KubeCon | CloudNativeCon

North America 2018

#kubeadm deep dive

Agenda

- Who?
- Mission
- GA
- Roadmap 2019
- Getting Involved
- Q/A

Who?

Who are we?

Timothy St. Clair
SIG Cluster Lifecycle co-lead
Steering Committee Member
Staff Engineer @Heptio/VMWare
@timothysc

Liz Frost
SIG Cluster Lifecycle Contributor
Kube Cuddle creator
SW Engineer @Heptio/VMWare
@liztio

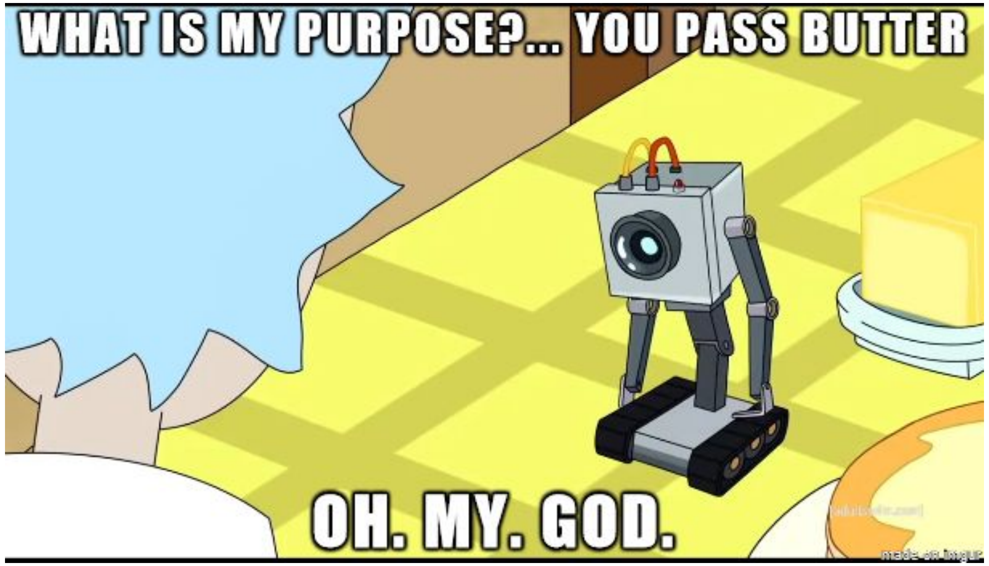
Who are we?

- 100s of contributors across several companies
- Smaller core group of active maintainers
  - VMWare
    - Lubomir, Ross
  - VMWare (née Heptio)
    - Tim, Liz, Jason, Chuck
  - Suse
    - Marek, Rafael
  - Intel
    - Alex, Ed
  - Other/Independent
    - Luxas, Fabrizio, Yago, Di
- Large user community on #kubeadm

Mission

What is our mission?

SIG Cluster Lifecycle's objective is to simplify creation, configuration, upgrade, downgrade, and teardown of Kubernetes clusters and their components.



<BRACE FOR RANT}

Why are we doing this?

- To prevent the mistakes of other open source cluster mgmt provisioning tools
    - Because…
        - Kubernetes is the beginning of the story, not the end
        - commoditizing the deployment of the core raises all boats and allows the community to focus on solving end user problems
        - “production grade” shouldn’t be firewalled by providers
        - It should “just work”
        - Because cross provider matters
- To make the management of (X) clusters across (Y) providers simple, secure, and configurable.

