

---

---

# Rancher 2.0

# Technical Deep Dive

— LINE Corporation —  
IT Service Center Verda 2  
Yuki Nishiwaki

---

---

# Who Are You: Yuki Nishiwaki

## Current Role

- Develop/Operate OpenStack based Private Cloud
- Plan/Develop/Operate Kubernetes as a Service



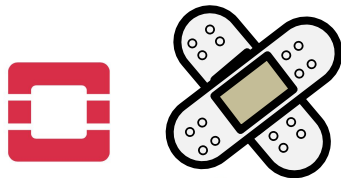
## Recent Talk

- Excitingly simple multi-path OpenStack Networking (May 2018, OpenStack Summit)

# Brief introduction of ourselves(Verda)

# The position against k8s from me

OpenStack based *Private Cloud*



openstack®



Server

Network

Database

.....

New Resource Type



Me

Operator/Developer for Private Cloud

# Verda Kubernetes as a Service - Background

- We've seen about 600 k8s node users deployed/used on our Private Cloud
- Many teams find easy way/struggle to use/operate k8s

## Problem description

- Operating k8s is not such a small burden every developer can handle in spare time
- Knowledge of k8s operation is fragmented



*New Resource Type  
For Verda User*



# Verda Kubernetes as a Service - Target

- Provide stable Kubernetes Cluster to Verda User
  - Don't have to automate everything but we will take responsibility to operate
- Provide API to Verda User
  - CREATE/DELETE Kubernetes Cluster (UPGRADE is not the target at the moment)
  - ADD/REMOVE Node
- Provide “Service Desk”
  - To advise/consider how to use with Verda User(Application developer)



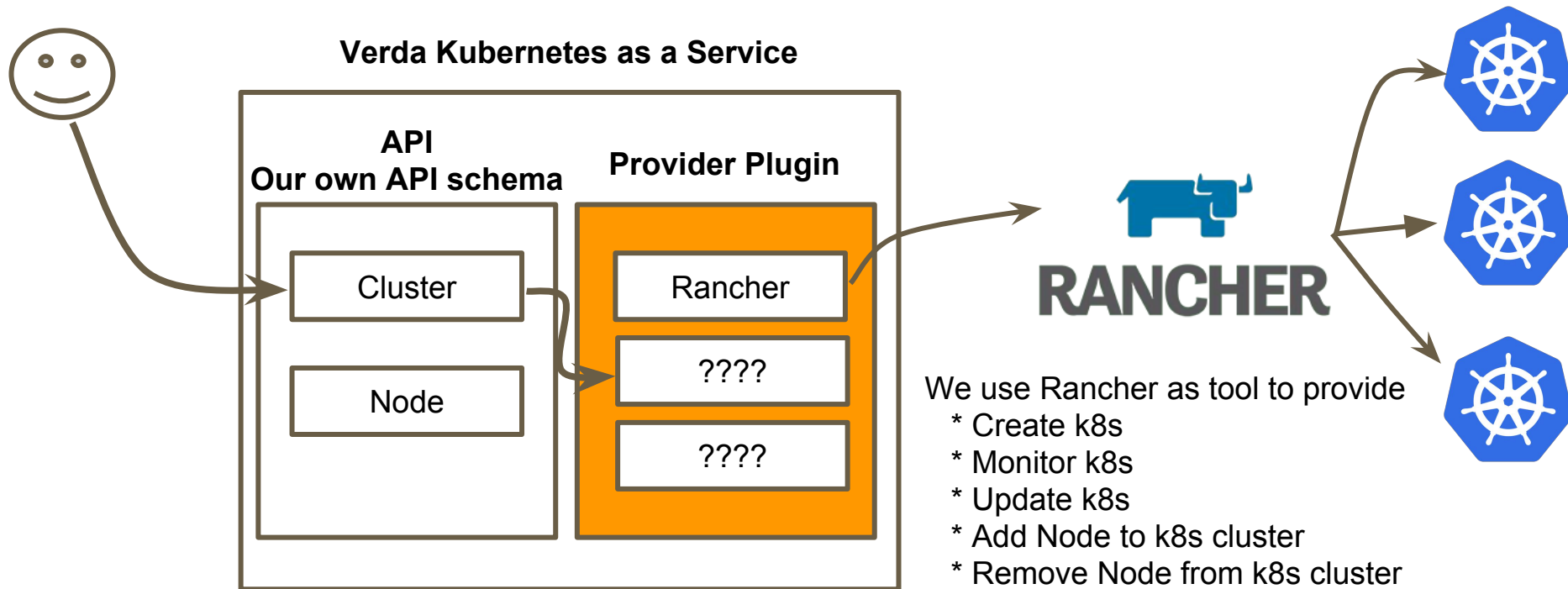
***New Resource Type  
For Verda User***



# Verda Kubernetes as a Service - Status of Project

- Start Project since May 2018
  - Pretty late, relatively
- Decided to utilize existing software(OSS)
  - Reduce lead time as much as possible
  - Rancher 2.0 is **one of the candidates** we will use
    - We are thinking to use **Rancher 2.0 for Phase 1**
- Still deciding which is good to use for managing k8s part
  - Or Will we have to develop from scratch?

# Less dependent design - Still considering





# Roadmap

## Phase 1

Phase1 (2018/09/01)

- \* This is First release
- \* No change for Kubernetes/Rancher
- \* Support only basic k8s cluster
- \* Support Limited Number of Cluster
- \* Train ourselves
  - \* For Rancher (because we depended)

## Phase 2

### Phase2 (Planning)

- \* Enhance VKS control plane(Tune Rancher)
  - \* Support More Clusters
  - \* Enhance monitoring item
  - \* Enhance GUI
- \* Enhance k8s support
  - \* Support Type Loadbalancer for in-house LB
  - \* Support Persistent Volume
  - \* Optimizing Container Networking
- \* Train ourselves
  - \* For Kubernetes, Etc

## Phase 3

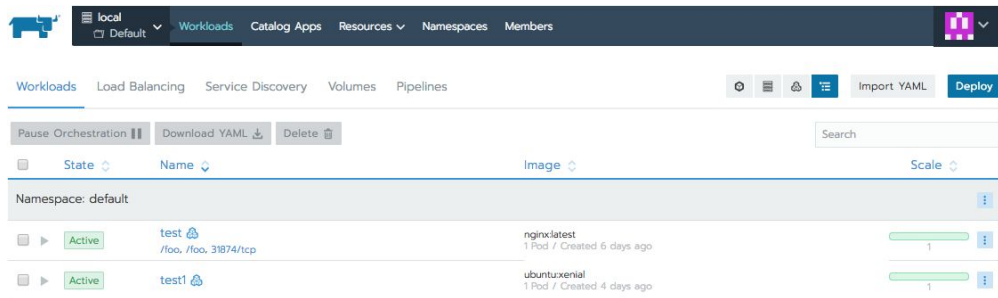
### Phase3 (Planning)

- \* Enhance k8s support
  - \* CRD/Controller for in-house Component
  - \* Prepare skeleton template to make it easy to start development
  - \* Consider solution about how k8s cover whole system including VM (kubevirt)

# About Rancher 2.0

# What's Rancher?

- Container Management Tool
- Support to deploy Container Orchestration Tool itself like Kubernetes
- Make “Container Orchestration itself” abstract and Provide rich UI
- UI allow you to deploy your container workload easier than native console
- UI allow you to use well-tested catalog



# Rancher 2.0 Released (May 1 2018)

- Focus on using Kubernetes as a Container Orchestration Platform
- Re-design to work on Kubernetes from scratch
- Re-Implement from scratch
- Introduce Rancher Kubernetes Engine (RKE)
- Unified Cluster Management including GKE, EKS... as well as RKE
- Application Workload Management

# Rancher 2.0

## Our Interest as a backend For our “k8s as a service”

- Focus on using Kubernetes as a Container Orchestration Platform
  - Re-design to work on Kubernetes from scratch
  - Re-Implement from scratch
  - Introduce Rancher Kubernetes Engine (RKE)
- Unified Cluster Management including GKE, EKS... as well as RKE
  - Application Workload Management

# Rancher 2.0

## Our Interest as a backend For our “k8s as a service”

- Focus on using Kubernetes as a Container Orchestration Platform
  - Re-design to work on Kubernetes from scratch
    - Don't have to understand multiple container orchestrators
  - Re-Implement from scratch
    - Readable amount of code (about 50,000~80,000 lines except for vendoring)
  - Introduce Rancher Kubernetes Engine (RKE)
    - Support Deploy/Upgrade/Monitor Kubernetes cluster
    - Less requirements for the environment to build k8s
- 
- Unified Cluster Management including GKE, EKS... as well as RKE
  - Application Workload Management

# As a context: backend for Verda K8s as a Service

- In our use case, User/Operator for Rancher is different
  - Operator: Cloud Operator (us)
  - User: Application Developer<sub>s</sub> for LINE Service
- Down time of Rancher affect to many users



Need to know well about How Rancher works

# But Documentation....

The screenshot shows a GitHub repository page for 'rancher / rancher'. The issue title is 'Add more documentation for developers of Rancher 2' with ID #13933. A comment by user 'jomeier' is highlighted with a red box. The comment text is: 'I'd love to contribute to this project but reverse engineering through the code is horrible :-)' followed by 'The Rancher 2 architecture document is a minimal starting point for getting a feeling how the API might work.' and 'The last days I worked on documentation and a few scripts which enable me to remote debug Rancher 2 from a GO IDE. I created a pull request for that: #13932'. The comment also includes a plea for more documentation and a Slack channel, signed 'Josef'. The right sidebar shows labels: 'area/documentation', 'kind/enhancement', and 'version/2.0'.

Search or jump to... Pull requests Issues Marketplace Explore

rancher / rancher Unwatch 560 Star 8,835 Fork 881

<> Code Issues 2,174 Pull requests 19 Projects 0 Wiki Insights

## Add more documentation for developers of Rancher 2

#13933 New issue

Open jomeier opened this issue 10 days ago · 3 comments

jomeier commented 10 days ago

I'd love to contribute to this project but reverse engineering through the code is horrible :-). The Rancher 2 architecture document is a minimal starting point for getting a feeling how the API might work.

The last days I worked on documentation and a few scripts which enable me to remote debug Rancher 2 from a GO IDE. I created a pull request for that:

[#13932](#)

Please please please create more documentation about how Rancher 2 works, how we can contribute to fix bugs and enhance it. Maybe it's also a good idea to create a Slack channel for Rancher Developers.

