KubeCon

CloudNativeCon

North America 2017

kubeadm Cluster Creation Internals

From Self-Hosting to Upgradability and HA

Lucas Käldström

7th December 2017 - KubeCon Austin

$ whoami

Lucas Käldström, Upper Secondary School Student, just turned 18

CNCF Ambassador and **Certified Kubernetes Administrator**

Speaker at **KubeCon Berlin** 2017 and now here at **KubeCon Austin**

Kubernetes Maintainer since April 2016, active in the community for +2 years

SIG Cluster Lifecycle co-lead and **kubeadm** maintainer

Driving **luxas labs** which currently performs contracting for Weaveworks

A guy that has never attended a computing class

Image credit: Dan Kohn

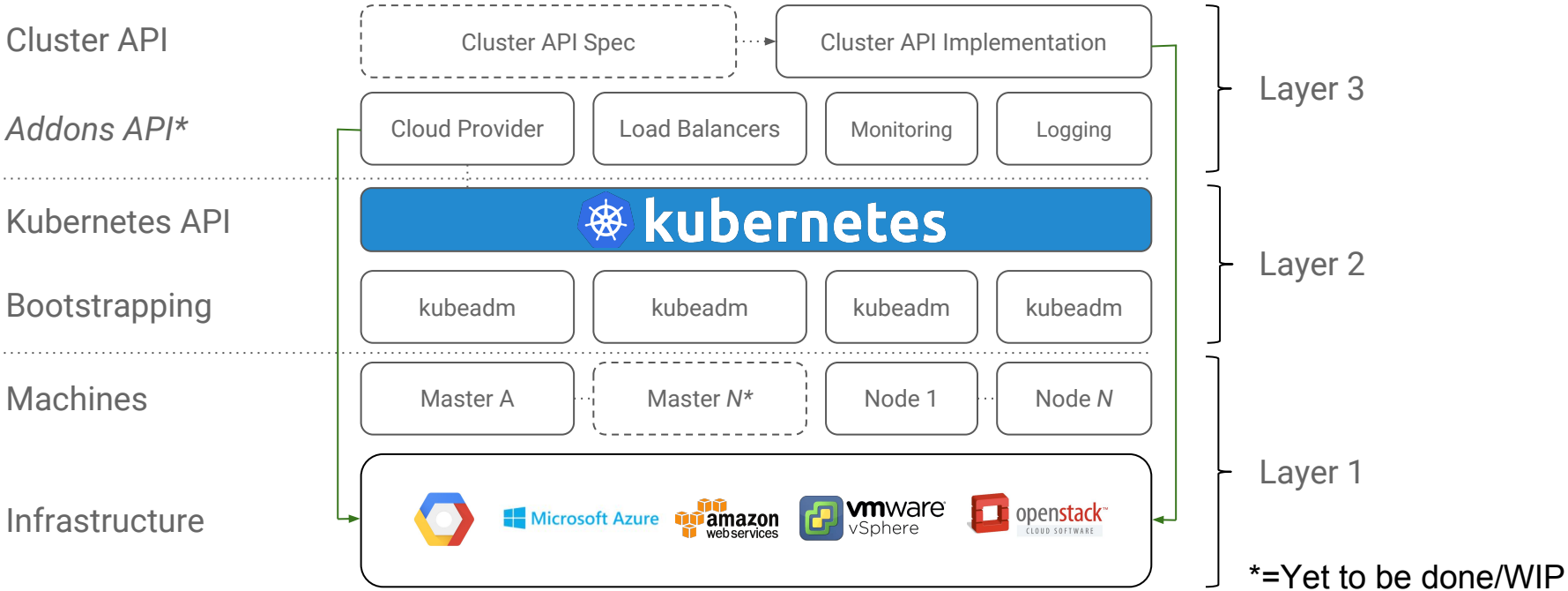
Agenda

1. What kubeadm is & how it fits into the ecosystem
2. What’s common for every TLS-secured Kubernetes cluster
3. What self-hosting is & how you can self-host your cluster
4. How kubeadm handles upgrades
5. How high availability can be achieved with kubeadm

This talk will dive deep into the technical aspects of creating Kubernetes clusters

