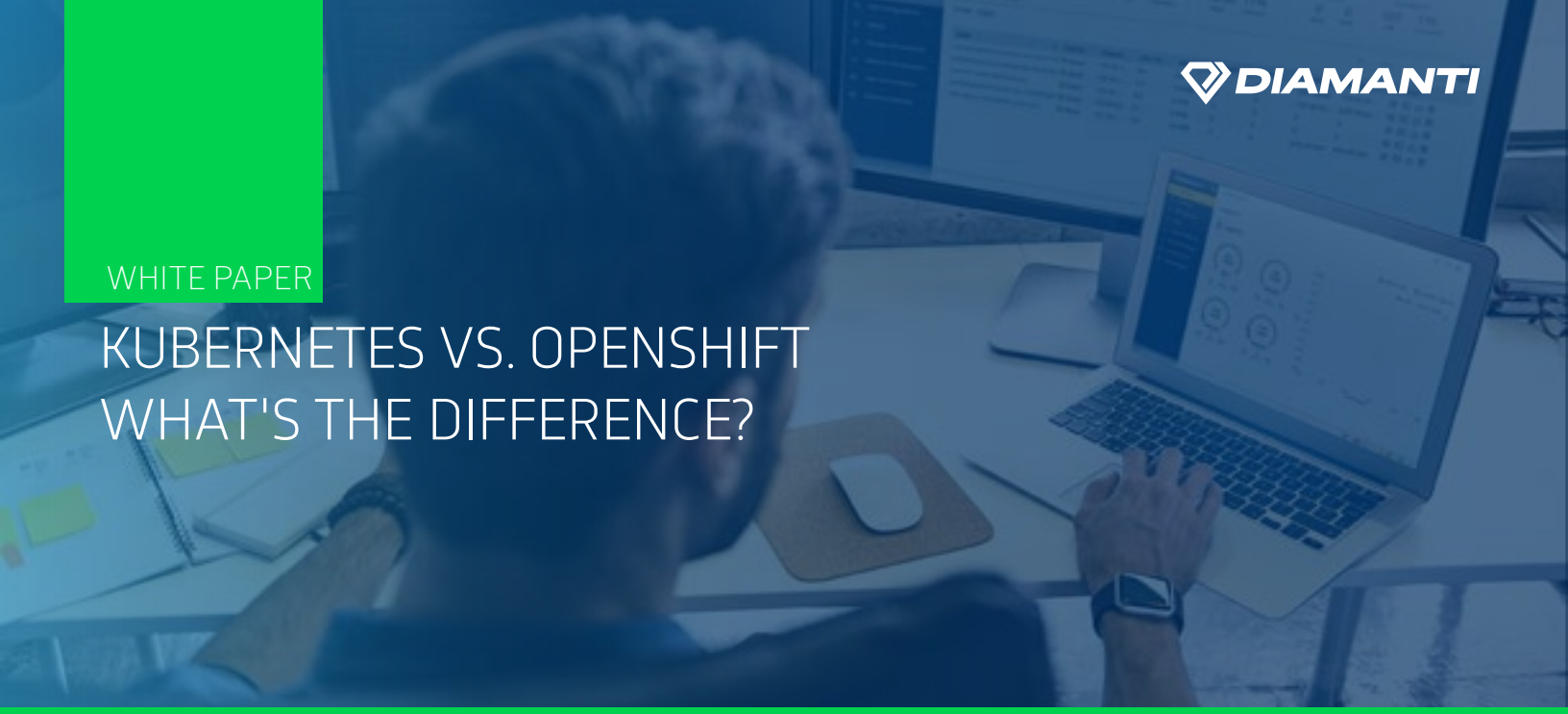

WHITE PAPER

KUBERNETES VS. OPENSHIFT WHAT'S THE DIFFERENCE?

Given the unprecedented growth of the container ecosystem in the span of just a few years, it's inevitable that there is still some confusion in the marketplace. If your team is adopting containers, you are probably working hard to understand all the options that are available to make the smartest choices for your needs.

