StorPool
DISTRIBUTED STORAGE

Persistent storage for scalable **bare-metal Kubernetes**

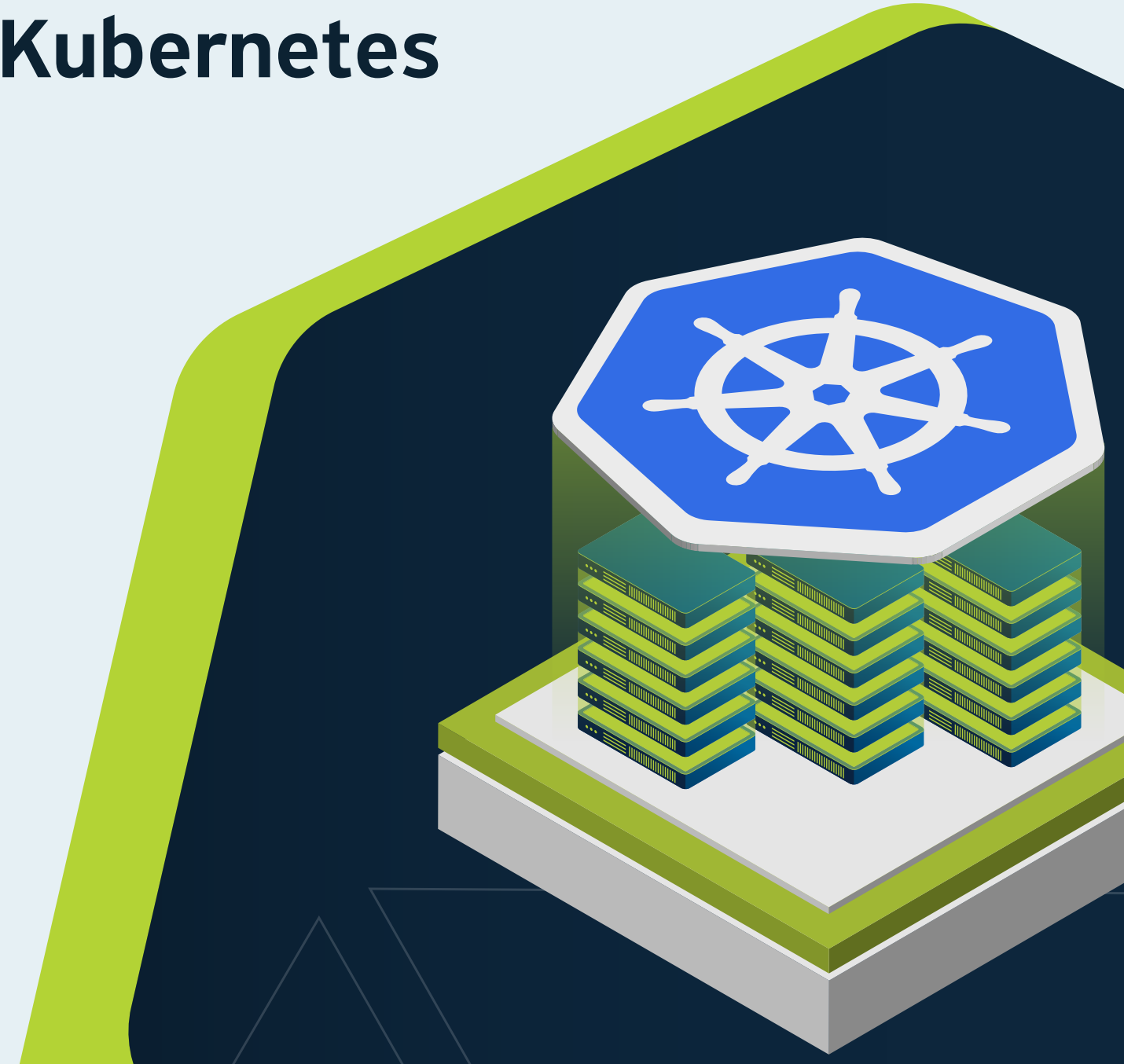


Table of Contents

Introducing Storage for Kubernetes

Kubernetes is the industry standard for deploying containers at a scale. With an unprecedented adoption rate, Kubernetes has become the foundation for modern infrastructure and New IT, as shown by the **CNCF 2019 SURVEY** - 78% of the respondents are using Kubernetes in production (up from 58% the year before).