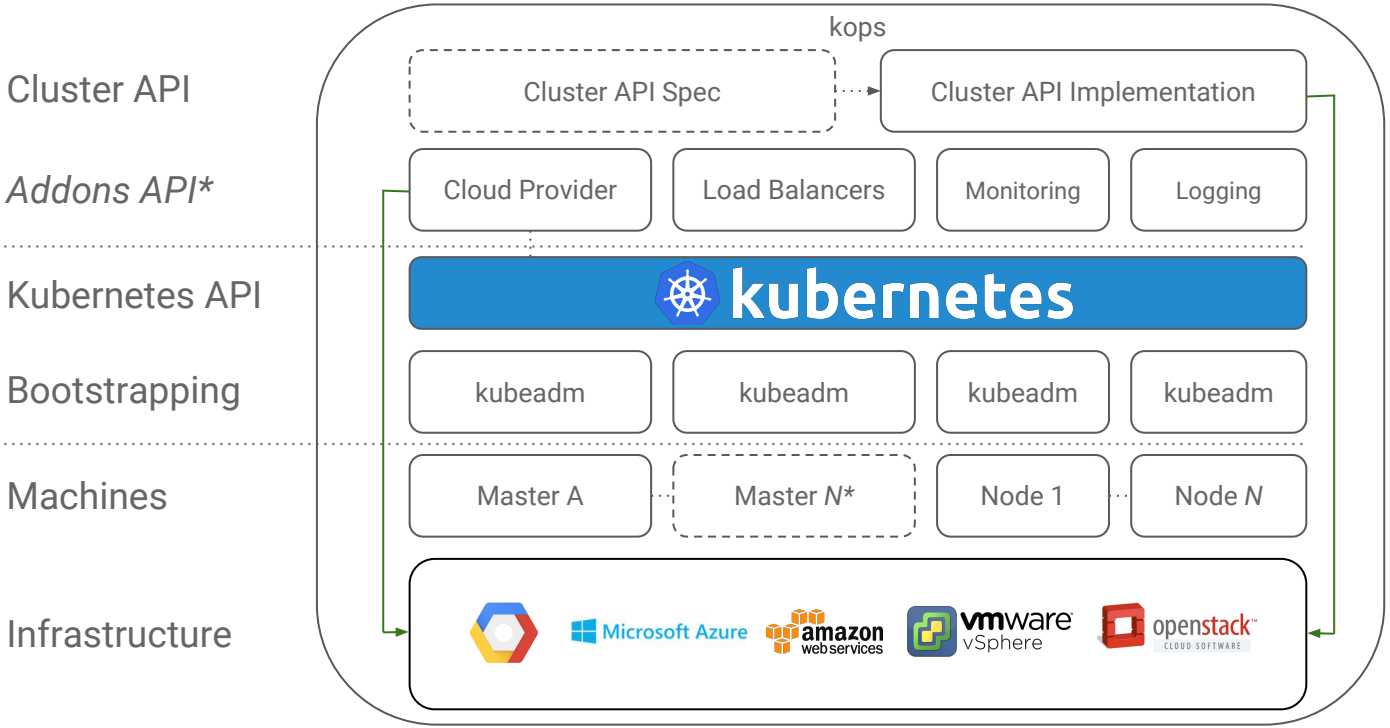
What is kubeadm and why should I care?

