Use following two commands to remove calico from your node:

```
kubectl delete -f
https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/ho
sted/rbac-kdd.yaml
```

```
kubectl delete -f
https://docs.projectcalico.org/v3.3/getting-started/kubernetes/installation/ho
sted/kubernetes-datastore/calico-networking/1.7/calico.yaml
```

Or **kubectl delete -f deployment_file.yaml** if custom configuration was used.

Calico stores some additional data in etcd.
If you're using Calico's Kubeadm manifest which runs etcd as a pod on the master, you'll need to run **rm -rf /var/etcd/calico-data** on the master as well.

Deploy CANAL with the below steps.

Installation

When deploying your Kubernetes cluster, please make sure it meets the requirements at the bottom of this page. An easy way to create a cluster which meets these requirements (bottom) is by following the official kubeadm guide.

> **Note:** If you are upgrading from the Kubernetes 1.6 or 1.5 manifests from the (deprecated) Canal repo to the manifests here it is necessary to migrate your Calico configuration data before upgrading. Otherwise, your cluster may lose connectivity after the upgrade.

RBAC

Before you install Canal, if your Kubernetes cluster has RBAC enabled, you'll need to create the following RBAC roles to allow API access by Canal.
Apply the following manifest to create these necessary RBAC roles and bindings.

```
kubectl apply -f
https://docs.projectcalico.org/v2.6/getting-started/kubernetes/installation/ho
sted/canal/rbac.yaml
```

Canal with the Kubernetes API datastore

The recommended Canal installation uses the Kubernetes API as the datastore, the manifest below installs Calico and flannel configured to use the Kubernetes API.

```
kubectl apply -f
https://docs.projectcalico.org/v2.6/getting-started/kubernetes/installation/ho
sted/canal/canal.yaml
```

(or) Canal with etcd datastore

We strongly recommend using the Kubernetes API manifests above, but if you have a need to use etcd we have provided an example etcd with TLS manifest canal-etcd.yaml.

When using an etcd datastore, the provided manifest allows you to specify the etcd endpoints for your etcd cluster, which must be configured independently.

By default, the manifest expects an etcd proxy to be running on each Kubernetes node at `http://127.0.0.1:2379`.

Requirements / Limitations

- This install assumes no other pod network configurations have been installed in /etc/cni/net.d (or equivalent directory).
- The Kubernetes cluster must be configured to provide service account tokens to pods.
- kubelets must be started with `--network-plugin=cni` and have `--cni-conf-dir` and `--cni-bin-dir` properly set.
  - If using kubeadm these will be set by default.
- The Kubernetes controller manager must be started with `--cluster-cidr=10.244.0.0/16` and `--allocate-node-cidrs=true`.
  - If using kubeadm, specifying `--pod-network-cidr=10.244.0.0/16` will ensure the above flags are set.
- The service CIDR must not overlap with the cluster CIDR.
  - The default service CIDR is `10.96.0.0/12`.
  - The expected cluster CIDR is `10.244.0.0/16`.
  - If using kubeadm, the service CIDR can be set with the flag `--service-cidr`.

KUBEADM Updaing the PODS_CIDR

View the current configuration

```
$ kubeadm config view
apiServer:
  timeoutForControlPlane: 4m0s
apiVersion: kubeadm.k8s.io/v1beta1
certificatesDir: /etc/kubernetes/pki
clusterName: kubernetes
controlPlaneEndpoint: ""
controllerManager: {}
dns:
  type: CoreDNS
etcd:
  local:
            dataDir: /var/lib/etcd
imageRepository: k8s.gcr.io
kind: ClusterConfiguration
kubernetesVersion: v1.13.1
networking:
  dnsDomain: cluster.local
  podSubnet: 192.168.0.0
  serviceSubnet: 10.96.0.0/12
scheduler: {}
```

To update the pods_cidr run the below command

```
$ kubeadm config upload from-flags --pod-network-cidr 10.244.0.0/16
```