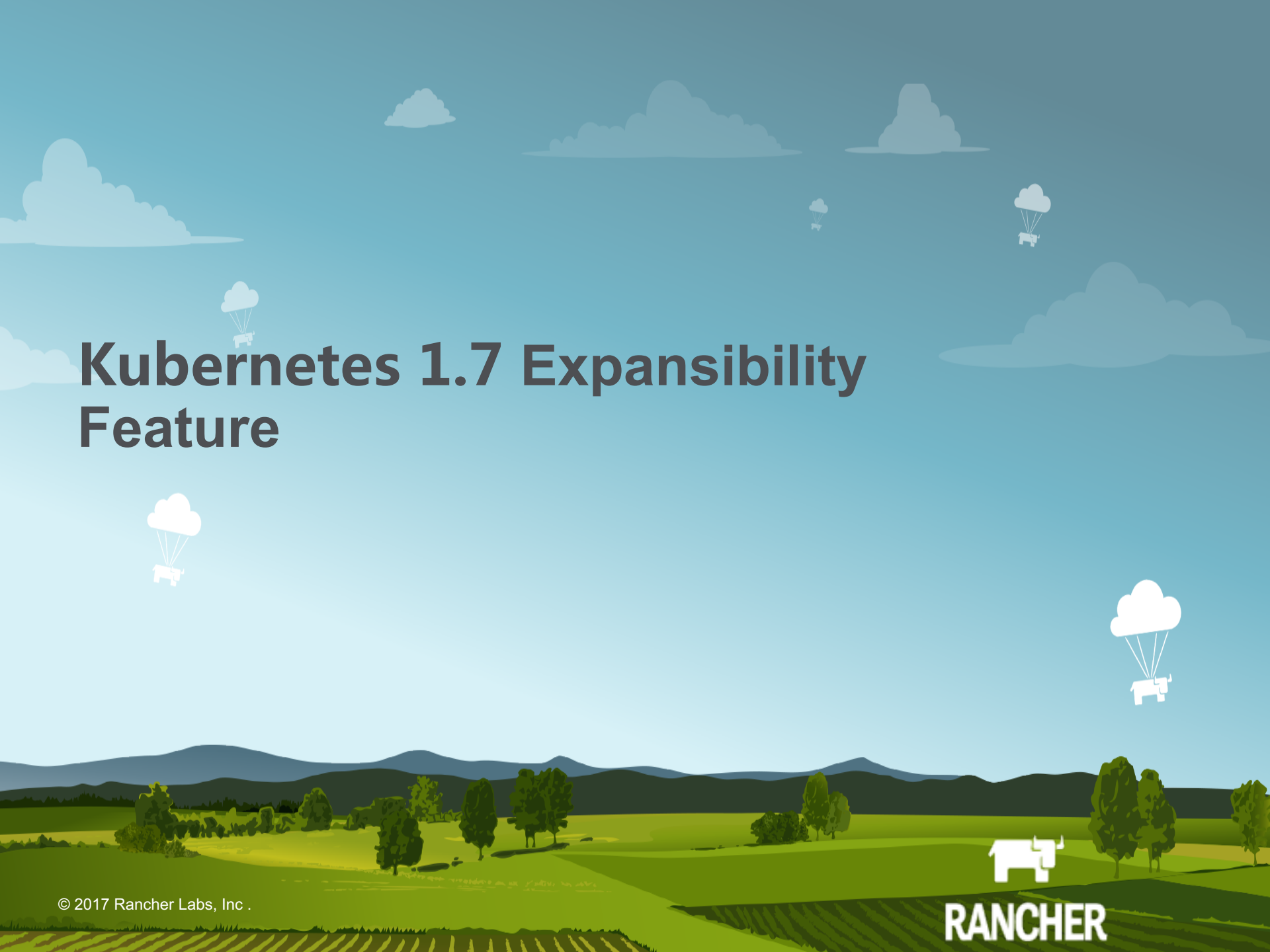
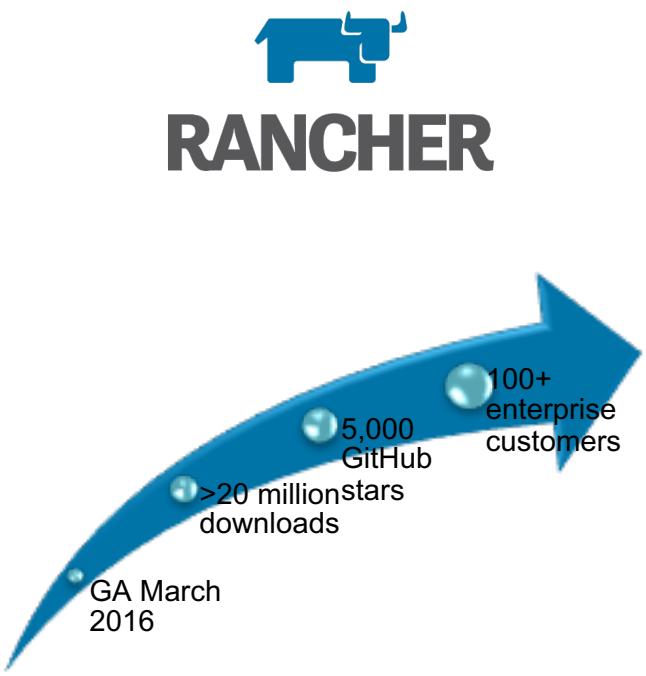
Kubernetes 1.7 Expansibility Feature