At Diamanti, one of the questions we are asked most frequently is the difference between Kubernetes and the Red Hat OpenShift platform. The simple answer is that Kubernetes is a core component of OpenShift, which bundles additional open-source software tools with Kubernetes to create a more complete container solution.

WHAT IS KUBERNETES?

By itself, a container platform such as Docker is not enough to run a production container environment at scale. Orchestration is needed to provide a cluster environment with the necessary capabilities to run containers including:

- Deployment
- Management
- Scaling
- Availability

LEARN MORE:

The State of the Kubernetes Ecosystem from The New Stack is a good introductory reference if you want to learn more about Kubernetes.

Since it was released as open-source software by Google in 2014, the Kubernetes system has quickly become the most popular framework for container orchestration. Docker itself recently began offering native support for Kubernetes as an alternative to Docker Swarm orchestration.

Kubernetes was built specifically for orchestrating container environments and provides the foundation for development environments that are container-centric. By grouping the containers that make up an application or service into units called pods, Kubernetes simplifies policy-based management and discovery in container environments.

Services running on Kubernetes are discoverable by name. Multiple pods supporting a mix of different services can be distributed across a cluster of Kubernetes nodes under the control of a Kubernetes master.

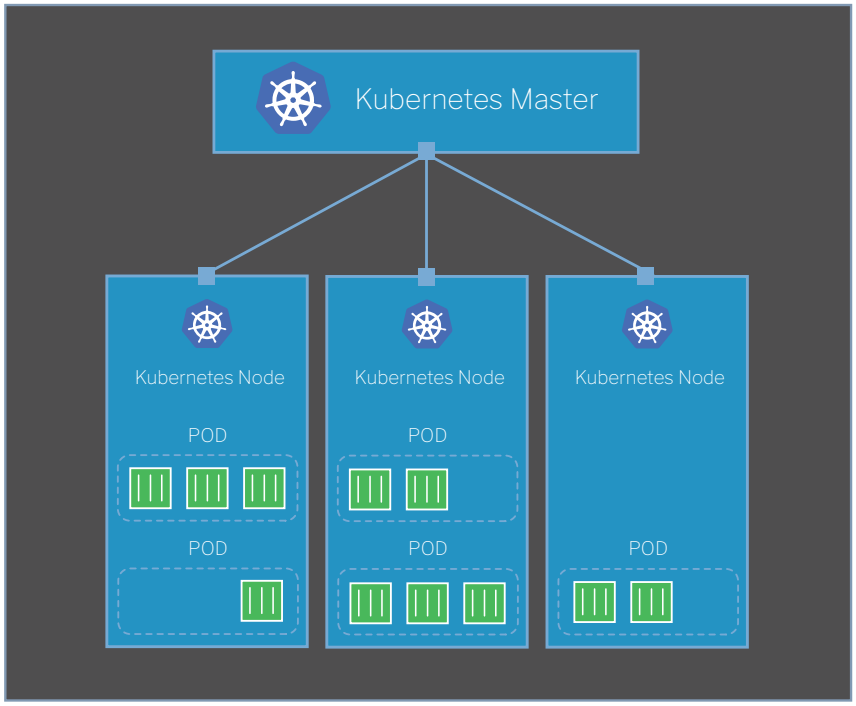

Figure 1: Kubernetes Master and Nodes

Kubernetes ensures that the pods you specify are kept running, with the desired number of instances of each pod. It also ensures that resource demands don't exceed the capabilities of the infrastructure, and regular "health checks" verify the status of running applications and services within the environment.

Many additional attributes contribute to the broad popularity of the Kubernetes framework. It is:

- Open and extensible with an extremely active community
- Built specifically for containers and associated workflows
- Tolerates failures well
- Automatically maintains desired state
- Scales to support large clusters

Although Kubernetes has been widely adopted, there are a few areas, in particular, that are the subject of continued development.

- Networking. Containers running within a cluster need to be able to communicate with one another. Client traffic from outside the cluster needs to be directed to the appropriate container(s) to service requests.
- Persistent storage. Because containers are ephemeral, special arrangements are needed to allow container instances to save and share data.

Packaged software distributions may add enhancements in these and other areas.

WHAT IS OPENSHIFT CONTAINER PLATFORM?