= A tool that sets up a minimum viable, best-practice Kubernetes cluster



kubeadm vs kops

Two different projects, two different scopes



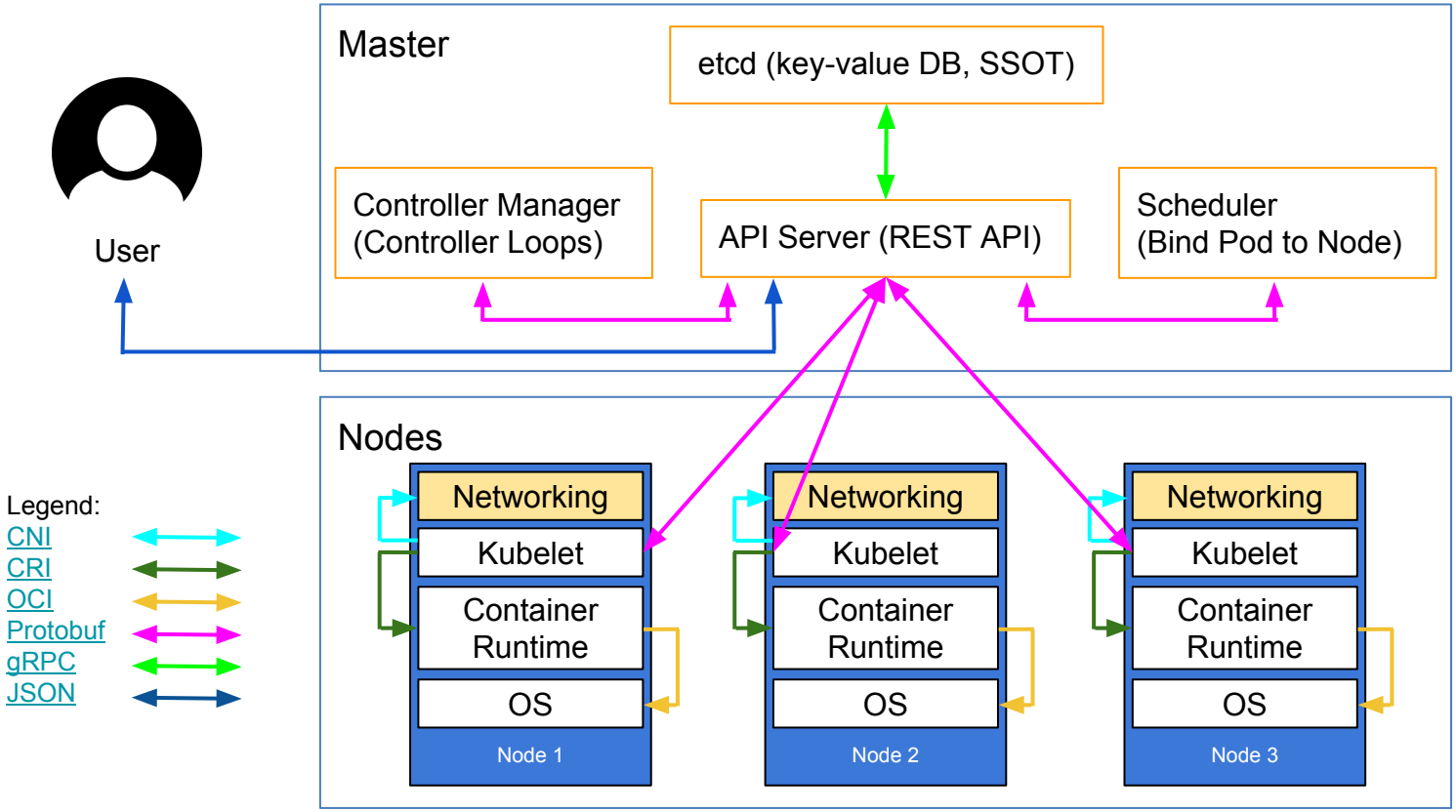
*=Yet to be done/WIP

Key design takeaways

- kubeadm's task is to set up a **best-practice cluster** for each *minor version*
- The user experience should be *simple*, and the cluster reasonably *secure*
- kubeadm's scope is limited; intended to be a **building block**
  - Only ever deals with the local filesystem and the Kubernetes API
  - Agnostic to ***how exactly*** the kubelet is run
  - Setting up or favoring a specific CNI network is **out of scope**
- Composable architecture with everything divided into **phases**

Audience: build-your-first-own-cluster users & higher-level tools like *kops & kubicorn*

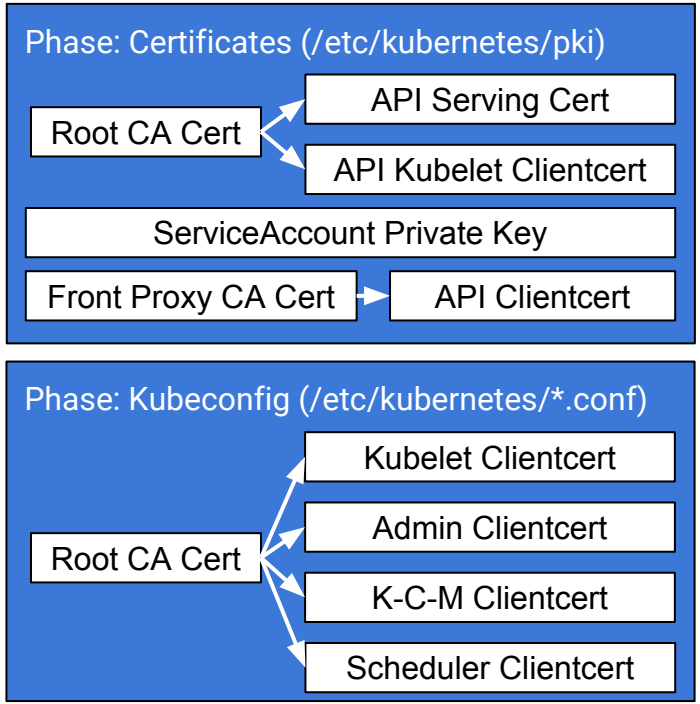
Kubernetes' high-level component architecture



Needed for all clusters

kubeadm-specific

What does `kubeadm init` really do -- part 1?