As the ecosystem develops, users realize they need a persistent shared storage for their containers. This is what has led to the creation and massive adoption of Persistent Volumes and CSI (Container Storage Interface). This evolution has happened in more mature ecosystems already - think Amazon AWS or OpenStack, both of these started with local disks for the VMs/instances and then realized they need a persistent shared storage backend - and currently all workloads run in this manner.

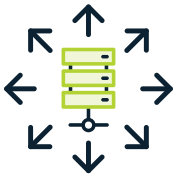
K8s follows the same learning curve, and eventually, persistent volumes will become the default way to run all container-powered workloads.

Challenges with Persistent Volumes

Even now, Persistent Volumes (PVs) are an essential part of the Kubernetes cluster. The expectations of the storage system, which provides PVs are, at least:

Reliable
Feature number zero

Modern SDS solution
No more hardware vendor lock-in



Distributed
No more single points of failure

Scalable
No more growth limits

Fast
Delivering tremendous performance,
to accelerate all containerized applications

In the public cloud, PVs are provided by the native service on the Public cloud of choice: EBS in Amazon ECS and Persistent Disk in Google Cloud's GKE. However, many companies reach a scale or have the need to run Kubernetes clusters outside of the public cloud and create on-prem K8S deployments. In this case, they need a fast, reliable, and scalable SDS.

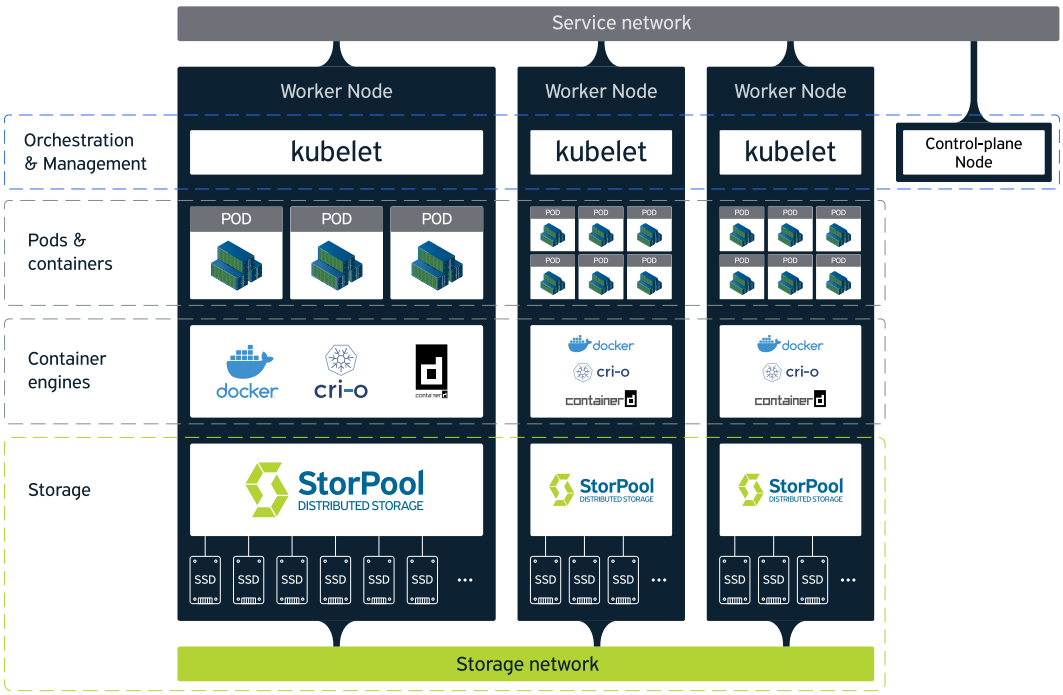
Moreover - not every use-case is suitable to be run in public clouds. Sometimes the data operational costs (upload, keep, download) could be very high like in **NASA forgot about the egress costs for its 247 petabyte data store on AWS** case. In other cases, the usage of public cloud services could be unacceptable or impossible due to regulations and local laws. Even, the service limitations (e.g. max number of CPU cores, RAM amount, etc.) could be a show-stopper.

StorPool's Solution

Meet StorPool: a best-in-class SDS, which allows you to re-create the public cloud stack in your datacenter and meets all the above core requirements of a modern Persistent Volumes storage for Kubernetes. Further, StorPool provides many unique advantages:

- Modern yet proven SDS storage solution;
- Fastest SDS on the market **with millions of IOPS and local SSD latency**;
- The most advanced **End-to-end data integrity** feature on the market;
- Efficient Copy-on-write snapshots and clones;
- Multi-stack/multi-platform support;
- Horizontal and vertical scalability;
- Built-in Backups and Disaster Recovery;
- Advanced Monitoring and Statistics;
- And much more.

