



Kubernetes Installation and Administration Course

Upscaling e High availability

Francesco Sinisi (francesco.sinisi@cnaif.infn.it)

IaaS side Improvements

How to increase the resilience of machines belonging to the cluster



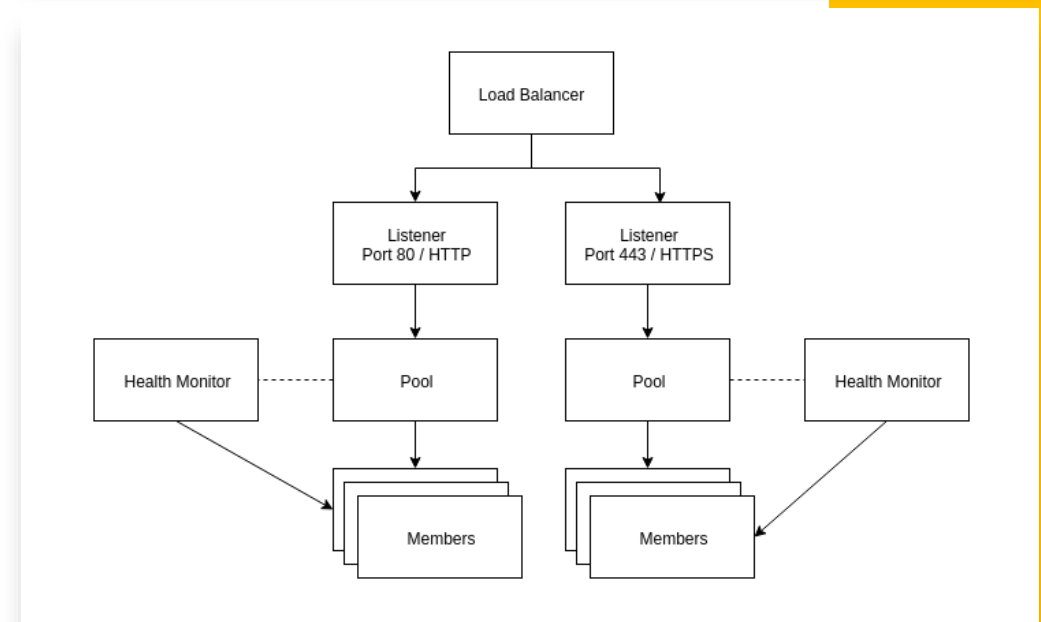
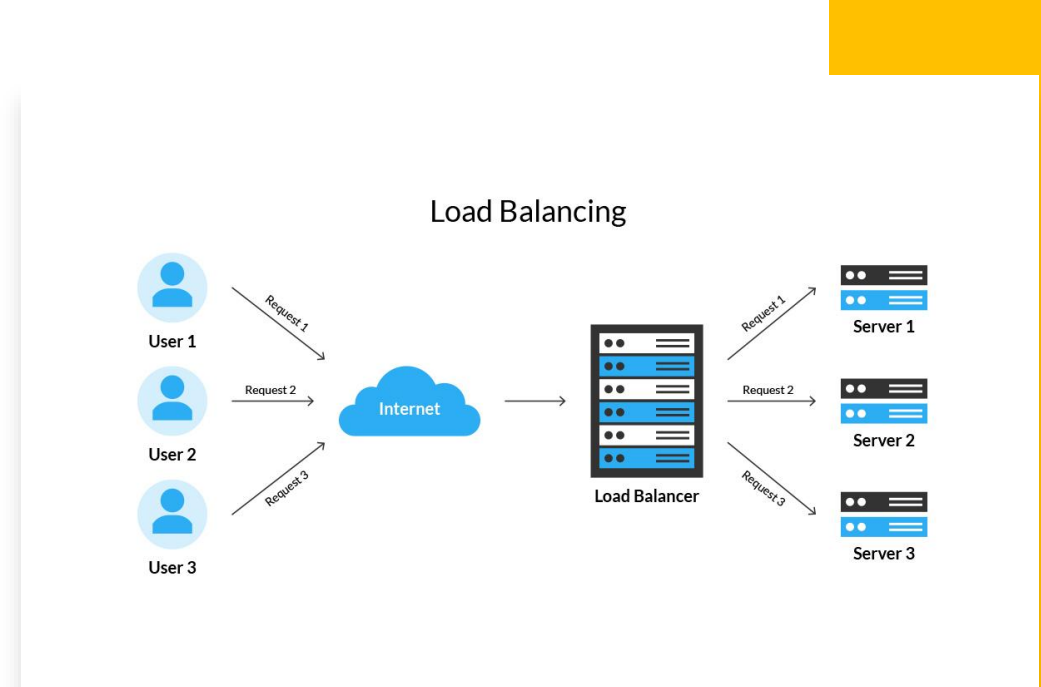
Cluster K8s, but even better

