# Introduction to Kubernetes/MiniKube (Lightning talk and MiniKube Demo)

By George Franklin

Our Main Sponsors




Limerick AI – (Applications and Games) Software Development MeetUp

http://meetup.com/LimerickAiSD

Twitter @LimerickAiSD

# Big Thanks To

- For Everyone Joining & Attending
- A Big thanks to Bank of Ireland for the BOI workbench meetup location in Limerick.
- Our Sponsors JetBrains and Redgate for their raffle product licenses.

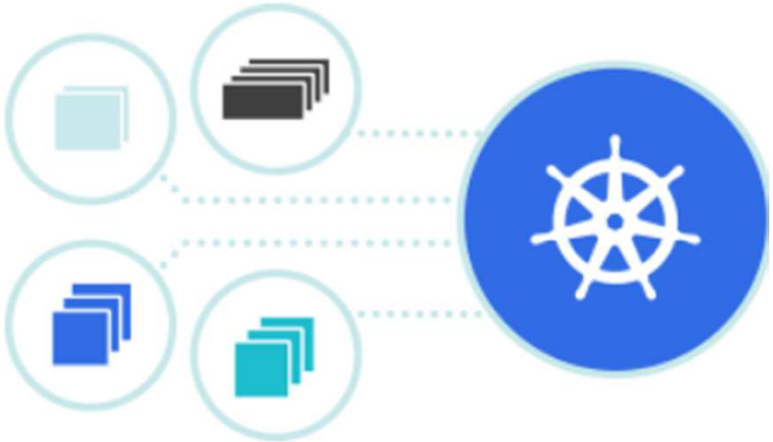
# What is Kubernetes

# So what will we cover

- What is Kubernetes
- What are containers / pod
- Why do containers need orchestration
- The main working processes/components of kubernettes
- It all about Pods and not Containers !!

- How we can control a cluster via cloud tools, the Apiserver and the client application – Kubectl.
- How to start a a statefulset and a simple pod using Kubectl

- We won’t have time to cover all the Controller types, that’s for another talk perhaps with a focus on building microservices or games services in kubenetes.



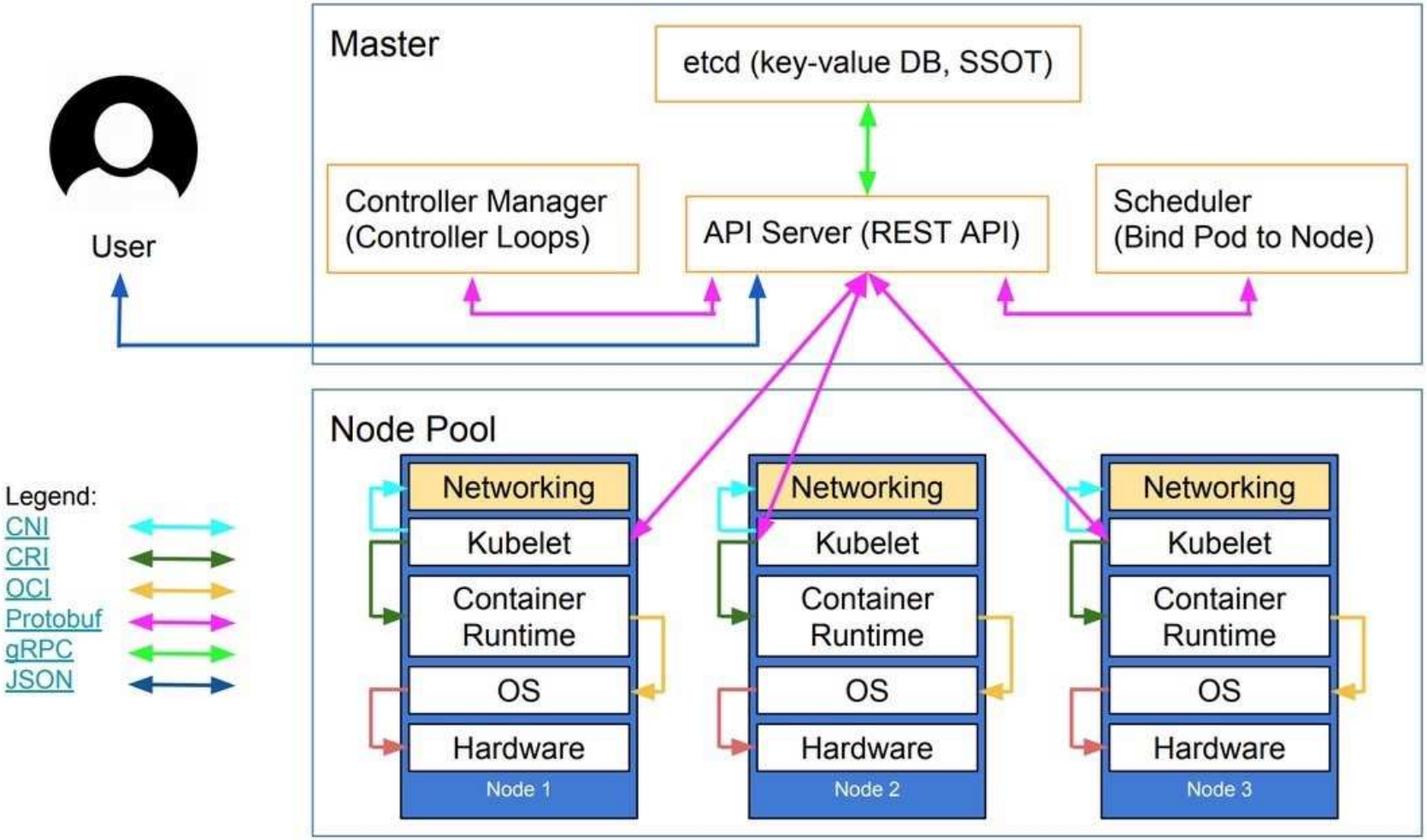
# Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.

# ***Kubernetes Cluster***

- We define **Kubernetes as a container  orchestration framework**.
- A **Kubernetes Cluster** is used to describe the **deployment of a master and worker nodes**. Users **application containers will run on the worker nodes**.
- **Kubernetes** is also a **Hybrid system** as part of the framework uses both containers as well as system processes to actually **manage the cluster  spread across the master and worker nodes**. So you could say it eats its own dog food.
- So we mentioned **containers** and specifically lets look at **Docker** the most popular container technology at the moment used by default in **Kubernetes**. (But you can use others i.e. **rkt**)
- Also ask the question - **Why do containers need orchestration** and the most common **uses cases of large deployments of containers**.