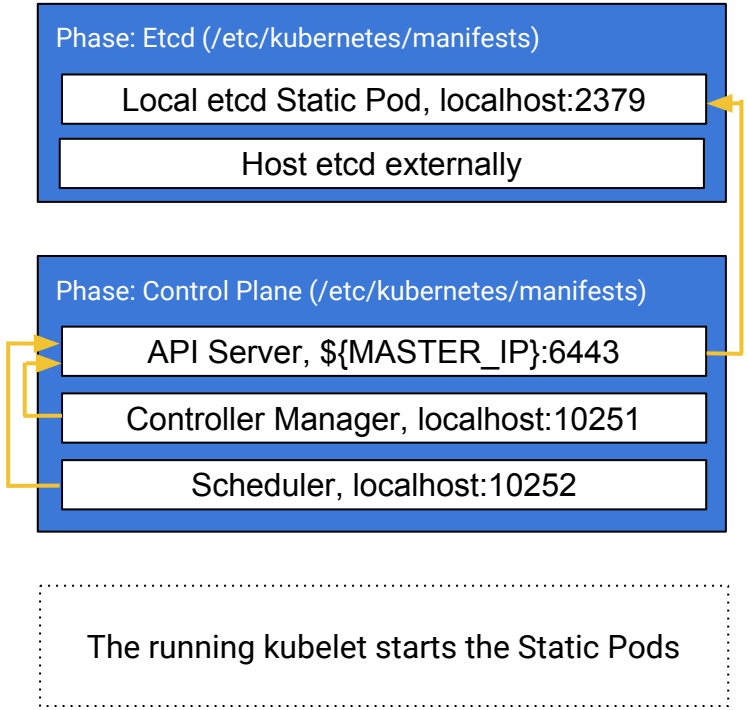
First kubeadm creates the necessary certificates for setting up a cluster with TLS-secured communication and API Aggregation support.

Then client certificates are generated in KubeConfig files for the first actors that need identities

Needed for all clusters

kubeadm-specific

What does `kubeadm init` really do -- part 2?



kubeadm generates Static Pod manifest files for etcd (or use external etcd) and control plane components

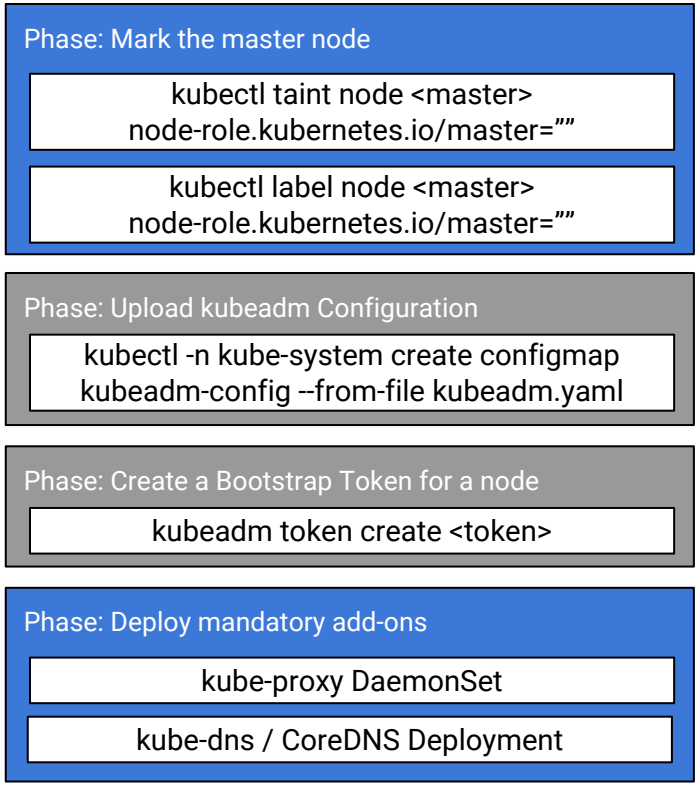
kubelet runs the Static Pods and hence kubeadm assumes a running kubelet

kubeadm waits for the kubelet to start the Static Pods, an operation that may fail in many ways

Needed for all clusters

kubeadm-specific

What does `kubeadm init` really do -- part 3?