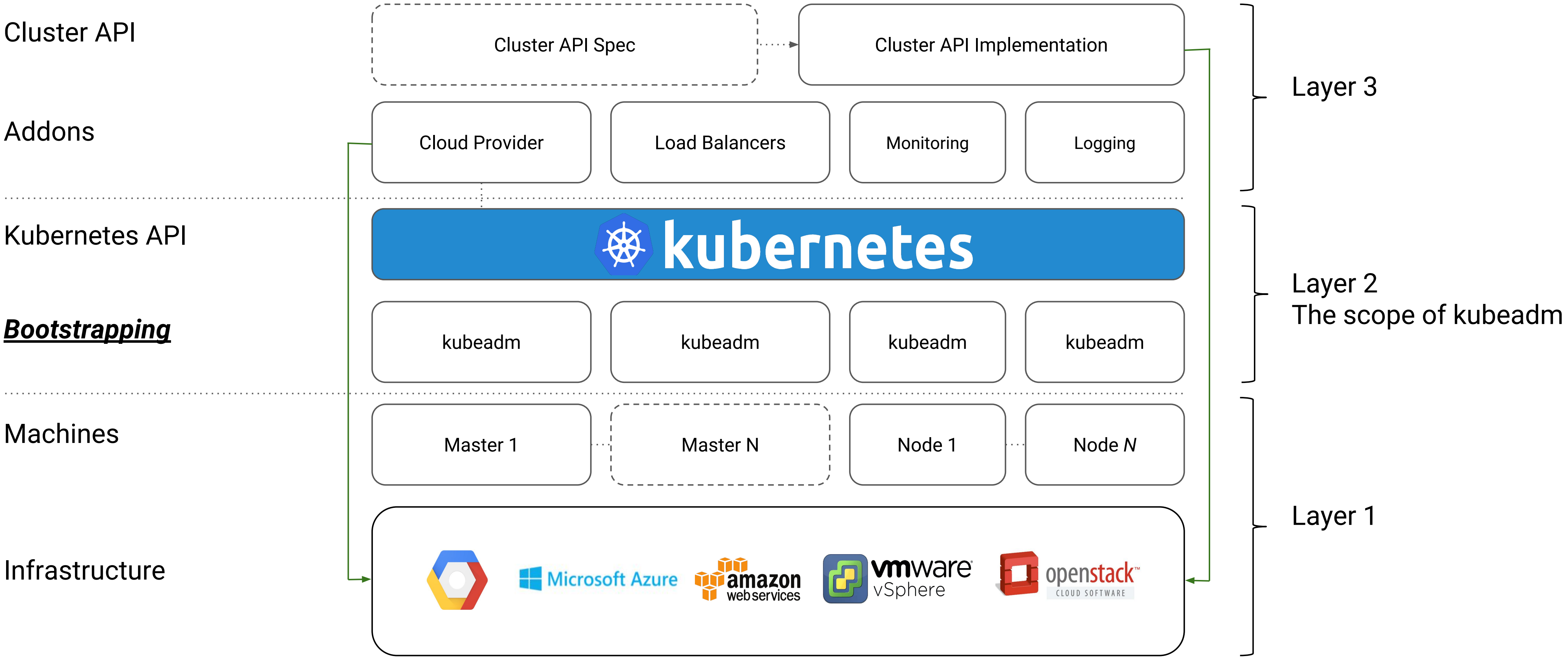
Why (unix philosophy)?

- Make each program do **one thing well**. To do a new job, build afresh rather than complicate old programs by adding new "features".
- Expect the **output** of every program to become the **input** to another, as yet unknown, program. **Don't clutter output with extraneous information**. **Don't insist on interactive input.**
- Design and build software, to be **tried early**, ideally within weeks. **Don't hesitate to throw away the clumsy parts and rebuild them**.
- Use tools instead of people to lighten a programming task, even if you have to detour to build the tools and expect to throw some of them out after you've finished using them.
  - Write down the “Hard Way” and optimize 80% UX Flow with override

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 ***composable*** **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**
- Versioned configuration

Component View



Kubeadm is GA!!!

What does GA mean?

- **Stable command-line UX** — The kubeadm CLI conforms to #5a GA rule of the Kubernetes Deprecation Policy, which states that a command or flag that exists in a GA version must be kept for at least 12 months after deprecation.
  - ***init/join/upgrade/config/reset/token/version***
- **Stable underlying implementation** — kubeadm now creates a new Kubernetes cluster using methods that shouldn’t change any time soon. The control plane, for example, is run as a set of static Pods, bootstrap tokens are used for the kubeadm join flow, and ComponentConfig is used for configuring the kubelet.
- **Upgrades between minor versions** — The kubeadm upgrade command is now fully GA. It handles control plane upgrades for you, which includes upgrades to etcd, the API Server, the Controller Manager, and the Scheduler. You can seamlessly upgrade your cluster between minor or patch versions (e.g. v1.12.2 -> v1.13.1 or v1.13.1 -> v1.13.3).

What does GA mean?

- **Configuration file schema** — With the new **v1beta1** API version, you can now tune almost every part of the cluster declaratively and thus build a “GitOps” flow around kubeadm-built clusters. In future versions, we plan to graduate the API to version **v1** with minimal changes (and perhaps none).
  - Examples and references are now in standard [Godoc format]()
  - Config is split into parts
    - InitConfiguration
    - ClusterConfiguration - stored on cluster in a configmap
    - JoinConfiguration

kubeadm: InitConfiguration

- Usage
  - “kubeadm init --config …”

- Why
  - Custom API endpoint address
  - Specify init bootstrap tokens
  - Pass custom kubelet flags
  - Set node name/taints

```
apiVersion: kubeadm.k8s.io/v1beta1
kind: InitConfiguration
localAPIEndpoint:
  advertiseAddress: "10.100.0.1"
  bindPort: 6443
nodeRegistration:
  criSocket: "/var/run/crio/crio.sock"
  kubeletExtraArgs:
    cgroupDriver: "cgroupfs"
bootstrapTokens:
    ...
```

kubeadm: Cluster Configuration

- Usage
  - “kubeadm init --config …”
- Why
  - Fine tune cluster defaults
  - Custom arguments and volume mounts to control plane components

```
apiVersion: kubeadm.k8s.io/v1beta1
kind: ClusterConfiguration
kubernetesVersion: "v1.12.2"
imageRepository: registry.example.com
networking:
  serviceSubnet: "10.96.0.0/12"
  dnsDomain: "cluster.local"
etcd:
  ...
apiServer:
  extraArgs:
    ...
  extraVolumes:
    ...
```

