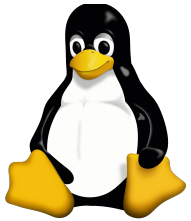
Kubernetes Fundamentals

greg@blacksintechnology:~$ whoami

Insight Digital Innovation

PER SCHOLAS

bit
BLACKS IN TECHNOLOGY



Greg Greenlee

Kubernetes Fundamentals Agenda

- Common terms
- What is Kubernetes (K8’s)?
- Why do we need Kubernetes?
- How does Kubernetes work?
  - K8’s components
  - K8’s objects
- How do we interact w/ Kubernetes?

Common Terms

- Microservices
- Imperative vs Declarative
- Current State vs Desired State
- Cloud native
- Immutable vs Mutable
- Containers
- Container run time
- Orchestration

Imperative vs Declarative

OUR
MIGHTY
BLOOM AND BLOSSOM
CHOCOLATE
CAKE

The Bloom and Blossom Juniors love getting messy in the kitchen. This chocolate cake recipe is an absolute favourite for bringing out their inner Bruce Bogtrotter.

Christina & Julia

INGREDIENTS

For the chocolate cake
- 200g dark chocolate chopped
- 200g butter, cubed
- 170g self-raising flour
- ¼ tsp bicarbonate of soda
- 400g golden caster sugar
- 25g cocoa powder
- 3 medium eggs
- 75ml buttermilk
- grated chocolate to decorate

For the icing
- 200g chocolate chopped
- 300ml double cream
- 2 tbsp golden caster sugar

METHOD

Heat the oven to 160c/fan oven 140c then line your round cake tin (approx. 20cm and 7.5cm deep).

With some help from an adult put the chocolate in a medium pan with the butter and add 125ml cold water. Warm through over a low heat until melted (Or melt in the microwave for about 5 minutes, stirring halfway through).

In a bowl mix the self-raising flour, bicarbonate of soda and golden caster sugar with the cocoa powder. In another bowl beat the eggs with the buttermilk.

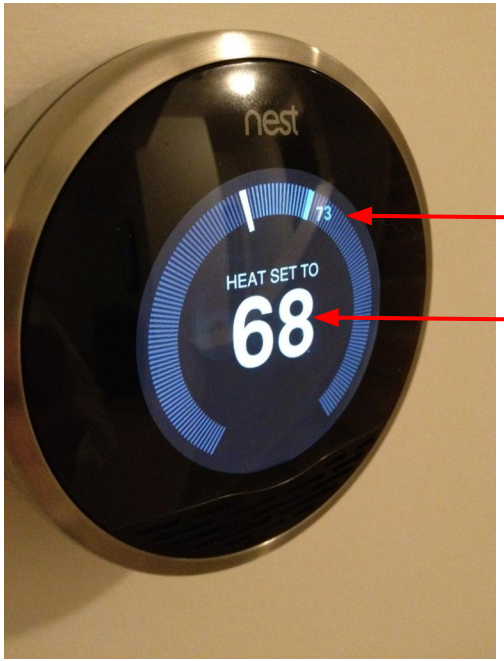
Pour the melted chocolate mixture and the egg mixture into the flour mixture and stir everything to a smooth, quite runny consistency.

Pour this into the tin and bake for 1hr 25 – 1hr 30 mins.

To make the icing, put the chocolate in a bowl. With the help of an adult carefully pour the cream into a pan, add the golden caster sugar and heat until it is about to boil. Take off the heat and pour it over the chocolate. Stir until the chocolate has melted and the mixture is smooth and let it cool until it is a little thicker.

Cut the cooled cake horizontally and sandwich the layers together with some icing, then pour the rest over the cake letting it fall down the sides, catch any drips…….
Then decorate with grated chocolate and eat.

Current State vs Desired State



Current State

Desired State

What is orchestration?