# Kubernetes' high-level component architecture




Picture Source : http://sysadvent.blogspot.com/2017/12/day-24-on-premise-kubernetes-with.html You can use the picture below for visualisation. Thanks to Lucas Käldström for creating this (@kubernetesonarm), used in his presentation on KubeCon

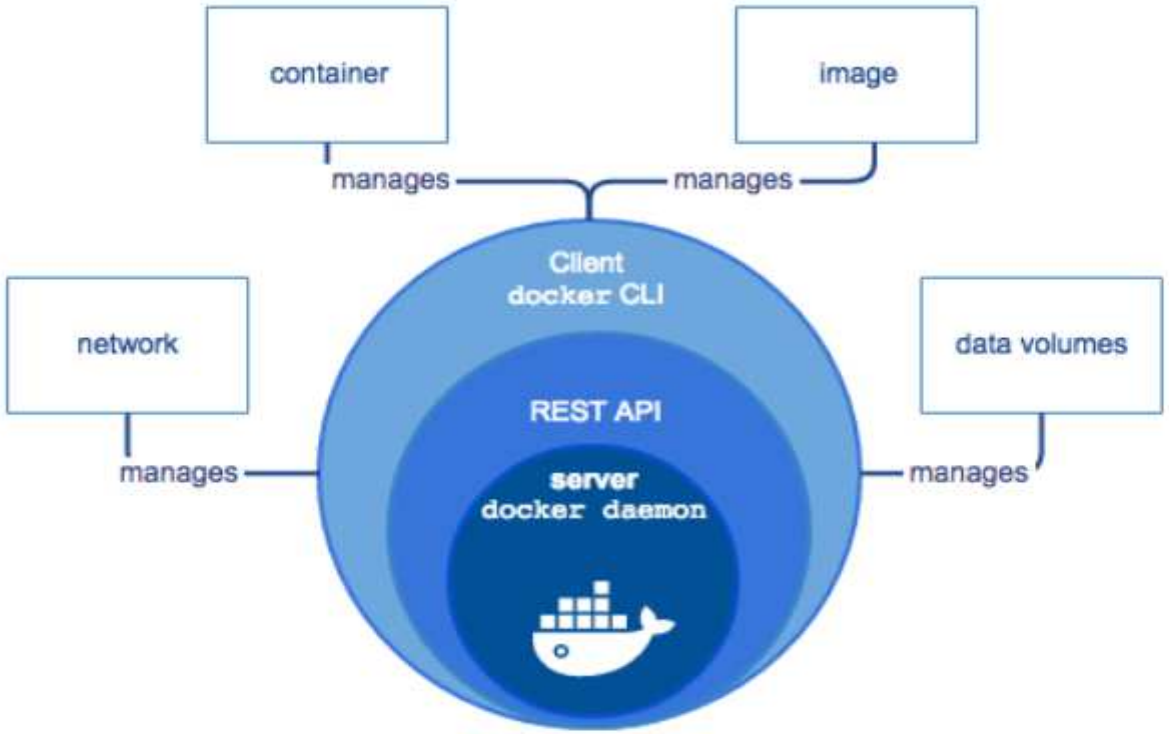
# ***Kubernetes and Docker***

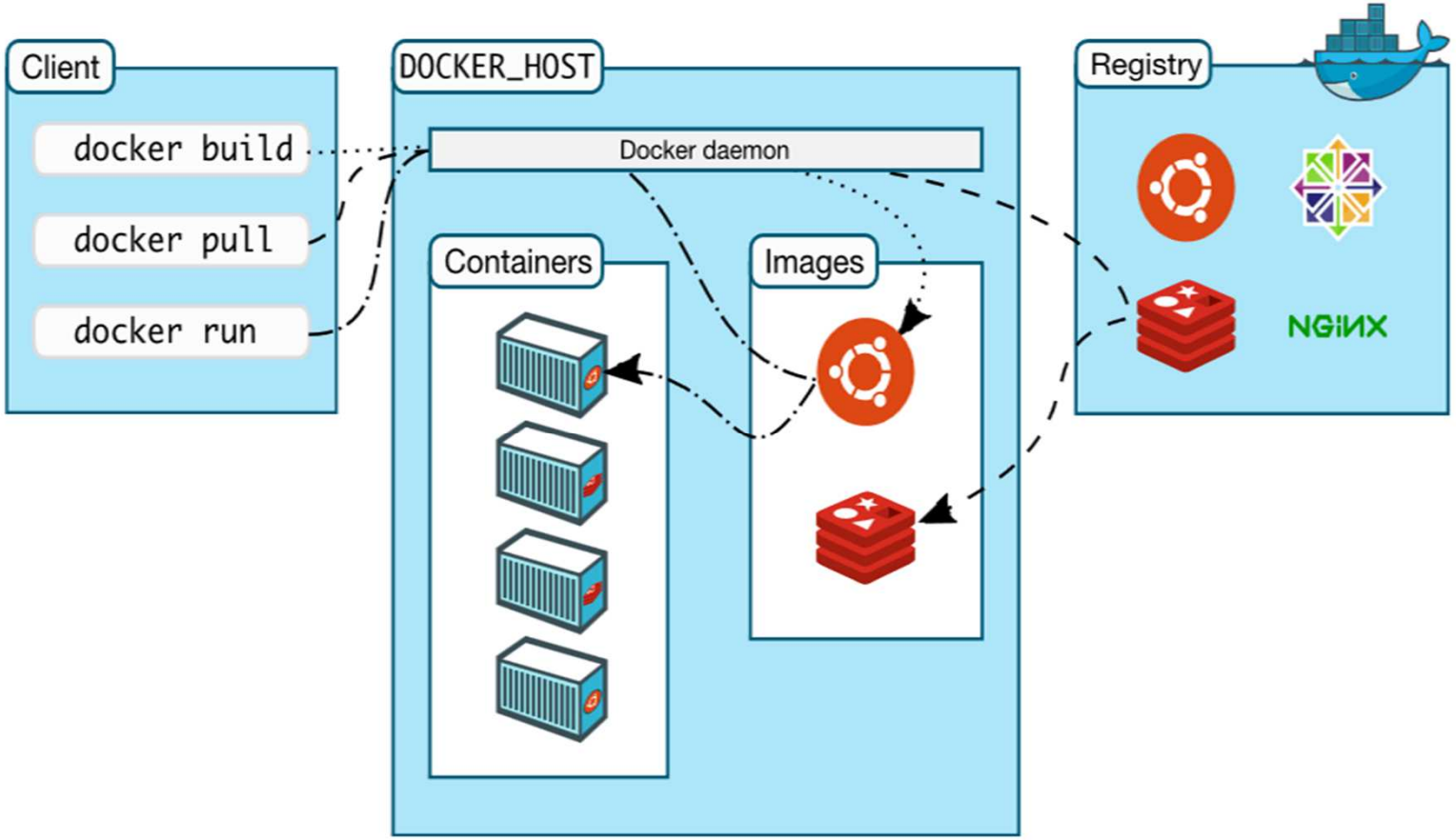
- Kubernetes uses docker as its default container runtime.
- Docker Initial release 13 March 2013
- Operating system Linux, Windows, macOS
- Written in Go
- Original author Solomon Hykes
- Website docker.com