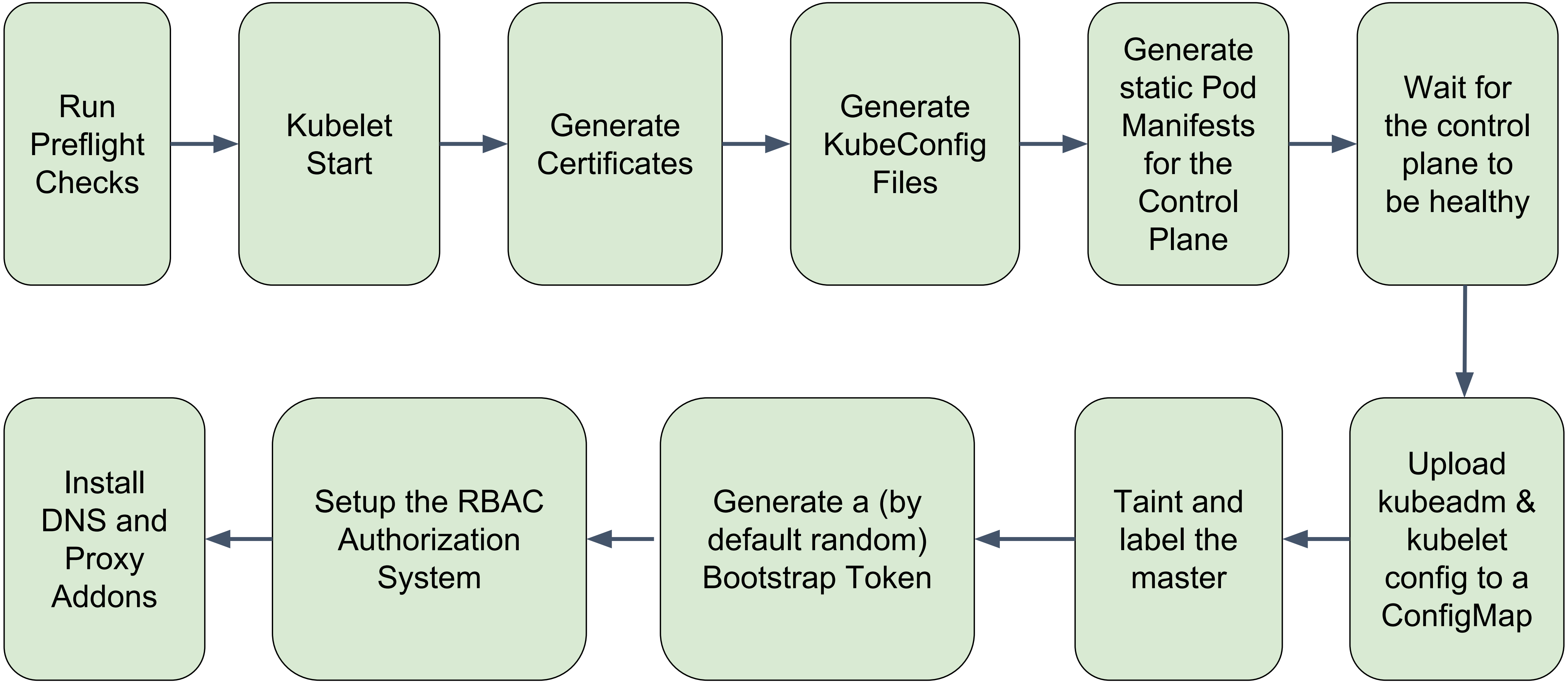
What does GA mean?

- **The “toolbox” interface of kubeadm** — Also known as **phases**. If you don’t want to perform all kubeadm init tasks, you can instead apply more fine-grained actions using the kubeadm init phase command (for example generating certificates or control plane Static Pod manifests).
  - Currently this only applies to `*kubeadm init*`
  - In 2019 - `kubeadm join phases`
- **etcd setup** — etcd is now set up in a way that is secure by default, with TLS communication everywhere, and allows for expanding to a highly available cluster when needed.

kubeadm: init phases

preflight	Run pre-flight checks
kubelet-start	Writes kubelet settings and (re)starts the kubelet
certs	Generates certificates for a Kubernetes cluster
kubeconfig	Generates all kubeconfig files for the control plane and the admin kubeconfig file
control-plane	Generates all static Pod manifest files necessary to establish the control plane
etcd	Generates static Pod manifest file for local etcd.
upload-config	Uploads the currently used configuration for kubeadm to a ConfigMap
mark-control-plane	Mark a node as a control-plane
bootstrap-token	Manage kubeadm-specific bootstrap token functions
addon	Installs required addons for passing Conformance tests