RANCHER

Rancher Labs简介



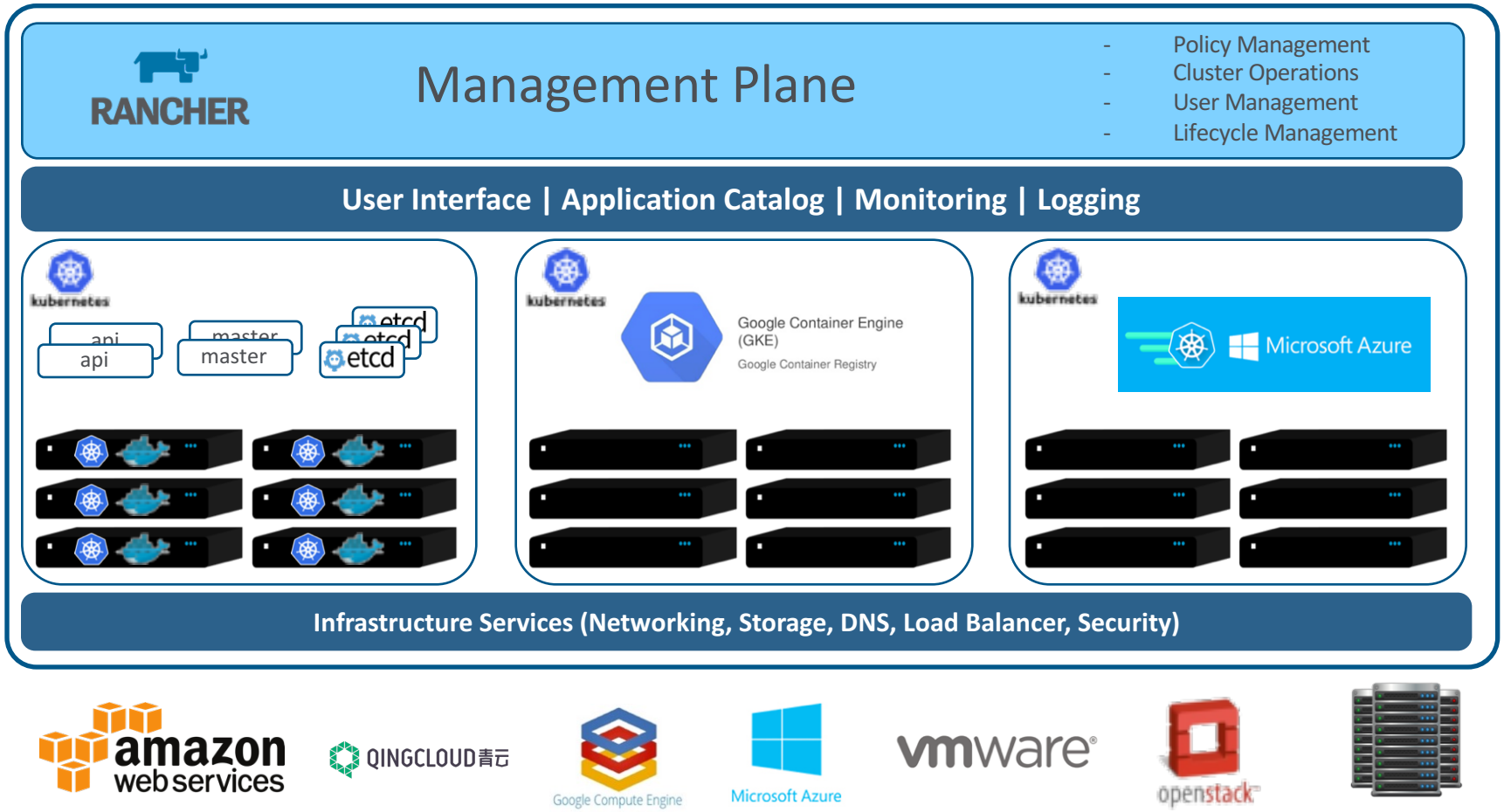
- Rancher Labs成立于2014年，总部位于加州的Cupertino，同时在亚利桑那州的Phoenix和中国设立研发中心
- 核心团队曾创立Cloud.com，并推出了CloudStack，历经从VM到容器的完整技术演进过程
- Rancher Server和Agent镜像在Docker Hub上的下载次数已经超过4000万次，全球Rancher的活动部署超过10,000个