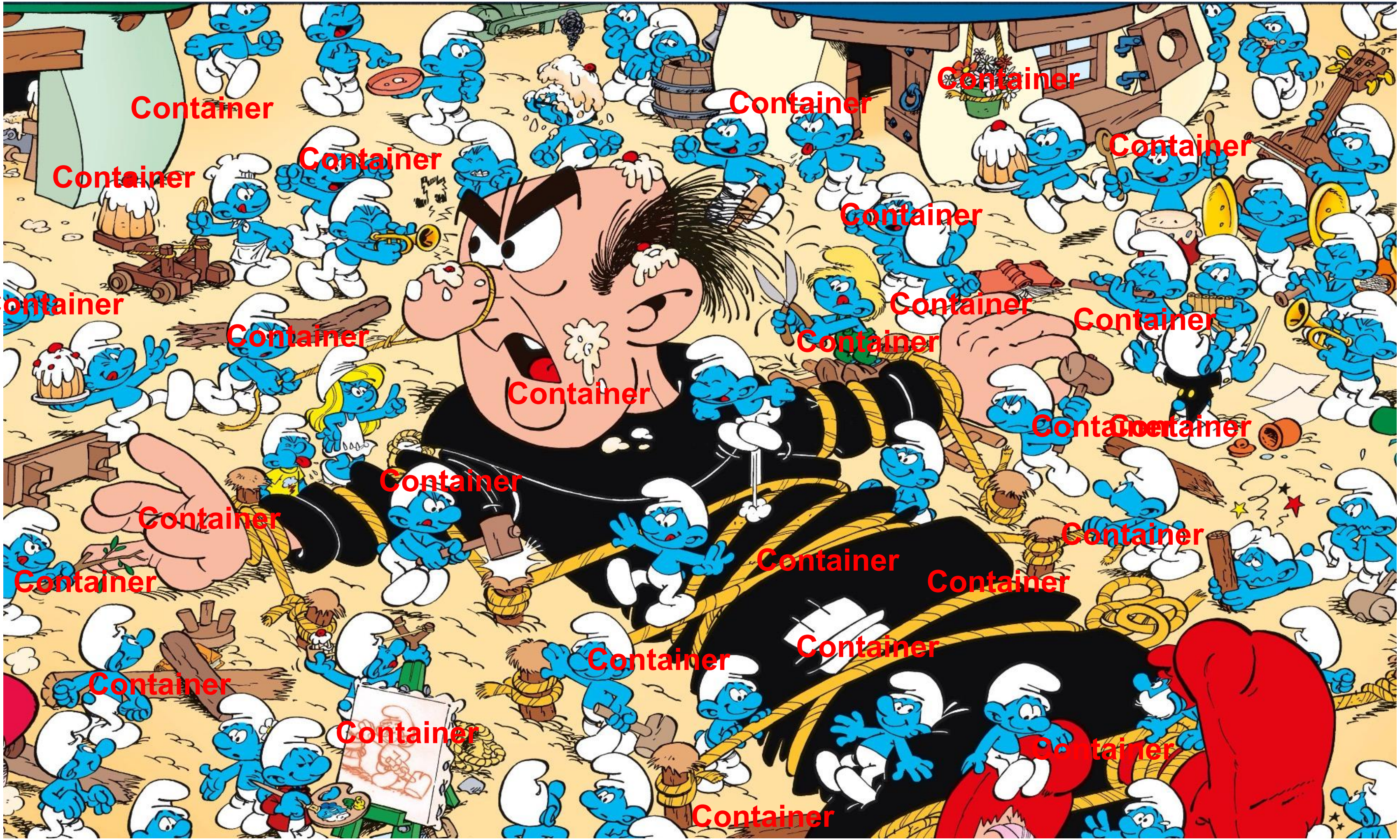
1. The arrangement ~~or scoring~~ of ~~music~~ containers for ~~orchestral~~ application and system performance.

2. The planning or coordination of the elements of a situation to produce a desired effect, especially surreptitiously
3.

What is Kubernetes (K8’s)?

- Open sourced in 2014 by Google (shares DNA with Borg and Omega)
- Kubernetes is an orchestration engine for automating deployments, scaling, managing, and providing the infrastructure to host containerized applications.
- Datacenter OS
- Abstraction - sit above cloud specific platforms

Why do we need Kubernetes?