kubeadm: init phases



Run Preflight Checks

Kubelet Start

Generate Certificates

Generate KubeConfig Files

Generate static Pod Manifests for the Control Plane

Wait for the control plane to be healthy

Upload kubeadm & kubelet config to a ConfigMap

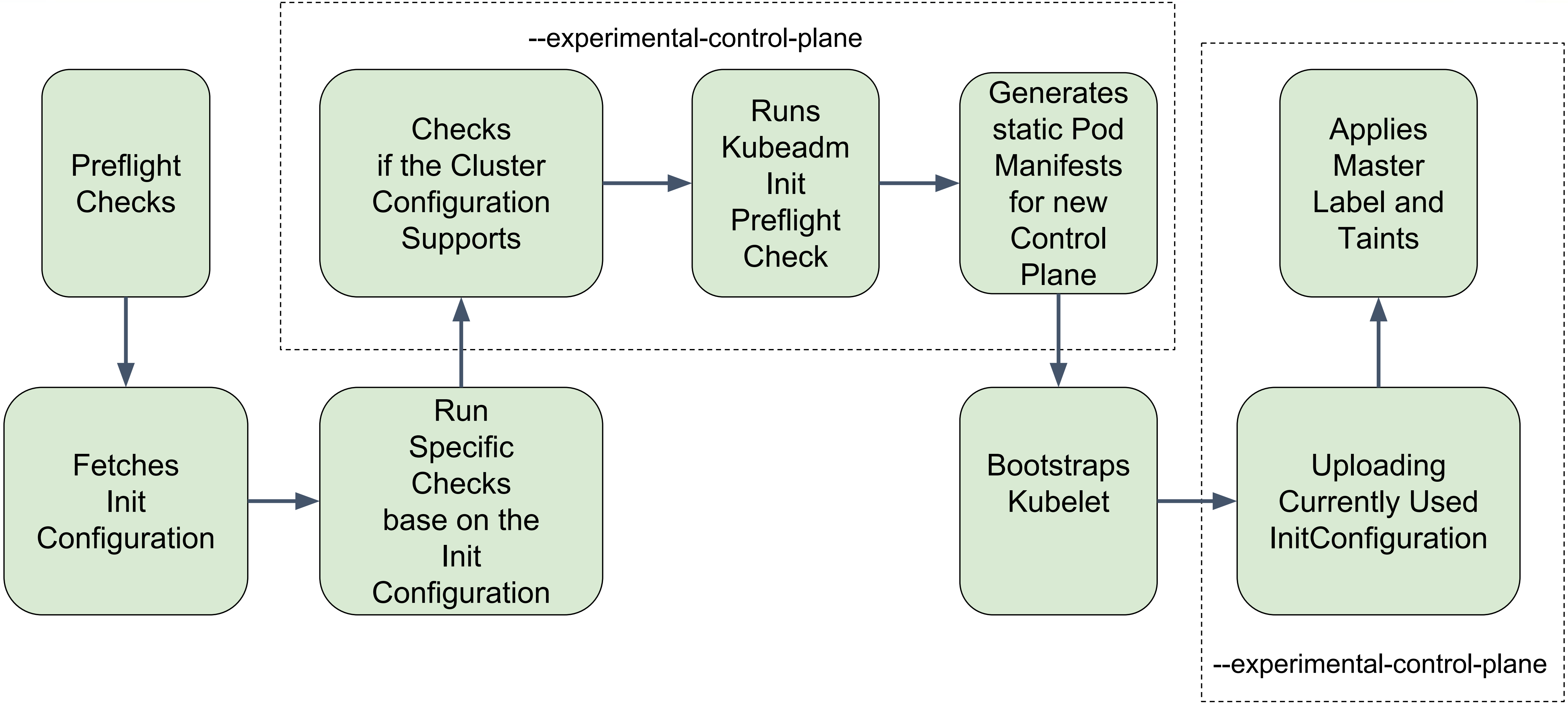
Taint and label the master

Generate a (by default random) Bootstrap Token

Setup the RBAC Authorization System

Install DNS and Proxy Addons

kubeadm join



Preflight Checks

Fetches Init Configuration

Run Specific Checks base on the Init Configuration

--experimental-control-plane

Checks if the Cluster Configuration Supports

Runs Kubeadm Init Preflight Check

Generates static Pod Manifests for new Control Plane

