Mastering Containers with OpenNebula

Version 1.2 – April 2021

Abstract

Application container technologies, like Docker and Kubernetes, are becoming the de facto leading standards for packaging, deploying, and managing applications with increased levels of agility and efficiency. Docker uses OS-level virtualization to deliver software in packages called containers, whereas Kubernetes is a widely used tool for the orchestration of containers on clusters. Although Kubernetes is a powerful tool, it doesn't necessarily work for every single use case nor does it solve all container management-related challenges an organization might face. Kubernetes is a very complex and demanding technology, and other open source alternatives may actually be the best solution for many use cases.

OpenNebula offers a simple but powerful approach for running containerized applications and workflows by directly using the **Docker official images** available from the Docker Hub and running them on **lightweight Firecracker microVMs** that provide an extra level of efficiency and security. This solution combines all the benefits of containers with the security, orchestration, and multi-tenant features of a solid Cloud Management Platform but without adding extra layers of management, thus reducing the complexity and costs, compared with **Kubernetes or OpenShift**. You can also run your containers on a cloud environment based on **LXC system containers** if you need full bare-metal performance and isolation is not a requirement. For those cases where Kubernetes is required or is the best fit, OpenNebula brings support for the deployment of Kubernetes clusters through a **CNCF-certified** Virtual Appliance available from the OpenNebula Public Marketplace or through the **K3s lightweight distribution** for resource-constrained and edge locations. Last but not least, OpenNebula also offers integration with other popular orchestration engines such as Docker Machine, Docker Swarm, and Rancher.

Contents

Glossary

ACL	Access Control List
CMP	Cloud Management Platform
K8S	Kubernetes
VDC	Virtual Data Center
VM	Virtual Machine
VMM	Virtual Machine Monitor

1. A High Level View of Containers

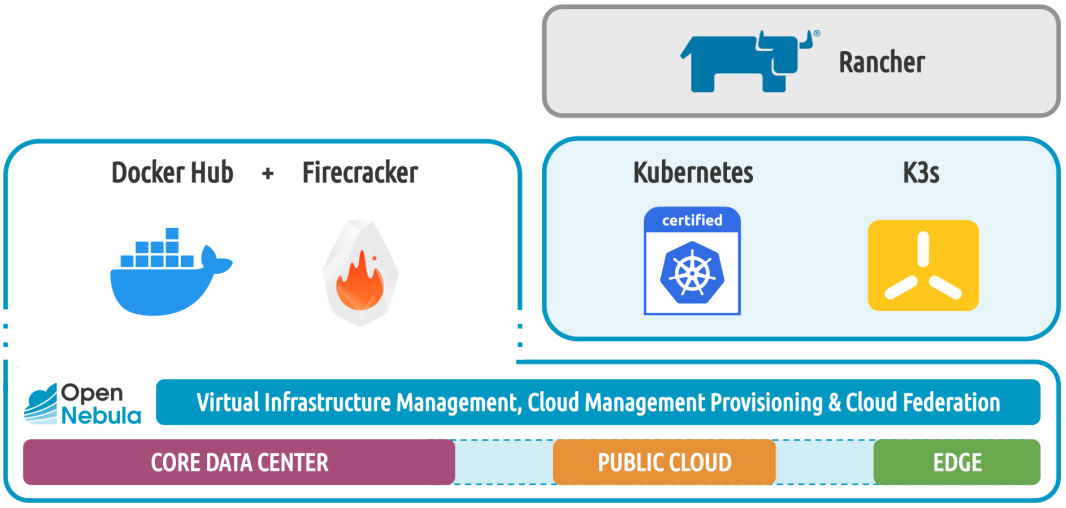


Figure 1. Available solutions for container orchestration with OpenNebula.

The Container Revolution

Information and Communication Technologies have evolved at a really fast pace over the past few years, dramatically changing the way information systems and applications are built. Software development has brought along many changes and revolutions, now allowing people to focus mainly on the business applications. The key drivers of this shift have been (1) the ability to **package and run applications anywhere** regardless of the underlying computing architecture, and (2) the means of **keeping applications isolated from each other** to avoid security risks and interference during operations or maintenance processes.

Keeping applications isolated on the same host or cluster can be difficult due to the packages, libraries, and other software components that are normally required to run them. Hardware virtualization was a solution to this problem, since applications could be kept apart on the same hardware by using Virtual Machines. Packaging an application within a VM also allowed it to run on any infrastructure that supported virtualization, which provided a lot of flexibility to the whole concept. However, Virtual Machines come with some serious limitations: moving them around is not that easy, since they are typically quite heavy and there are always difficulties associated with maintaining and upgrading applications running within a VM.

In recent years, we have all witnessed how container technologies have revolutionized the way enterprise and distributed applications are being developed and deployed. Containers clearly offer a more portable and flexible way of packaging, maintaining and running applications. They allow admins to deploy, move and replicate workloads more quickly and easily than using Virtual Machines. While containers as a concept have been around for a while, **Docker** was the technology that introduced several crucial changes to the existing container technology, making containers more portable and flexible to use. This resulted in a turning point towards the adoption of containerization and microservices in software development (e.g. cloud-native development).

