




Cloudless k8s: Lessons after 3 years

Who am I

- Nadia Santalla Fdez. (she/her)
 -  <https://nadia.moe>
 -  `mailto:nadia@nadia.moe`
- Senior software engineer @ Grafana
- I read books and collect machine tools
-  All things infrastructure
 - K8s



Agenda

1. Intro to self-managed k8s
2. `kubeadm`
3. `kine`
4. `cilium`
5. `MetaLB`
6. `txqueuelen/stateless-dns`
7. Others

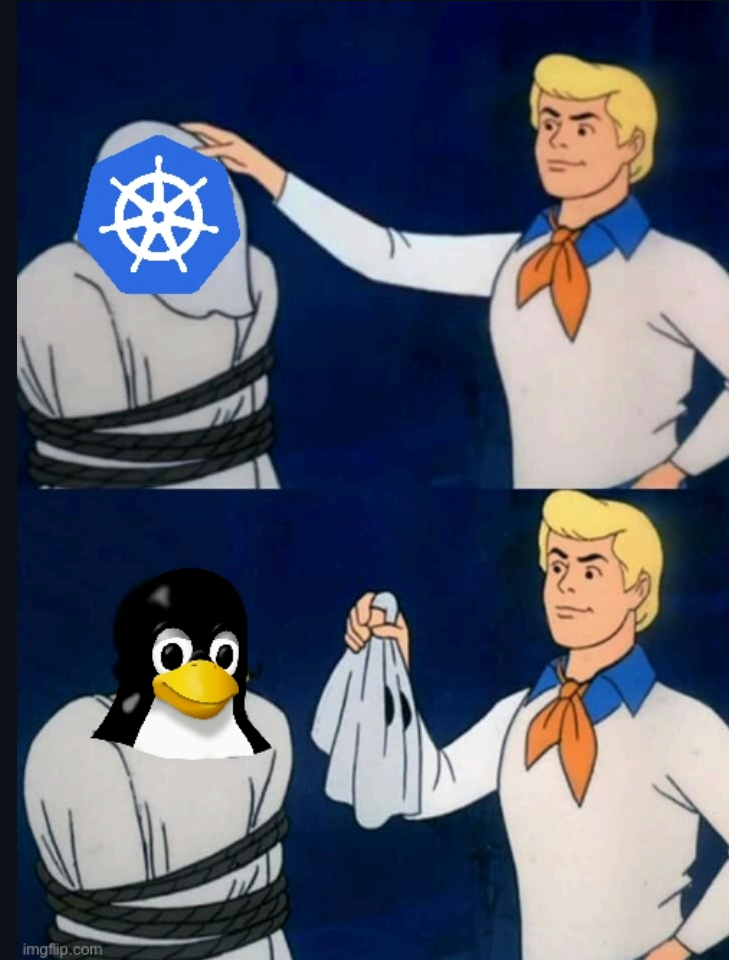
Intro to self-managed k8s

Out of the cloud? That's when

- You manage your own hardware
- You manage your own control plane
- You do not rely on external services (DNS)