Rancher 与 Kubernetes

- Rancher提供Kubernetes分发版
- Rancher是Kubernetes社区的活跃成员
- Rancher目前支持Kubernetes 1.6.6
- Rancher计划在Rancher 1.6.6版本支持Kubernetes 1.7.0

RANCHER

Rancher 与 Kubernetes



Kubernetes 1.7的扩展特性

- API aggregation(beta)
- CustomResourceDefinitions(beta)
- Support for extensible admission controllers
- Pluggable cloud providers
- Container runtime interface (CRI) enhancements

CustomResourceDefinition(CRD)

- What CRD provides
  - Very flexible way to extend managed resource into a current Kubernetes cluster
  - Auto-generated API in Kubernetes API server
  - Customized resource controller to implement your business logic of managed resource
  - Natural Kubernetes experience for operating your own resource with Kubernetes RBAC and authentication.

- What it comes from
  - From ThirdPartyResource in Kubernetes 1.6
  - Create CRD with spec in Kubernetes 1.7

Example of CRD and Resource Item

```yaml
apiVersion: apiextensions.k8s.io/v1beta1
kind: CustomResourceDefinition
metadata:
  # name must match the spec fields below, and be in the form: <plural>.<group>
  name: crontabs.stable.example.com
spec:
  # group name to use for REST API: /apis/<group>/<version>
  group: stable.example.com
  # version name to use for REST API: /apis/<group>/<version>
  version: v1
  # either Namespaced or Cluster
  scope: Namespaced
  names:
    # plural name to be used in the URL: /apis/<group>/<version>/<plural>
    plural: crontabs
    # singular name to be used as an alias on the CLI and for display
    singular: crontab
    # kind is normally the CamelCased singular type. Your resource manifests use this.
    kind: CronTab
    # shortNames allow shorter string to match your resource on the CLI
    shortNames:
    - ct
```

