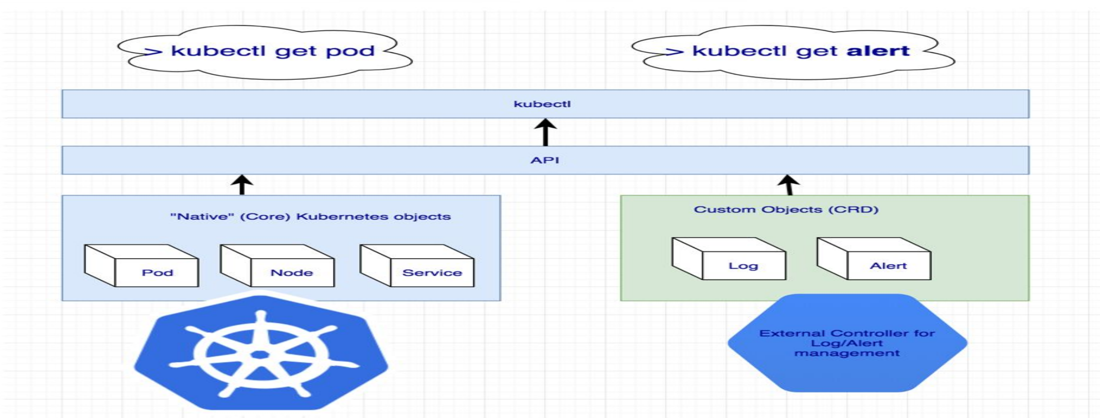
FREE FRESHERS AND EXPERIENCED DEVOPS/SRE JOB READY COURSE

- DAY 4 - KUBERNETES CUSTOM RESOURCE DEFINITION
- RECORDED VIDEOS ON YOUTUBE
  WEEK BY WEEK PROJECTS AND VIDEOS
- YOUTUBE - https://www.youtube.com/praveensingampalli
- TELEGRAM - https://t.me/DevOpsSREbootcamp3
- INSTAGRAM – https://www.instagram.com/singam4devops/
- WEBSITE - www.praveensingampalli.com



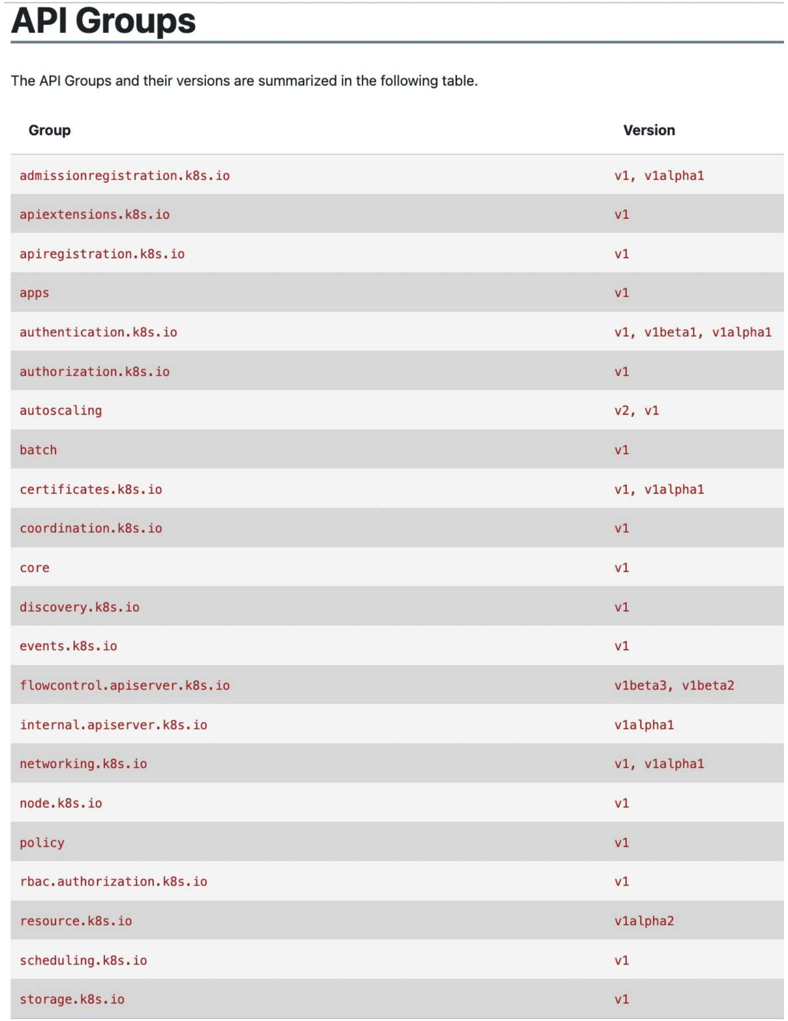
Kubernetes API

The core of Kubernetes' control plane is the API server. The API server exposes an HTTP API that lets end users, different parts of your cluster, and external components communicate with one another.