In order to make the master exclusive and identifiable, it is tainted and labelled with a common-practice label

For `kubeadm upgrade` to remember the config passed to `kubeadm init`, the config is uploaded to the cluster

A Node Bootstrap Token is created and granted privileges to add a node

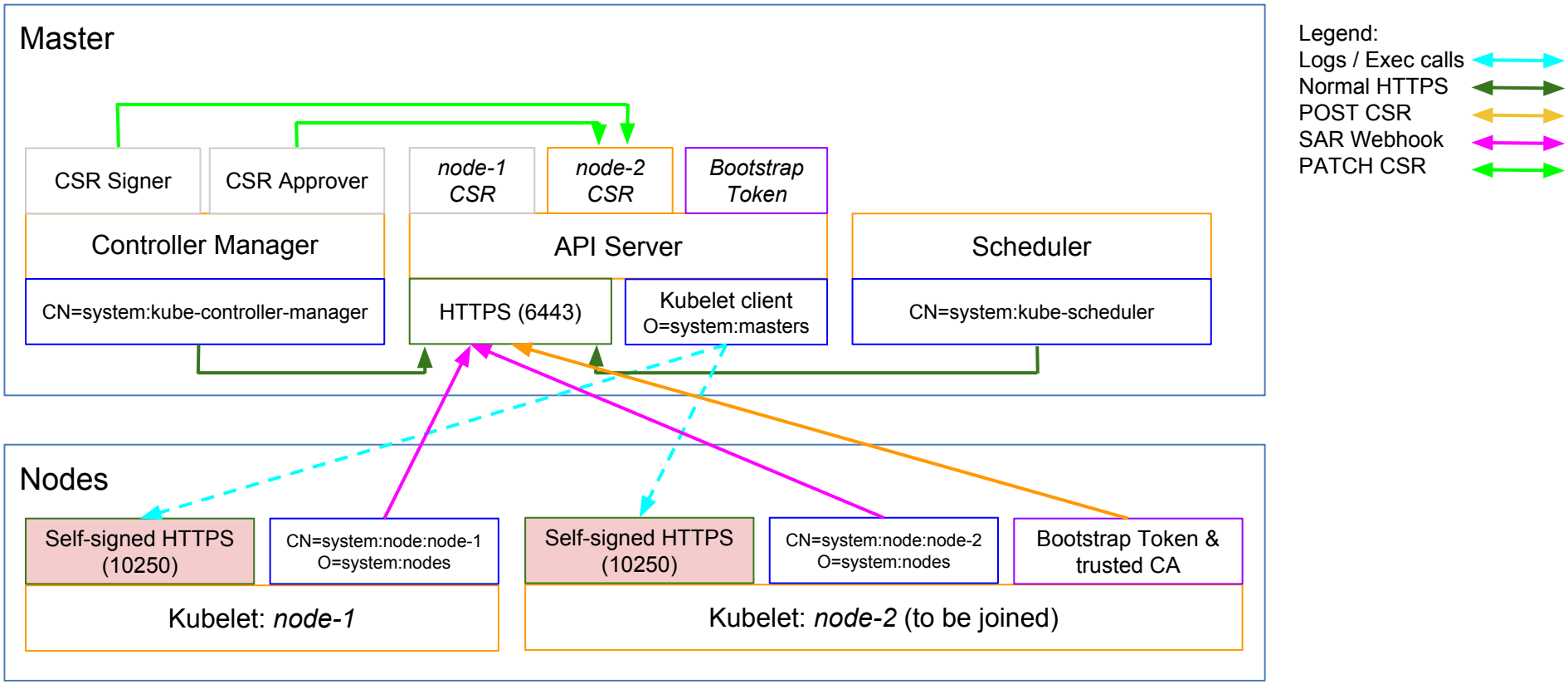
Lastly, kube-proxy and kube-dns / CoreDNS are deployed

Build your cluster from scratch with `kubeadm phase`

The `kubeadm phase` command lets you invoke atomic sub-tasks of a full installation operation. You don’t have to choose the “full meal deal” anymore (kubeadm init). This is what `kubeadm init` looks like phase-wise:

```
$ kubeadm alpha phase certificates all
$ kubeadm alpha phase kubeconfig all
$ kubeadm alpha phase etcd local
$ kubeadm alpha phase controlplane all
$ systemctl start kubelet
$ kubeadm config upload from-flags
$ kubeadm alpha phase mark-master $(kubectl get no --no-headers | awk ‘{print $1}’)
$ kubeadm alpha phase bootstrap-token cluster-info /etc/kubernetes/admin.conf
$ kubeadm alpha phase bootstrap-token node allow-post-csrs
$ kubeadm alpha phase bootstrap-token node allow-auto-approve
$ kubeadm token create
$ kubeadm alpha phase addons kube-dns
$ kubeadm alpha phase addons kube-proxy
```