This project is awesome and we want to get our hands on it. Give us more documentation, please :-)!

Josef

1

Assignees  
No one assigned

Labels  
area/documentation  
kind/enhancement  
version/2.0

Projects  
None yet

Milestone  
No milestone

Notifications

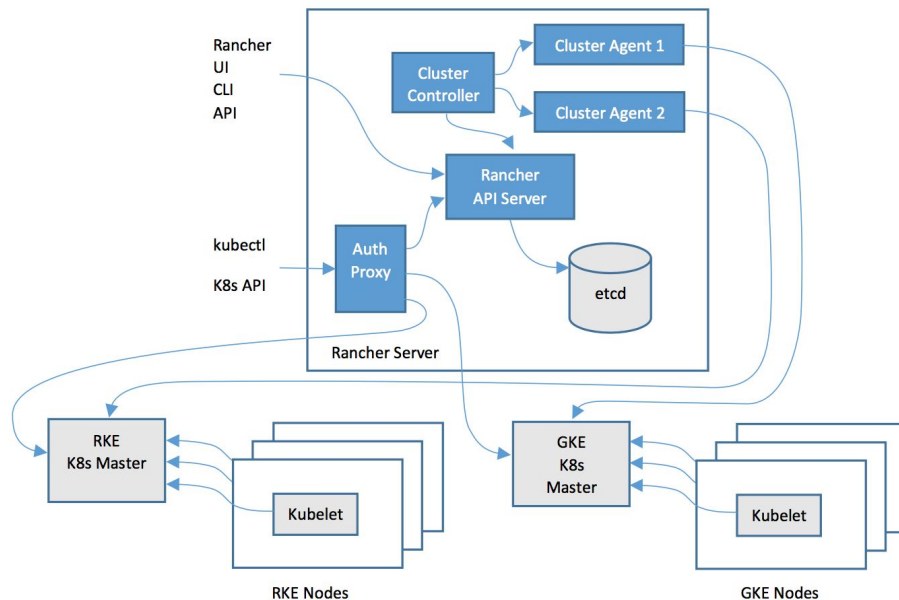


# Only High-level Architecture is available

## High-level Architecture

Like Rancher 1.0, majority of Rancher 2.0 software runs on the Rancher server. Rancher server includes all the software components used to manage the entire Rancher deployment.

Figure 2 illustrates the high-level architecture of Rancher 2.0. The figure depicts a Rancher server installation that manages two Kubernetes clusters: one Kubernetes cluster created by RKE and another Kubernetes cluster created by GKE.



To be honest,  
It's difficult to operate Rancher 2.0 with  
just official documents



# Let's unbox Rancher 2.0

## <v2.0.0>

本資料は、調査資料を一部抜粋して作成しています。  
フルバージョンをご覧になりたい方は、後日slideshareでご確認をお願いします。

# 1. Rancher Overview

## 1. Rancher Overview

- 1.1. Casts in Rancher 2.0
- 1.2. What Server does?
- 1.3. What Agent does?
- 1.4. Summary

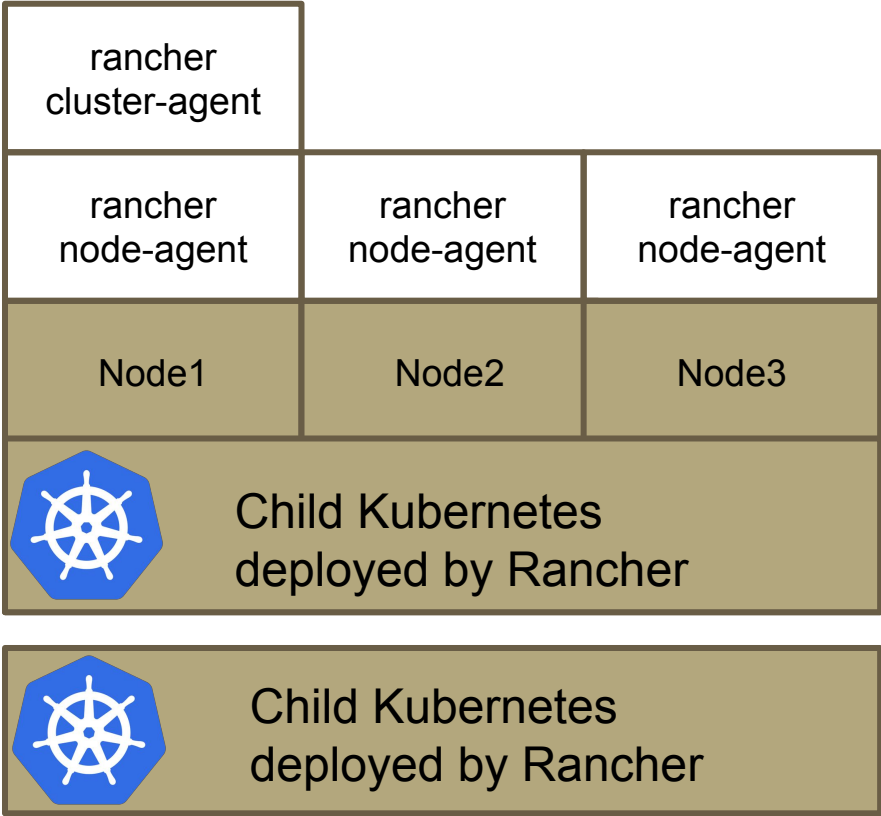
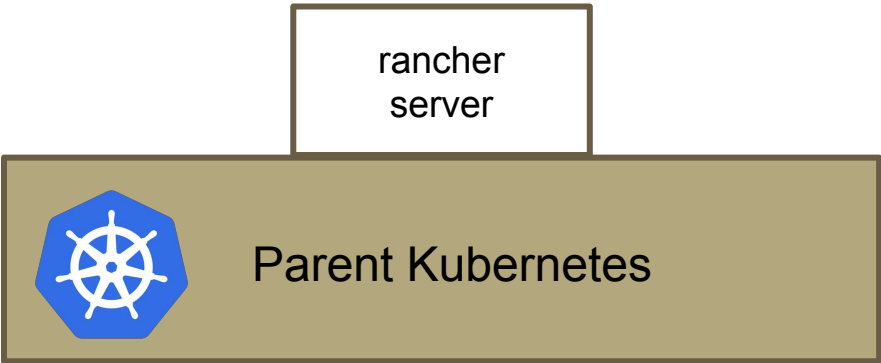
## 2. Rancher Server Internal