The Kubernetes API lets you query and manipulate the state of API objects in Kubernetes (for example: Pods, Namespaces, ConfigMaps, and Events)

API Discovery

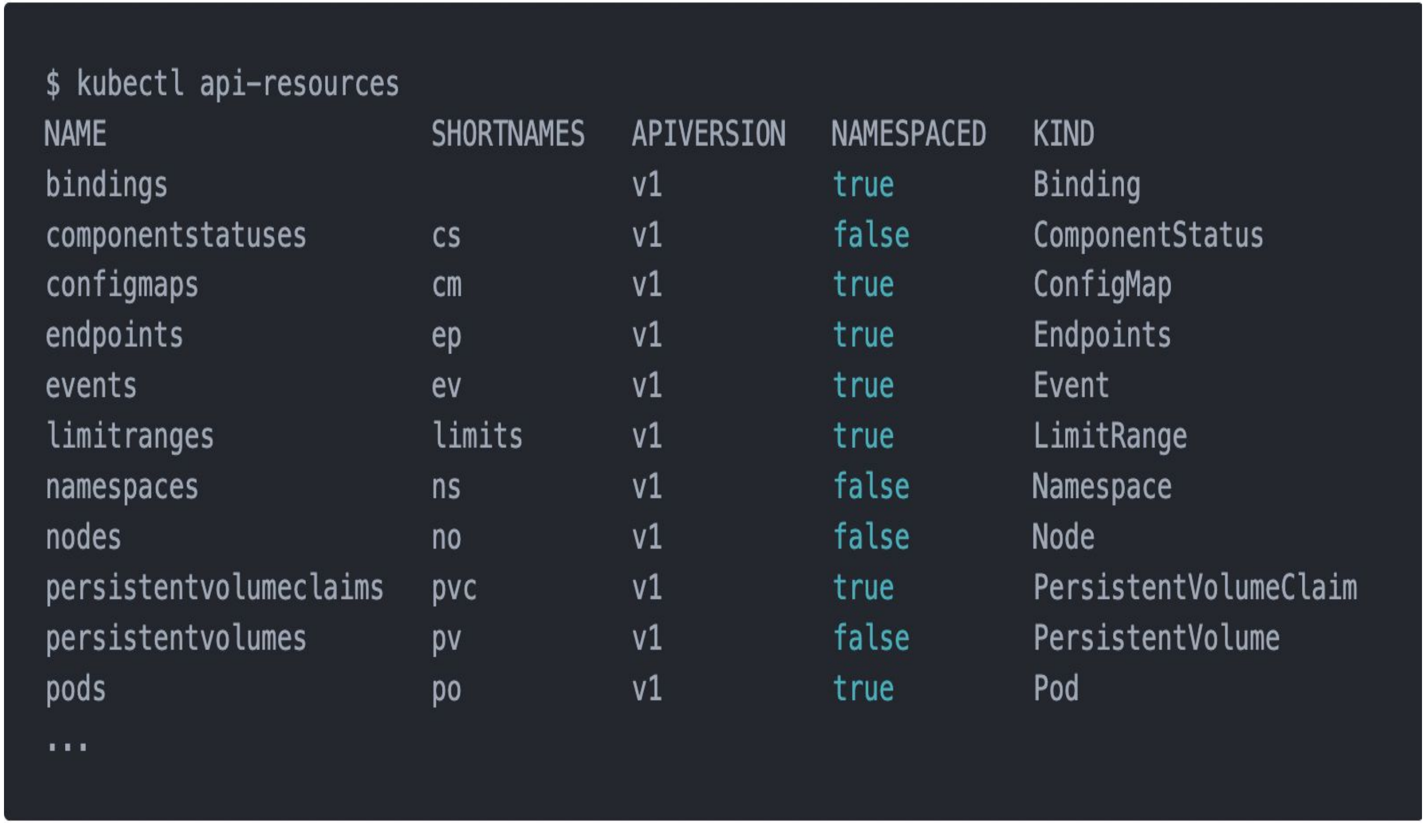
A list of all group versions supported by a cluster is published at the /api and /apis endpoints. Each group version also advertises the list of resources supported via **/apis/<group>/<version>** (for example: /apis/rbac.authorization.k8s.io/v1alpha1). These endpoints are used by kubectl to fetch the list of resources supported by a cluster.