Immutability,

Declarative

SPEED, SAFETY & RELIABILITY

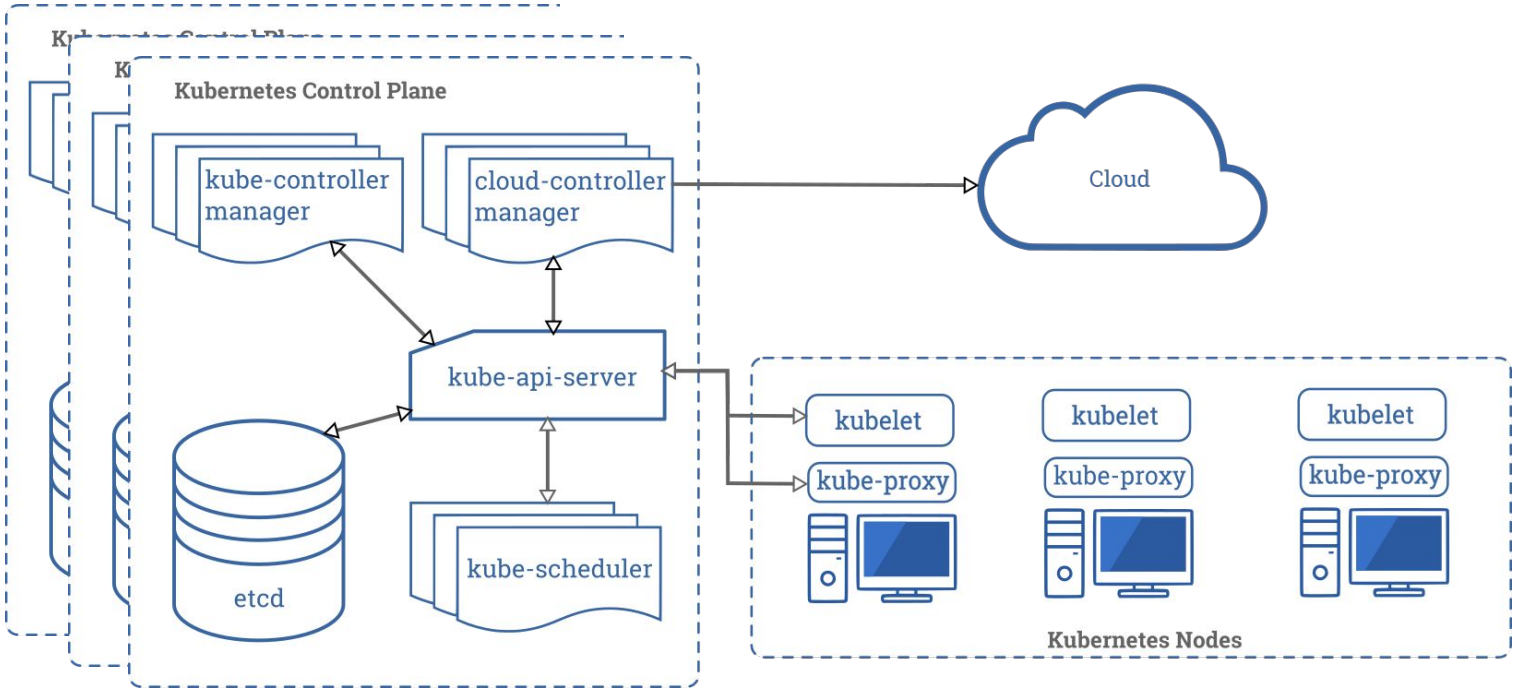
Online self healing

zero downtime updates/rollouts

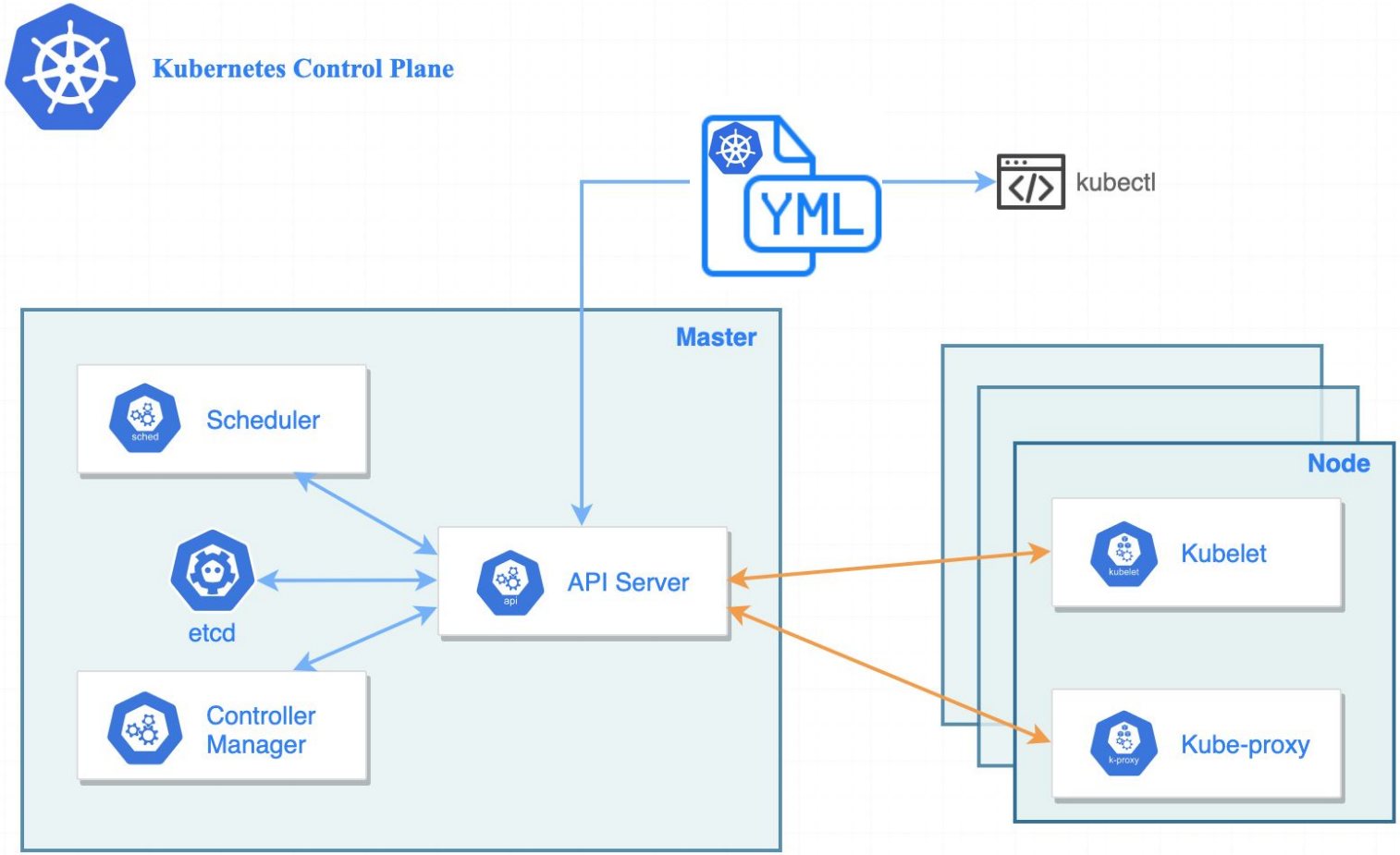
k8’s components

- Write apps as microservices in a language of your choosing
- Package each service in it’s own container (Dockerfile)
  - Build image
  - Upload image to a container registry
- Wrap each container in it’s own Pod
- Deploy pod to the cluster via Deployment, Daemonsets, Statefulsets, CronJobs, etc
  - kubernetes manifest