# Docker Engine

*Docker Engine* is a client-server application with these major components:

- A server which is a type of long-running program called a daemon process (the `dockerd` command).
- A REST API which specifies interfaces that programs can use to talk to the daemon and instruct it what to do.
- A command line interface (CLI) client (the `docker` command).




Client
docker build
docker pull
docker run
DOCKER_HOST
Docker daemon
Containers
Images
Registry
NGINX

## ***The LifeCycle of a Docker Container I***

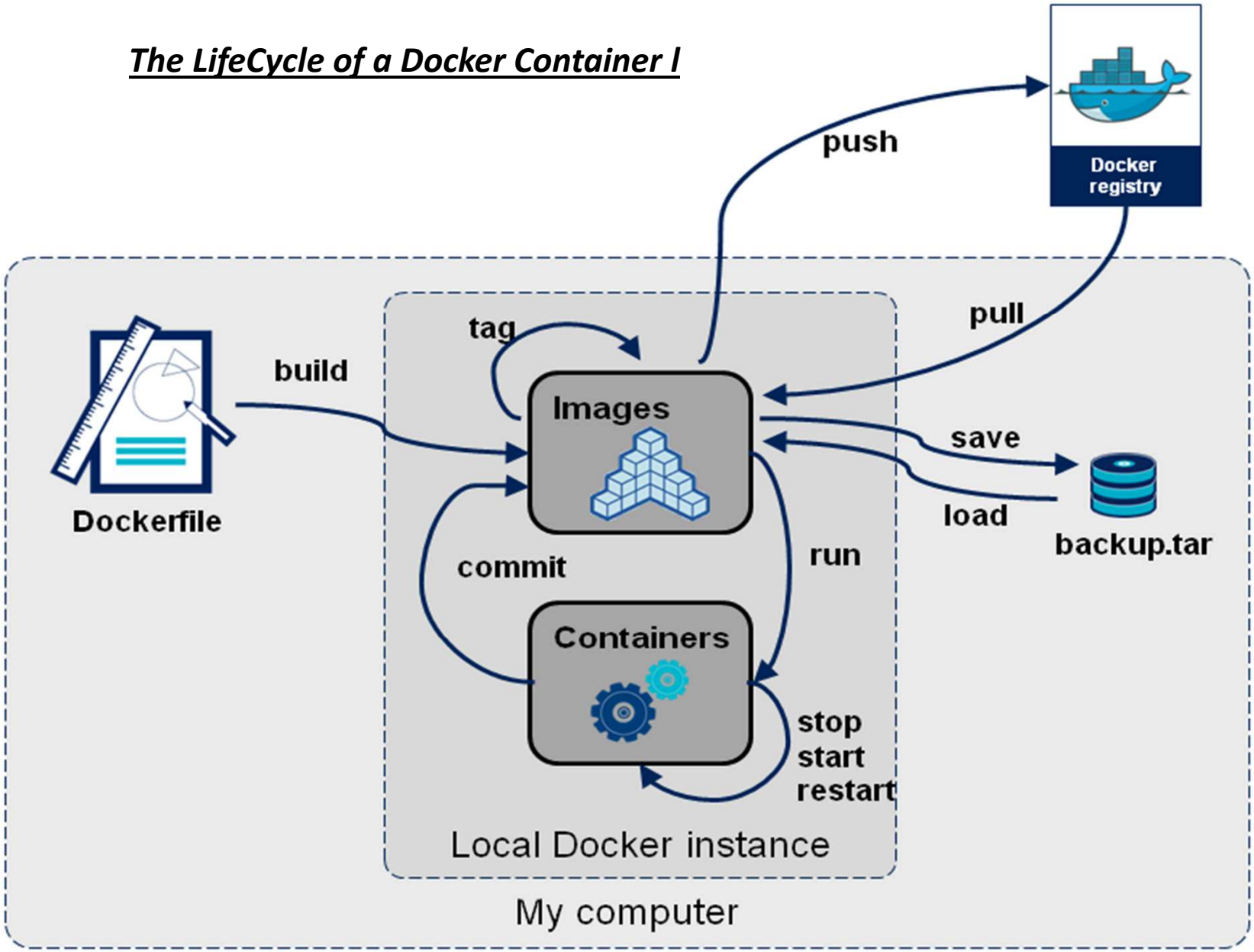



Image Source : https://stackoverflow.com/questions/23735149/what-is-the-difference-between-a-docker-image-and-a-container

# Docker Editions

## Docker Community Edition

Get started with Docker and experimenting with container-based apps. Docker CE is available on many platforms, from desktop to cloud to server. Build and share containers and automate the development pipeline from a single environment. Choose the Edge channel to get access to the latest features, or the Stable channel for more predictability.

Learn more about Docker CE

## Docker Enterprise Edition

Designed for enterprise development and IT teams who build, ship, and run business critical applications in production at scale. Integrated, certified, and supported to provide enterprises with the most secure container platform in the industry to modernize all applications. Docker EE Advanced comes with enterprise add-ons like UCP and DTR.

