

---

# How to manage stateful applications in Kubernetes

Devoxx France - 2018

---

# Florian Woerner

DevOps Engineer / Cloud Architect

Onmyown

@woernfl

[florian.woerner@onmyown.io](mailto:florian.woerner@onmyown.io)





---

# Agenda

- Context
- Volume provider
- Volumes lifecycle management
- Demo
- Conclusion



---

# Why should I choose Kubernetes over VMs?



**Because Kubernetes is a  
distributed system  
management framework**





---

# Why would we need to manage data persistence?

**Because, stateless  
architecture... it does not exist**

---

---

# Types of volumes

---

# Types of Volumes - Kubernetes Internals

- configMap
- downwardAPI
- emptyDir
- gitRepo
- hostPath
- local
- persistentVolumeClaim
- projected
- Secret
- csi



---

# Types of Volumes - Public Cloud

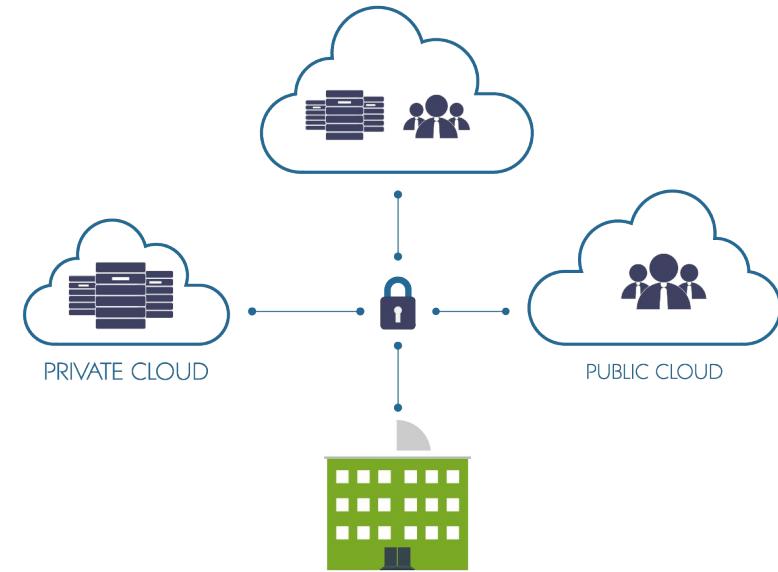
- awsElasticBlockStore
- azureDisk
- azureFile
- gcePersistentDisk



---

# Types of Volumes - Non Cloud Dependant

- cephfs
- fc (fibre channel)
- flocker
- glusterfs
- nfs
- iscsi
- portworxVolume
- quobyte
- rbd (Rados Block Device)
- scaleIO
- storageos
- vsphereVolume





---

# Any issues?



# CSI to the rescue





# What is CSI?

- Container Storage Interface
- Adopted by Kubernetes, Mesos and Cloud Foundry (for the moment)
- Supported as beta in Kubernetes 1.10 (planed to fall in stable for 1.12)
- Is a standardized way of exposing storage to a container



# Why is it so important?

- Common interface to multiple platforms (standardisation)
- Allow storage provider to release out of tree

---

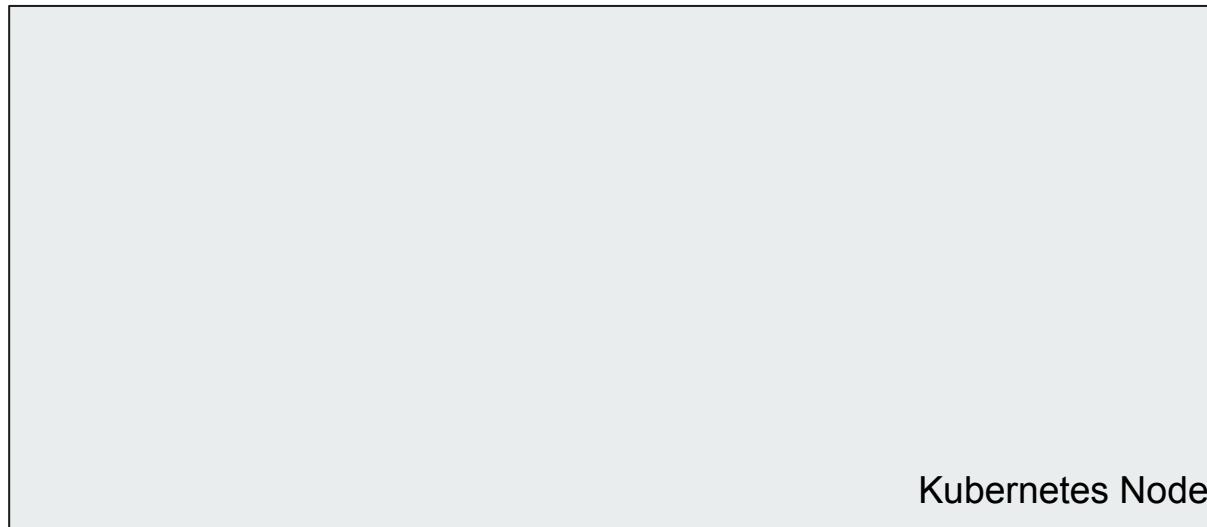
Real world scenario with **Bob** the dev and **John** the Sysadmin