- 2.1. Rancher API
  - 2.2. Rancher Controllers
  - 2.3. Example Controllers
-

Parent k8s: k8s working with rancher  
Child k8s: k8s deployed by rancher

# 1.1. Casts in Rancher 2.0

- Rancher Server
- Rancher Cluster Agent
- Rancher Node Agent



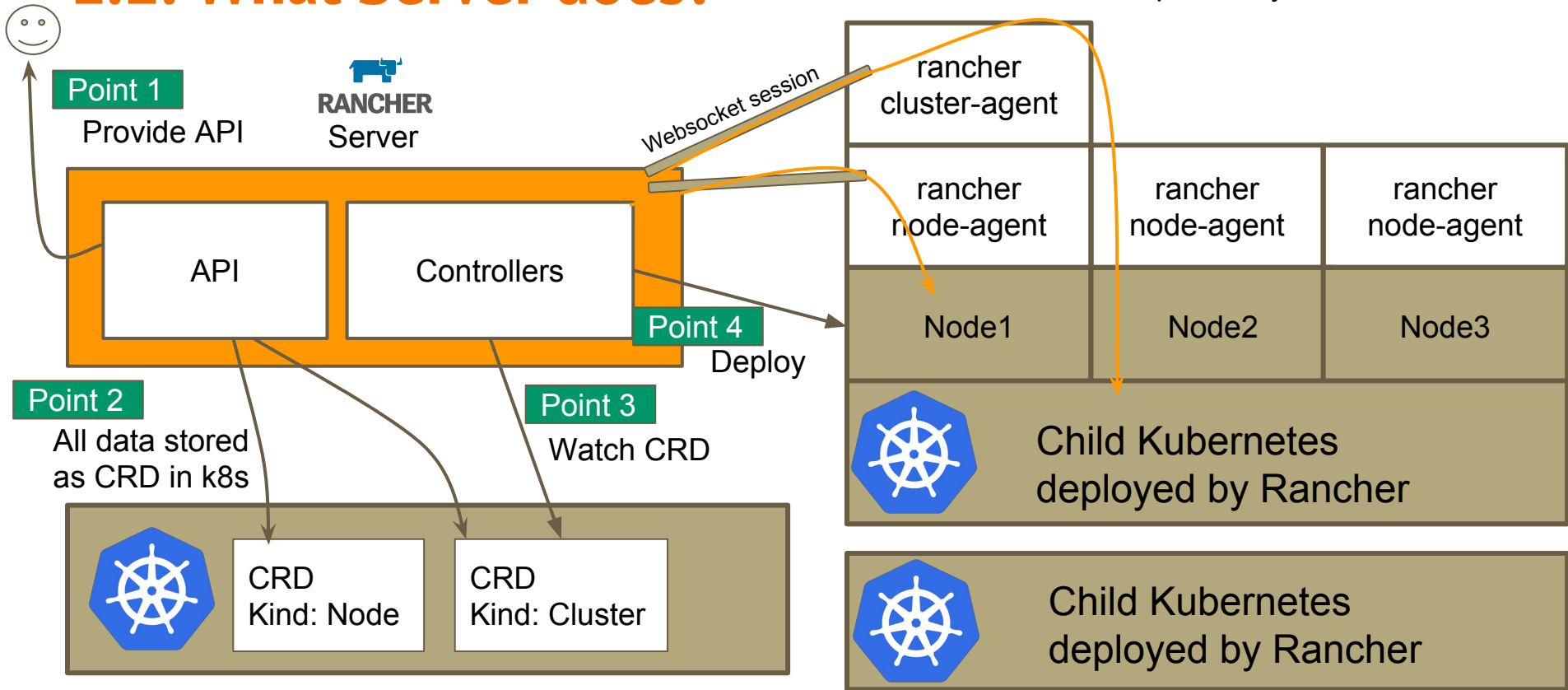
## Point 5

Monitor Cluster/Sync Data

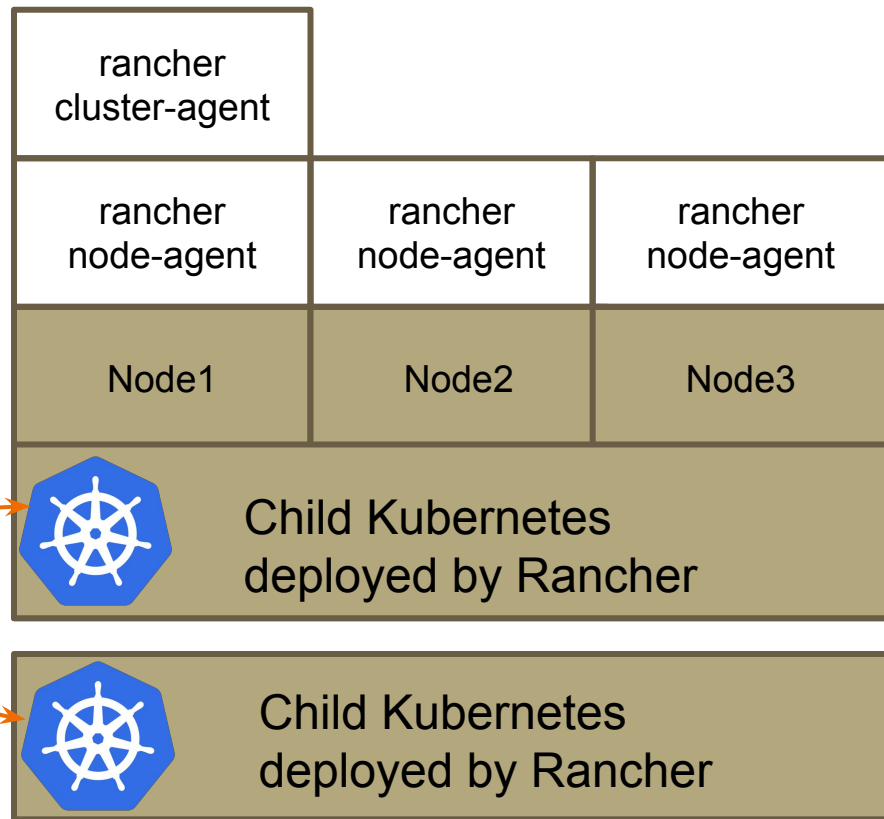
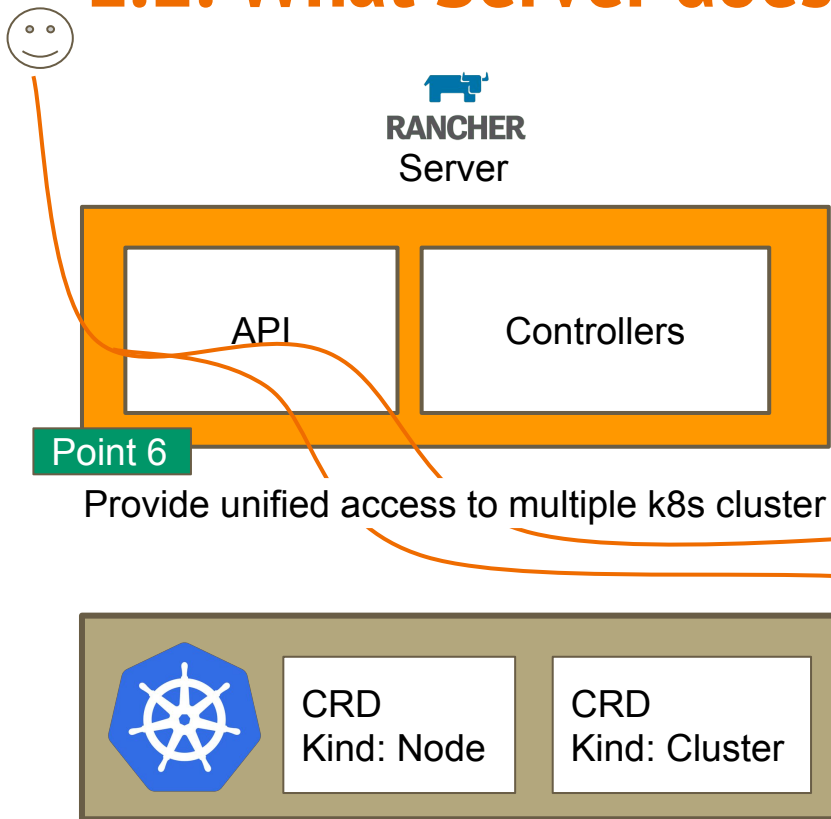
Call docker/k8s API via websocket, If need.

Don't access to docker/k8s api directly from rancher server

# 1.2. What Server does?



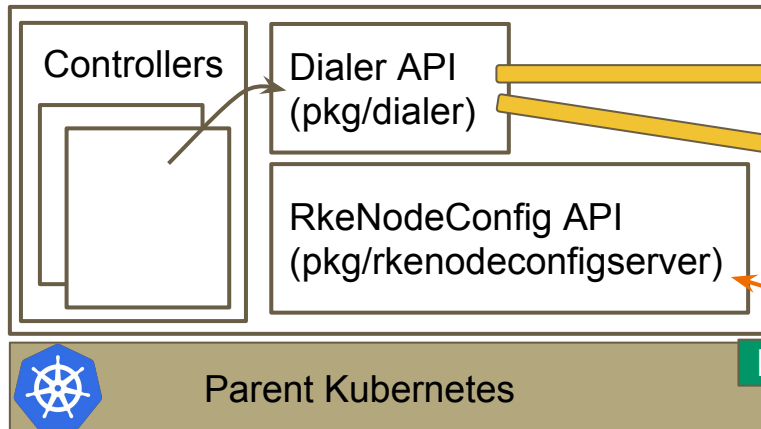
## 1.2. What Server does?



# 1.3. What Agent does?

Use session  
For access  
(k8s, docker)

  
**RANCHER**  
Server



Point 1

Establish websocket session

websocket session  
(/v3/connect)



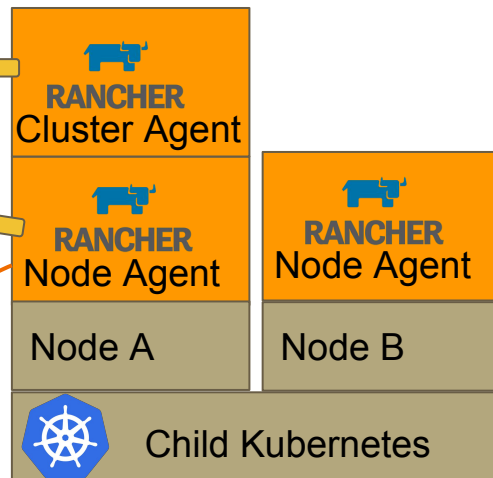
/v3/connect/config

Point 3

Check If file,container need to  
create/run or not periodically

Point 2

Provide TCP Proxy  
via websocket



Rancher Agent basically establishes websocket to provide TCP Proxy and just checks NodeConfig periodically. Almost all configurations will be done/triggered by controllers through websocket

Parent k8s: k8s working with rancher  
Child k8s: k8s deployed by rancher

## 1.4. Rancher 2.0 overview summary

Almost all logics are in Rancher Server and Agent is just sitting as a TCP Proxy

If we want to know more about How Rancher maintain Kubernetes Cluster, It's enough to see just Rancher Server. Because Agent is just to provide proxy.

- b. Check periodically if node need to create something file or run something container



## 2. Rancher Server Internal

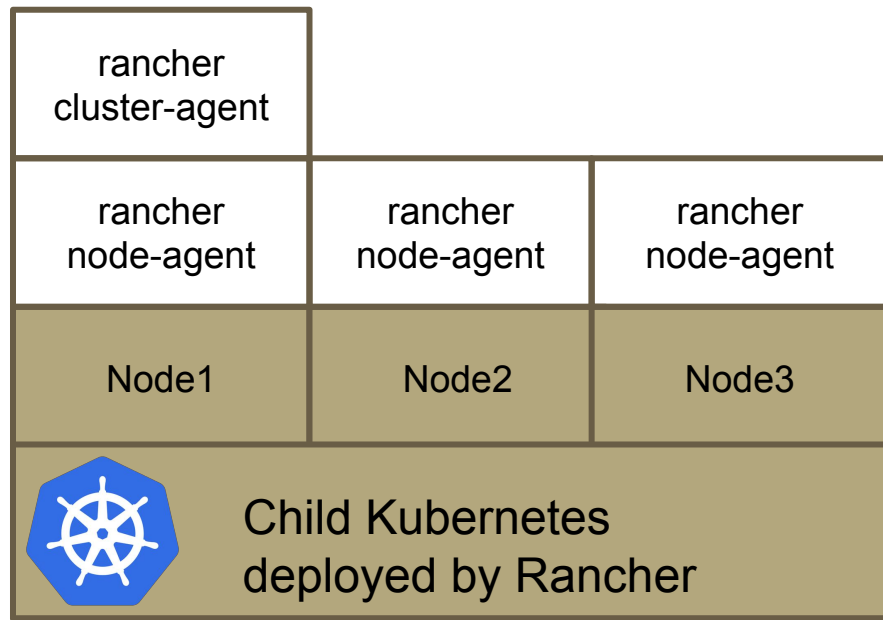
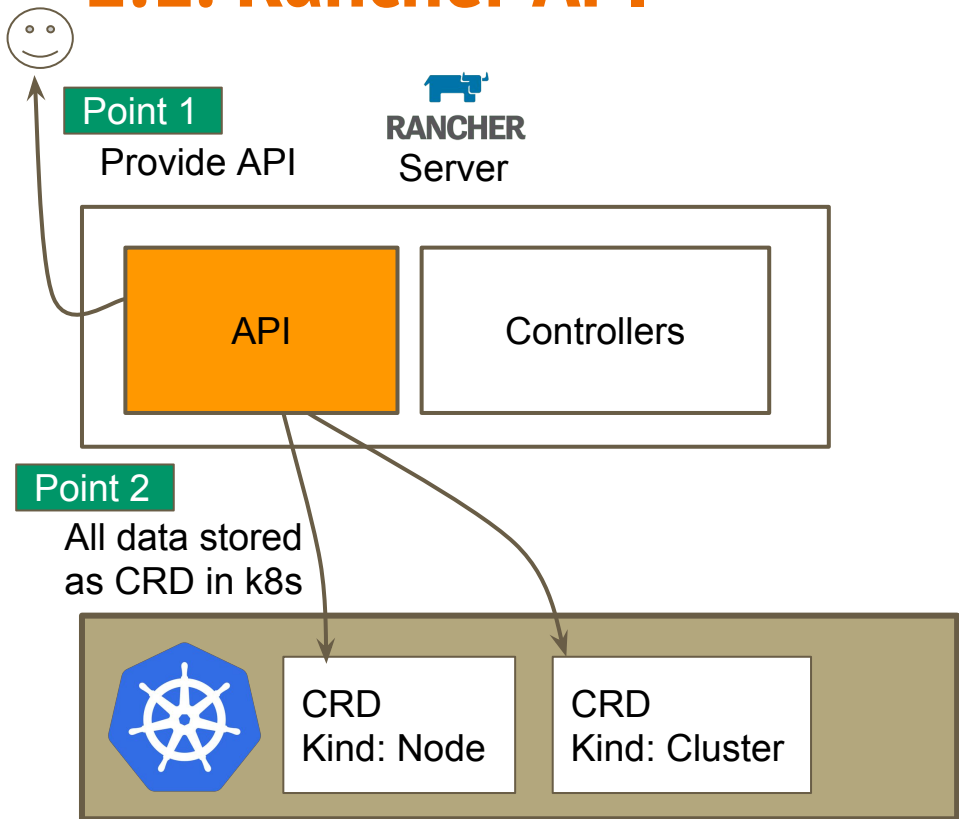
### 1. Rancher Overview