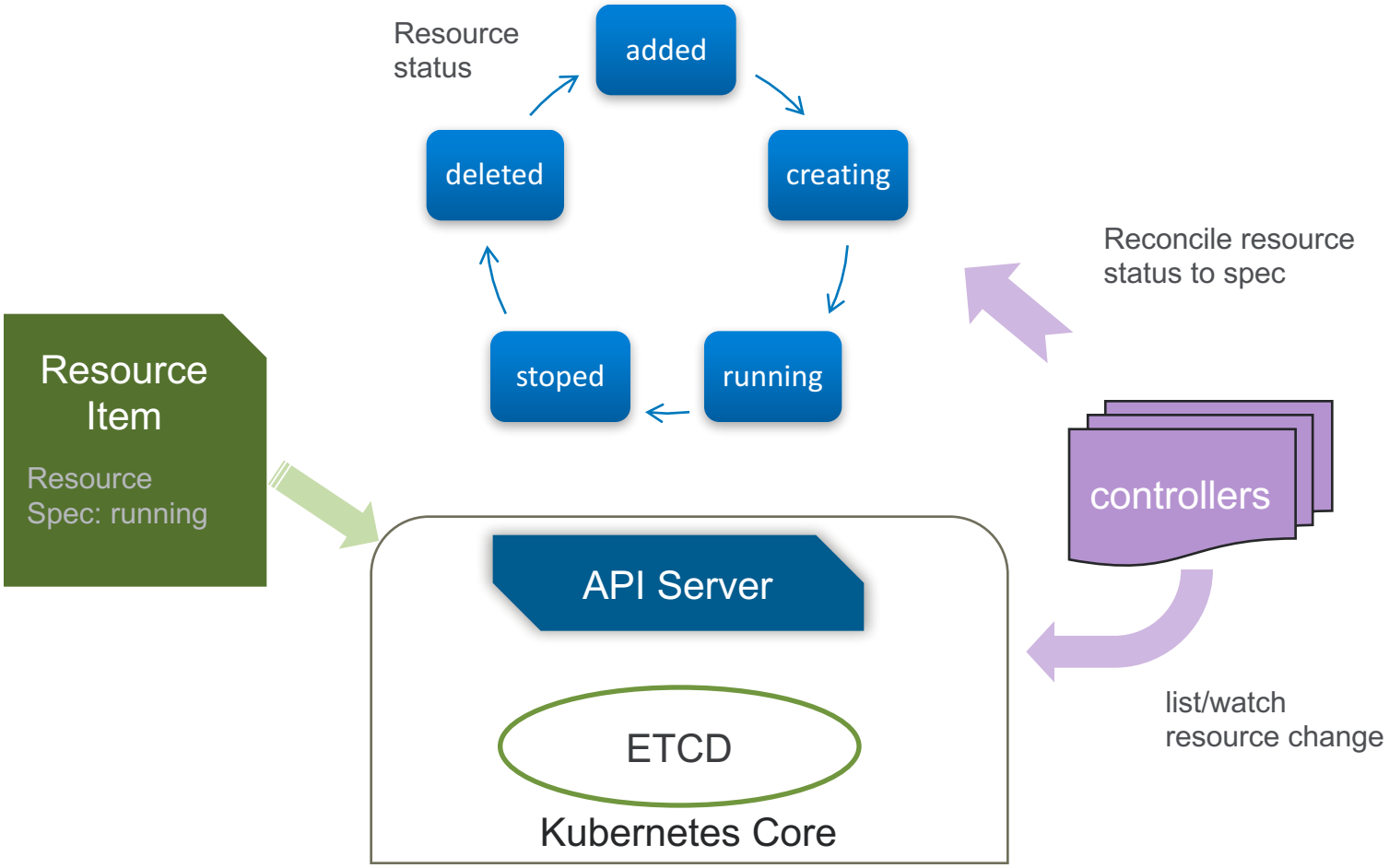
```yaml
apiVersion: "stable.example.com/v1"
kind: CronTab
metadata:
  name: my-new-cron-object
spec:
  cronSpec: "* * * * /5"
  image: my-awesome-cron-image
```

my-crontab.yaml

```
kubectl create -f resourcedefinition.yaml
```

```
/apis/stable.example.com/v1/namespaces/*/crontabs/...
```