Docker gave us an easy way to create container-based applications and to package them in portable images containing the specifications for the software components the container will run. Docker's technology brought cloud-like flexibility to any infrastructure capable of running containers, with their container image tools allowing developers to build libraries of images, compose applications from multiple images, and launch those containers and applications on local and remote infrastructures alike.

Orchestrating Containers

Nowadays, many companies have embraced a **cloud-native paradigm** in developing applications and have shifted from a "monolithic" approach to a microservice approach. While deploying a single container can be an easy task, things get a bit more complicated when deploying multi-container applications on distributed hosts given that in these cases a Docker Engine alone is not enough. This is where container orchestrators (like **Kubernetes** or **Docker Swarm**) play an important role in scheduling containers to run on different servers, moving containers to a new host when the host becomes unhealthy, restarting containers when they fail, managing overlay networks to allow containers on different hosts to communicate, orchestrating storage to provide persistent volumes to stateful applications, and so on.

However, container technologies (e.g. Docker, Kubernetes) also come with some serious limitations, such as for example **security** (application containers share the kernel OS) and **multi-tenant environments**. In order to provide a multi-tenant and secure environment to deploy containerized applications, one has to provision different "virtual environments" to each user or group of users, typically by deploying several isolated Kubernetes clusters on top of a Cloud Management Platform. The CMP is then responsible for managing and orchestrating the underlying virtual resources (i.e. Virtual Machines, virtual networks and storage) that are used by the different Kubernetes deployments that are in charge of scheduling application containers within those isolated environments. This approach **adds an extra control layer** that ends up increasing management complexity, resource consumption, and operational costs.

Running Containers on MicroVMs

But what if we could remove one of those layers? Cloud Management Platforms such as OpenNebula have been implemented with multi-tenancy and security by design, providing already powerful orchestration features but for VM-based applications (e.g. networking, storage blocks, high availability with live/cold migrations, etc.). Now **Firecracker** has been incorporated into OpenNebula as a new supported virtualization technology. This microVM technology, developed by **Amazon Web Services (AWS)** and widely used as part of its **Fargate** and **Lambda** services, has been especially designed for creating and managing secure, multi-tenant container and function-based services. By taking this step, OpenNebula has managed to **bridge the gap between two technological worlds**, leaving behind the old dilemma of whether to use containers—lighter, but with weaker security—or Virtual Machines—with strong security but high overhead.

By adopting the increasingly popular approach of running microservices based on containerized applications, and thanks to its **seamless integration with Docker Hub**, OpenNebula has now become a powerful alternative to deploy and orchestrate containers as **secure and fast Firecracker microVMs**. Section 2 of this white paper offers a general introduction to OpenNebula, while Section 3 provides insight into the technical foundations of **OpenNebula's new native model for container orchestration**.

No single size fits all and there are use cases that still require the running of a Kubernetes cluster, including those cases in which your target application has been defined as a Helm Chart and so you may need to use Kubernetes on your OpenNebula cloud. For those situations, as described in Section 4, OpenNebula provides support for the deployment of Kubernetes clusters through a **Virtual Appliance** available from the OpenNebula Public Marketplace. Other cases may require alternative orchestration engines like **Docker Swarm** or **Rancher**. The integration with these tools is described in Section 5 of this white paper.

2. What is OpenNebula?

OpenNebula is **a simple, open source solution to build and manage enterprise clouds** that combines existing virtualization technologies with advanced features for multi-tenancy, automatic provision, and elasticity to offer on-demand virtualized services and applications. It provides a single, feature-rich and flexible platform with **unified management of IT infrastructure and applications** that **avoids vendor lock-in** and **reduces complexity, resource consumption, and operational costs**.