As is common with open-source software, several vendors have packaged Kubernetes into distributions that make it easier to get the software installed and running, with support provided by the vendor. One of the best known of these distributions is the Red Hat OpenShift Container Platform. In addition to Kubernetes, OpenShift includes Docker, Red Hat Linux, and a number of other software additions and enhancements for capabilities such as:

- Authentication (LDAP and SSO)
- Logging and metrics
- Software-defined networking (SDN)
- Router
- Web-Console
- Red Hat JBoss Middleware Service images and templates

- Source-to-Image (S2I), a tool for building and release management
- Jenkins for continuous integration and release management
- Integration with Eclipse, JBoss Developer Studio, and Visual Studio

These development tools can be useful in many environments, especially for those that are new to both containers and DevOps. However, if you already have a preferred development platform, these tools may not be needed. You could end up deploying them in locations where development is not taking place, with part of your maintenance fee going to the support of these tools whether you use them or not.

NETWORKING AND PERSISTENT STORAGE

For networking inside a Kubernetes cluster, OpenShift SDN creates an overlay network using Open vSwitch (OVS), enabling communication between containers in the cluster. Third-party SDN plugins are also supported.

OpenShift Routers take the place of Kubernetes Ingress Controllers. An HAproxy or F5 plugin provides load balancing and routing of external traffic destined for services running in the Kubernetes cluster.

For persistent storage, OpenShift does not provide anything beyond the persistent storage capabilities of Kubernetes: volumes, persistent volumes, and dynamic provisioning.

OPENSHIFT DEPLOYMENT

For on-premises deployment of OpenShift Container Platform you have two options: bare-metal or virtualized—typically using VMware. For optimum container performance and density, bare-metal deployments are considered the gold standard. However, for sites already running VMware, deploying containers on top of VMware may be a good starting point, allowing VMs and containers to run side-by-side and share the same infrastructure.

BARE METAL DEPLOYMENT

With bare metal, your team is responsible for selecting and deploying all the physical hardware needed—including servers, storage, and networking—and then installing all software.

FOR MORE INFORMATION:

The Diamanti white paper Choosing the Right Container Infrastructure for Your Organization discusses container infrastructure options in more detail.

For each host, you must first install Linux (Red Hat Enterprise Linux 7 or Linux Atomic Host 7), ensure the host is properly licensed and registered, install the required base packages, install Docker software, and configure Docker storage.

Once the hosts are ready, you install the OpenShift software, which includes Kubernetes, using either the quick installation or advanced installation option. Both approaches utilize Ansible to execute the necessary installation processes, introducing an additional open-source tool into your environment.

The quick install option only requires that you provide hostnames, IP addresses, and a few other details. The advanced install option provides fine-grained control, but it requires you to give significantly more thought to the environment you are going to install and configure.

With either option, there are a number of additional tasks that need to be carried out such as the configuration of:

- OpenShift container registry
- OpenShift router
- LDAP
- SDN
- Persistent storage

Therefore, you should expect the initial deployment process to be complex and take significant time. Some aspects of the install may require additional work to make the most informed configuration decisions. If your team is new to containers, staff resources are limited, or time is of the essence, a professional services engagement may be needed.

VIRTUALIZED DEPLOYMENT WITH VMWARE

Deploying OpenShift Container Platform on VMware is similar to a bare-metal deployment in many respects. Linux and Docker must first be installed inside VMs, which serve as Kubernetes nodes, taking the place of physical servers in a bare-metal install. For VMware deployments, OpenShift also defines two different types of nodes:

- Infrastructure nodes are used to run router and registry pods
- Application nodes run pods associated with applications and services

RedHat provides a 100-page reference architecture detailing the process of deploying and managing OpenShift Cloud Platform on VMware.

KUBERNETES WITH DIAMANTI

Enterprise IT teams face increasing business demands, including the need to introduce new digital services more rapidly. In response, many teams are working to streamline operations, move away from complex infrastructure deployment and management, and adopt a buy versus build approach.