Learn more about Docker EE

# Run Docker anywhere

Docker for Mac

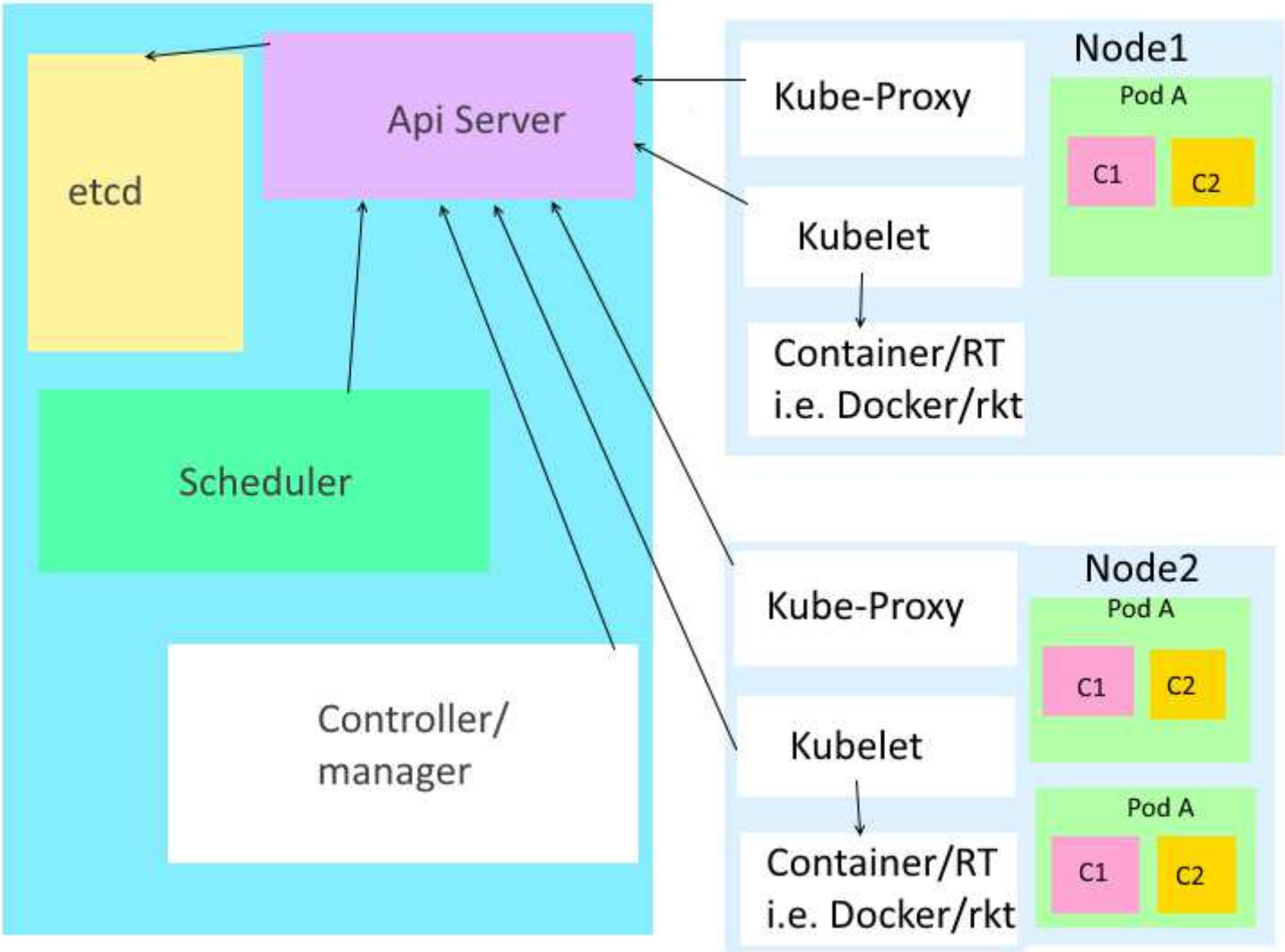
Docker for Windows

Docker for Linux

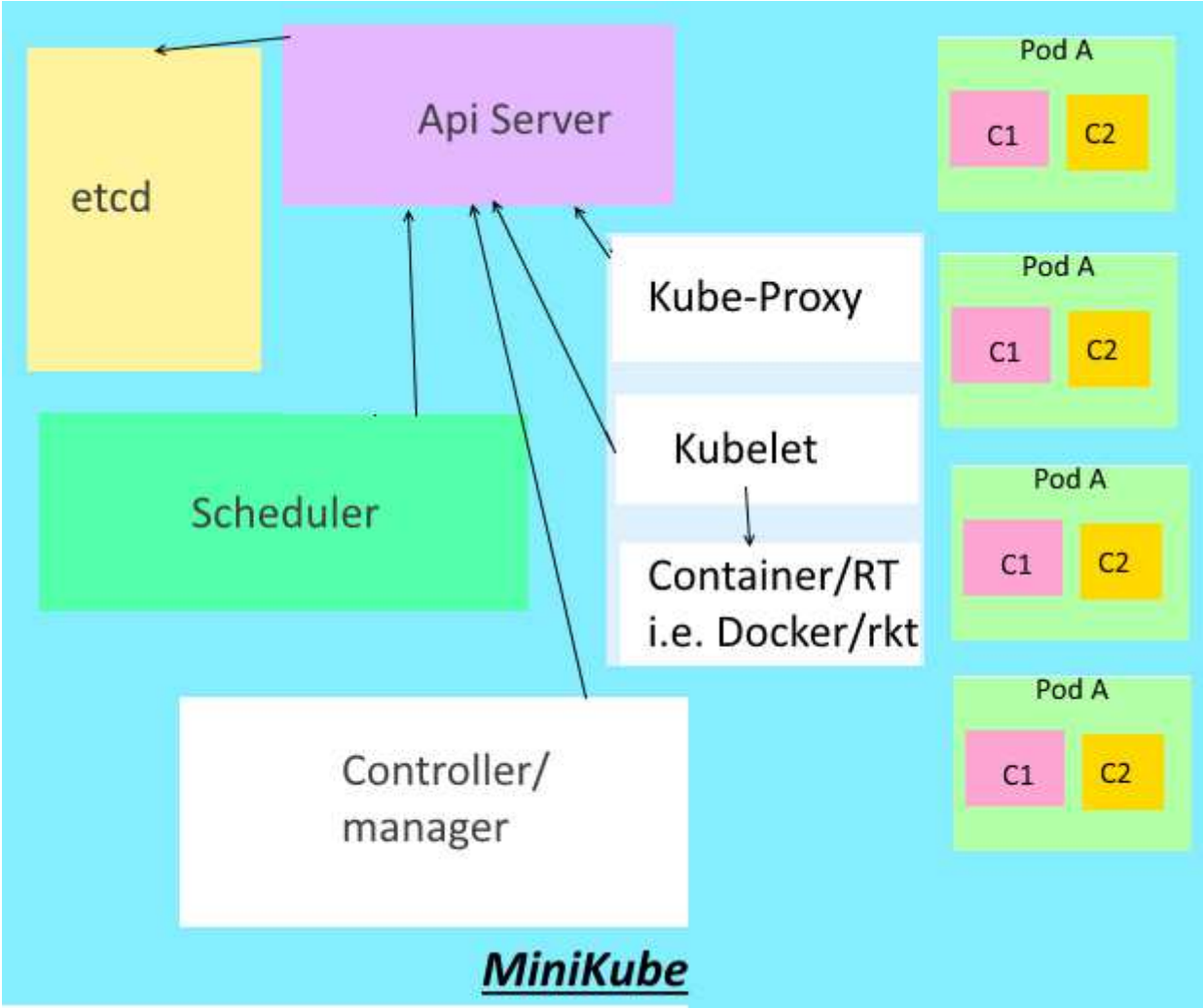
# ***Kubernetes Resources***

- NameSpaces
- ReplicationController
- ReplicaSet
- Job
- CronJob
- DaemonSet
- Deployment
- Service, Endpoints , Ingress
- ConfigMap, Secret
- PesistentVolume, PersistentVolumeCliam
- StorageClass