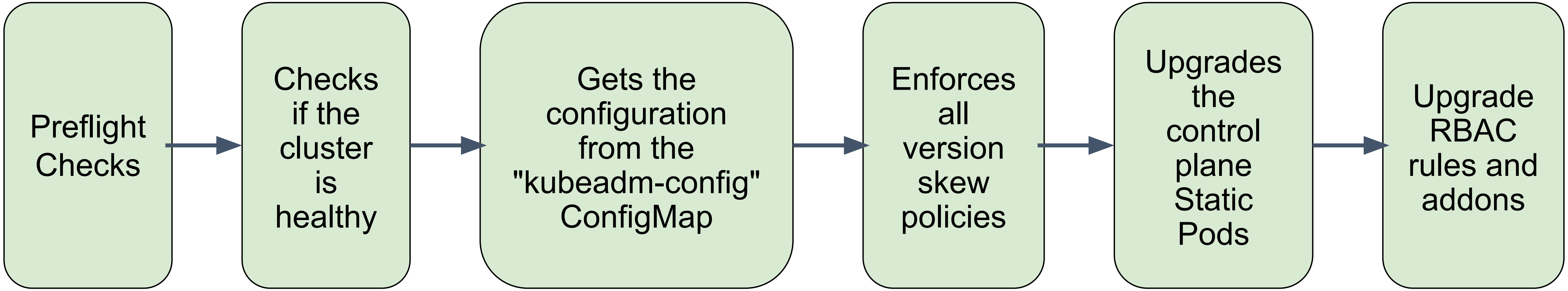
Bootstraps Kubelet

Uploading Currently Used InitConfiguration

Applies Master Label and Taints

--experimental-control-plane

kubeadm upgrade: Control Plane



Preflight Checks

Checks if the cluster is healthy

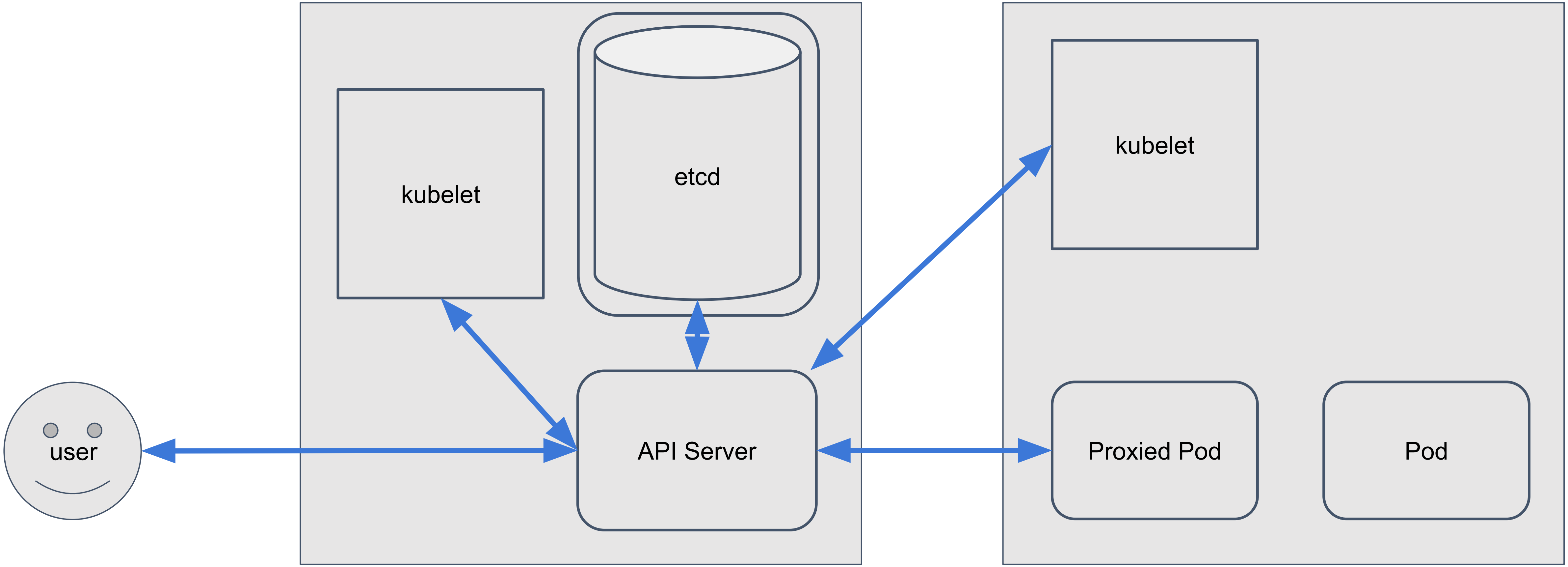
Gets the configuration from the "kubeadm-config" ConfigMap

Enforces all version skew policies

Upgrades the control plane Static Pods

Upgrade RBAC rules and addons

Certificate Management



Certificate Management

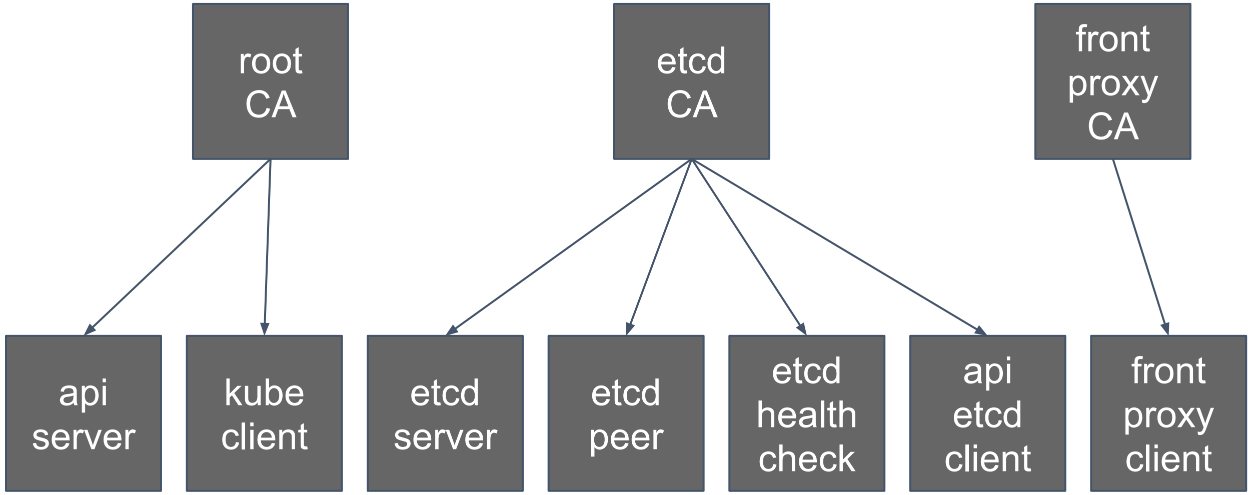
- apiserver
- apiserver-kubelet-client
- front-proxy-client
- etcd-server
- etcd-peer
- etcd-healthcheck-client
- apiserver-etcd-client
- ~~user certificates~~