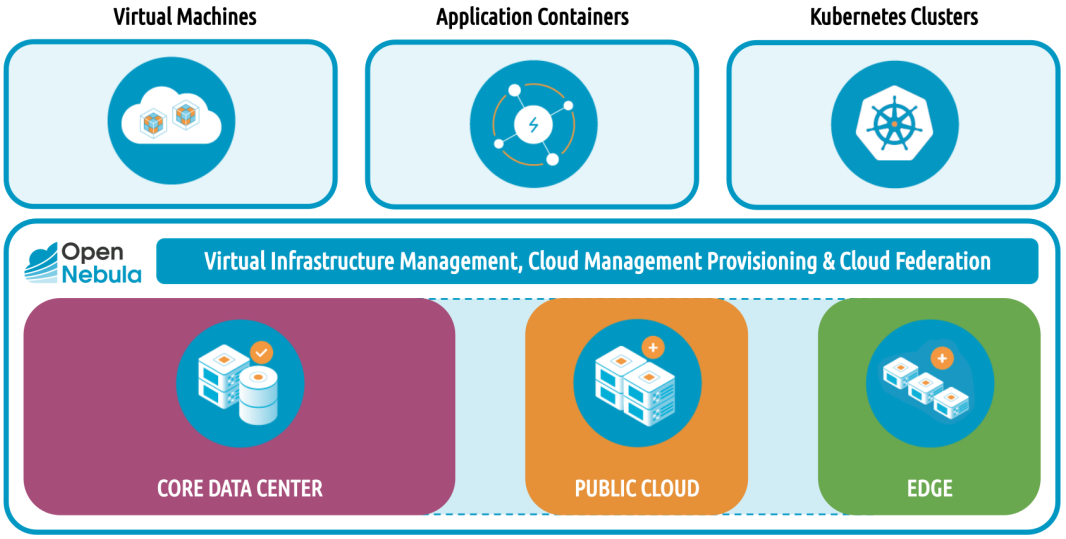


Figure 2. Available solutions for container orchestration with OpenNebula.

OpenNebula manages:

- **Any Application**: Combine containers with Virtual Machine workloads in a common shared environment to offer the best of both worlds: mature virtualization technology and orchestration of application containers.
- **Any Infrastructure**: Unlock the power of a true hybrid and multi-cloud platform by combining edge, public, hosted, and private cloud operations.
- **Any Virtualization**: Integrate multiple types of virtualization technologies to meet your workload needs, from a fully virtualized environment to system containers and serverless deployments.

OpenNebula provides the necessary tools for running containerized applications from Kubernetes and Docker Hub, while ensuring enterprise requirements for your DevOps practices. It helps organizations to easily embrace Hybrid and Edge Computing, allowing them to grow their Enterprise Cloud on-demand with infrastructure resources from third-party Public Cloud and bare-metal providers such as AWS, Microsoft Azure, and Equinix Metal. OpenNebula supports a number of virtualization technologies, including VMware and KVM Virtual Machines for fully virtualized clouds, LXC system containers for container clouds, and Firecracker microVMs for serverless deployments.

This white paper describes the new native features for container orchestration that are being developed by OpenNebula, and how it integrates with third-party technologies like Docker, Kubernetes and Rancher. If you are interested in designing and deploying an OpenNebula cloud on top of VMware vCenter, please refer to our VMWare Cloud Reference Architecture.[1] If you are interested in an OpenNebula cloud fully based on open source platforms and technologies, please refer to our Open Cloud Reference Architecture.[2]

[1] https://support.opennebula.pro/hc/en-us/articles/206652953-VMware-Cloud-Reference-Architecture-White-Paper
[2] https://support.opennebula.pro/hc/en-us/articles/204210319-Open-Cloud-Reference-Architecture-White-Paper

OpenNebula brings the provisioning tools and methods needed to dynamically grow a private cloud infrastructure on-demand with resources running on remote cloud and edge providers to enable powerful, true hybrid and multi-cloud computing, and support all major clouds. This disaggregated cloud approach allows a seamless transition from centralized private clouds to distributed edge-like cloud environments. Companies are able to grow their private cloud with resources at cloud and edge data center locations to meet peaks in demand or the latency and bandwidth needs of their workload. This approach involves a single management layer where organizations can continue using the existing OpenNebula images and templates, keep complete control over the infrastructure and avoid vendor lock-in.

OpenNebula allows the deployment of a fully operational **Edge Cluster**[3] in a remote provider and the management of its full life-cycle, starting with its provision and maintenance, until the unprovision. Each cloud or edge location (the "**provision**") is defined as a group of physical hosts allocated from the remote bare-metal or virtual provider. They are fully configured with the user-selected hypervisor and enabled in the cloud stack for the end-users.

Recommended Configurations for Application Container Workflows

Use Case	Hypervisor	Edge Cluster
Execute Docker Hub application containers on **system containers** within cloud **Virtual Machine instances**	LXC	VIrtual
Execute Docker Hub application containers on **system containers** within cloud **bare-metal servers** for **enhanced performance** over cloud Virtual Machine instances	LXC	Metal
Execute containers on **microVMs** within cloud **bare-metal servers** for **enhanced security and workload isolation** over system containers, while maintaining the speed and resource efficiency of containers	Firecracker	Metal

Recommended Configurations for Kubernetes Clusters

Use Case	Hypervisor	Edge Cluster
Execute **CNCF-certified k3s** lightweight kubernetes on **microVMs** within cloud **bare-metal servers** for resource-constrained and **edge locations**	Firecracker	Metal
Execute **CNCF-certified k8s** marketplace app on **VMs** within cloud **bare-metal servers**	KVM	Metal

The development of OpenNebula follows a bottom-up approach driven by the real need of sysadmins, DevOps and corporate users. OpenNebula is an open source product with a healthy and active community, commercially supported by OpenNebula Systems through its OpenNebula Subscription.[4] Releases are produced on a regular basis and delivered as a single package with a smooth migration path. More information on the benefits of running an OpenNebula cloud can be checked on the key features page.[5]

[3] https://support.opennebula.pro/hc/en-us/articles/360050302811-Edge-Cloud-Architecture-White-Paper
[4] https://opennebula.io/subscriptions/
[5] https://opennebula.io/discover/

3. Container Workflows with OpenNebula

The Best of Both Worlds

OpenNebula brings new exciting features to the container orchestration ecosystem by providing an innovative open source solution for organizations that need to build and manage a secure, self-service, multi-tenant cloud for serverless computing. Users of an OpenNebula cloud can now easily run isolated containers without the need to provision and manage servers or additional control layers, thus allowing them to focus on designing and building their applications instead of managing the underlying infrastructure. OpenNebula's pioneering approach towards container orchestration is based on the innovative integration of two main technologies:

- **Firecracker** as the Virtual Machine Monitor (VMM) that provisions and orchestrates microVMs.
- **Docker Hub** as the marketplace for application containers from which users can obtain and seamlessly deploy Docker images as microVMs.