With StorPool, as the storage layer, you now have the tools to create your own, on-premise Kubernetes clusters, outside of AWS or Google Cloud.

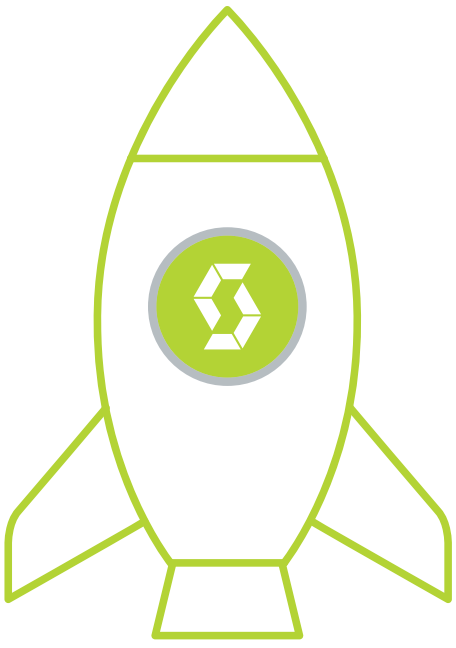


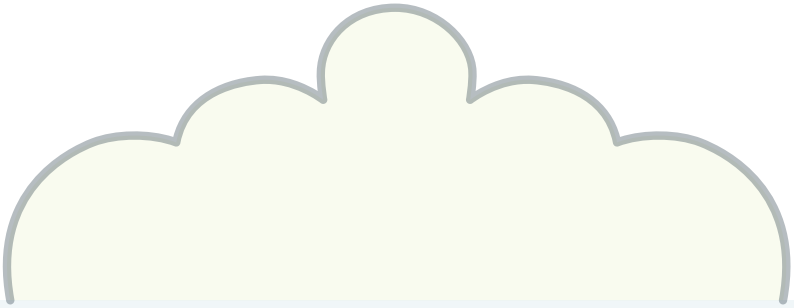
And it's simple to build such a K8S cluster!

Deploying StorPool

The installation of StorPool is effortless and straightforward - drop us an email and we will perform all the necessary steps. The whole process includes validation and test of the chosen hardware, software deployment with all dependencies and configurations, and all fine tunings and optimizations after, to achieve maximum performance of the cluster. StorPool is not just a piece of software, but a fully-managed storage solution, with included active monitoring and on-line non-disruptive upgrades. For more information, visit **StorPool Software-Defined Storage for building Private Clouds**.

For redundancy and optimal performance, StorPool requires two independent high-throughput Ethernet networks (10/ 25/ 40/ 100 Gbps). The redundancy scheme for the storage traffic is integrated into StorPool and there is no need for complicated network designs to achieve it. The management of the storage cluster could be through dedicated VLAN, on top of that Ethernet networks. The same scheme, or even the same VLAN, could be used for Kubernetes management too. For more details, see the network diagram in **StorPool Pre-installation Checklist**.





Initial Kubernetes Setup

For isolation and security, the Kubernetes control plane is installed in a VM, hosted on the same cluster. The virtual drive of that VM is backed by dedicated StorPool volume. The high-availability of the control plane VM is achieved by StorPool daemons which periodically verify the status of the control-plane VM. In case of a need, the VM can be live-migrated to any available server.

In case of hardware failure or if the physical host, where the VM runs, crashes, the VM control-plane will be redeployed and started on one of the other available nodes in a matter of seconds (the time needed to boot the VM). Because the drive of the VM is backed by a persistent StorPool volume, the control-plane data will be intact.

With control-plane VM prepared, the installation of Kubernetes can be performed in several ways.

Steps to deploy a Kubernetes cluster using kubeadm tool are:

- On each physical server and in control-plane VM - install Docker following the instructions for **CentOS/RHEL 7.4+** or **Ubuntu 16.04+**
  Note: StorPool uses cgroupfs for control groups which is the default for Docker too. Because of this, the configuration option `native.cgroupdriver=systemd` should be removed or replaced with `native.cgroupdriver=cgroupfs` in /etc/docker/daemon.json

- On the control-plane VM - install the kubeadm toolbox following the **Installing kubeadm instructions**.
  Note: When using Docker, kubeadm will automatically detect the cgroup driver for the kubelet. To verify:

```
# fgrep cgroupDriver /var/lib/kubelet/config.yaml
...
cgroupDriver: cgroupfs
...
```

- To finish the deployment, follow the procedure described on **Creating a single control-plane cluster with kubeadm** page.