Setting up a dynamic TLS-secured cluster



CSR=Certificate Signing Request, SAR=Subject Access Review

kubeadm & self-hosted Kubernetes cluster

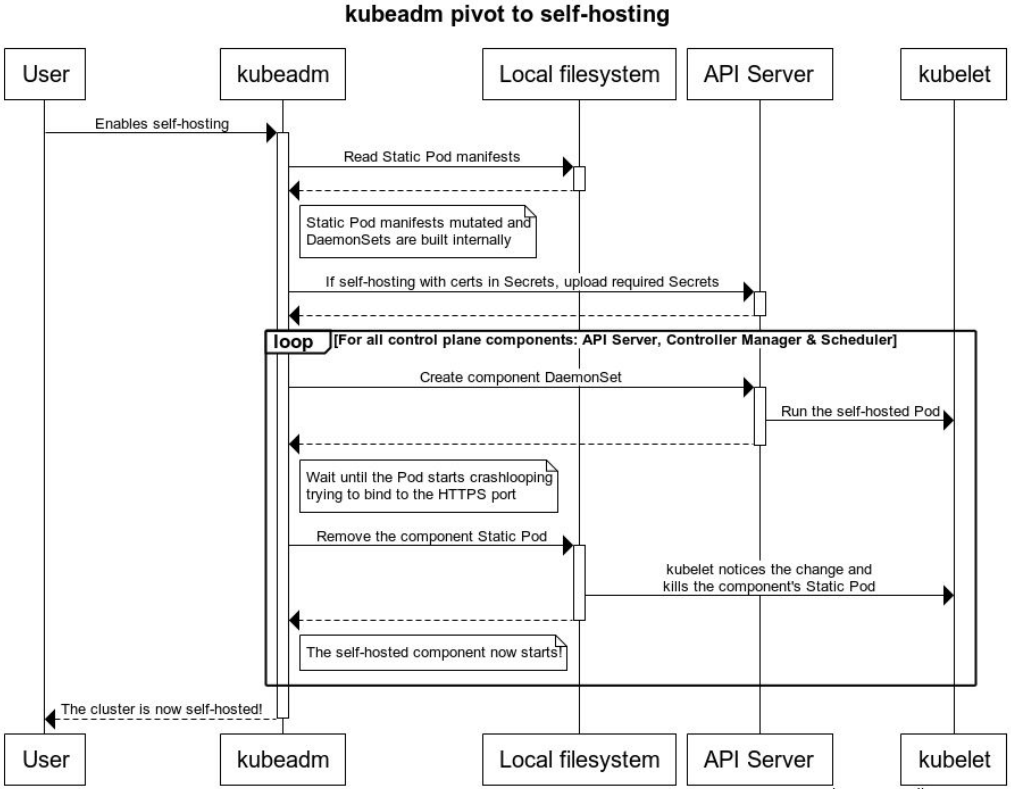
Self-hosting
=using Kubernetes primitives (e.g DaemonSets, ConfigMaps) to run and configure the control plane itself

The self-hosting concept was initially developed by CoreOS and the bootkube team. Also see the other self-hosting talk here at KubeCon.

We're now in the process of "upstreaming" the work done in bootkube, in a way that makes it easier for ***any*** Kubernetes cluster to be *self-hosted*.

Building blocks that can be used for pivoting to self-hosting exist in *kubeadm*

How is a self-hosted cluster bootstrapped?



kubeadm either hosts the control plane in Static Pods managed in the filesystem or as self-hosted DaemonSets.

The process is modular; it creates the self-hosted control plane from the current state of the world of Static Pods.