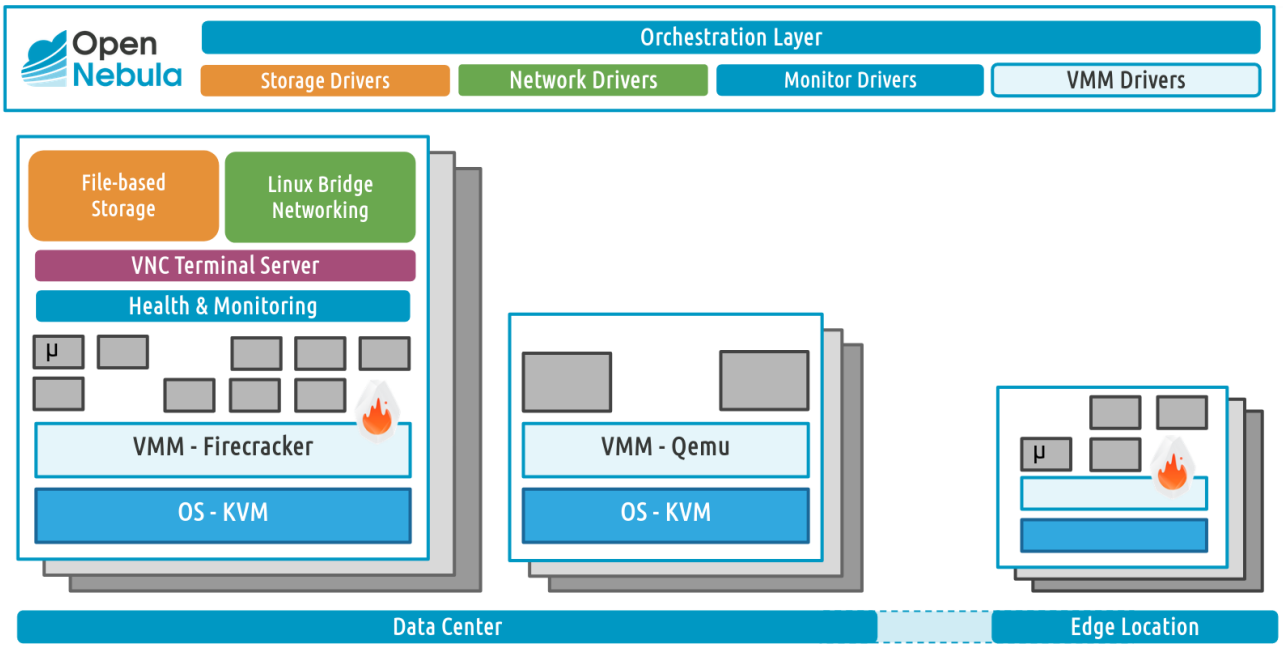


Figure 3. Integration of Firecracker as a new virtualization technology officially supported by OpenNebula.

Firecracker is an open source virtualization technology developed by **Amazon Web Services (AWS)** that makes use of KVM to launch lightweight Virtual Machines—called microVMs—for enhanced security, workload isolation, and resource efficiency. It is widely used by AWS as part of their Fargate and Lambda services. Firecracker opens up a whole new world of possibilities as the foundation for serverless solutions that need to quickly deploy critical applications as containers while keeping them in secure isolation. With the recent integration of Firecracker as a new supported virtualization technology, OpenNebula now provides an innovative solution to the classic dilemma of whether to use containers—lighter, but with weaker security—or Virtual Machines—with strong security but high overhead. You can also run your containers on a cloud environment based on **LXC system containers** if you need full bare-metal performance, and isolation is not a requirement.

OpenNebula's integration of **Docker Hub** as a new native marketplace provides users with immediate access to Docker Hub official images. Through this integration, Docker images can be easily imported into an OpenNebula cloud, following a process similar to that of OpenNebula's Public Marketplace. The OpenNebula context packages are installed during the import process so that, once an image is imported, it

becomes fully functional (including auto IP configuration, SSH key management, and custom scripts). The Docker Hub marketplace also creates a new VM template associated with the imported image; this template can then be customized (e.g. by adding the desired kernel, tuning specific parameters, etc.).