An alternative option to deploy a Kubernetes cluster on bare-metal servers is to use Kubespray. The procedure is described in detail in **Installing Kubernetes with Kubespray**. For the current case, just check and verify that:

- Container runtime is configured to be Docker
- Docker is configured to use cgroupfs for control groups

Extend Kubernetes Cluster with StorPool Storage

The next step, after the Kubernetes cluster is up and running, is to add the StorPool CSI plugin. The procedure can be found in **StorPool Kubernetes Integration** article in the public **StorPool Knowledge Base**. Basically the steps are:

1. Create ServiceAccount, ClusterRole, and ClusterRoleBinding for the StorPool provisioner. An example configuration could be found in a server with StorPool installed, in */usr/share/doc/storpool/examples/kubernetes/storpool-provisioner.yaml* file.

2. Create a configuration for the StorPool management daemon for authorization in the Kubernetes API service of a cluster with `$CLUSTER_NAME` . The steps are covered in **StorPool Kubernetes Integration**.

3. Add the Kubernetes cluster in StorPool using CLI

```
# storpool kubernetes add name $CLUSTER_NAME
```

4. On all Kubernetes workers, enable and start StorPool's external-attacher

```
# systemctl enable --now storpool_kubcsi.service
```

At this point, the new bare-metal Kubernetes cluster with StorPool is ready to use. Kubernetes' Persistent Volumes are backed by StorPool volumes, which are presented as block devices to the workers.

StorPool as Kubernetes Storage Provider

StorPool is a multi-tier storage system which means that in the same cluster and hardware it is possible to create multiple Kubernetes Storage-Classes with a different kind (e.g. HDD or SSD/NVMe) - either through using different drives or through enabling IOPS and MB/s limits, e.g. storage **Quality of Service (QoS) features**.

Each Kubernetes Storage Class is created in a separate StorPool template. More about StorPool volumes and templates can be found in the **StorPool User Guide** (sections **Volume Management** and **Templates**).

To start using this new Kubernetes cluster:

- Create a StorPool template for each Storage Class. For example - to create a template placed on all NVMe drives:

```
# template $TEMPLATE_NAME replication 3 placeAll nvme
```

- Create and apply a Storage Class object backed by this template:

```
# cat storageClass.yaml
...
kind: StorageClass
metadata:
  name: $STORAGE_CLASS_NAME
provisioner: csi-driver.storpool.com
parameters:
  template: "$TEMPLATE_NAME"
...
```

- Create Persistent Volume Claim from this StorageClass:

```
...
kind: PersistentVolumeClaim
metadata:
  name: $PVC_NAME
spec:
  storageClassName: $STORAGE_CLASS_NAME
...
```

- Deploy a Pod that consumes a Persistent Volume from the Claim:

```
...
kind: Pod
spec:
  Volumes:
    claimName: $PVC_NAME
...
```

When these configurations are applied, a new StorPool volume will be created and attached to the node where Kubernetes schedules the Pod to run, and the volume will be mounted in the Pod when it starts.

For brevity, a lot of object settings are skipped in the above example. A more comprehensive usage example, with all details, can be found at **StorPool Integration with Kubernetes - Usage Example**.

Additional Benefits

By following this guide, it is easy to create your own on-premise Kubernetes cluster with StorPool, as a powerful persistent volumes backend. This will protect your data against data loss and ensure data is accessible wherever the containers are and regardless of what hardware failures occur in the cluster. On top of this, StorPool provides a wide set of **data management features**, matching a high-end SAN / All-Flash Array storage system.

This example is for the creation of a new hyper-converged cluster, but the storage from StorPool could be added to existing infrastructure, as a stand-alone storage system too.

Managing new-age IT and modern infrastructure requires multi-platform support. While providing persistent volumes for Kubernetes, StorPool can also present storage to multiple different IT platforms and cloud management systems, such as OpenNebula, OpenStack, VMware, Hyper-V, and more.

About StorPool

StorPool is the fastest software-defined block storage on the market, used by public and private cloud builders, enterprises, MSPs, SaaS, hosting, and cloud providers. It comes as software, plus a fully-managed data storage service that transforms commodity hardware into a fast, highly available, and scalable shared-storage system.

If you have any questions & comments or would like to learn more about StorPool's Persistent Volumes - do contact us at info@storpool.com.

Get in Touch

+1 415 670 9320

+44 (0) 20 7097 8536

info@storpool.com

sales@storpool.com

Get Started | Book a Demo