- 1.1. Casts in Rancher 2.0
- 1.2. What Server does?
- 1.3. What Agent does?
- 1.4. Summary

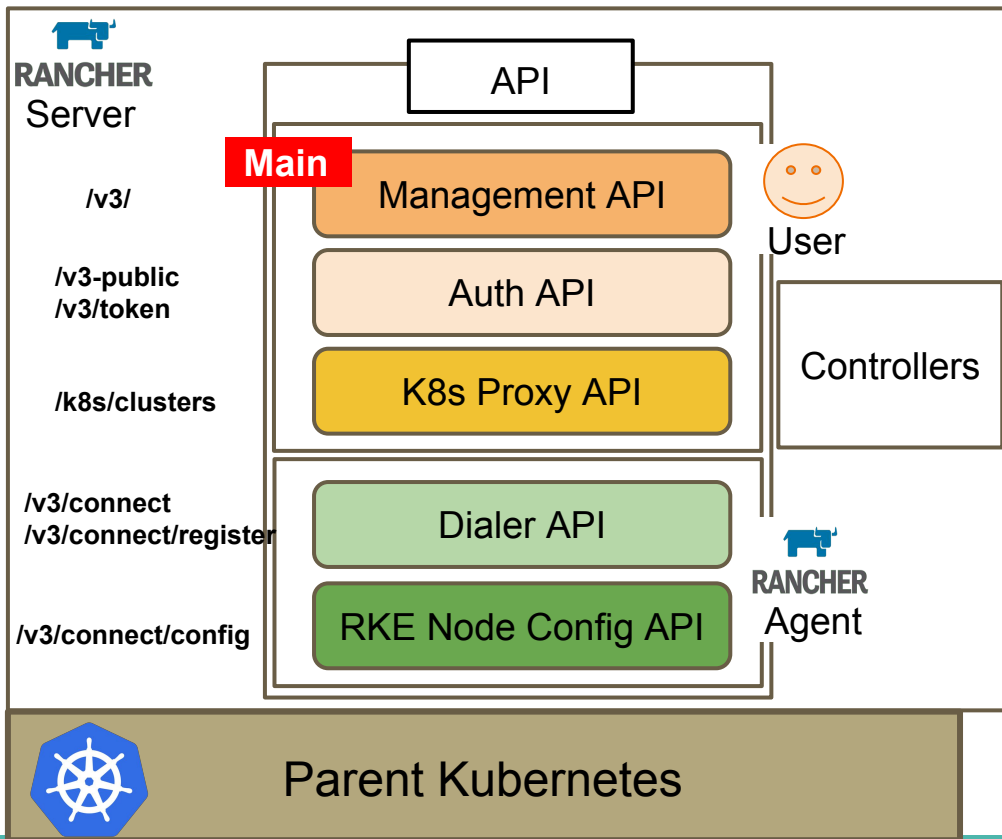
### 2. Rancher Server Internal

- 2.1. Rancher API
  - 2.2. Rancher Controllers
  - 2.3. Example Controllers
-

## 2.1. Rancher API



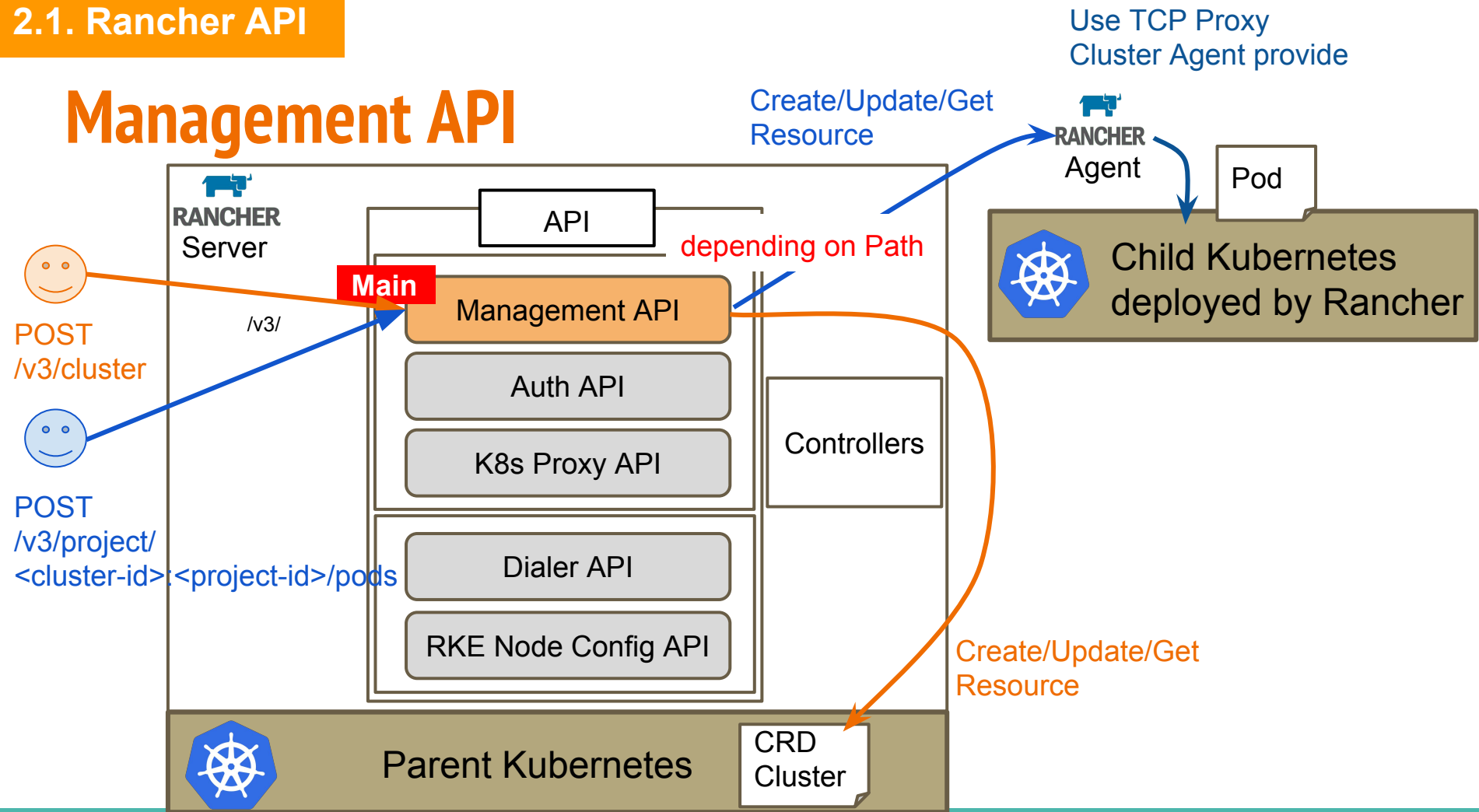
# 5 types of API



- API can be classified into 5 types
- Some API is for only Agent
  - API for user
    - Management
    - Auth
    - K8s Proxy
  - API for agent
    - Dialer
    - RKE Node Config