Ready for Launch? 🚀

Using the Docker Hub Marketplace to Deploy Container-based Applications

SCREENCAST + TUTORIAL

How to Use miniONE to Deploy a Firecracker Cloud Integrated with Docker Hub on AWS

SCREENCAST + TUTORIAL

An Enterprise Cloud of Containerized Applications

OpenNebula brings to the container world a series of unique features to build your own enterprise cloud for serverless computing. Its unique approach speeds up the deployment of containerized applications and multi-tier services within your development workflow. In particular, OpenNebula provides:

- **Fast startup times**: With OpenNebula you can start containers in seconds by deploying them as Firecracker microVMs, without the need to provision and maintain Dockerized hosts or complex Kubernetes infrastructures.
- **Direct access to applications**: Deploying a container as a microVM allows your container to directly access your networks through the Firecracker microVM's IP address. Furthermore, the microVM can be accessed via SSH and VNC by providing interactive modes that help with application development and troubleshooting.
- **Hypervisor-level security**: OpenNebula guarantees that your containerized application will run within a Firecracker microVM which, in turn, provides VM-grade security and isolation in a multi-tenant environment while preserving the efficiency of lightweight containers.
- **Persistent storage**: OpenNebula's datastores can be used as persistent data volumes for containers by attaching them to your Firecracker microVMs. Read-only config or data files can be provided by using the OpenNebula file datastore, which can be used by the application within the microVM.
- **Network access**: Virtual Networks defined within an OpenNebula cloud (i.e. IPv4, IPv6, Dual Stack, Ethernet) can be easily configured so that containerized applications get direct access to the internet without requiring additional components (e.g. ingress controllers, load balancers, etc.). With OpenNebula Virtual Routers it is also possible to connect different virtual networks, allowing applications and Virtual Machines attached to different virtual networks to communicate with each other. It is possible to use security groups for ensuring network security for container applications.
- **Application horizontal scaling**: Applications deployed as complex, multi-tier services can be scaled up and down "manually" but also (via the OneFlow component) in an automatic manner, based on user-defined metrics and pre-defined criteria. An init script can be defined to send application metrics to OneGate, OpenNebula's metadata server.
- **Complete multi-tenant environments** with ACLs, users, groups, resource UNIX-like permissions and VDCs, in which cloud admins can easily adapt OpenNebula to their organization's infrastructure and DevOps requirements, and set up independent sets of resources for specific purposes or groups of users (i.e. development, testing, integration, and production).

- **Geo-distributed applications**: Thanks to its elastic cloud infrastructure feature (OneProvision), OpenNebula allows users to build on-demand geo-distributed infrastructures for execution of containerized applications at the Edge.

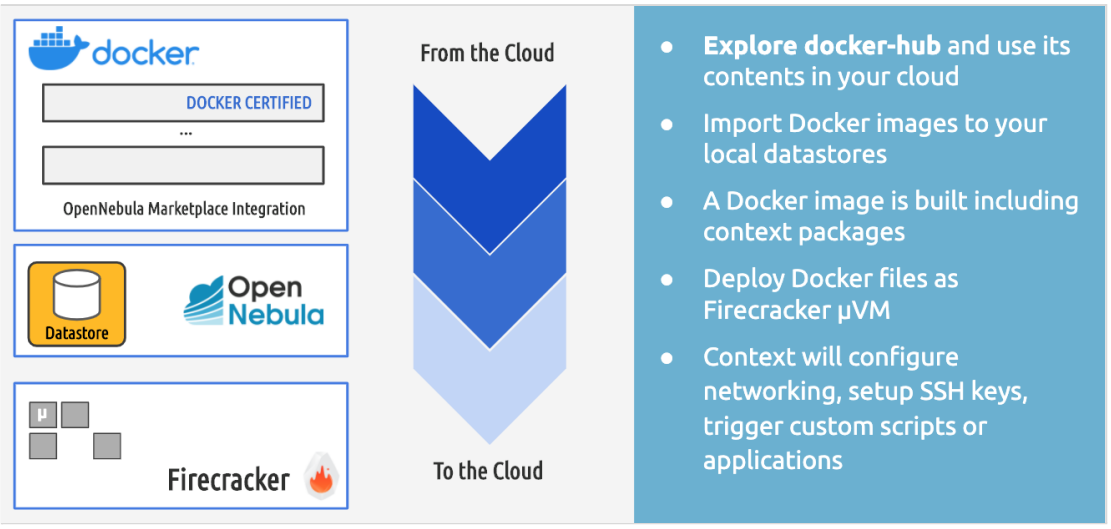


Figure 4. The process of deploying a Docker image from Docker Hub as a Firecracker microVM.

Orchestrating Your Containerized Apps with OpenNebula or Kubernetes?

Although many IT departments providing container execution services have decided to implement their DevOps requirements on top of Kubernetes, that doesn't mean that betting only on one horse is the wisest thing to do. Organizations requiring container orchestration capabilities should first of all state what their main objective is and be careful not to make apples-to-oranges comparisons or to fall into unexpected costs or vendor lock-ins. Kubernetes is a very complex and demanding technology, and—temporary fashions aside—other technologies may actually be the best solution for some use cases.