Certificate Hierarchy

- root CA
  - apiserver
  - apiserver-kubelet-client
- front-proxy CA
  - front-proxy-client
- etcd CA
  - etcd-server
  - etcd-peer
  - etcd-healthcheck-client
  - apiserver-etcd-client

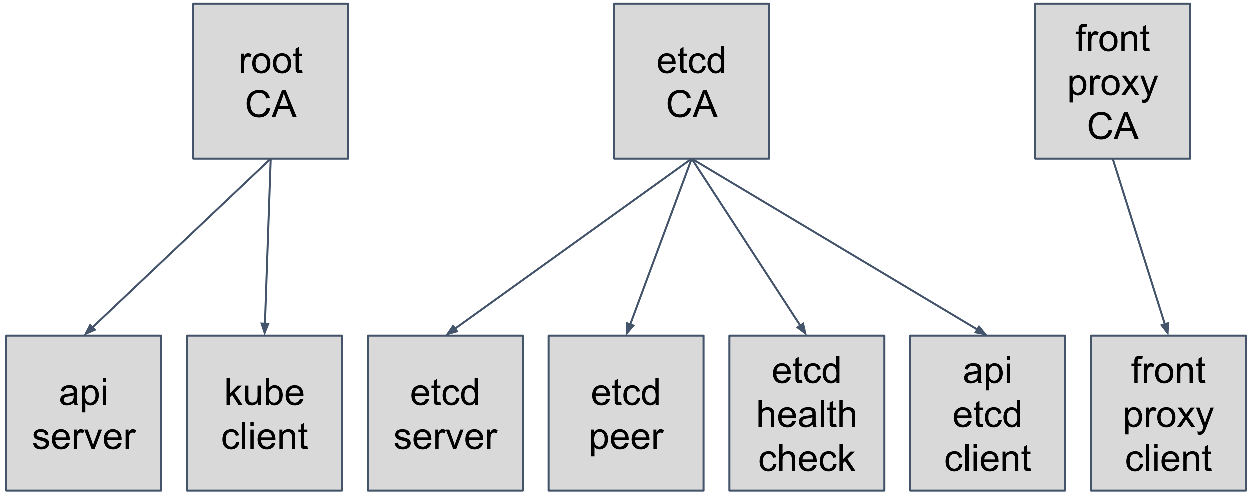
Certificate Hierarchy

- root CA
  - apiserver
  - apiserver-kubelet-client
- front-proxy CA
  - front-proxy-client
- etcd CA
  - etcd-server
  - etcd-peer
  - etcd-healthcheck-client
  - apiserver-etcd-client