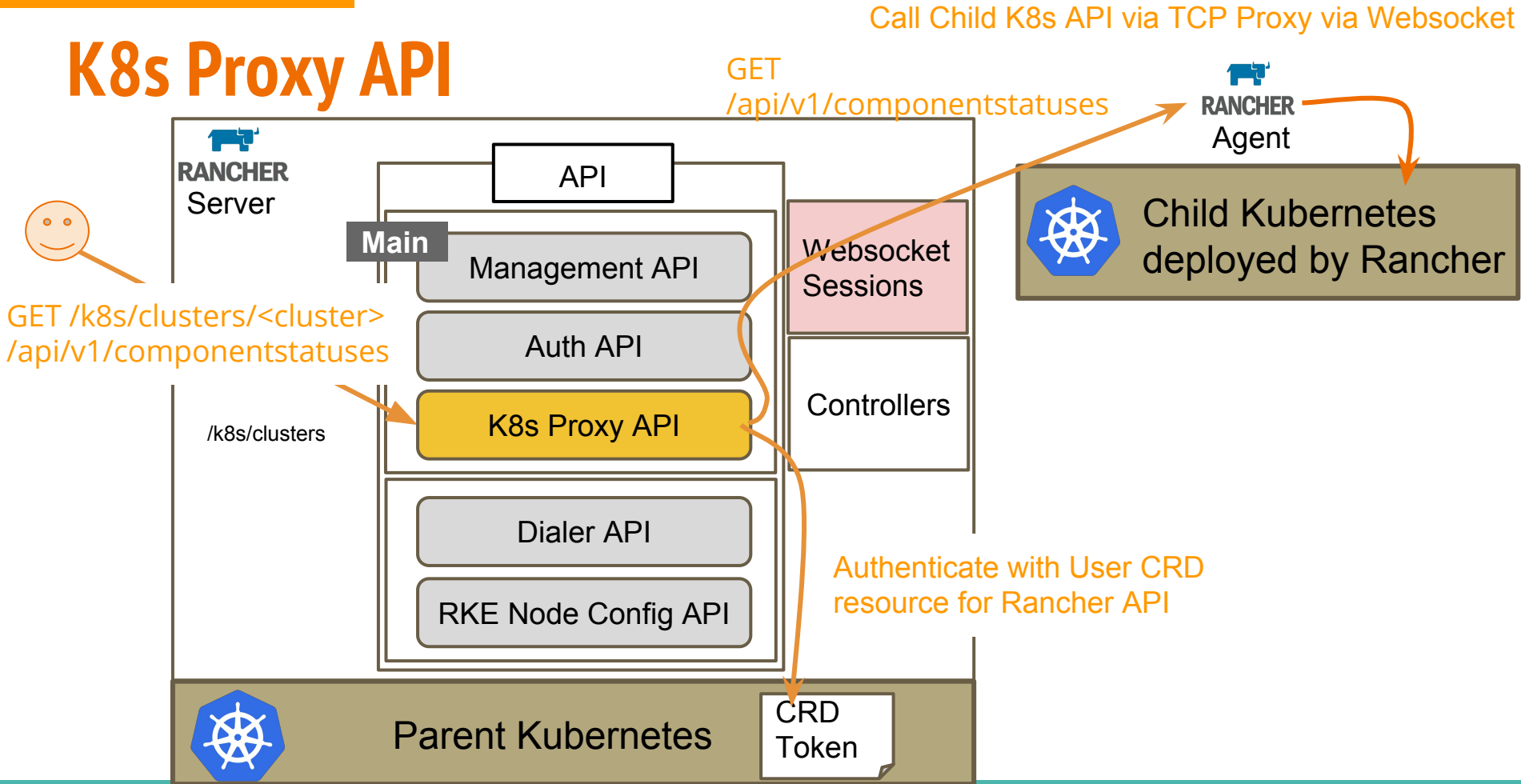
## 2.1. Rancher API

# Management API

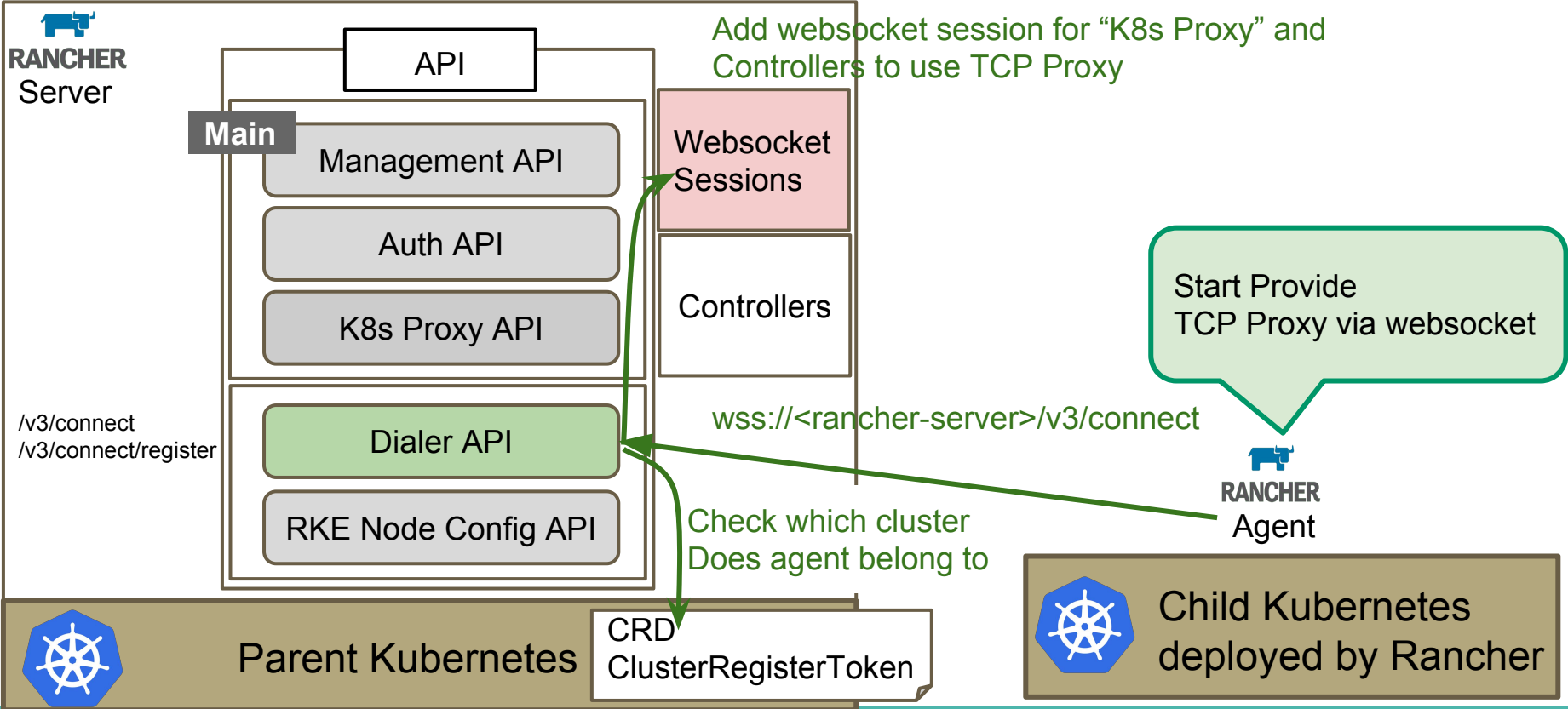


## 2.1. Rancher API

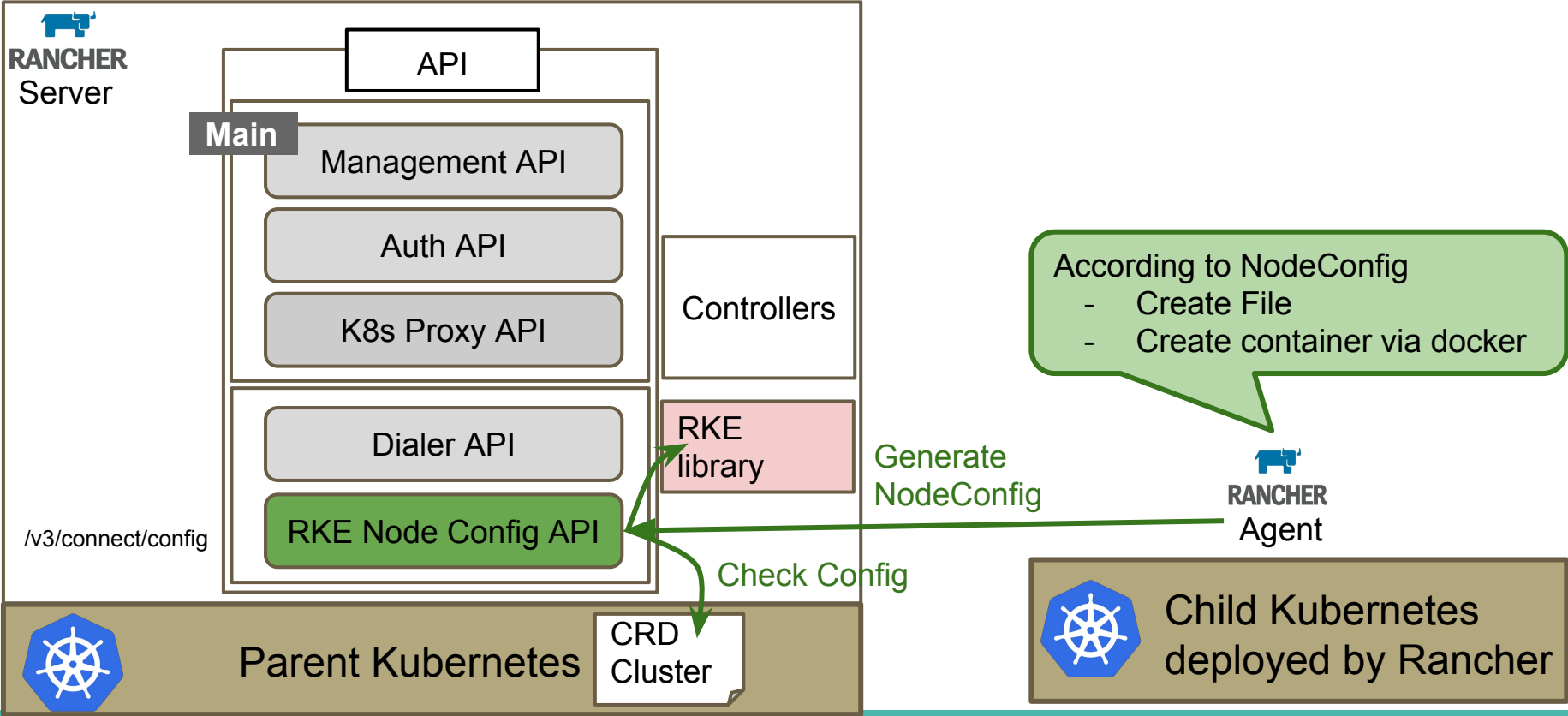
# K8s Proxy API



## Dialer API



# RKE Node Config API



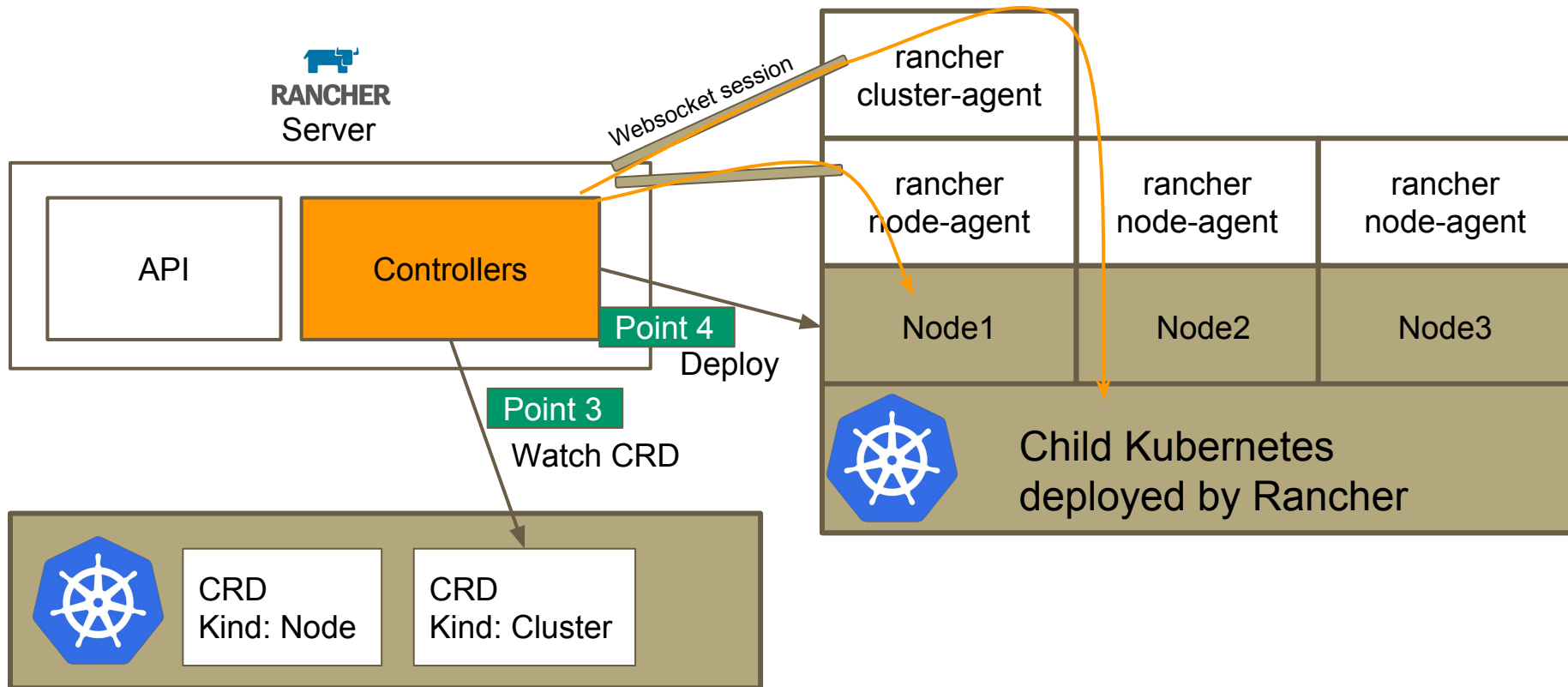
## Point 5

Monitor Cluster/Sync Data

Call docker/k8s API via websocket, If need.

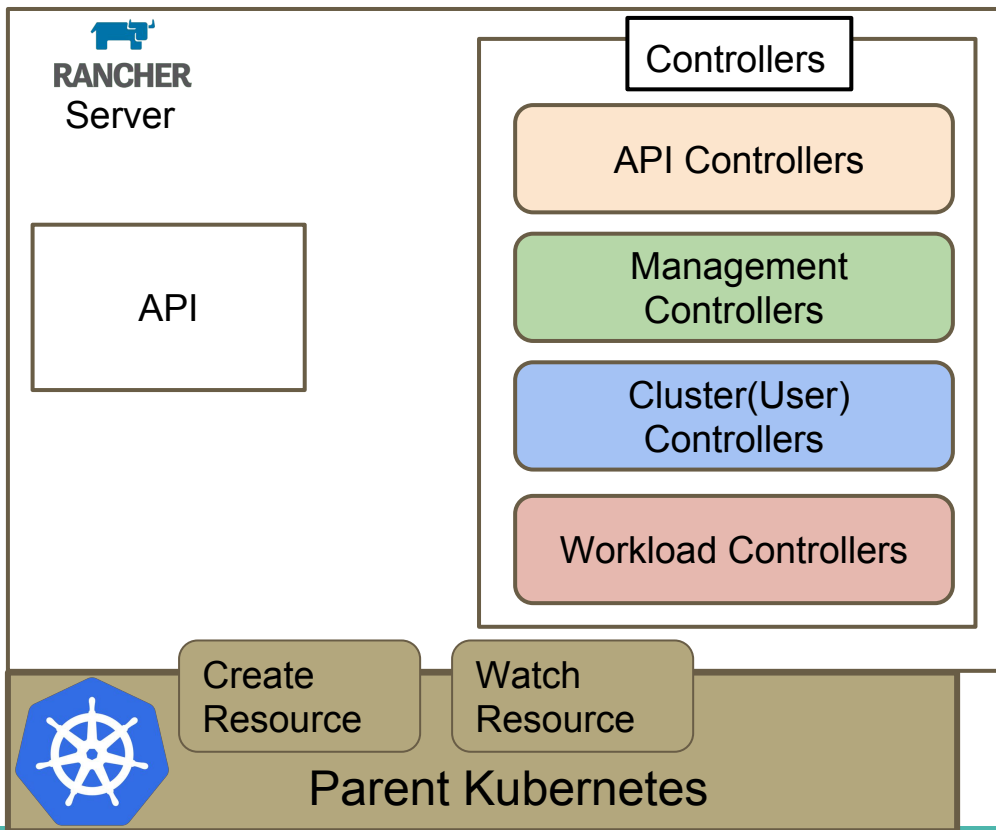
Don't access to docker/k8s api directly from rancher server