We can use kubectl – the client application to talk to the Api server (an Restful service) or via directly by https. The other way to talk to the Api server is via containers we have deployed in our in Pods. Since the Api server is a restful implementation so can support different versions of a restful Api Service.



Minikube provides us with a Kubernetes Cluster in a Box i.e. a virtual Machine. So we combine the components of a Master and a worker Node. So hence we have no additional nodes. We can kubectl the client application to create our pods and services the same as with a full cluster(masters & nodes).



# ***Install/Get Minikube and KubeCtl***

- We will install/get minikube and kubectl for windows (hyperv)
- Make a directory i.e. c:\k8\bin

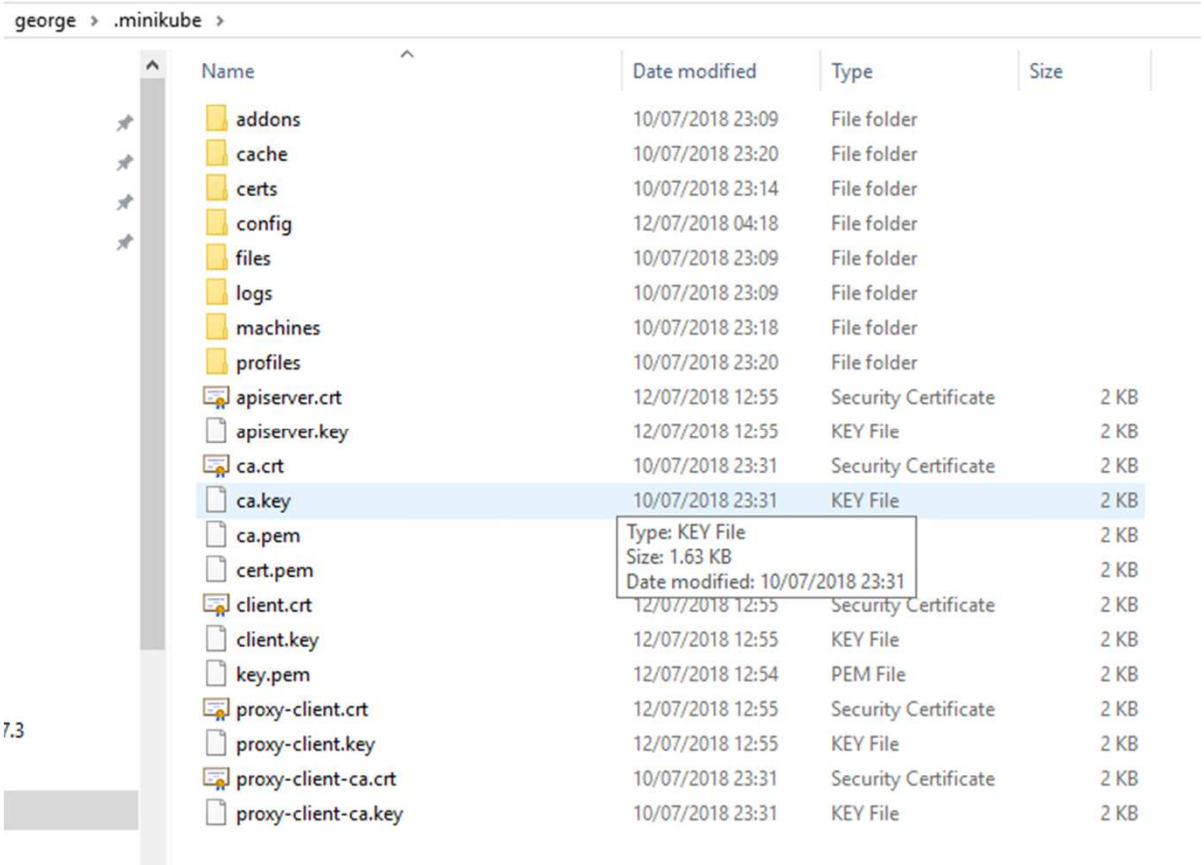
- Next Download minikube-windows-amd64.exe from https://storage.googleapis.com/minikube/releases/v0.28.0/minikube-windows-amd64.exe

- Rename minikube-windows-amd64.exe to minikube.exe and move it into c:\k8\bin

- Next download kubectl.exe from https://storage.googleapis.com/kubernetes-release/release/v1.10.3/bin/windows/amd64/kubectl.exe
- move minikube.exe into c:\k8\bin
- Next add c:\k8\bin to your path ( please note docker also now bundles kubectl.exe in his bin folder so decide if you want to use that or use you own. Put your bin folder first in the path if you want to make sure of using your own.
- by default MiniKube install into your home directory

# ***MiniKube default home directory***

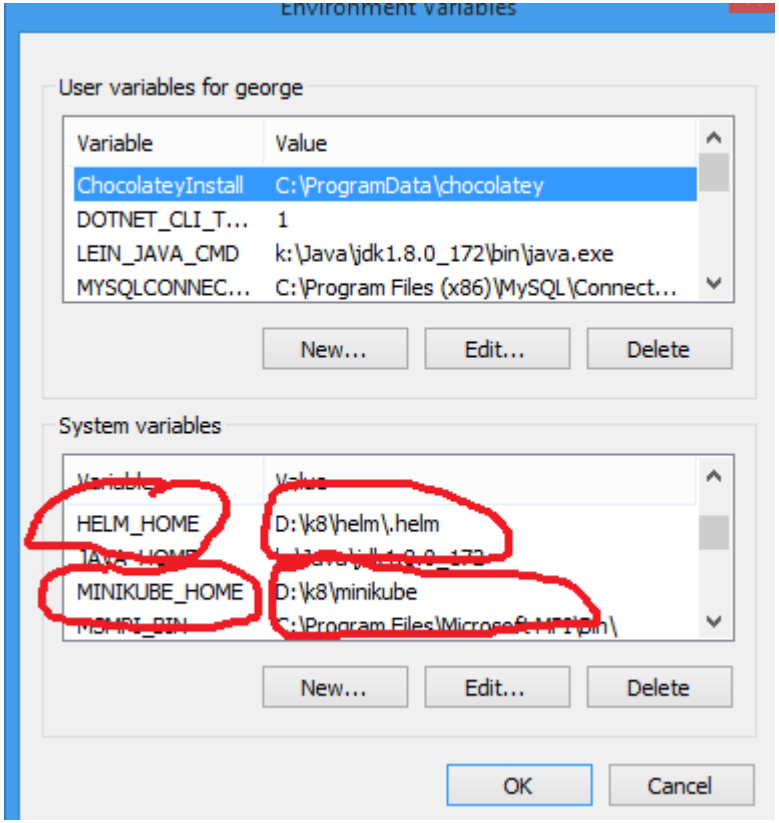
- by default MiniKube installs into your home directory under directory .minikube




