec2017.

[3] Please see https://vmware.github.io/admiral/as of 01Jan2017.

[4] Please see https://github.com/cloudfoundry/bosh as of 07Dec2017.

[5] Please see https://github.com/kubernetes/kubernetes as of 07Dec2017.

[6] Please see https://12factor.net as of 11Dec2017.

[7] Multi-AZ support using multiple vSphere clusters is not supported on VxRack SDDC but is available on VxRail.