---

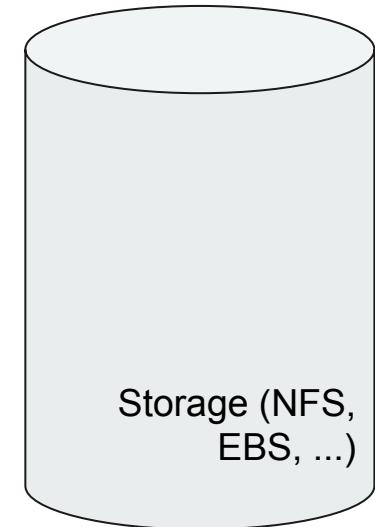
# Scenario: Persistent Volumes and Claims

---

## Scenario: Persistent Volumes and Claims



Kubernetes Node

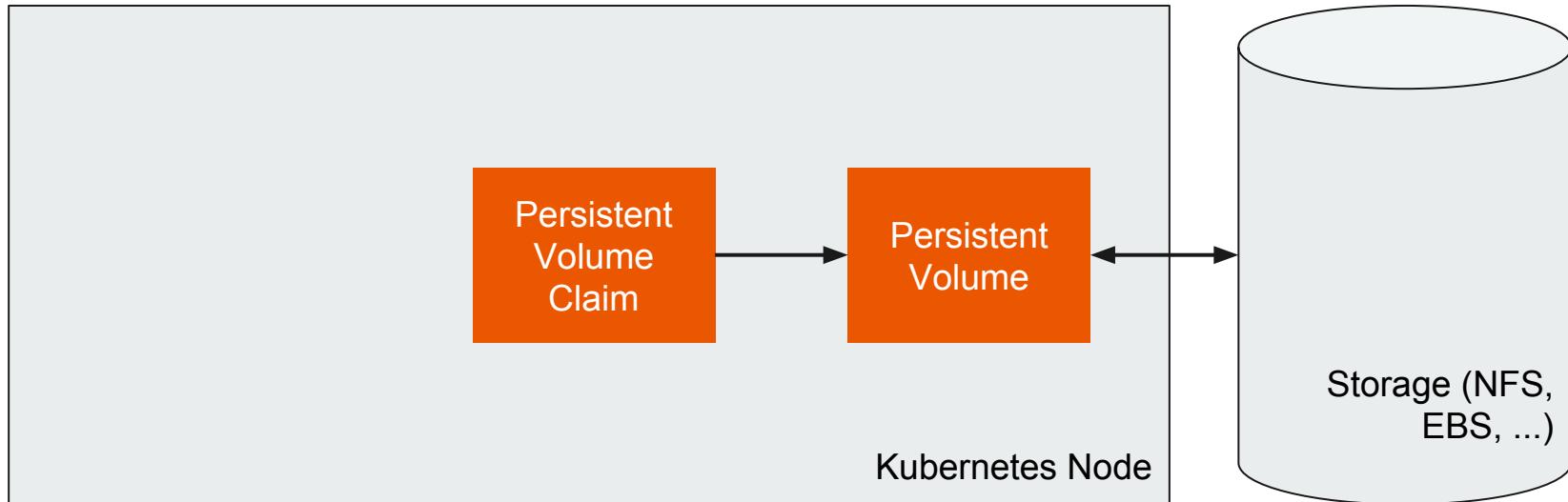


Storage (NFS,  
EBS, ...)