But you can set a windows environment variable to point to another location i.e. if you have limited space on your c disk or want the vm that MiniKube creates to run another disk for performance.

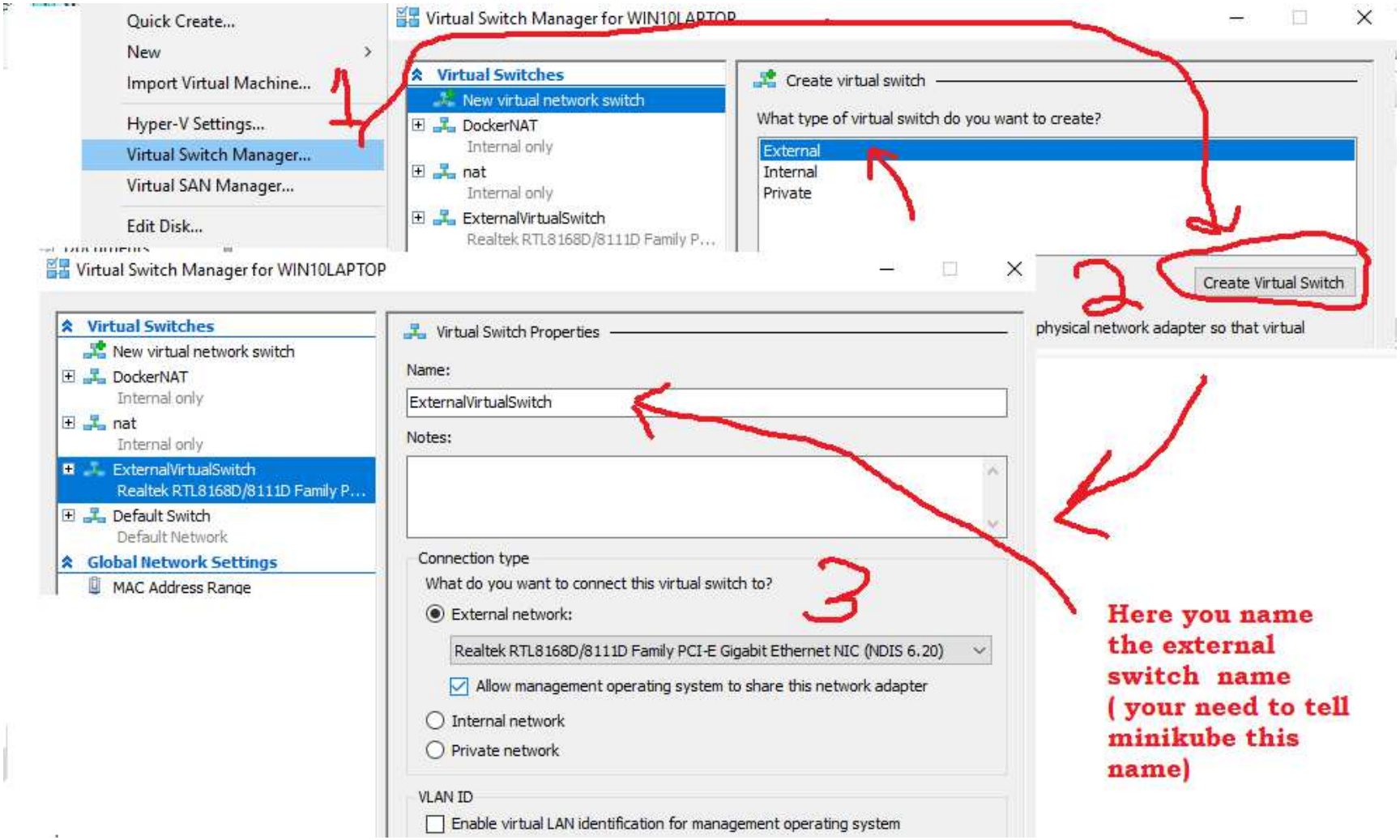
# ***Changing the MiniKube home directory***

- Set MINIKUBE_HOME to your new location – the install process will place it there instead




You will also notice you can set the Helm home directory from the default of your user home directory, as in this case we moved it to D\:k8\helm\.helm Helm is a kudenetes packages manager, we haven't discussed that yet. But just for reference encase you need to.

# ***How to install MiniKube on HyperV (First create a Hyperv Virtual External Switch)***




By default Minikube default to creating a Virtualbox VM as it default. If we want to use HyperV we must first create a external virtual switch in Hyperv (see graphics) you have have spaces or _ but best to keep it simple unlike my example !!!

# ***How to install MiniKube on Hyperv***

```
Windows Command Processor - minikube  start --vm-driver hyperv --hyperv-virtual-switch "External_Virtual_Switch"
Microsoft Windows [Version 10.0.17134.112]
(c) 2018 Microsoft Corporation. All rights reserved.

 Convert Virtual Disk
    40%
    [oooooooooooooooooooooooooooooooooooooooooooooooooooooo

C:\Windows\System32>d:

D:\>cd d:\k8\bin

d:\k8\bin>minikube start --vm-driver hyperv --hyperv-virtual-switch "External_Virtual_Switch"
Starting local Kubernetes v1.10.0 cluster...
Starting VM...
Downloading Minikube ISO
 153.08 MB / 153.08 MB [=============================================] 100.00% 0s
```

We tell Minikube to use hyperv to create our vm machine and tell it the external virtual switch to use. (note my virtual switch has underscores in it, that’s because I did several installations on different machines and named then different names that’s all).