API Groups

The API Groups and their versions are summarized in the following table.

Group	Version
admissionregistration.k8s.io	v1, v1alpha1
apiextensions.k8s.io	v1
apiregistration.k8s.io	v1
apps	v1
authentication.k8s.io	v1, v1beta1, v1alpha1
authorization.k8s.io	v1
autoscaling	v2, v1
batch	v1
certificates.k8s.io	v1, v1alpha1
coordination.k8s.io	v1
core	v1
discovery.k8s.io	v1
events.k8s.io	v1
flowcontrol.apiserver.k8s.io	v1beta3, v1beta2
internal.apiserver.k8s.io	v1alpha1
networking.k8s.io	v1, v1alpha1
node.k8s.io	v1
policy	v1
rbac.authorization.k8s.io	v1
resource.k8s.io	v1alpha2
scheduling.k8s.io	v1
storage.k8s.io	v1



```
$ kubectl api-resources
NAME                     SHORTNAMES   APIVERSION   NAMESPACED   KIND
bindings                              v1           true         Binding
componentstatuses        cs           v1           false        ComponentStatus
configmaps               cm           v1           true         ConfigMap
endpoints                ep           v1           true         Endpoints
events                   ev           v1           true         Event
limitranges              limits       v1           true         LimitRange
namespaces               ns           v1           false        Namespace
nodes                    no           v1           false        Node
persistentvolumeclaims   pvc          v1           true         PersistentVolumeClaim
persistentvolumes        pv           v1           false        PersistentVolume
pods                     po           v1           true         Pod
...
```

KUBERNETES CUSTOM RESOURCE DEFINITION

What are custom resources?

custom resources are extensions of the Kubernetes API. But, unlike a normal resource, custom resources are not necessarily available in a default Kubernetes installation.

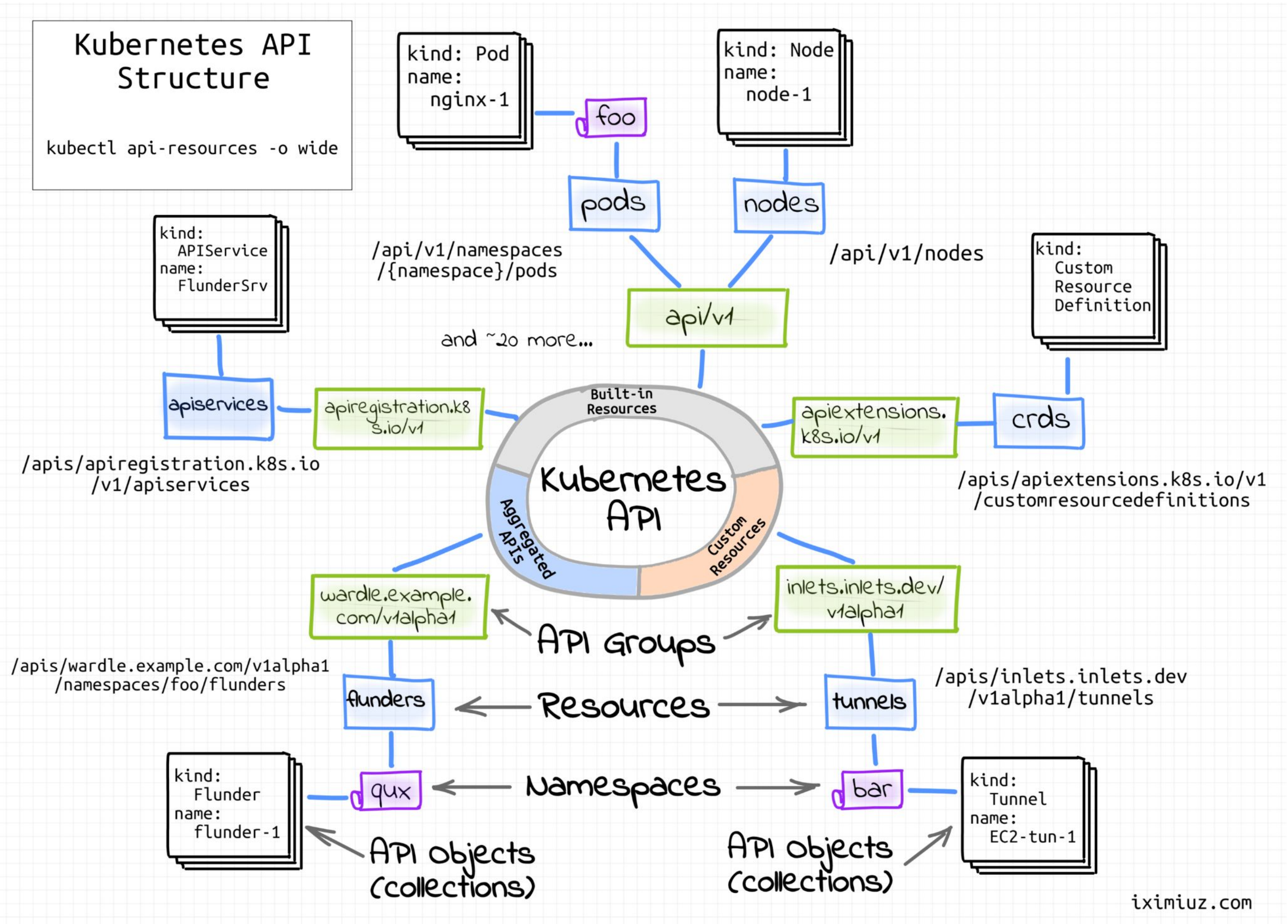
Custom resources are instead registered dynamically to a cluster. Once the custom resource is registered, end users can create, update and delete its object using kubectl, similar to how users interact with built-in resources, like pods, deployments and services.