Architecture Overview



Kubernetes Control Plane

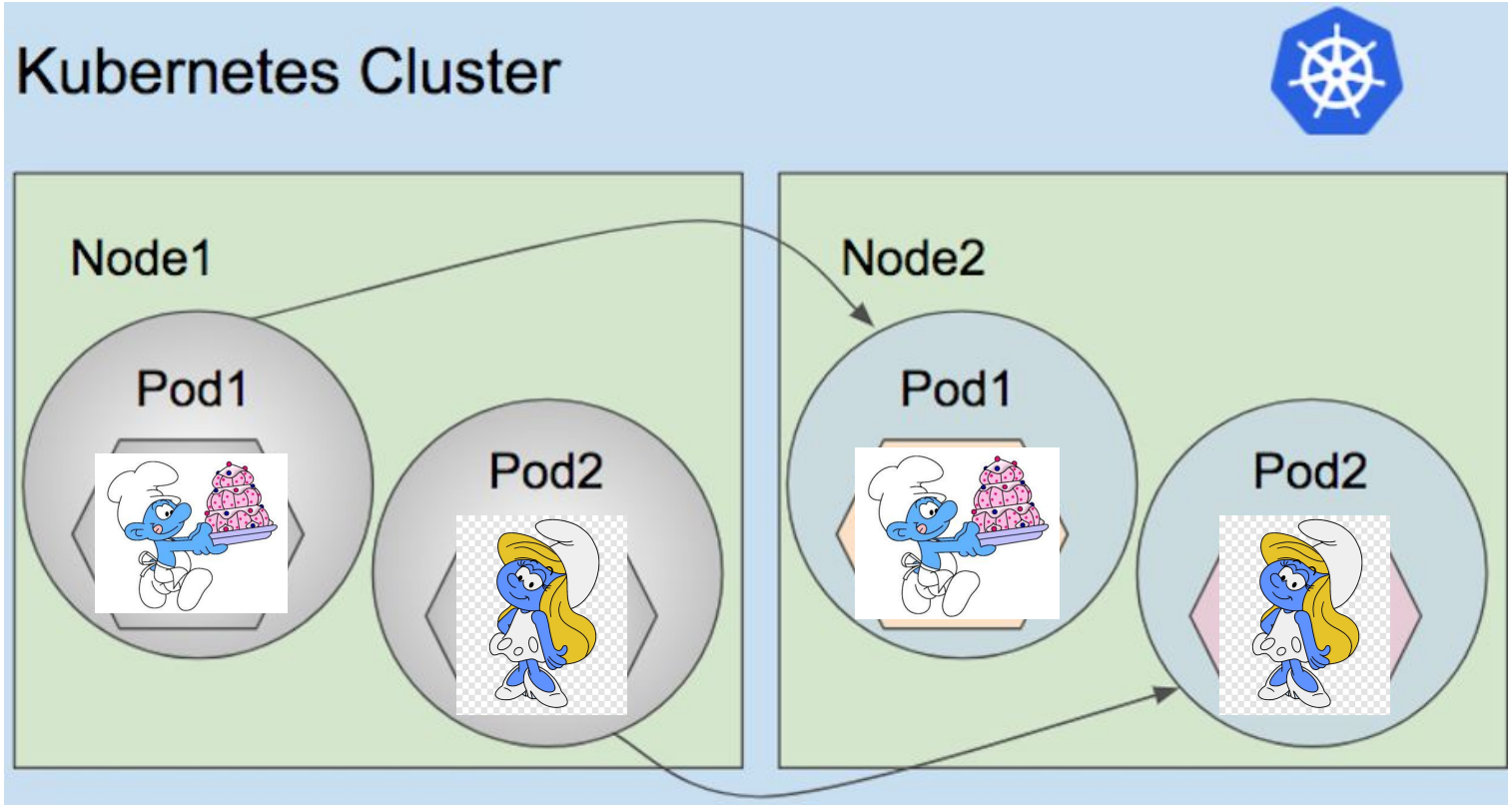

Kubernetes Control Plane
YML
kubectl
Master
Scheduler
sched
etcd
API Server
api
Controller
Manager
Node
Kubelet
kubelet
Kube-proxy
k-proxy

k8’s Objects

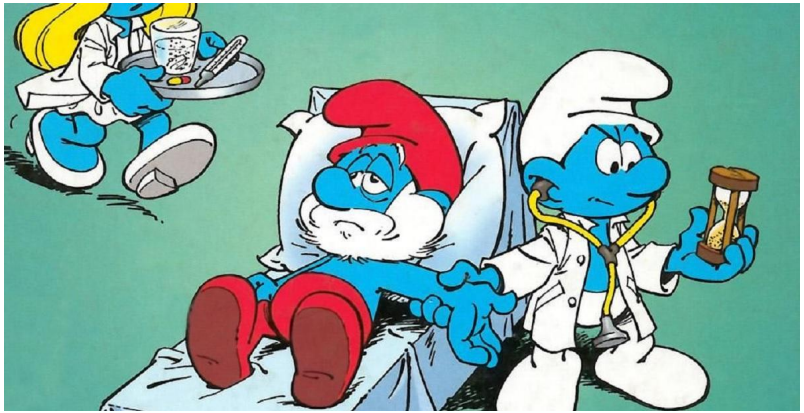
Namespaces

Pods

- Kubernetes API object
- Smallest atomic object
- Encapsulates containers
- Can house one or more containers
  - Best practice is have one container per pod unless the containers are tightly coupled
    - Logging container
    - Colocated on same server
    - Share same networking namespace (IP address and port)
    - Same hostname