# 2.2. Rancher Controllers



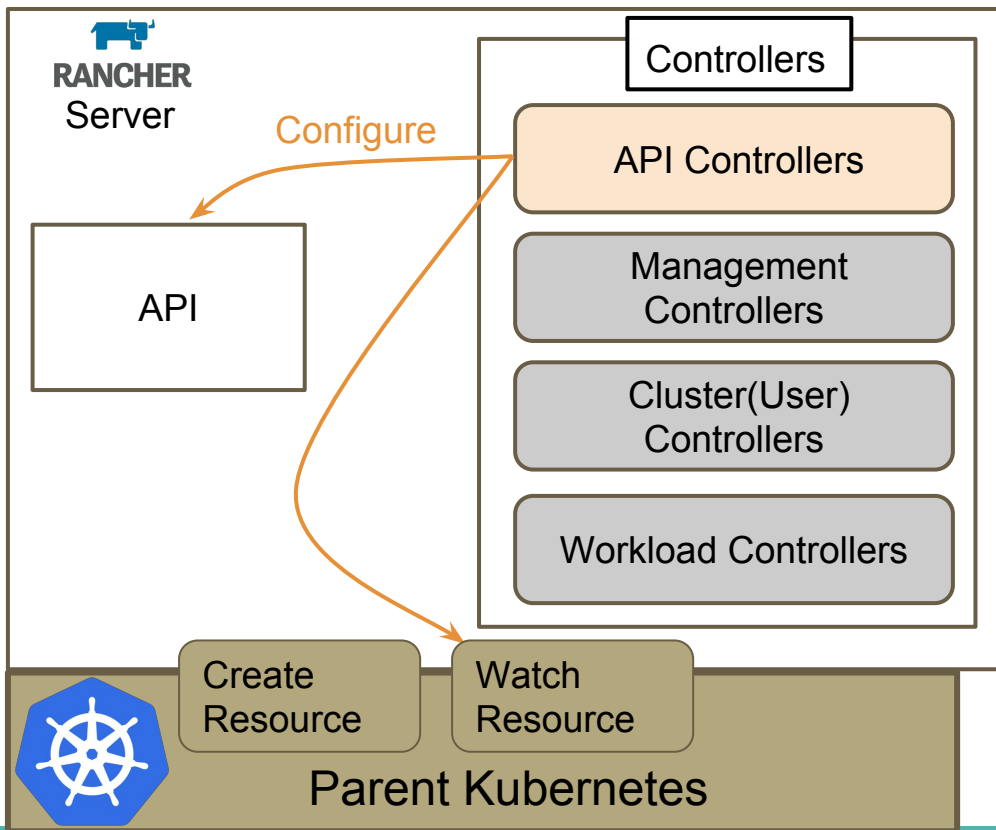


# 4 types of Controllers



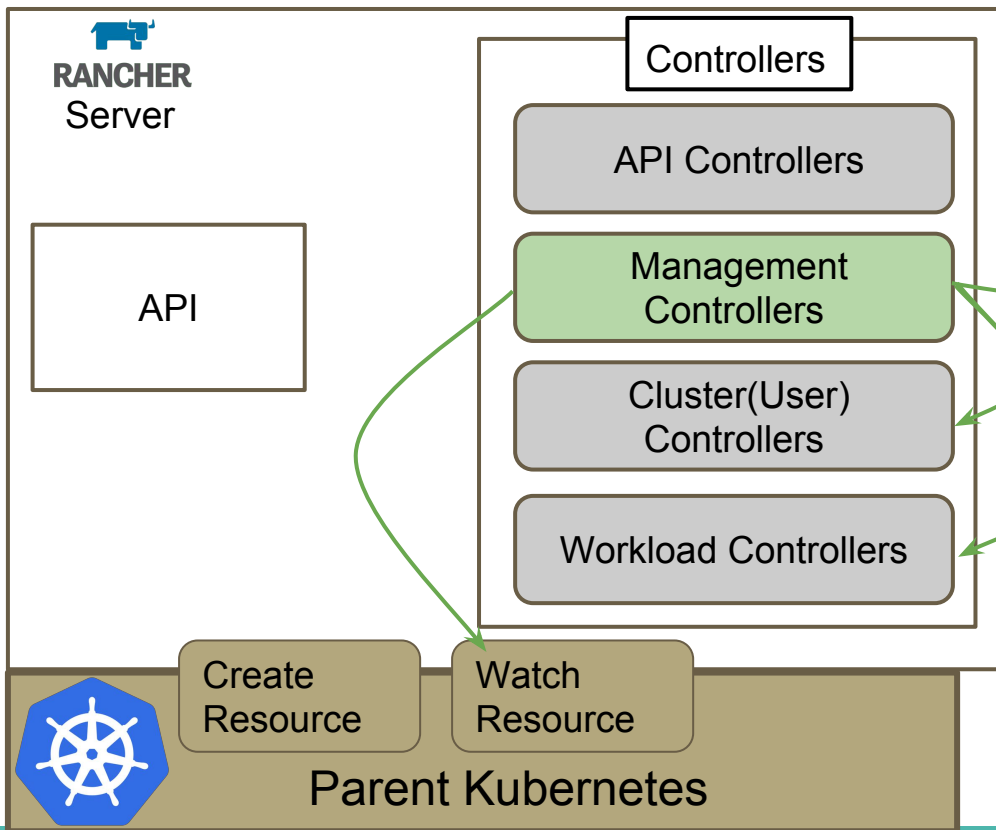
- Rancher Controllers can be classified into 4 types of group
- Each group have own trigger to start
- Triggered when Server start
  - API Controllers
  - Management Controllers
- Triggered when new Cluster detected
  - Cluster(User) Controllers
  - Workload Controllers