To increase the resilience and/or performance of the cluster, the following measures can be used:

- Update the **operating system** of the cluster VMs in turn;
- Use a **bastion**, open to the outside, to access the cluster VMs;
- Distribute workers and especially masters **geographically** (or at least on different racks of the same DataCenter);
- **Add/resize** workers quickly thanks to the cloud paradigm;
- Add a **Load Balancer** (Octavia in Openstack);
- Have a **multi-master** cluster (**odd number**) to increase the redundancy of the main components (especially etcd **quorum**) and allow an update **without interrupting** the provision of services;
- The nodes hosting the ETCD should have **fast disks** (or save data in RAM with periodic backups) due to the **high I/O**;
- Possibility to **separate** master nodes and nodes hosting the ETCD (maybe it is better to have 5 masters?).

Octavia Load Balancer

- Automatic creation of 2 VMs (active + backup).
- Health monitor integrated and pool composed of multiple members.



Octavia Load Balancer

- Ability to easily configure connections (protocol, timeout, CIDR, etc.)
- Lower consumption of FloatingIP

Provide the details for the listener.

Name	<input type="text" value="https"/>	Description	<input type="text"/>
Protocol *	<input type="text" value="TCP"/>	Port *	<input type="text" value="443"/>
Client Data Timeout	<input type="text" value="50000"/>	TCP Inspect Timeout	<input type="text" value="0"/>
Member Connect Timeout	<input type="text" value="5000"/>	Member Data Timeout	<input type="text" value="50000"/>
Connection Limit *	<input type="text" value="-1"/>	Default Pool ID	<input type="text" value="03812a0c-6e25-4f15-8f4c-1fe669a7a44e(https)"/>
Allowed Cidrs	<input type="text" value="0.0.0.0/0"/>		
Admin State Up	<input type="button" value="Yes"/> <input type="button" value="No"/>		



ETCD quorum

An [etcd](#) cluster must be comprised of an **odd number** of server nodes for etcd to maintain **quorum**. For a cluster with n servers, quorum is $(n/2)+1$. For any odd-sized cluster, adding one node will always increase the number of nodes necessary for quorum. Although adding a node to an odd-sized cluster appears better since there are more machines, the fault tolerance is worse. Exactly the same number of nodes can fail without losing quorum, but there are now more nodes that can fail.