  - Will these containers work correctly if they land on different nodes
- Rarely create individual pods directly - even single pods


Kubernetes Cluster
Node1
Pod1
Pod2
Node2
Pod1
Pod2

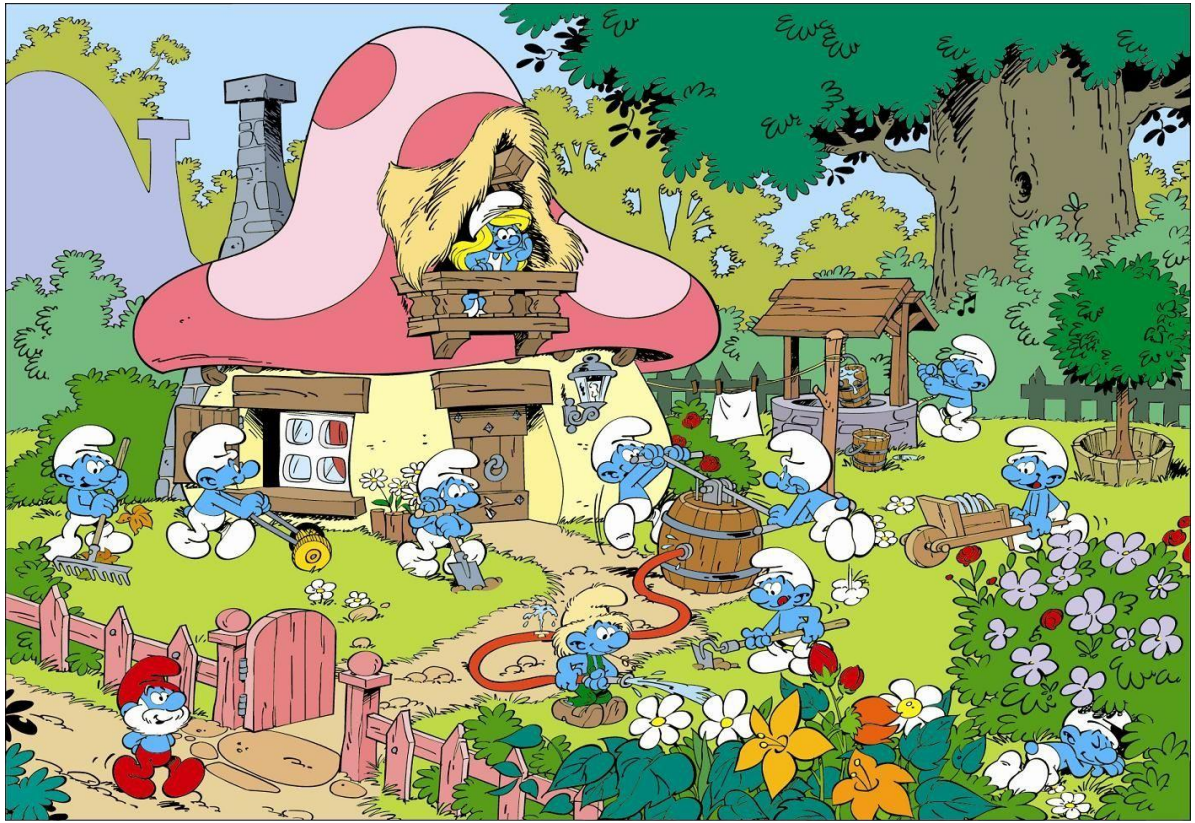
Pod Health and Liveness probes



```yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    test: liveness
  name: liveness-http
spec:
  containers:
  - name: liveness
    image: k8s.gcr.io/liveness
    args:
    - /server
    livenessProbe:
      httpGet:
        path: /healthz
        port: 8080
        httpHeaders:
        - name: Custom-Header
          value: Awesome
      initialDelaySeconds: 3
      periodSeconds: 3
```