Some of the features provided by Kubernetes (such as its declarative model, self-healing, automated rollout and rollbacks, secret and configuration management, service discovery and load balancing) make it ideal for companies that need complete container orchestration services for the deployment and management of containerized workflows in a production environment. Yet, these organizations have to be able to cope with high operational costs if what they want is to build and manage a corporate Kubernetes deployment. Kubernetes support for multi-tenancy is actually quite limited and it cannot guarantee perfectly secure isolation between tenants. The only way to run Kubernetes is by providing different teams with their own clusters. Upgrading clusters and patching vulnerabilities is not a quick and easy task, which in the end requires building an expensive, full-time admin team. Moreover, Kubernetes does not offer cloud-like self-service provision features for users, nor accounting and advanced authorization features for administrators. Cloud providers and management tools, like Amazon Elastic Kubernetes Service (EKS) or Google Kubernetes Engine (GKE), try to bridge these gaps by offering managed Kubernetes-as-a-Service platforms. What these solutions do is to add an extra control layer that ends up increasing management complexity, resource consumption, and associated costs.

OpenNebula, on the other hand, is an ideal solution for companies that need to build multi-tenant Container-as-a-Service environments, but with lower operational costs. In this way, users and business units can develop and deploy applications easily and very fast, without their organizations having to manage Dockerized hosts or complex orchestration infrastructures such as Kubernetes or OpenShift. With the release of its version 5.12 "Firework", OpenNebula became a real alternative for implementing an agile and serverless cloud paradigm for containerized workloads in production environments.

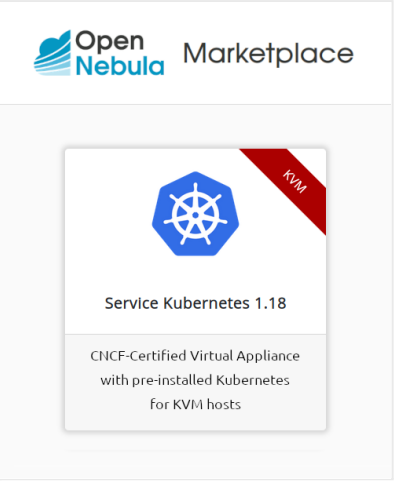
	Container as a Service	Container Orchestration
Purpose	Create a CaaS multi-tenant Enterprise Cloud for containerized applications	Manage a cluster of Linux containers as a single system to accelerate development and simplify operations
Use	Deliver shared resources to groups of users for secure execution of their container workloads	Deployment, scaling, and operations of containers across a cluster of hosts of VMs for a single user or group of users
Applications	Containerized distributed applications	Containerized distributed applications
Access to Applications	Easy SSH, VNC access to the microVM	Container exec bash access. Hard to troubleshoot
Orchestration Approach	Imperative	Declarative
Application Management	User-driven life-cycle application management (create, delete, stop, resume)	Application rollback and updates, self-healing, service discovery & load balancing
Application Scheduling & Resource Optimization	Application microVMs are placed according to resource requirements (CPU and Memory), affinity rules, custom heuristics…	Automatic bin packing is used to place containers based on their resource requirements
Network Access	Full support of IPv4, IPv6, Dual Stack, Ethernet networks and security groups	Allocation of IPv4 and IPv6 addresses to Pods and services
Persistent Storage	Datastore persistent data block images can be mounted (via start_script) as storage volumes for application	Local, network, and public cloud storage systems can be automatically mounted as Pod Volumes
Lifespan	Short-term	Long-term
Tenancy	Multiple tenants	Single tenant
Self-Provision	Existing catalogs like Docker Hub	Existing catalogs like Helm Hub (for Kubernetes-ready applications) and Docker Hub (for Pod containers)
Security	Hypervisor-level security for multi-tenant environments	Container-level isolation (shared kernel)

Table 1. Main differences between OpenNebula's native model and Kubernete-based solutions.

4. Using Kubernetes with OpenNebula

Running Your Kubernetes Clusters on OpenNebula

When companies need complete container orchestration services based on Kubernetes for the deployment and management of containerized workflows, OpenNebula provides them with a simple "press-of-a-button" option to create and deploy a fully functional Kubernetes cluster thanks to the **Kubernetes Appliance** available from the OpenNebula Public Marketplace.[6]



This virtual appliance supports multiple contextualization parameters to bend to your needs and to your required configuration. Simplicity and versatility have been enhanced by using the OneFlow[7] service so that the appliance can work as a:

- Single-node cluster
- Manually managed multi-node cluster
- Automatically managed multi-node cluster

This virtual appliance provides you with a Kubernetes cluster, including one master node and several worker nodes (these are optional, of course). Every node is then managed by OpenNebula as a regular VM (and you can always add more nodes to the cluster at any time using the contextualization process), but OpenNebula does not manage containers or pods inside the Kubernetes cluster. The Kubernetes cluster exposes the **Kubernetes API** (on a designated IP address of the master node) so that you can then access it via *kubectl* or *UI dashboard* to create pods, deployments, services, etc. Before deploying this Kubernetes Service, we recommend you to check out the associated documentation.[8]