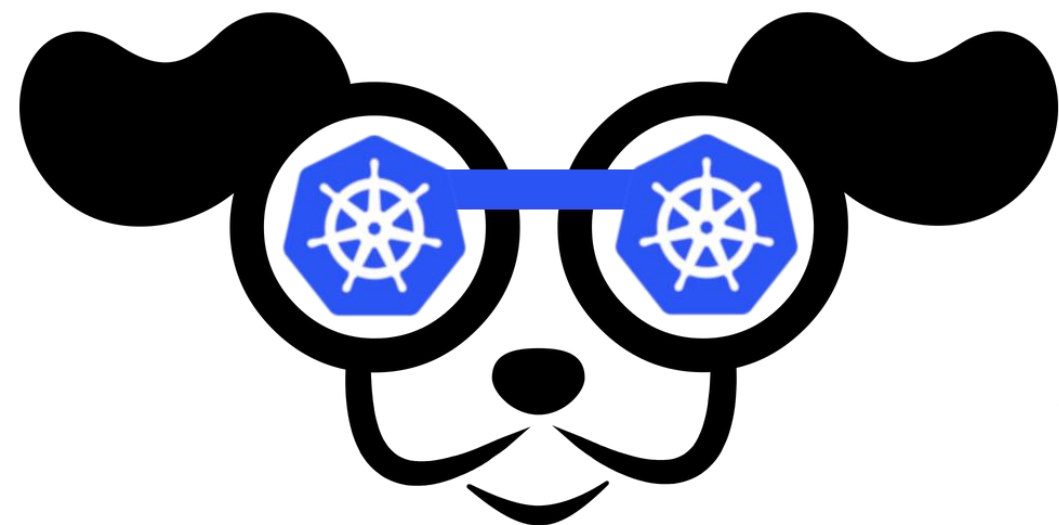


K8s side Improvements

Increase resilience and management of a service thanks to K8s components



API server, but even better with...



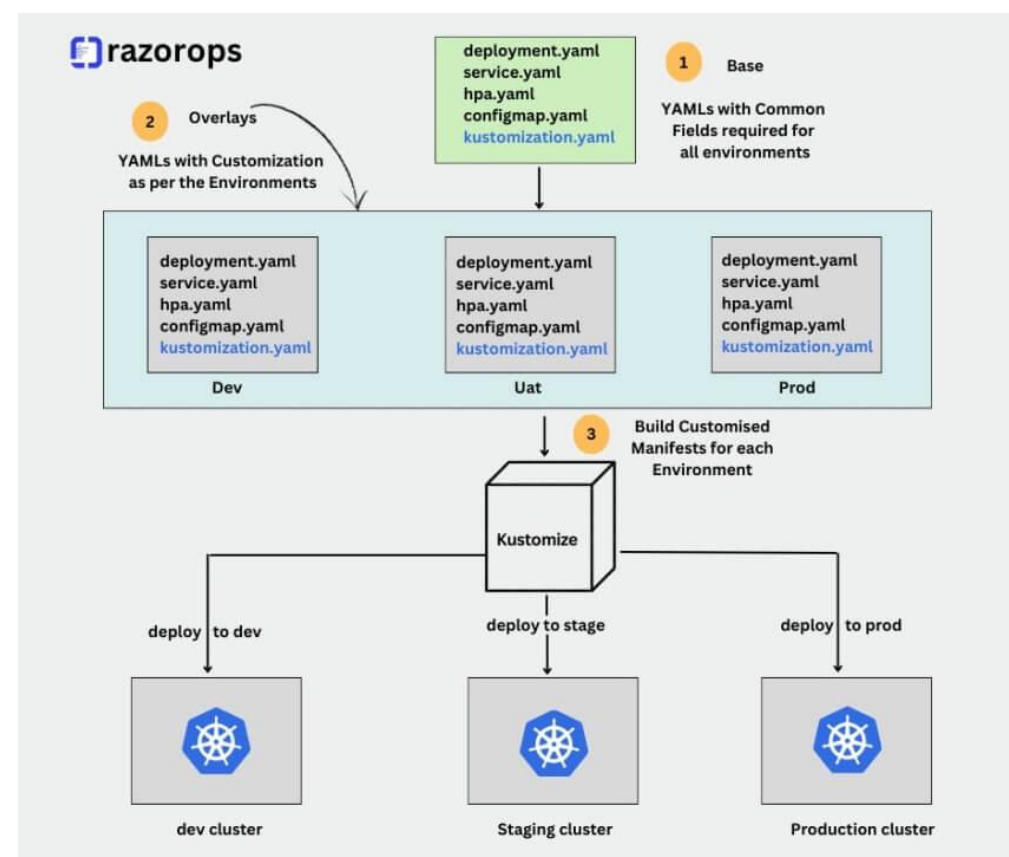
- Bash [auto-completion for K8S](#)
 - Install and enable bash-completion

```
$ dnf install bash-completion
```
 - Insert the following lines into the `.bashrc` file

```
source /usr/share/bash-completion/bash_completion
source <(kubectl completion bash)
alias k=kubectl
complete -F __start_kubectl k
```
- [K9s](#) is a terminal-based UI to interact with your Kubernetes clusters. The aim of this project is to make it easier to navigate, observe and manage your deployed applications in the wild. K9s continually watches Kubernetes for changes and offers subsequent commands to interact with your observed resources.