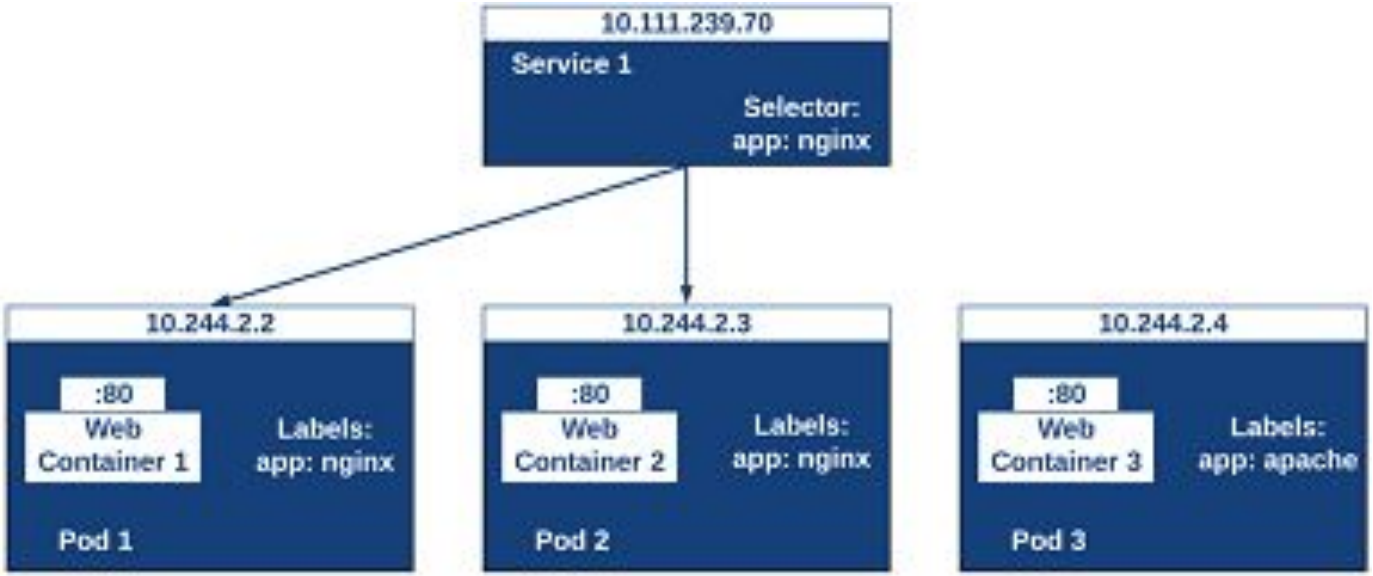

LET'S PRAY
TO THE DEMO GODS
makeameme.org

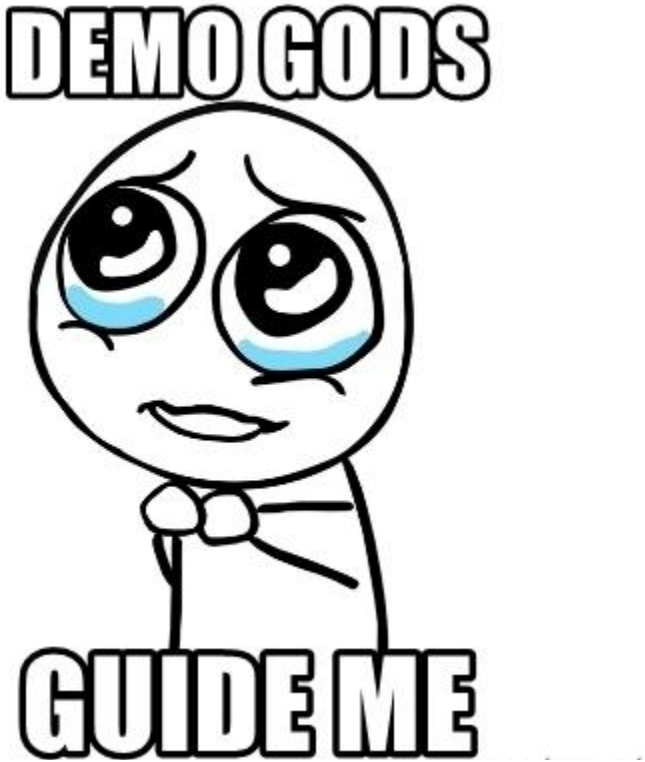
Services



Services

- Kubernetes API object
- Exposes an application running on a set of Pods
- Provides stable IP to pods
- Provides DNS
- Service Types
  - **ClusterIP** - Exposes the Service on an internal IP in the cluster. This type makes the Service only reachable from within the cluster.
  - **NodePort** - Exposes the Service on the same port of each selected Node in the cluster using NAT. Makes a Service accessible from outside the cluster using `<NodeIP>:<NodePort>`. Superset of ClusterIP.