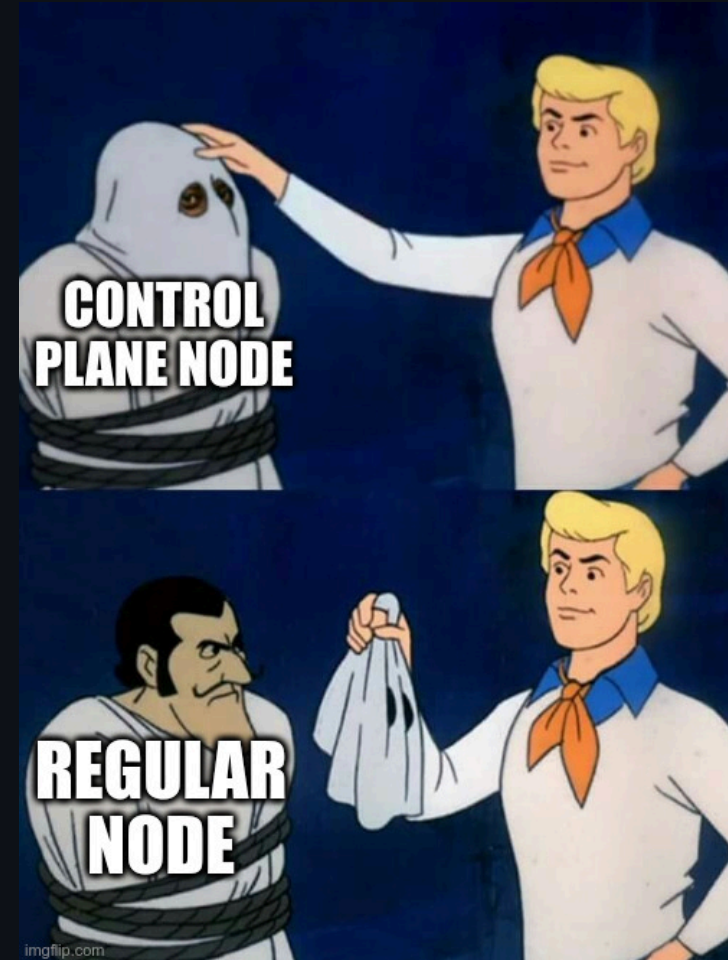
What is a Kubernete? (I mean, node)

- A properly configured, running `kubelet` process
 - Connects to the k8s API and looks for `Pods` with a specific `nodeName`
- The `kubelet` binary:
 - Lives on a host (linux) filesystem
 - Has configuration flags and files
 - Depends on other binaries
 - Should be ran at boot



What is a control plane?

- A series of services that make Kubernetes work
 - API
 - Database
 - Bunch of clients using the API
- These services often run on Kubernetes itself
- Often run machines dedicated to them (QoS)
- A control plane node is just a node where (only) the control plane runs
 - Plus some conventional taints and labels



How does a control plane look in-cluster?

- A bunch of pods in `kube-system`
 - DB, API server, and controllers
 - Persist data on disk (`hostPath`)
- These pods are responsible for
 - Serving the k8s API
 - Turning `Deployments` into `Pods`
 - Setting `nodeName` in `Pod` objects
- When this happens, the `kubelet` creates containers for them
- 🤔 Who creates the pods that create the pods?

```
$> kubectl get pods -n kube-system
NAME                                STATUS
cilium-dd1l4                        Running
cilium-envoy-vk2h1                  Running
cilium-operator-555cfb9b88-tbz6d    Running
coredns-7c65d6cfc9-lsvnh           Running
etcd-ubuntu-4gb-nbg1-1              Running
kube-apiserver-ubuntu-4gb-nbg1-1    Running
kube-controller-manager-ubuntu-4gb-nbg1-1 Running
kube-scheduler-ubuntu-4gb-nbg1-1    Running
```


Static manifests!

- Static manifests are just files on disk
- The `kubelet` blindly spins these pods up
 - (i.e. it creates containers for them)
- In parallel, it tries to connect to the apiserver
 - When connected, registers these pods there (mirror pods)
- Restrictions apply, but it's enough

```
# cat /etc/kubernetes/manifests/kube-apiserver.yaml
apiVersion: v1
kind: Pod
metadata:
  labels:
    component: kube-apiserver
    tier: control-plane
    name: kube-apiserver
    namespace: kube-system
spec:
  containers:
  - command:
    - kube-apiserver
    - --advertise-address=10.0.0.30
    # ...
  volumes:
  - hostPath:
    path: /etc/kubernetes/pki
  # ...
```

So what's Kubernetes, really?

- A daemon running on a linux box
 - Any distro will work
 - (As long as you are okay taking care of it)
- Fleet management tools like Ansible help
- You'll encounter linux problems. Use a linux distro you're comfortable with.
- Make sure your K8s distribution is not too opinionated or inflexible.