Ready for Launch? 🚀

Kubernetes Appliance Demo

SCREENCAST

OpenNebula vs Kubernetes:
Comparing Two Container Orchestration Models

ARTICLE

Benefits of Kubernetes on OpenNebula

A leading-edge private cloud is not just about containers, and containers are not appropriate for all workloads and use cases. Hosting a virtualized container orchestration framework like Kubernetes on an OpenNebula cloud:

[6] https://marketplace.opennebula.io
[7] http://docs.opennebula.io/5.12/advanced_components/application_flow_and_auto-scaling/appflow_use_cli.html
[8] http://docs.opennebula.io/appliances/service/kubernetes.html

- Provides a flexible infrastructure environment with dynamic allocation and partitioning of physical resources, resizing of the virtual resources on the fly, or overprovisioning if necessary.
- Allows you to encompass containers with other virtualized workloads.
- Provides a multi-tenant environment for the execution of multiple container clusters on a shared physical infrastructure.
- Enhances security thanks to the additional layer provided by the hardware virtualization to isolate different resource pools (Virtual Machines) on the same host.
- Offers abstraction from physical hardware, allowing you to rebuild a whole container cluster from scratch without messing with the physical hosts. Grow your Kubernetes cluster with on-prem and remote bare-metal providers.
- Automatically deploys and manages multiple Kubernetes clusters across edge and cloud locations to enable truly multi-tenant and large-scale container orchestration.

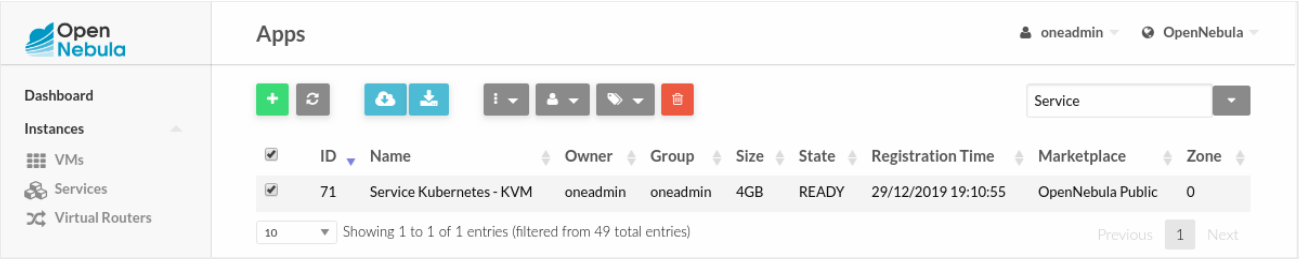


Figure 5. Downloading the K8s virtual appliance from the OpenNebula Public Marketplace.

Additionally, the Kubernetes appliance available from our Marketplace:

- Provides an auto-configured cluster with information exchanged over additional OpenNebula services (OneGate), managed as one entity (OneFlow).
- Dynamically increases/decreases the K8s cluster based on hypervisor and/or application metrics.
- Enables the provisioning of managed K8s clusters and application containers on demand with just one click. Easily deploys different architectures for different users and applications.
- Runs anywhere with a built-in configuration of components (e.g. networking) selected to deal with restrictions on the end-user side.

5. Other Approaches

You can use different container orchestration platforms in your OpenNebula cloud.

Docker Machine and Docker Swarm

The OpenNebula Marketplace includes an OpenNebula Docker Engine appliance that you can easily download and register in the cloud datastore. OpenNebula provides cloud users with two approaches to use the Docker Engine instances hosted by these virtualized Docker hosts:

- The simpler approach is to directly instantiate and access the OpenNebula Docker image, and manage the Dockerized hosts by using the Command Line Interface (CLI) or OpenNebula's WebGUI, Sunstone.
- OpenNebula also provides a driver for Docker Machine which allows the remote provision and management of Docker hosts, and the execution of Docker commands on the remote host from your Docker client.

A cluster of Docker Engines can be defined in OneFlow, and the autoscaling mechanisms of OneFlow can be used to automatically grow/decrease the number of Docker Engines based on application metrics. Another alternative is to use Docker Swarm, the native clustering for Docker. With Docker Swarm you can aggregate a group of Docker Engines (in this case, running as VMs in OpenNebula) into a single Virtual Docker Host. Docker Swarm delivers many advantages like scheduling, high availability, etc. Docker Swarm will make a cluster out of a collection of Docker Engine VMs deployed in OpenNebula with Docker Machine.

Rancher

Rancher is an open source software stack to run and manage containers and Kubernetes clusters while providing other integrated tools to enhance DevOps workflows. Rancher simplifies the creation of Kubernetes clusters as it allows you to create them through the Rancher's UI by dynamically provisioning nodes in cloud providers and then installing Kubernetes on them.

You can use the OpenNebula Docker Machine driver for creating Kubernetes clusters on OpenNebula using Rancher's UI. Using Rancher, you can create pools of nodes based on a node template that defines the parameters used to launch nodes in OpenNebula.