  - **LoadBalancer** - Creates an external load balancer in the current cloud (if supported) and assigns a fixed, external IP to the Service. Superset of NodePort.


10.111.239.70
Service 1
Selector:
app: nginx
10.244.2.2
:80
Web
Container 1
Labels:
app: nginx
Pod 1
10.244.2.3
:80
Web
Container 2
Labels:
app: nginx
Pod 2
10.244.2.4
:80
Web
Container 3
Labels:
app: apache
Pod 3


DEMO GODS
GUIDE ME

ReplicaSet

ReplicaSets

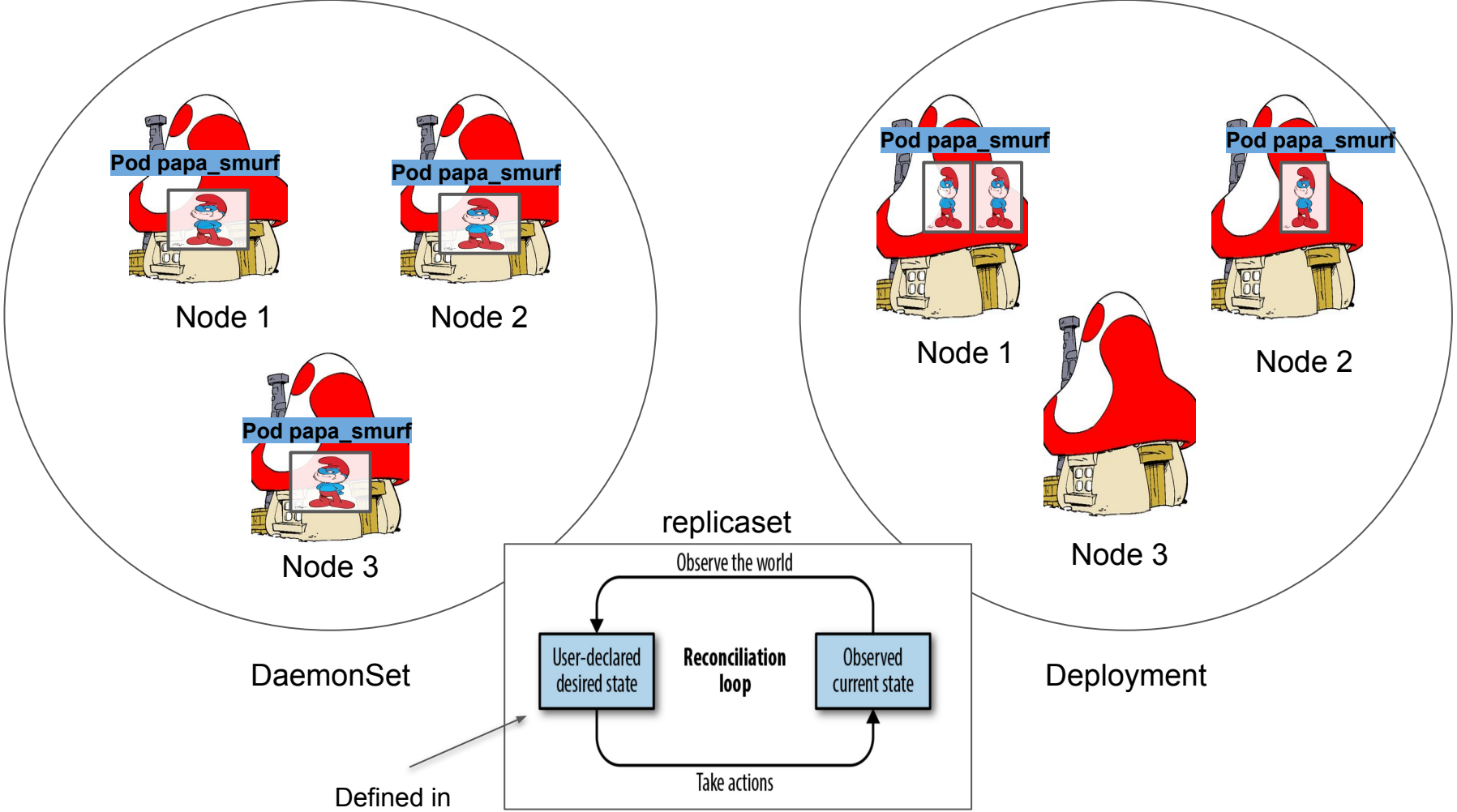
- Kubernetes API object
- Cluster wide pod manager
- Ensures right number and type of pods are running at all time
- Provides underpinnings of self healing, scaling up and down for apps