How Does The Controller Work



RANCHER

API Aggregation

- What API aggregation provides
  - Extended with additional APIs
  - Build your own API server

- Requirements of aggregation layer
  - Running Kubernetes 1.7 Cluster
  - Enable apiserver flags

Setup an Extension API Server

- Use apiserver-builder to build your own API server
  - https://github.com/Kubernetes-incubator/apiserver-builder
- Download and install the latest version of apiserver-builder
- Create project path in your GOPATH
- Go into your project path and init your project

```
apiserver-boot init repo --domain <your-domain>
```

'your-domain' would be like your private tenant name.

- Then initialize your own resource group, version and kind.

```
apiserver-boot create group version resource --group <your-group> --version <your-version> --kind <your-kind
```

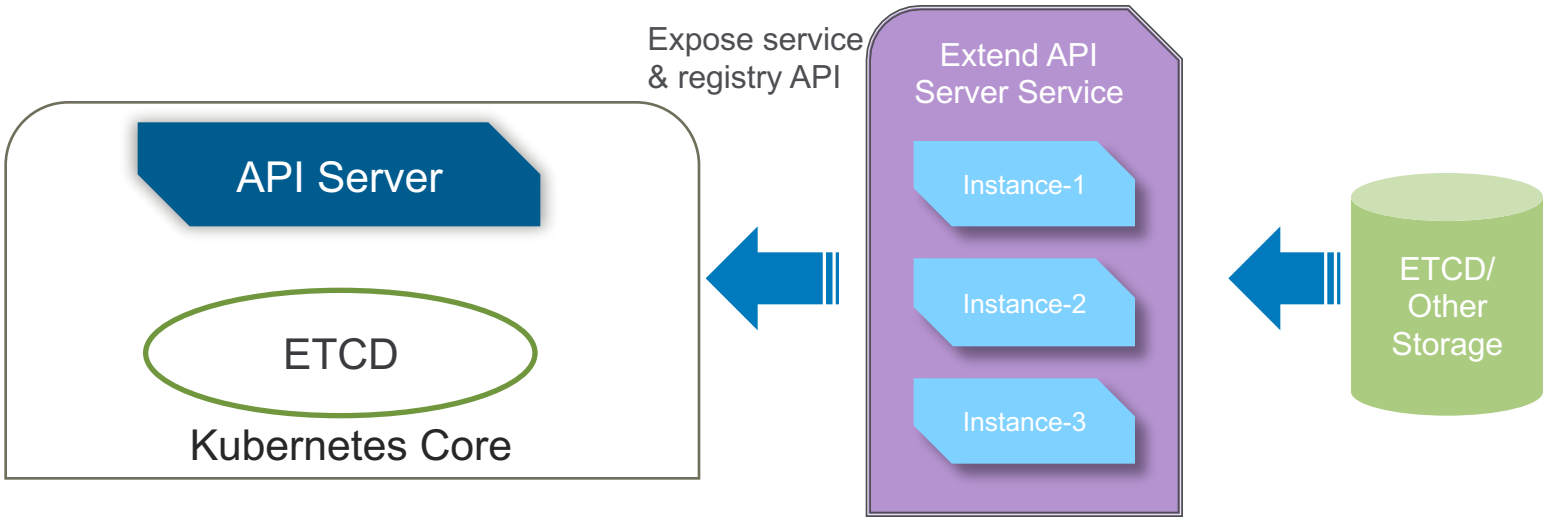
- Your API server could be build and run now

```
apiserver-boot build executables
```

- Build as an image and run in a cluster

```
apiserver-boot run in-cluster --name nameofservicetorun --namespace default --image gcr.io/myrepo/myimage:my
```

API Server Aggregation Architecture



RANCHER

Thanks!

Yuxing

2017-8-5

RANCHER