Kustomize

- [Kustomize](#) is an open-source configuration management tool designed for Kubernetes. It empowers developers and operators to customize, patch, and manage Kubernetes manifests in a declarative manner.



Pod controller

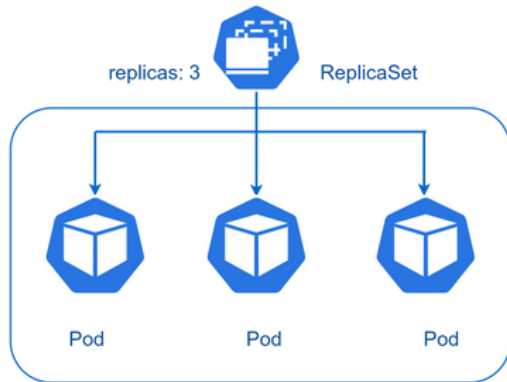


Fig: ReplicaSet

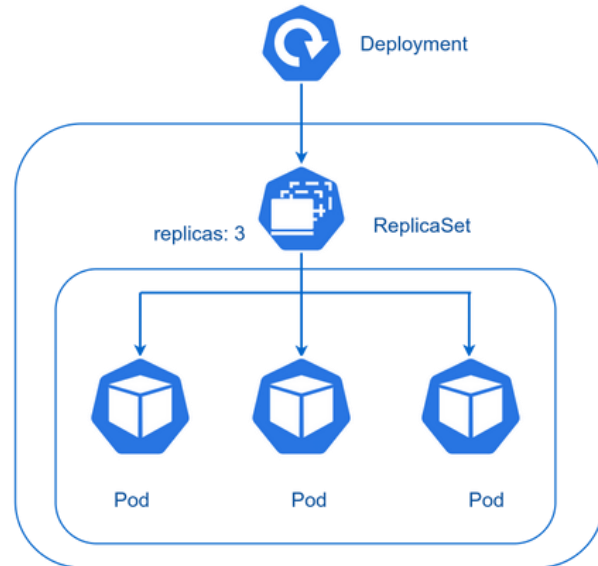


Fig: Deployment #gogslides

The nature of Pods is ephemeral, so [controllers](#) have been created to monitor their **number** and **health status**.

- **ReplicaSet (RS)**: the purpose is to multiply the number of copies of a certain application (Pod). This serves to increase the reliability of the application in the event of a replica failure and/or to manage requests that would not be digestible by a single instance.
- **Deployment**: ensures that there are a certain number of replicas (using the RS) and allows rollout strategies. Thanks to the rollout it is possible to gradually replace the Pods belonging to the old version of the Deployment, with the new ones. Furthermore, it is always possible to perform a rollback to the previous version.

Horizontal Pod Autoscaler



- [Horizontal Pod Autoscaler](#) (HPA) automatically scales the number of Pods in a replication controller, deployment, replicaSet or statefulSet based on observed CPU or Memory (RAM) utilization.
- To create an HPA

```
$ kubectl autoscale <kind> <name> --cpu-percent=<value> --min=<value> --max=<values>
$ kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
```