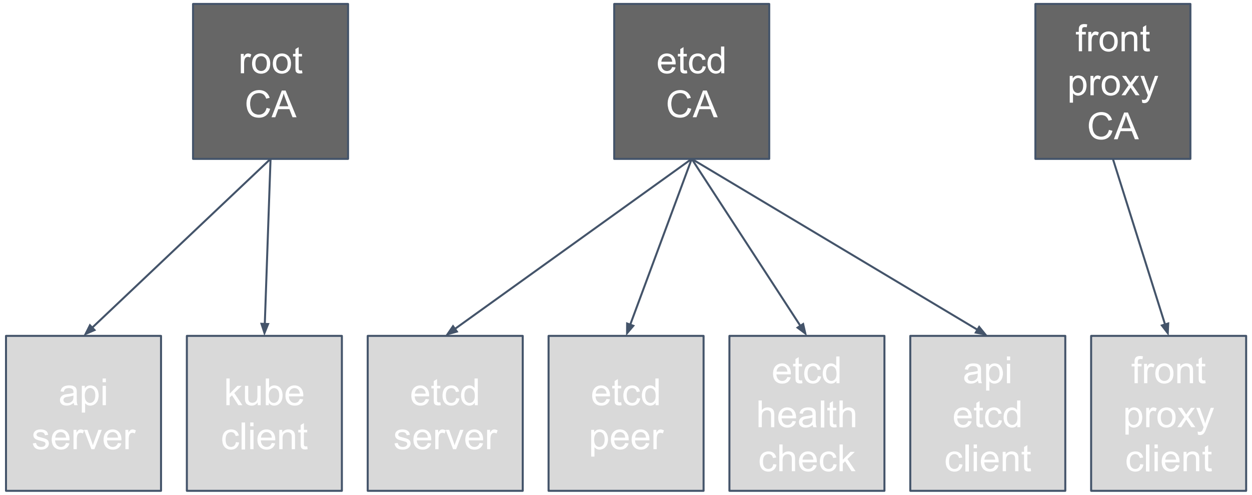
Certificate Generation

- From Scratch



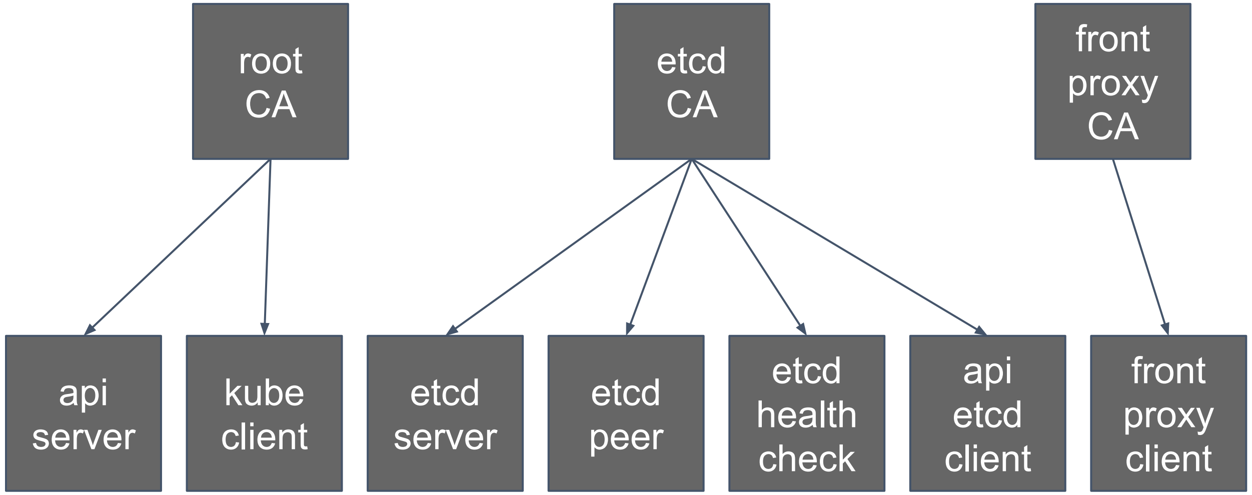
Certificate Generation

- From Scratch
- Provided CAs (+ keys)



Certificate Generation

- From Scratch
- Provided CAs (+ keys)
- All External (keys optional)