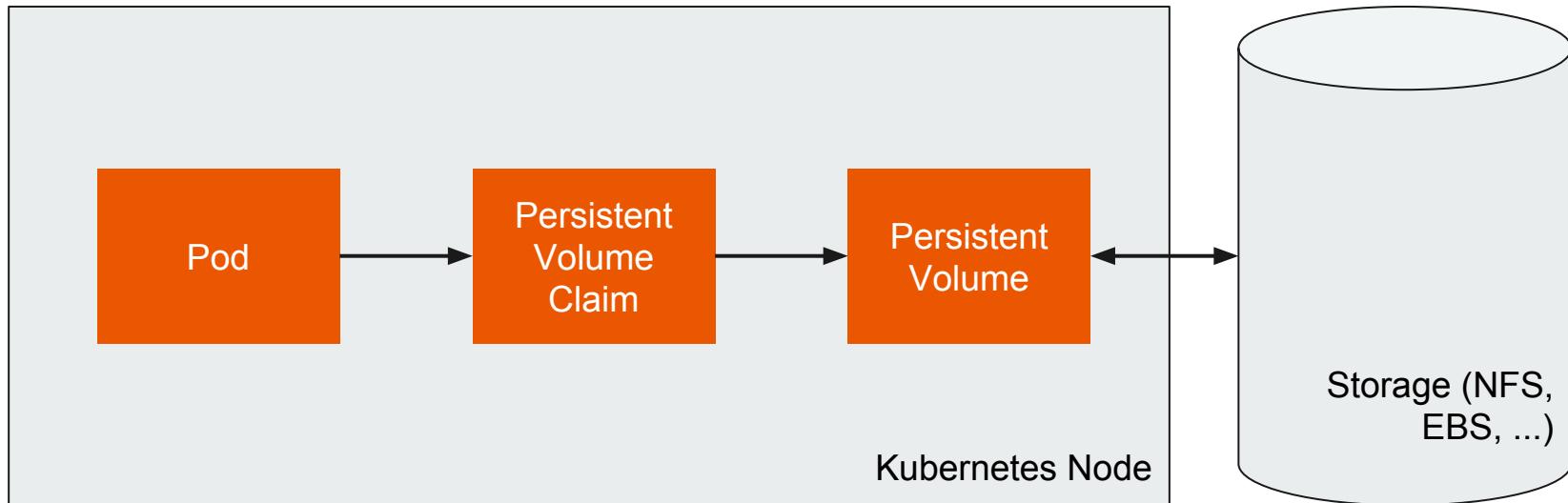
## Scenario: Persistent Volumes and Claims



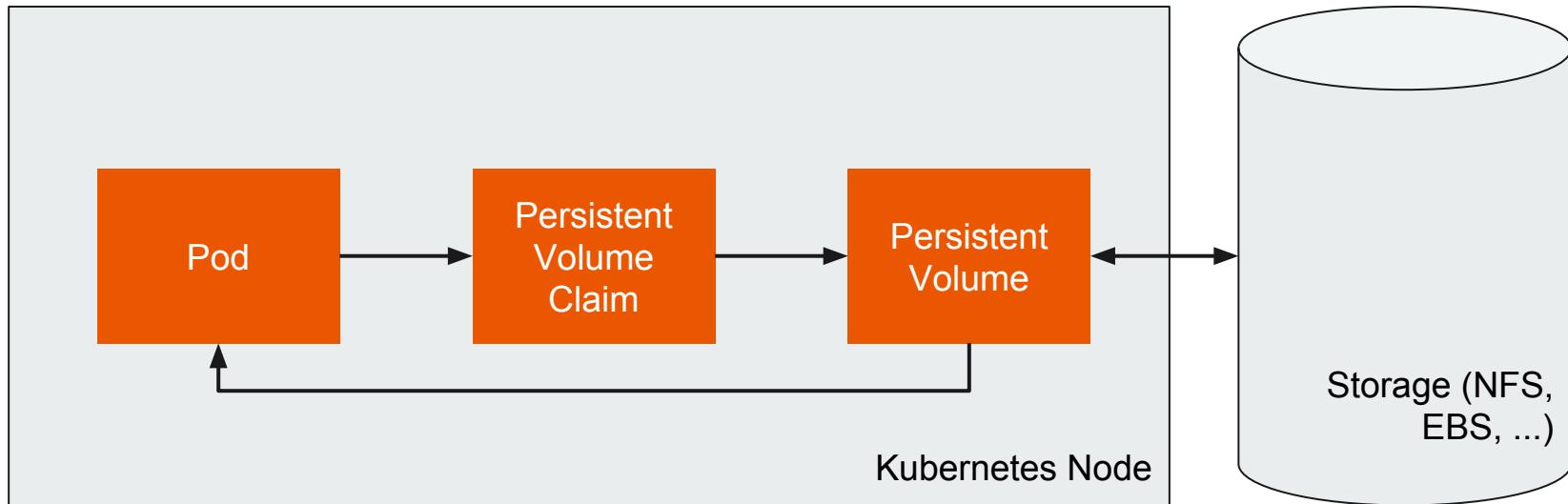
# Scenario: Persistent Volumes and Claims