# ***How Stop and Start MiniKube on Hyperv***

```
Shutting Down
   0%
   [

D:\k8\bin>minikube stop
Stopping local Kubernetes cluster...
```

Administrator: Windows PowerShell

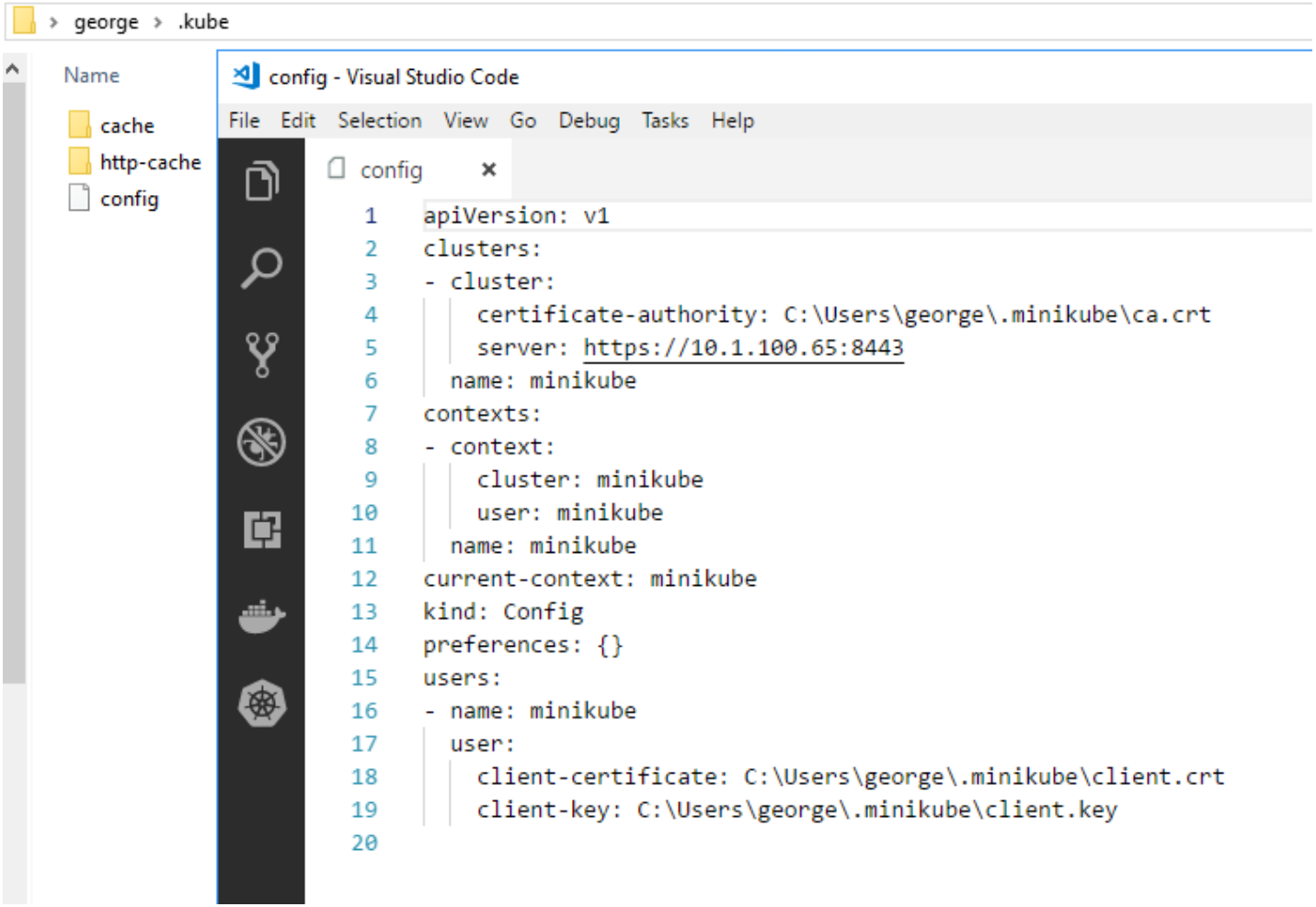
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> minikube start
Starting local Kubernetes v1.10.0 cluster...
Starting VM...
Getting VM IP address...
Moving files into cluster...
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Starting cluster components...
Kubectl is now configured to use the cluster.
Loading cached images from config file.
```

Be warned this never seems to work without errors in the current version of 0.28 I was using. After we have installed Minikube and it is up and running we can just use minikube start , as it creates a config file in your home directory under .kube

# ***MiniKube Creates a Cluster Config file***

After we have installed Minikube , Minikube creates a config file in your home directory under .kube called config. Lets have a look at it.




The Purpose of this is so that client tools that need to work with the cluster have a way to find out how to connect to the cluster and hence the kubectrl client tool can send your command to the Kubernetes clusters Api server. ( we can cluster context with kubectl config set-context)

# ***MiniKube addons***

Minikube has the option to enable/disable various addons – there are issues around addons and you need to make sure the various configurations files are updates. i.e. perhaps needing to restart Minikube . But this is just a passing note to make you aware of the addon feature.

```
C:\Windows\System32>minikube addons list
- addon-manager: enabled
- coredns: disabled
- dashboard: enabled
- default-storageclass: enabled
- efk: disabled
- freshpod: disabled
- heapster: disabled
- ingress: disabled
- kube-dns: enabled
- metrics-server: disabled
- registry: disabled
- registry-creds: disabled
- storage-provisioner: enabled
```

# ***Kubernetes namespaces***

A Kubernetes cluster has namespaces lets have a look and use the kubectl to talk the api server and send some commands.