root CA

etcd CA

front proxy CA

api server

kube client

etcd server

etcd peer

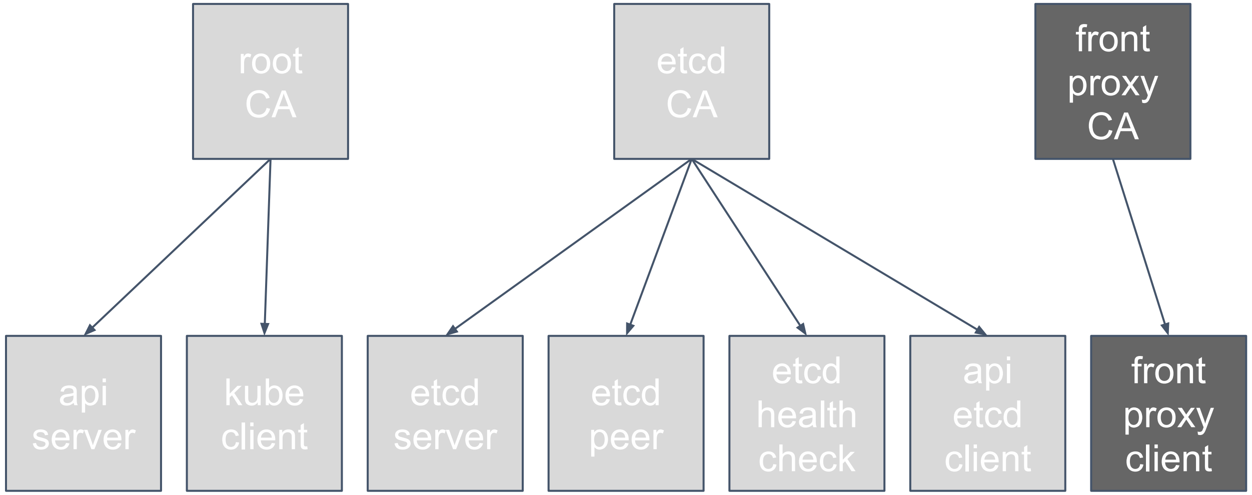
etcd health check

api etcd client

front proxy client

Certificate Generation

- From Scratch
- Provided CAs (+ keys)
- All External (keys optional)
- Mixed



Other Certificate Options

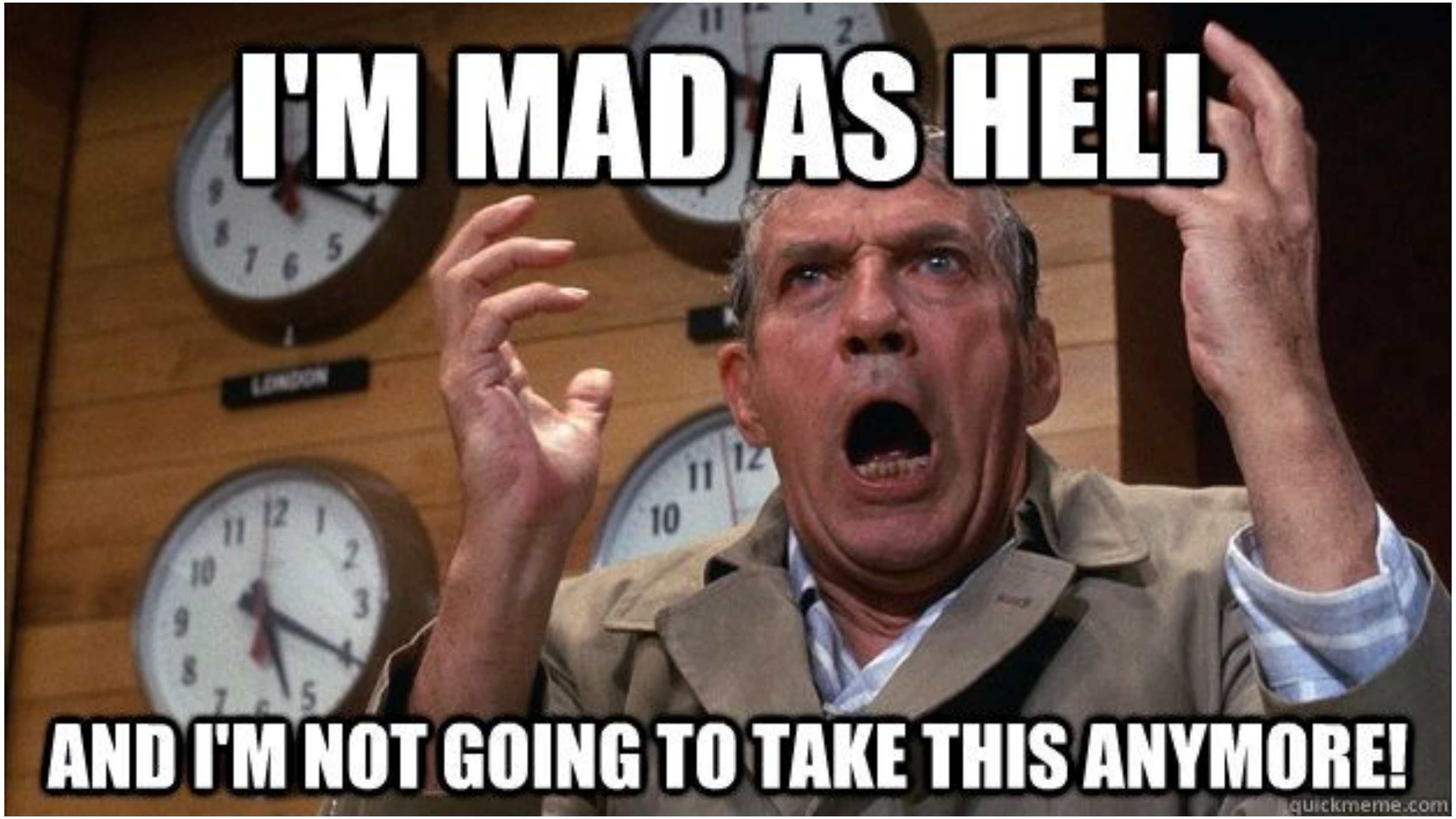
- Generate CSRs!
- `kubeadm alpha certs renew`
- Certificates API requests

2019 Roadmap

2019 Roadmap

- Config to v1
- HA to GA
  - Full test automation
- Continued promotion of alpha phases to subcommands
  - e.g. join phases
- Grand unified field theory on ComponentConfiguration
  - Working group being formed.
- Incorporate ***etcdadm*** and ***bundles*** when stable
- Test and release automation …

Testing and release tooling



2019 Roadmap - CI + Release

- CI
  - KIND as the only PR blocking job
  - Move all SCL jobs to periodics
  - CI = release artifacts
  - Kill `kubernetes-anywhere` with extreme prejudice
- Release
  - Move all package building into k/k
    - .deb/rpm build artifacts
  - Keep signing and publishing separate in the release repo
  - Work with k8s-infra team
    - Want -devel and -stable repos & registries

Getting Involved

http://bit.ly/kubeadm-survey

How can you contribute

- Contributing to SIG Cluster Lifecycle documentation
- We’re working on growing the contributor/reviewers pool; scaling the SIG
- We have “Office Hours” for our projects: weekly for kubeadm, bi-weekly for kops and kubespray…
- Cluster API office hours weekly for both US West Coast and EMEA
- Full list of SIG meetings and links to minutes and recordings can be found on SIG page
- Attend our Zoom meetings / be around on Slack
- Look for “**good first issue**”, “**help wanted**” and “**sig/cluster-lifecycle**” labeled issues in our repositories

Logistics

- Follow the SIG Cluster Lifecycle YouTube playlist
- Check out the meeting notes for our weekly office hours meetings
- Join #sig-cluster-lifecycle, #kubeadm channels
- Check out the kubeadm setup guide, reference doc and design doc
- Read how you can get involved and improve kubeadm!

KubeCon | CloudNativeCon

North America 2018

Thank You!

Q/A