Upgrading clusters with kubeadm

1. Upgrades made easy with `kubeadm upgrade plan` and `kubeadm upgrade apply`
2. In v1.8, the upgrades basically shifted Static Pod files around on disk
3. Self-hosted upgrades in HA clusters work by modifying normal Kubernetes resources
4. From v1.9 we're supporting automated downgrades as well
5. In a future release we'll look at "runtime reconfiguration"

    i.e. invoking an upgrade with new config but the same version number
6. Read the proposal

The “just works” kubeadm HA feature request

Adding support for joining masters is the most popular feature request

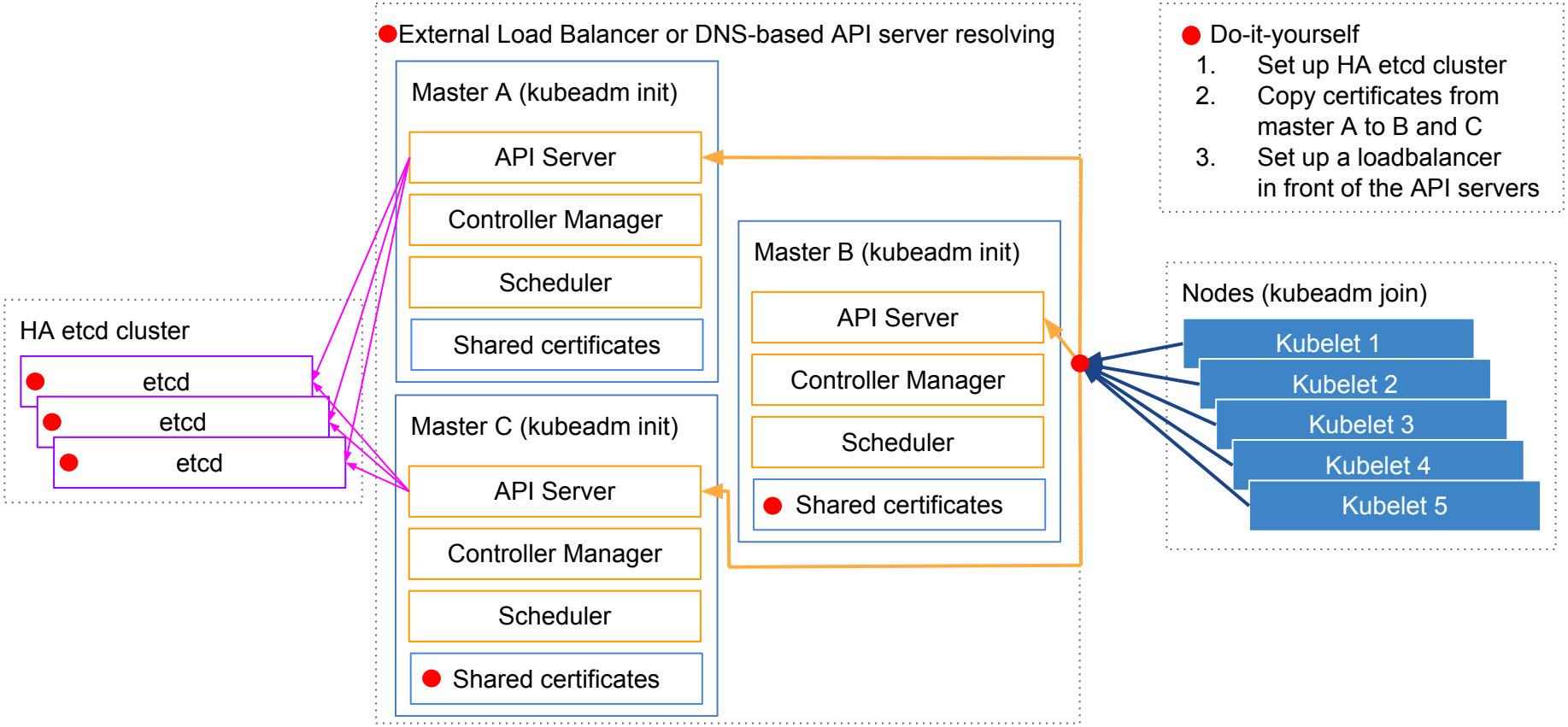
What challenges do we face in making a slick multi-master UX like this?

```
$ kubeadm init
...
 - Add node: kubeadm join --token abcdef.0123456789abcdef 192.168.1.100:6443
 - Add master: kubeadm join master --token fedcba.fedcba09876543210 192.168.1.100:6443
```