# Scenario: Persistent Volumes and Claims



# Scenario: Persistent Volumes and Claims

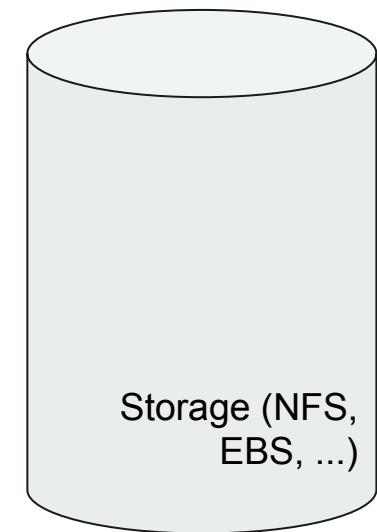
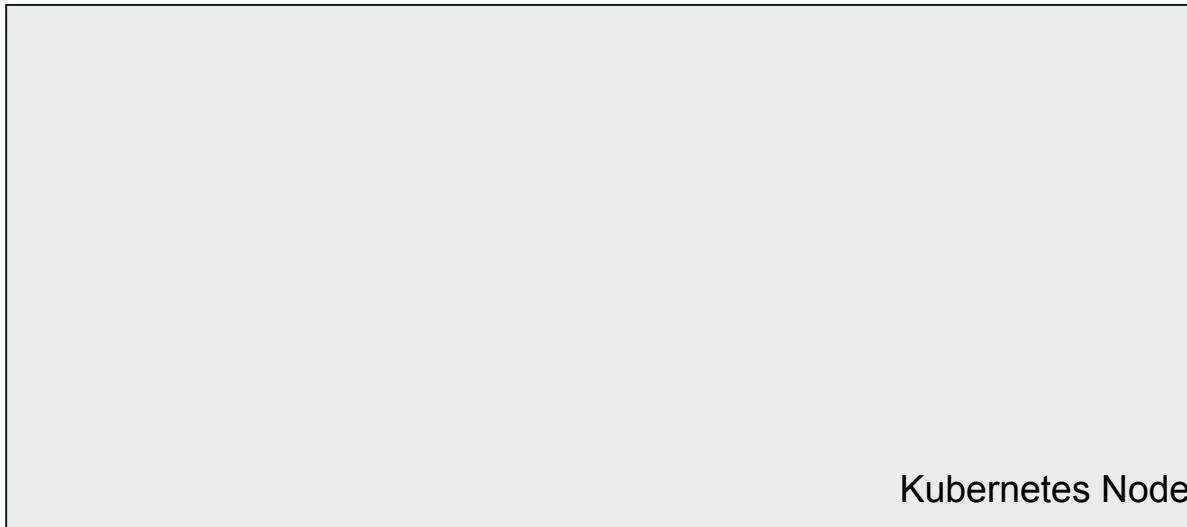