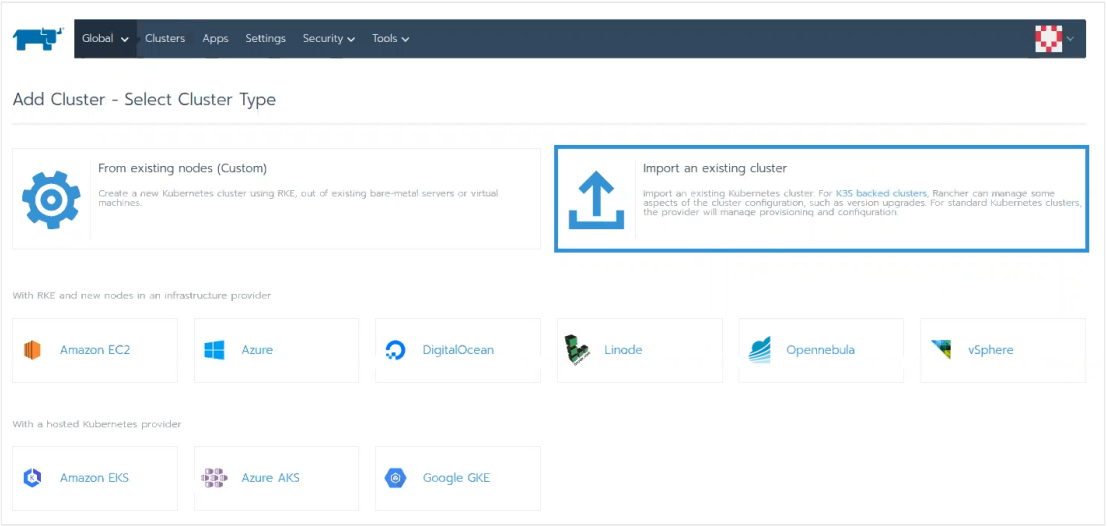


Figure 6. Using the OpenNebula Provider from Rancher's UI.

Once the OpenNebula driver is activated, you can proceed to create node templates and eventually node pools for your OpenNebula Kubernetes clusters using Rancher. Please refer to the documentation to see how to operate Rancher on top of your OpenNebula cloud to manage your K8s/K3s deployments.[9]

6. Ready for a Test Drive?

You can evaluate OpenNebula and build a cloud in just a few minutes by using **miniONE**,[10] our deployment tool for quick installation of an OpenNebula Front-end inside a Virtual Machine or a physical host, and then adding a remote Edge Cluster based on KVM, LXC or Firecracker.

7. Conclusions

There are different options to use containers with OpenNebula. We always recommend to start using our native integration, based on our seamless integration with **Docker Hub** and the new support to **Firecracker**, the virtualization technology based on microVMs developed by Amazon Web Services (AWS). This solution combines all the benefits of application containers with the security, orchestration, and multi-tenant features of a solid Cloud Management Platform, and all that without adding extra (and costly) layers of complexity like those of **Kubernetes or OpenShift**. You can also run your containers on a cloud environment based on **LXC system containers** if you need full bare-metal performance, and isolation is not a requirement. For those cases where Kubernetes is a requirement, OpenNebula also provides support for the deployment of **K8s/K3s clusters**. And last but not least, OpenNebula offers integration with other popular orchestration engines such as **Docker Machine**, **Docker Swarm,** and **Rancher**.

If you require any assistance in implementing any of these solutions within your organization or in adapting these technologies to your specific **DevOps** requirements, don't hesitate to reach out to our OpenNebula Services Team for consultation and support.

[9] https://docs.opennebula.io/5.12/advanced_components/applications_containerization/rancher_integration.html
[10] https://minione.opennebula.io

LET US HELP YOU DESIGN, BUILD, AND OPERATE YOUR CLOUD

CONSULTING & ENGINEERING

Our experts will help you design, integrate, build, and operate an OpenNebula cloud infrastructure

OPENNEBULA SUBSCRIPTION

Get access to our Enterprise Edition and to our support and exclusive services for Corporate Users

MANAGED SERVICES

Our team of experts can fully manage and administer your OpenNebula cloud for you

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement ONEedge 880412

ONEedge is an OpenNebula project developing innovative features to bring private cloud computing to the Edge (ONEedge.io)

Sign up for updates at OpenNebula.io/getupdated

© OpenNebula Systems 2021. This document is not a contractual agreement between any person, company, vendor, or interested party, and OpenNebula Systems. This document is provided for informational purposes only and the information contained herein is subject to change without notice. OpenNebula is a trademark in the European Union and in the United States. All other trademarks are property of their respective owners. All other company and product names and logos may be the subject of intellectual property rights reserved by third parties.

LINUX FOUNDATION SILVER MEMBER

CLOUD NATIVE COMPUTING FOUNDATION SILVER MEMBER

LF EDGE GENERAL MEMBER

GAIA-X data-infrastructure.eu

Rev2.2_20210420