~~Lessons learned~~

Things I do



Kubeadm

- Kicking the tires on that binary is not trivial
 - Generating static manifests
 - Generating consistent configuration files
 - Adding conventional labels and taints
 - Creating RBAC objects
 - Creating TLS certificates
 - Deploy addons (kube-proxy, coredns)
 - Renovating certificates
- Kubeadm can do all this!

```
apiVersion: kubelet.config.k8s.io/v1beta1
kind: KubeletConfiguration
# Use a resolv.conf file without the curry domain/search to prevent ndots spam.
resolvConf: /etc/resolv-k8s.conf
# Wait this duration for pods to gracefully terminate.
shutdownGracePeriod: 40s
# Of the duration above, reserve this for critical pods.
shutdownGracePeriodCriticalPods: 20s
---
apiVersion: kubeadm.k8s.io/v1beta3
kind: ClusterConfiguration
clusterName: Curry
controlPlaneEndpoint: '{{ apiserver_host }}:6443'
networking:
  dnsDomain: cluster.local
  podSubnet: 10.185.0.0/16
  serviceSubnet: 10.96.0.0/12
etcd:
  external:
    endpoints:
      - 'https://{{ etcd_host }}:2379'
    certFile: /etc/kubernetes/pki/apiserver-etcd-client.crt
    keyFile: /etc/kubernetes/pki/apiserver-etcd-client.key
    caFile: /etc/kubernetes/pki/etcd/ca.crt
---
apiVersion: kubeadm.k8s.io/v1beta3
kind: InitConfiguration
nodeRegistration:
  ignorePreflightErrors:
    - ExternalEtcdVersion
skipPhases:
  # Cilium replaces kube-proxy.
  - addon/kube-proxy
```

Kine instead of etcd

<https://github.com/k3s-io/kine>

- etcd is great software built with certain tradeoffs in mind
 - SSD life does not seem to be one of them
- etcd chewed 10% of my SSD lifespan in a year
- Postgres chews nearly nothing
 - *This is anecdotal evidence
- Kine allows using SQL DBs in place of etcd
- Kine can be deployed easily with kubeadm

```
# cat /etc/kubernetes/kubeadm.yaml
# ...
etcd:
  external:
    endpoints:
      - https://10.0.0.9:2379
    certFile: /etc/kubernetes/pki/apiserver-etcd-client.crt
    keyFile: /etc/kubernetes/pki/apiserver-etcd-client.key
    caFile: /etc/kubernetes/pki/etcd/ca.crt
```

```
# cat /etc/kubernetes/manifests/kine.yaml
apiVersion: v1
kind: Pod
spec:
  hostNetwork: true
  containers:
    - name: kine-postgres
      image: docker.io/postgres:15.8-alpine
      # ...
    - name: kine-kine
      image: docker.io/rancher/kine:v0.11.12-amd64
      # ...
```

cilium as the CNI plugin

- Easily configurable through Helm chart
- Extremely well documented
- Works without `kube-proxy` (Kubeadm does too!)
- Less hard to debug
- Lots of knobs to tweak for metal
 - Interface names, encapsulation, custom routes, etc.
- You may want to delete nodeports
 - <https://github.com/txqueuelen/cilium-nodeport-deleter>

cilium egress gateway

- Egress gateways allow to control which IP address is used for outgoing traffic
- Useful for multitenancy
 - Map tenants to IPs instead of nodes
 - Isolating IP reputation problems
- Designated egress nodes

```
apiVersion: cilium.io/v2
kind: CiliumEgressGatewayPolicy
metadata:
  name: nadia
spec:
  selectors:
    - namespaceSelector:
        matchLabels:
          owner: nadia
  destinationCIDRs: [ "0.0.0.0/0" ]
  egressGateway:
    egressIP: 88.99.146.130 # Nadia's IP.
    nodeSelector: {} # Required even if it's empty.
```

You'll probably need a load balancer

- Pod and service IPs are not world-reachable
- Your nodes may have public IPs (or a gateway)
- Traffic from the internet cannot easily reach your pods, svcs, and ingress controller
- SVC depends on a single IP being reachable
- NodePort
 - Scheduling constraints
- ExternalIP
 - Which node's? SPOF
- ... Or a load balancer

```
networking:  
  dnsDomain: cluster.local  
  podSubnet: 10.185.0.0/16  
  serviceSubnet: 10.96.0.0/12
```


