Getting Kubernetes

1. Playgrounds

- easiest way
- not for production
- Play with Kubernetes, Docker Desktop, Minikube, k3d, etc.

2. Hosted Kubernetes

- Cloud platforms offer hosted Kubernetes service
- Not all services are equial
- Great on-ramp to lets you to focus on your applications
- Close to a zero-effort production-grade Kubernetes cluster.
- Example: Google Kubernetes Engine:
    - deploy
    - high performance
    - highly available
    - security best-practices out-of-the-box
- AWS: Elastic Kub Service EKS
- Azure: Azure Kub Service AKS
- Civo CLoud Kubernetes
- DigitalOcean: DOKS
- Google CLoud Platform: Google Kub Engine GKE
- Linode: Linode Kubernetes Engine LKE

3. DIY Kubernetes Clusters

- Hardest way to get Kubernetes Cluster
- build by yourself
- Most flexibility and control

Playground Example: Play With Kubernetes

- Quick and simple.
- Requeriments: computer, internet connection, DockerHub and GitHub account.
- suffer from capacity and performance issues
- It last four hours
- FREE

1. labs.play-with-k8s.com
2. ADD NEW INSTANCE
3. This is a Kubernetes node
4. Run commands to see components pre-installed on the node

```
docker --version
kubectl version --output=yaml
```

5. Run prpovided `kubeadm init` to initialize a new cluster.

```
kubeadm init --apiserver-advertise-address $(hostname -i) --pod-network-
cidr...
```

The node is initialized as the control plane node. The output of kubeadm init give a list of commands, buy PWK has already configured.

6. Verify de nodes `kubectl get nodes`. The status NotReady is because the missing configured Pod Network.
7. Initialice Pod network (cluster networking). copy from kubeadm init the `kubetctl apply -f https...`.
8. verify the cluster to see status Ready
9. copy the `kubeadm join` from command kubeadm init
10. join command includes the cluster join-token, IP socket, API server.
11. Add new instance
12. Paste the `kubeadm join` command into the terminal of node2
13. `kubeadm join --token ...`
14. Switch back to the node1 and run `get nodes`

node1 was initialized as control plane node and additional nodes will join as worker nodes.

2. Google Kubernetes Engine GKE

GKE is a hosted Kubernetes service that runs on the Google Cloud Platform

- Fast and east way to get a production-grade Kubernetes cluster
- Managed control plane
- Itemized billing

kubectl

- `kubectl` is the main Kubernetes command-line tool.
- converts commands to HTTP REST requests with JSON content required by the Kubernetes API server.
- uses a config file to know wich cluster and API endpoint to send commands to
- config file is on `$HOME/.kube/config`.
    - defines Clusters, Users credentials and Contexts.

Clusters is the list of Kub clusters. It makes it possible for a single kubectl workstation to manage multiple clusters: name, certificate info, API server endpoint

Users define users access on each cluster, as dev or ops users with different permissions: name, username, credentials.

Contexts group clusters and users undera friendly name

```
kubectl config view
```

```
apiVersion: v1
kind: Config
clusters:
  - cluster:
  name: shield
    certificate-authority: C:\Users\nigel'.minikube/ca.crt
    server: https://191.168.1.77:8443
users:
- name: coulson
  user:
    client-certificate: ...
    client-key: ...
context:
- context:
    cluster: shield
    user: coulson
  name: director
current-context: director
```

```
kubectl config current-context
kubectl use-context holamundo
>> switched to context "holamundo"
```