What are custom resource definitions?

CRDs enable IT admins to introduce unique objects or types into the Kubernetes cluster to meet their custom requirements. A Kubernetes CRD acts like any other Kubernetes object: It uses all the features of the Kubernetes ecosystem -- for example, its command-line interface (CLI), security, API services and role-based access control.

Kubernetes API Structure

```
kubectl api-resources -o wide
```



```
kind: Pod
name:
  nginx-1
```

```
kind: Node
name:
  node-1
```

```
kind:
  APIService
name:
  FlunderSrv
```

```
kind:
  Custom
  Resource
  Definition
```

```
kind:
  Flunder
name:
  flunder-1
```

```
kind:
  Tunnel
name:
  EC2-tun-1
```

/api/v1/namespaces
/{namespace}/pods

/api/v1/nodes

/apis/apiregistration.k8s.io
/v1/apiservices

/apis/apiextensions.k8s.io/v1
/customresourcedefinitions

/apis/wardle.example.com/v1alpha1
/namespaces/foo/flunders

/apis/inlets.inlets.dev
/v1alpha1/tunnels

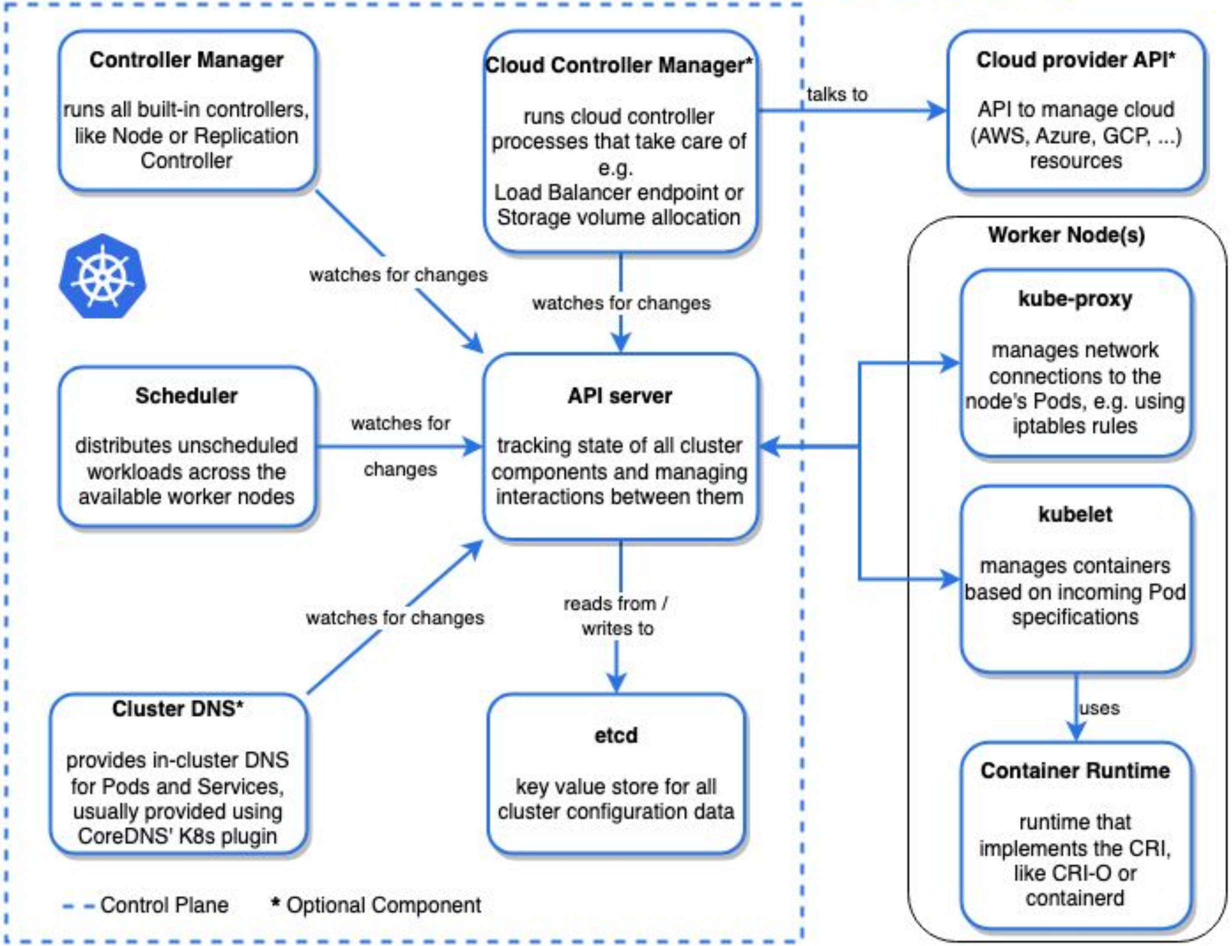
Built-in Resources

Aggregated APIs

Custom Resources

iximiuz.com

Kubernetes Architecture


Controller Manager
runs all built-in controllers, like Node or Replication Controller
Cloud Controller Manager*
runs cloud controller processes that take care of e.g.
Load Balancer endpoint or Storage volume allocation
talks to
Cloud provider API*
API to manage cloud (AWS, Azure, GCP, ...) resources
watches for changes
watches for changes
Worker Node(s)
kube-proxy
manages network connections to the node's Pods, e.g. using iptables rules
Scheduler
distributes unscheduled workloads across the available worker nodes