DaemonSets

- Kubernetes API object
- Ensures a copy of pod is running across a set of nodes in a k8’s cluster
  - Log collectors
  - Monitoring agents
- Managed by a reconciliation loop
  - New node is added to cluster it makes sure pod is created on the new node
- nodeSelector
  - Limits the daemonset to specific nodes

Deployments

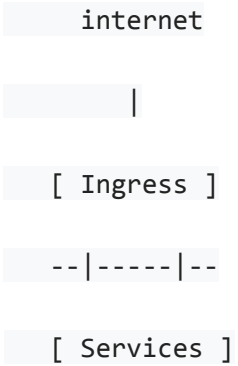
- Kubernetes API object
- Enables you to easily move from one version of code to the next version
  - Application rollouts
- Deployments managed by a Deployment controller
  - You describe a *desired state* in a Deployment, and the Deployment Controller changes the actual state to the desired state at a controlled rate.
- Builds off of ReplicaSets
  - Deployments manage ReplicaSets
- Updates/Rollouts/Rollbacks


Pod papa_smurf
Pod papa_smurf
Node 1
Node 2
Pod papa_smurf
Node 3
DaemonSet
Pod papa_smurf
Pod papa_smurf
Node 1
Node 2
Node 3
Deployment
replicaset
Observe the world
User-declared desired state
Reconciliation loop
Observed current state
Take actions
Defined in Kubernetes manifest


LET'S PRAY
TO THE DEMO GODS

Ingress

- Ingress Object (Kubernetes API object)
- Ingress rules
- Ingress Controller
  - Nginx
  - Traefik
  - AKS Application Gateway Ingress Controller
  - HAProxy Ingress
  - AWS ALB Ingress Controller
-



```yaml
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: minimal-ingress
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
  - http:
      paths:
      - path: /testpath
        pathType: Prefix
        backend:
          service:
            name: test
            port:
              number: 80
```

Configmaps

- API object used to store non-confidential data in key-value pairs.
- Pods can consume ConfigMaps
  - Command line arguments to the entrypoint of a container
  - Environment variables for a container
  - Add a file in read-only volume, for the application to read
  - Write code to run inside the Pod that uses the Kubernetes API to read a ConfigMap

Secrets

- Kubernetes API object
- Contains small amount of sensitve data
- Can be used in a pod via:
  - As files in a volume
  - Mounted on one or more of its containers.
  - As container environment variable.
  - By the kubelet when pulling images for the Pod


DEMO GODS
I SMURF TO YOU
imgflip.com

How do we interact w/ Kubernetes

+

kubectl

- Interacts with the k8's API
- Manages most k8's objects
- Explore, Verify, Create, Update and Destroy
  - kubectl <verb> <object type> <object>
    - kubectl describe
    - kubectl get
    - kubectl delete
    - kubectl create
    - kubectl apply

Resources

- Kubernetes docs (https://kubernetes.io/docs/home/)
- Minikube (https://github.com/kubernetes/minikube)
- Kubernetes up and running
  https://www.amazon.com/Kubernetes-Running-Dive-Future-Infrastructure/dp/1492046531
- https://labs.play-with-k8s.com/
- Kubernetes cheat sheet
  (https://kubernetes.io/docs/reference/kubectl/cheatsheet/)