# API Controllers



- Watch CRD resource related to API Server Configuration
  - settings
  - dynamicschemas
  - nodedrivers
- Configure API server according to the change of resource

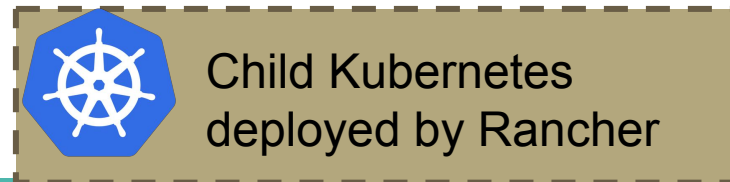
# Management Controllers



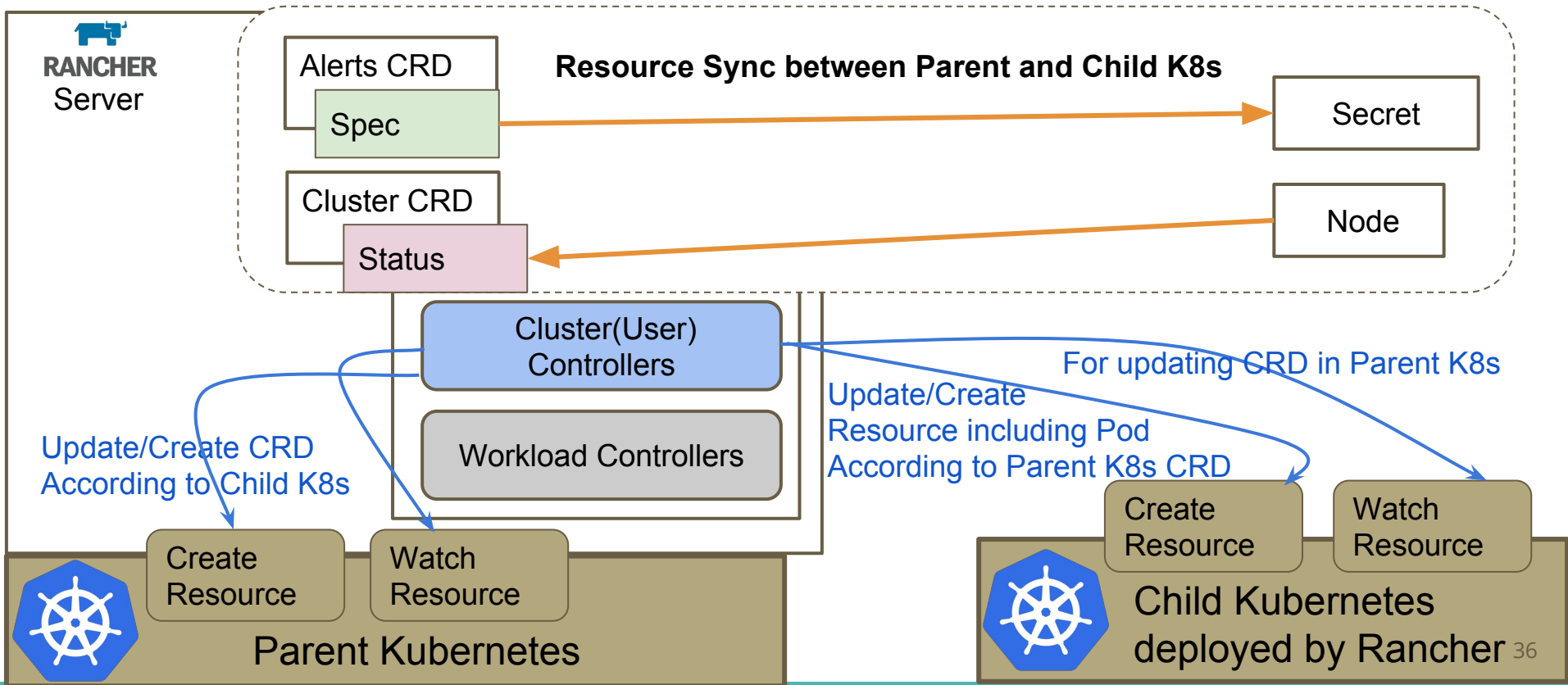
- Watch Cluster/Node related CRD
- Provision/Update Cluster according to the change of resource
- After Provision, Start Cluster(User), Workload Controllers to start data sync and monitor

*Provisioning/Update Cluster*

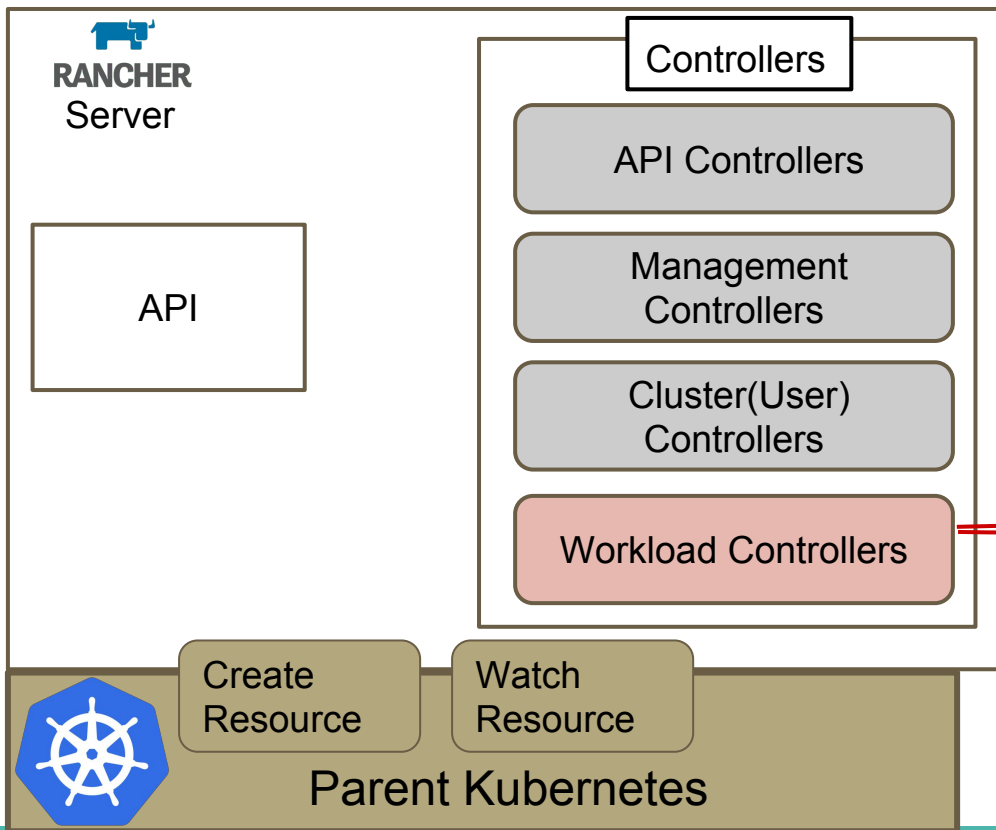
*Start Cluster(User),  
Workload Controllers*



# Cluster(User) Controllers

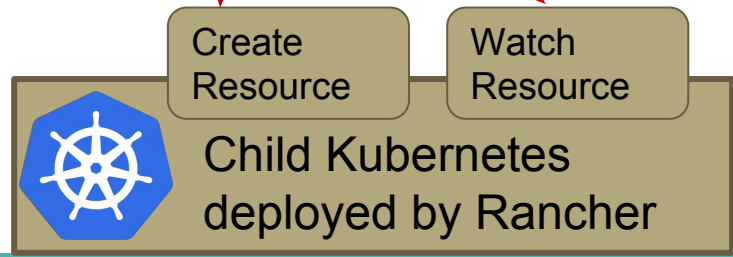


# Workload Controllers



- Watch only resource@Child K8s
- Create/Update additional resource
- These Controller are more like enhancing K8s feature itself