watches for changes
API server
tracking state of all cluster components and managing interactions between them
kubelet
manages containers based on incoming Pod specifications
watches for changes
reads from / writes to
uses
Cluster DNS*
provides in-cluster DNS for Pods and Services, usually provided using CoreDNS' K8s plugin
etcd
key value store for all cluster configuration data
Container Runtime
runtime that implements the CRI, like CRI-O or containerd
- - Control Plane
* Optional Component

Create the custom resource definition

USE CASE FOR CRD: The IT operations team wants to build an application CI/CD platform and expose it as a Kubernetes custom resource. This custom resource requires user input to build and deploy an application on top of Kubernetes. The first section of the tutorial below displays the minimum input for setup:

- **appId** is the application's unique name, presented as a string identifier.
- **code language** is any predefined language supported in the framework, such as C#, Python or Go.
- **OS** options are restricted to Windows or Linux.
- **instanceSize** should be "T-shirt size" -- predefined CPU and memory sizes, e.g. "small" could mean 100m CPU and 512 mebibyte memory -- for the pods that are spun up. Allowed values are small, medium and large.
- **environmentType** is metadata that classifies the type of environment for the app. Allowed values are dev, test and prod.
- **replicas** is the minimum number of app replicas to maintain, which should be set to 1.
- Checkout the USeCase1 Implementation here - https://github.com/praveen1994dec/Custom_Resource_Definition.git