---

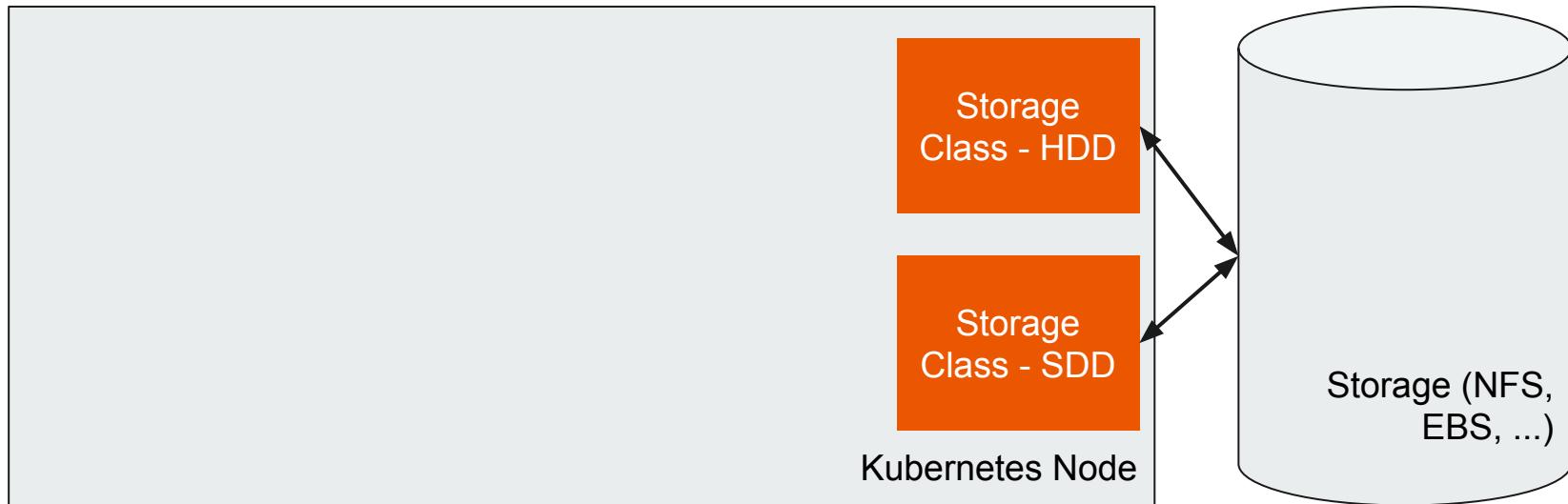
# Scenario: Dynamic Provisioning

---

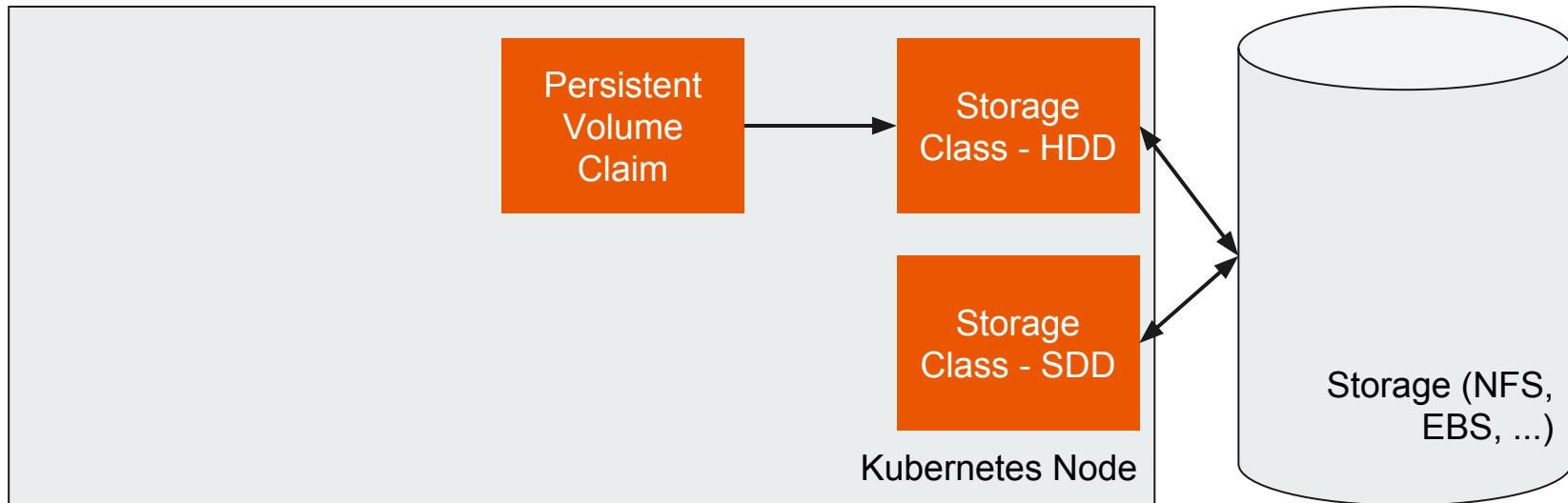
## Scenario: Dynamic Provisioning



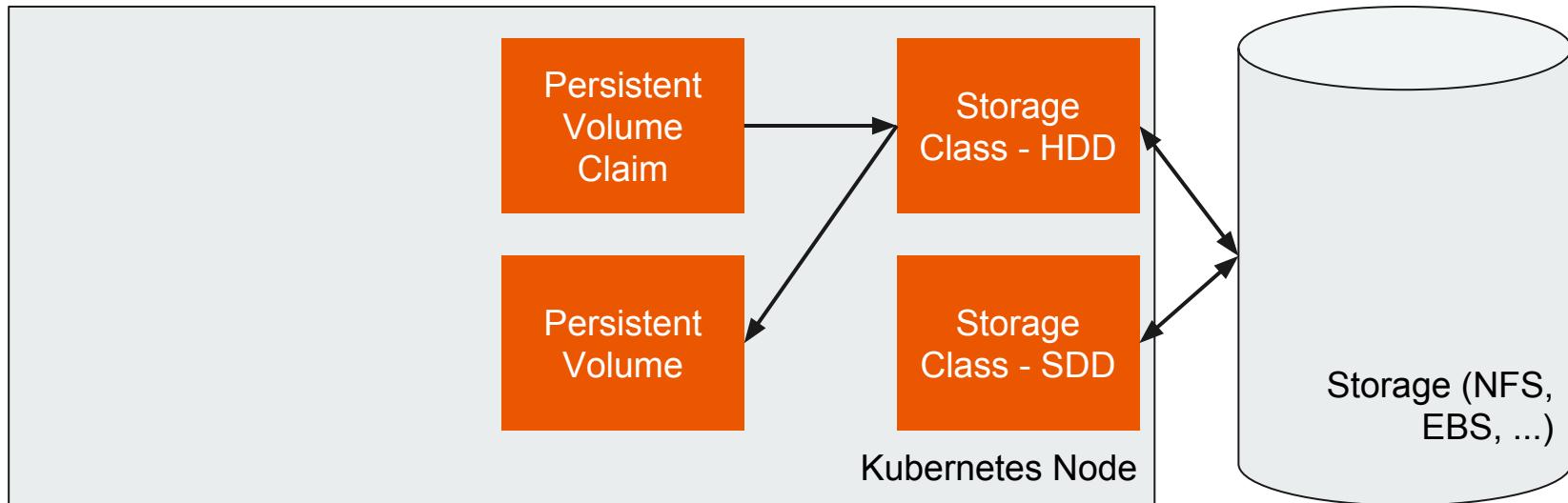
# Scenario: Dynamic Provisioning



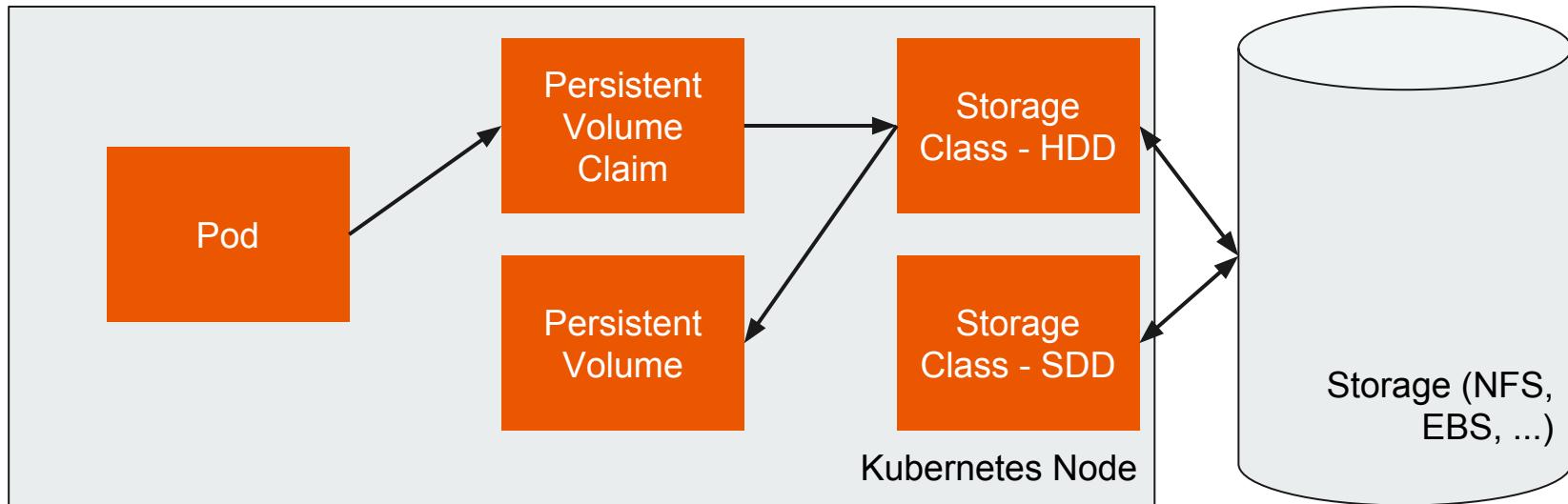
# Scenario: Dynamic Provisioning



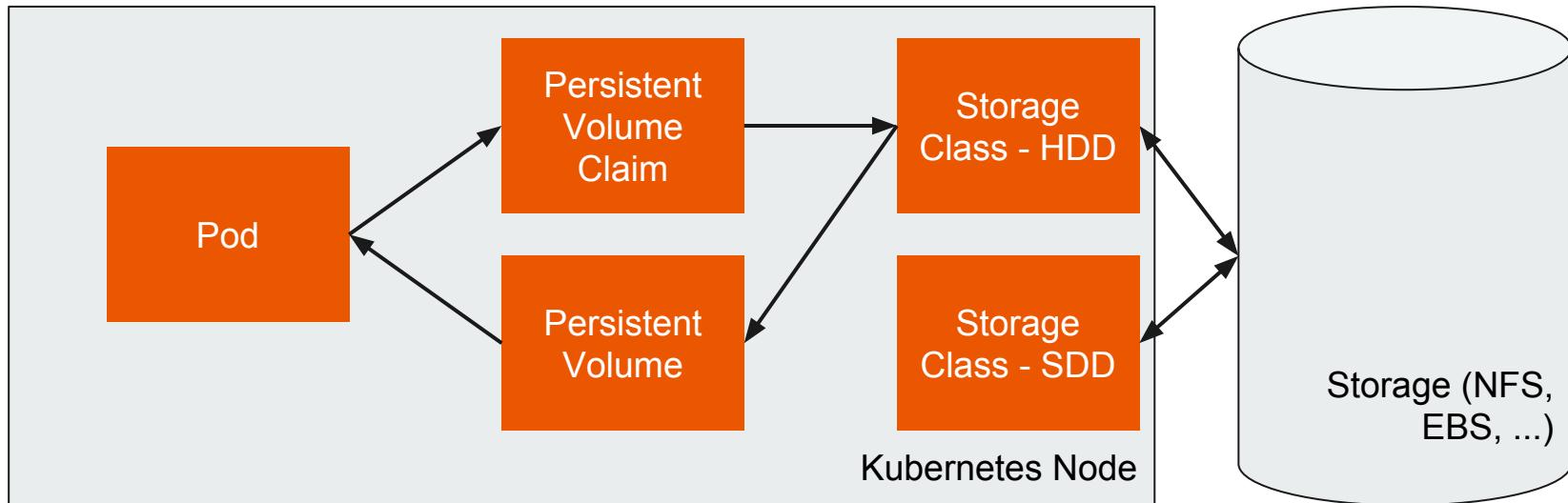
# Scenario: Dynamic Provisioning



# Scenario: Dynamic Provisioning



# Scenario: Dynamic Provisioning



---

# Demo



---

# Conclusion

---

# References

- Volumes Kubernetes Doc: <https://kubernetes.io/docs/concepts/storage/volumes/>
- Kubernetes Tasks: <https://kubernetes.io/docs/tasks/>
- Configure a Pod to Use a PersistentVolume for Storage:  
<https://kubernetes.io/docs/tasks/configure-pod-container/configure-persistent-volume-storage/>
- Run a Single-Instance Stateful Application:  
<https://kubernetes.io/docs/tasks/run-application/run-single-instance-stateful-application/>
- Run a Replicated Stateful Application:  
<https://kubernetes.io/docs/tasks/run-application/run-replicated-stateful-application/>
- StatefulSets Kubernetes Doc:  
<https://kubernetes.io/docs/concepts/workloads/controllers/statefulset/>

---

# Thank you!

Link to the repos: <https://gitlab.com/woernfl/k8s-stateful-demo>