📍 MetalLB as a load balancer

<https://metallb.io/>

- Allows defining you any number of virtual IPs
- Nodes take turns to announce VIPs
 - No need to bind them to nodes
- CNF routes input traffic through the cluster
- Implements failover using Gossip
 - https://en.wikipedia.org/wiki/Gossip_protocol
- Also supports BGP

```
---
apiVersion: metallb.io/v1beta1
kind: IPAddressPool
metadata:
  name: outofcloud
  namespace: metallb
spec:
  addresses:
    - 116.203.0.9/32
---
apiVersion: metallb.io/v1beta1
kind: L2Advertisement
metadata:
  name: outofcloud
  namespace: metallb
spec:
  ipAddressPools:
    - outofcloud
```

MetalLB demo

LoadBalancers and externalTrafficPolicy

<https://kubernetes.io/docs/tasks/access-application-cluster/create-external-load-balancer/#preserving-the-client-source-ip>

- LoadBalancers are great for exposing services
- But are ultimately designed for balancing load
- By default, traffic is spread across all nodes
 - externalTrafficPolicy: Cluster
- MetalLB will announce from any node
- Routing to other nodes requires natting
- Some peculiar services will not like this

```
apiVersion: v1
kind: Service
metadata:
  name: transmission-torrent
  annotations:
    metallb.universe.tf/address-pool: curry-torrent
spec:
  type: LoadBalancer
  # ⬇️ Avoid obscuring source IP.
  externalTrafficPolicy: Local
  selector:
    app.kubernetes.io/name: transmission
  ports:
    - protocol: TCP
      name: torrent-tcp
      port: 31415
    - protocol: UDP
      name: torrent-udp
      port: 31415
```

Kubernetes-aware DNS for external names

- As the number of applications you host grows, so does the number of domains
 - Wildcard records may fail short
- External-dns creates A and AAAA records for
 - Ingresses
 - `LoadBalancer` services
 - Custom resources
- ... but external DNS is not a DNS server



DNS with txqueue1en/stateless-dns

stateless-dns is a Helm chart that combines external-dns and a stateless setup of PowerDNS

<https://github.com/txqueuelen/stateless-dns> (@kilokang)

- Makes PowerDNS stateless
 - Create DNS zones at boot
 - Configure nits & grits of serial updates
 - Configures pdns API
- Preconfigures external-dns to point to it
- Allows to define zones as text
- Highly available*

```
# values.yaml
service:
  dnsUdp:
    enabled: true
    type: LoadBalancer
    annotations:
      external-dns.alpha.kubernetes.io/hostname: >-
        ns1.nadia.moe,ns2.nadia.moe
  pdns:
    apiKeySecret:
      create: true
      valuePlain: hunter2
  zones:
    nadia.moe: |
      $ORIGIN .
      nadia.moe 900 IN SOA ns1.nadia.moe nadia.nadia.moe 1 900 600 172800 60
      nadia.moe 900 IN NS ns1.nadia.moe
      nadia.moe 900 IN NS ns2.nadia.moe
```

stateless-dns demo

Misc tools & approaches

- Gitops, gitops, gitops
 - Outsource your brain to git
 - ArgoCD & FluxCD are awesome
 - Renovatebot ❤️ Gitops
 - Forgejo works fantastically
- Secrets management
 - 💔 Sops, SealedSecrets
 - Separate repository with YAMLS
- Databases
 - CNPG <https://cloudnative-pg.io/>
 - Provisioning, user management, failover
- Manifests over Helm
 - All manifests have the same schema
 - Every `values.yaml` is different
 - Do not depend on upstream for tweaks
- Storage
 - Statically generated `hostpath` PV(C)s

Things I'd love to try

- OS
 - Something with A/B updates
 - Talos, Flatcar
- Storage
 - MinIO & AWS Mountpoint (PVs on S3)
- Architecture
 - Multi-region, wireguard-based networking
 - Egress on (disposable) cloud

Q&A

Thank you! 