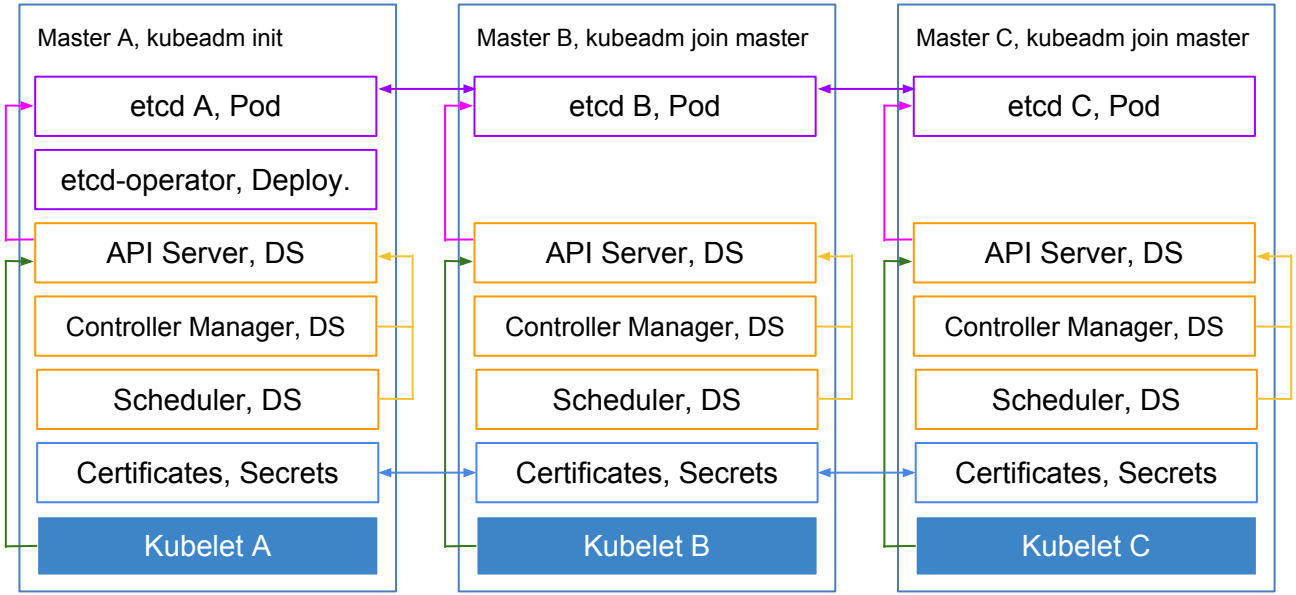
Three primary challenges:

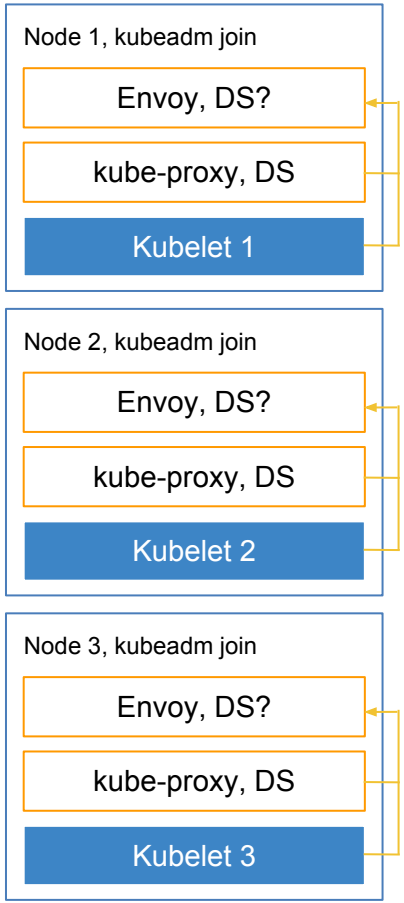
- Managing state reliably in an HA etcd cluster with TLS communication
- Sharing & rotating the common certificates from master A to master B
- Making the kubelets able to address all masters & dynamic reconfig

How achieve HA with kubeadm today?



One possible (?) solution





[Read the full proposal document]

DS=DaemonSet

What now?

Follow the SIG Cluster Lifecycle YouTube playlist

Check out the meeting notes for our weekly SIG meetings in Zoom

Join the #sig-cluster-lifecycle (for dev) and #kubeadm (for support)

Check out Diego Pontoriero' talk "Self-Hosted Kubernetes: How and Why"

Read the two latest SIG updates on the Kubernetes blog in January and August

Check out the kubeadm setup guide, reference doc and design doc

Read what the kubeadm contribution / work cycle looks like and start contributing to kubeadm!

Thank you!

@luxas on Github
@kubernetesonarm on Twitter
lucas@luxaslabs.com