Diamanti looked at the rapidly growing container ecosystem and saw an unfilled need for container infrastructure that is fast and easy to deploy. The Diamanti Container Converged Platform extends the benefits of hyperconverged infrastructure for a containerized world, providing fast installation, simple management, and bare-metal container, network, and storage resources integrated with Docker and Kubernetes. Diamanti provides:

- Docker and Kubernetes preloaded, preconfigured, and fully supported
- Traditional enterprise network interfaces without requiring a network overlay
- Dynamically provisioned, persistent, highly available distributed NVMe storage
- Quality of Service (QoS) with granular SLAs for compute, network, and storage
- Per-container reporting
- Role-based access control
- Supported upgrade process
- Self-service portal

Diamanti offers the only true turnkey, full-stack container solution on the market today, providing a cloud-like experience for companies that need to deploy containers on-premises quickly.

Ready to go out of the box and operations-focused, Diamanti appliances are for IT teams that don't want to install and troubleshoot open-source software. Diamanti provides full-stack support for Kubernetes, Docker Runtime, and the underlying Linux OS, in addition to Diamanti hardware. You no longer have to worry about which part of the stack is at fault before placing a service call.

We are also unique in providing developer-defined SLAs in Kubernetes for both networking and storage. Policies follow containers from development to production.

Diamanti networking provides:

- Seamless enterprise network integration
- Ability to use existing network services
- Easy network segmentation
- Per-container SLAs

Your existing enterprise networks and associated operational processes should operate smoothly when you adopt containers. Diamanti preserves the well-understood VM networking model in which each VM has its own virtualized network interface, MAC address, and IP address—or multiple interfaces if needed. By utilizing SR-IOV, each container can use DHCP and DNS just like a VM. Existing services such as load balancers and firewalls continue to work as expected.

Diamanti also provides predictable high-performance persistent storage with built-in SLAs via the Kubernetes FlexVolume plugin, which was developed and contributed to Kubernetes by Diamanti engineering. FlexVolume enables vendors to develop their own backend drivers and has enabled Diamanti to expose the full performance of our NVMe storage technology to applications running inside containers.

HOW DIAMANTI AND OPENSHIFT STACK UP

Both the Diamanti Container Converged Platform and OpenShift Container Platform are based on Docker and Kubernetes. Diamanti adds a high-performance hyperconverged infrastructure appliance that is highly available with distributed storage infrastructure and workload affinity to storage. With QoS for each workload, Diamanti can guarantee SLAs.

OpenShift requires an overlay network for all configurations. Diamanti plugs directly into existing enterprise networking and also supports overlay networks as needed. Diamanti supports multi-homing with per-network SLAs.

OpenShift may lock your team in with non-Kubernetes-specific objects, resources, concepts, and commands. Even though it is based on Kubernetes, OpenShift makes many changes on top of Kubernetes and requires additional education. This can create three additional problems:

- **Migration.** If you are running OpenShift, you may have a hard time migrating to a "pure" Kubernetes distribution.
- **Compatibility.** You may experience compatibility issues with open-source tools that work well with vanilla Kubernetes.
- **Hybrid cloud integration.** It may be difficult to federate an OpenShift cluster with other Kubernetes clusters. OpenShift doesn't officially support this yet.

Openshift does include some software services that Diamanti does not, such as developer tools and a CI/CD pipeline. However, when necessary these software requirements are addressed by Diamanti reference architectures, that specify how to deploy and run additional tools and third-party software. These reference architectures result in solution stacks that maintain compatibility with vanilla Kubernetes and your current practices.

	Red Hat OpenShift	Diamanti
Deployment model	Installed Platform	Appliance
Service and support	Software Only	Full Stack
Docker	√	√
Kubernetes	√	√
LDAP	√	√
Logging and metrics	√	√
Time to deploy	Weeks	Minutes
Management complexity	Medium	Low
Pre-installed software		√
Bare metal	√	√
Network QoS		√
Storage QoS		√
Web console	√	√
CI/CD pipeline	√	3rd-party Solutions
Other development tools	√	3rd-party Solutions

Table 1: Red Hat OpenShift compared to Diamanti Container Converged Platform

For more information, visit www.diamanti.com. To request a demo, email demo@diamanti.com.