```
D:\k8\bin> kubectl get pods --all-namespaces
NAMESPACE     NAME                                    READY     STATUS    RESTARTS   AGE
kube-system   etcd-minikube                           1/1       Running   0          12m
kube-system   kube-addon-manager-minikube             1/1       Running   1          1h
kube-system   kube-apiserver-minikube                 1/1       Running   0          12m
kube-system   kube-controller-manager-minikube        1/1       Running   0          12m
kube-system   kube-dns-86f4d74b45-hq45c               3/3       Running   1          1h
kube-system   kube-proxy-vflz5                        1/1       Running   0          12m
kube-system   kube-scheduler-minikube                 1/1       Running   0          1h
kube-system   kubernetes-dashboard-5498ccf677-pcth4   1/1       Running   4          1h
kube-system   registry-h6xcw                          1/1       Running   0          28m
kube-system   storage-provisioner                     1/1       Running   2          1h

D:\k8\bin> kubectl get services --all-namespaces
NAMESPACE     NAME                   TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)         AGE
default       kubernetes             ClusterIP   10.96.0.1      <none>        443/TCP         1h
kube-system   kube-dns               ClusterIP   10.96.0.10     <none>        53/UDP,53/TCP   1h
kube-system   kubernetes-dashboard   NodePort    10.110.92.97   <none>        80:30000/TCP    1h
kube-system   registry               ClusterIP   10.98.77.99    <none>        80/TCP          28m

D:\k8\bin> kubectl get rc --all-namespaces
NAMESPACE     NAME       DESIRED   CURRENT   READY     AGE
kube-system   registry   1         1         1         29m

D:\k8\bin> kubectl get pods
No resources found.
```

# ***Lets ssh into the MiniKube vm***

Use minikube  ssh

To ssh into your minikube – be warned if you have the windows 10 ssh comand on the path that won't work. I use git's ssh and put that on the path .

# ***Lets run Statefulset of poker game tables***

I will talk through the various stages and will demo this if there is time.

```
PS C:\k8\bin> .\kubectl.exe delete sts poker
statefulset.apps "poker" deleted
PS C:\k8\bin> .\kubectl.exe get pods
No resources found.
PS C:\k8\bin> .\kubectl.exe create -f files/simplepodexamples/pokerstatefulset.yaml
statefulset.apps "poker" created
PS C:\k8\bin> .\kubectl.exe get pods
NAME      READY     STATUS    RESTARTS   AGE
poker-0   1/1       Running   0          4s
poker-1   1/1       Running   0          2s
poker-2   1/1       Running   0          0s
PS C:\k8\bin> .\kubectl.exe get sts
NAME      DESIRED   CURRENT   AGE
poker     3         3         2h
PS C:\k8\bin> .\kubectl.exe get sc
NAME                 PROVISIONER                AGE
standard (default)   k8s.io/minikube-hostpath   1d
PS C:\k8\bin> .\kubectl.exe get rs
No resources found.
PS C:\k8\bin> .\kubectl.exe get rc
No resources found.
PS C:\k8\bin> .\kubectl.exe get jobs
No resources found.
PS C:\k8\bin> .\kubectl.exe get cronjobs
No resources found.
```

# ***Lets run Statefulset of poker game tables***

Lets talk Yaml – The yaml layout is very important, with regards to space and indention. Here we run a Statefulset , which controls a service with 3 replicas. This allows our game tables to have a consistent identify, so if an pod gets restarted then the pod gets the same identify. Also I am using my own local registry with a self signed cert – for the image cpucode.local/pokerv1.

pokerstatefulset.yaml

```yaml
apiVersion: apps/v1beta1
kind: StatefulSet
metadata:
  name: poker
spec:
  serviceName: poker
  replicas: 3
  template:
    metadata:
      labels:
        app: poker
    spec:
      hostAliases:
      - ip: "10.1.100.133"
        hostnames:
        - "cpucode.local"
      - ip: "10.1.100.150"
        hostnames:
        - "anotherhost.local"
        - "andanotherhost.local"

      containers:
      - name: pokerv1
        image: cpucode.local/pokerv1
#         ports:
#        - name: http
#          containerPort: 8080
#        volumeMounts:
#        - name: data
#          mountPath: /var/data
#  volumeClaimTemplates:
#  - metadata:
#     name: data
#    spec:
#     resources:
#        requests:
#          storage: 1Mi
#      accessModes:
#      - ReadWriteOnce
```

# *Lets run a single poker game table pod*

When we run a single pod that is not controlled by a controller, then we don't get any automatic behaviour to restart that pod if i.e. the node that the pod was running on in the cluster fails. Basically we want to control our pods via services and those services by controllers. So that we can scale up and down the resources as need. In terms of microservices we may want to do blue/green deployment. We have a Deployment controller that does actually that for us.

pokerstatefulset.yaml  cpucodepoker.yaml ...\persis

```yaml
apiVersion: v1
kind: Pod
metadata:
    name: poker-pod
spec:
    hostAliases:
    - ip: "10.1.100.133"
      hostnames:
      - "cpucode.local"
    - ip: "10.1.100.150"
      hostnames:
      - "anotherhost.local"
      - "andanotherhost.local"

    containers:
    - image: cpucode.local/pokerv1
      name: pokerv1
```

```
PS C:\k8\bin> .\kubectl.exe run singlepokertable --image=cpucode.local/pokerv1
deployment.apps "singlepokertable" created
PS C:\k8\bin> .\kubectl.exe get pods
NAME                                READY   STATUS    RESTARTS   AGE
poker-0                             1/1     Running   0          3h
poker-1                             1/1     Running   0          3h
poker-2                             1/1     Running   0          3h
singlepokertable-5dc677fbcd-29vxg   1/1     Running   0          21m
PS C:\k8\bin>
```

Here we use the " kubectl run " to run the pokertable pod. We couldhave used "kubectrl create -f filename" to run it but just demo the run option.

Here we run a single pod see (Kind: pod)

# *Installing Helm into our Cluster*

```
D:\k8\bin>echo %HELM_HOME%
D:\k8\helm\.helm

D:\k8\bin>helm init
Creating D:\k8\helm\.helm\repository
Creating D:\k8\helm\.helm\repository\cache
Creating D:\k8\helm\.helm\repository\local
Creating D:\k8\helm\.helm\plugins
Creating D:\k8\helm\.helm\starters
Creating D:\k8\helm\.helm\cache\archive
Creating D:\k8\helm\.helm\repository\repositories.yaml
Adding stable repo with URL: https://kubernetes-charts.storage.googleapis.com
Adding local repo with URL: http://127.0.0.1:8879/charts
$HELM_HOME has been configured at D:\k8\helm\.helm.
Warning: Tiller is already installed in the cluster.
(Use --client-only to suppress this message, or --upgrade to upgrade Tiller to the current version.)
Happy Helming!

D:\k8\bin>
```

1

2

3

Helm has the client side which and a server side called Tiller that get installed into your cluster. As noted above you use ( helm init ) . You can now install Helm charts ( i.e. names to do with ships) charts are packages of Yaml files so you can deploy large or complex applications i.e. Mongodb when you want to install a Mongodb replicaset etc.

# The End for Now

- This was just a 15 minute lighting talk with Demo of a scaling a poker game. The talk was more focused on installing and setting up Minikube. I would have loved to discussed deploying microservices via the different types of controllers in  Kuternetes. But that is a different talk for another time perhaps.

- Thanks – See you next time at the
  Limerick Ai Software Development meetup