The Simple Custom Controller to extend K8s



## 2.3. Examples: Controllers

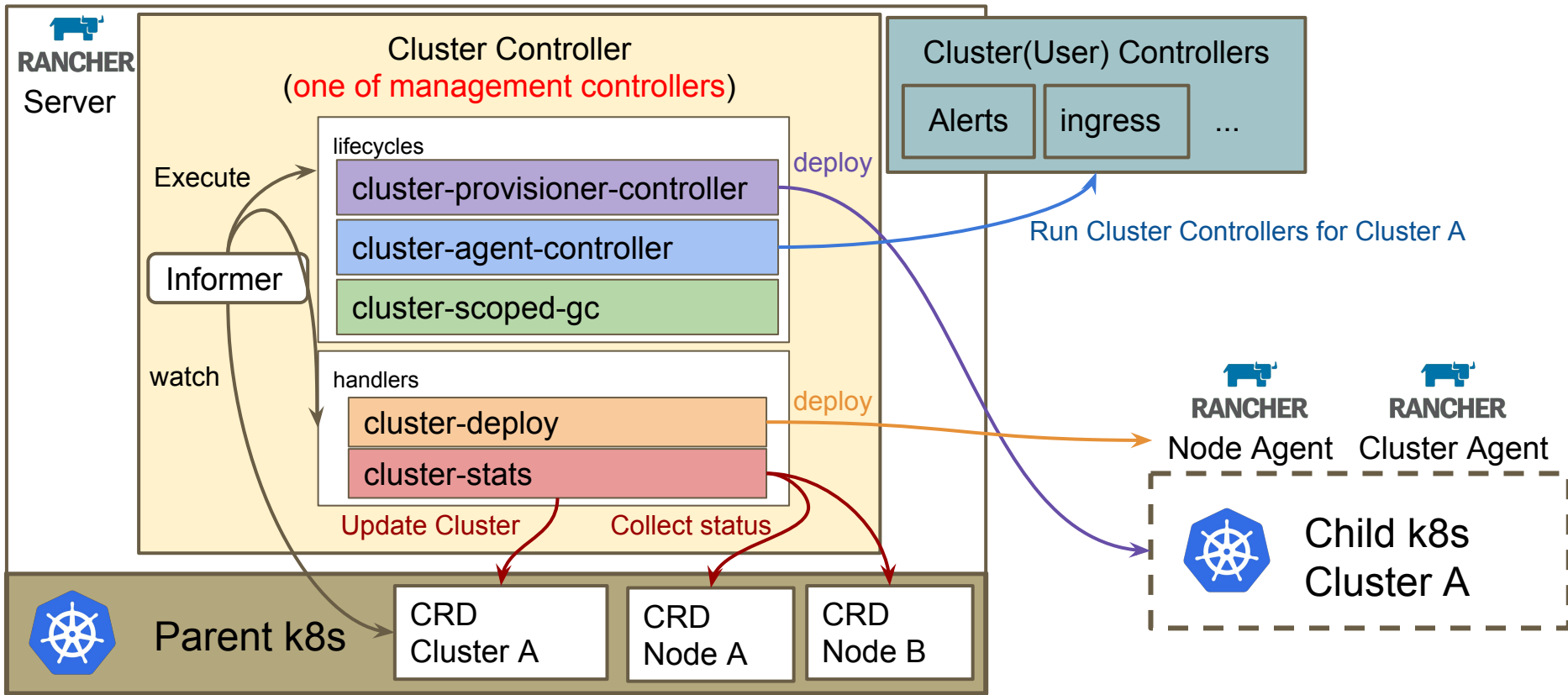
### Management Controllers

- Cluster Controller
- Node Controller

### Cluster(User) Controllers

- ClusterLogging Controller

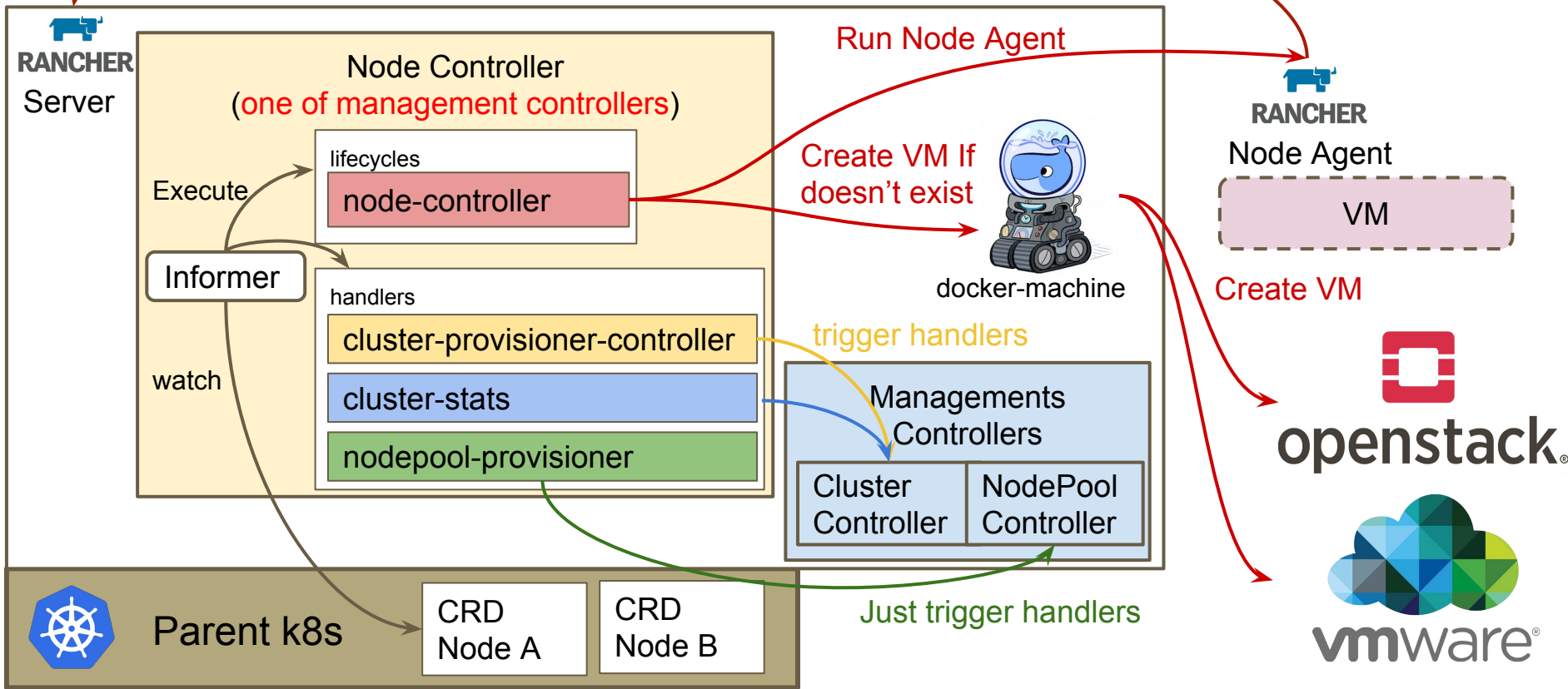
# Cluster Controller Implement (pkg/controllers/management)



## 2.3. Example Controllers

Call `wss://<server>/v3/connect/register`  
To register node into specific cluster

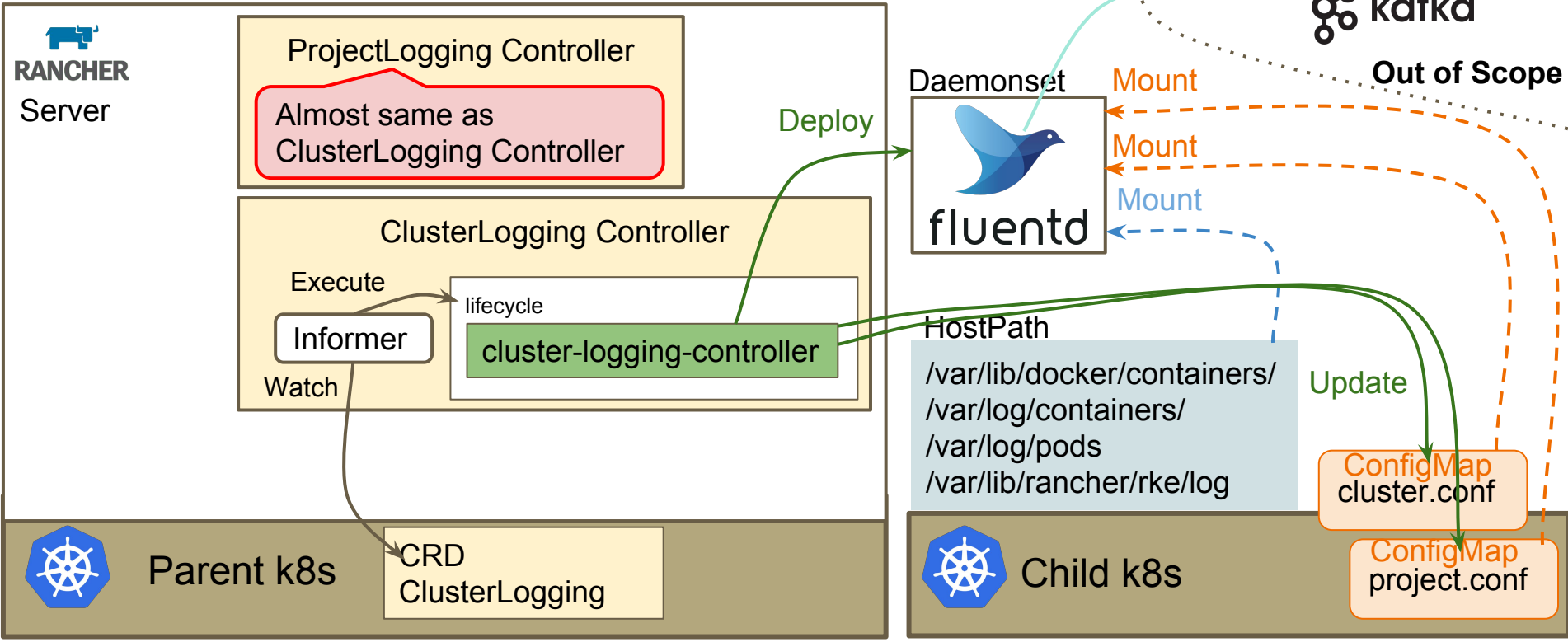
# Node Controller Implement (pkg/controllers/management)





# Cluster(Project)Logging Controller Implement

(pkg/controller/user/logging/)



**How I look Rancher 2.0**

**In the context of backend for Verda k8s as a Service**

# Good thing: we are thinking to utilize

- Less requirement for environment to run
  - but this cause some scalability limitation at the same time though...
- There are some interesting Controllers like alert, logging, eventsync ...
  - We can utilize these feature to manage K8s Cluster
- Easy to modify/add Rancher behaviour thanks to Norman Framework
  - We will utilize this framework to extend even for k8s

# Not good thing: we are thinking to improve

- Poor Document (Currently reading code is only way to know)
  - Norman Framework that Rancher actively used is also less document
- Doesn't support Active-Active HA
  - Scalability limitation cannot be avoided
- 1 binary that have ton of features make it difficult to do performance tuning
- Even for K8s Proxy API, we can not deploy multiple process because that feature depend on websocket session to cluster-agent
- Poor monitoring relying on kubelet and componentstatus
- Upgrading Strategy is just to replace old container with new one. Is it enough?

<https://github.com/rancher/rke/blob/master/services/kubeapi.go#L15> , <https://github.com/rancher/rke/blob/master/docker/docker.go#L72>

# **Future Work**

## **Verda Kubernetes as a Service**

# Future Works

- Use Rancher 2.0 as a backend to manage k8s without any change at Phase 1
  - Modify Rancher and Give feedback to community in the long run after Phase 1 release
    - Enhance scalability
    - Enhance monitoring
    - Cut(or Disable) many unneeded features to us
- Enrich Kubernetes deployed by RKE
  - Support Type Loadbalancer for our XDP based Loadbalancer
  - Support Persistent Volume
  - Add CRD/Controller to support our In-house Component like Kafka, Database as a Service
    - We want Kubernetes to be orchestration tool for System not for Container
- Need more k8s/etcd itself knowledge
  - !!!!Read Code!!!! Not only just books/documents!!
    - Kubernetes
    - Etcd

# We are hiring people!!!

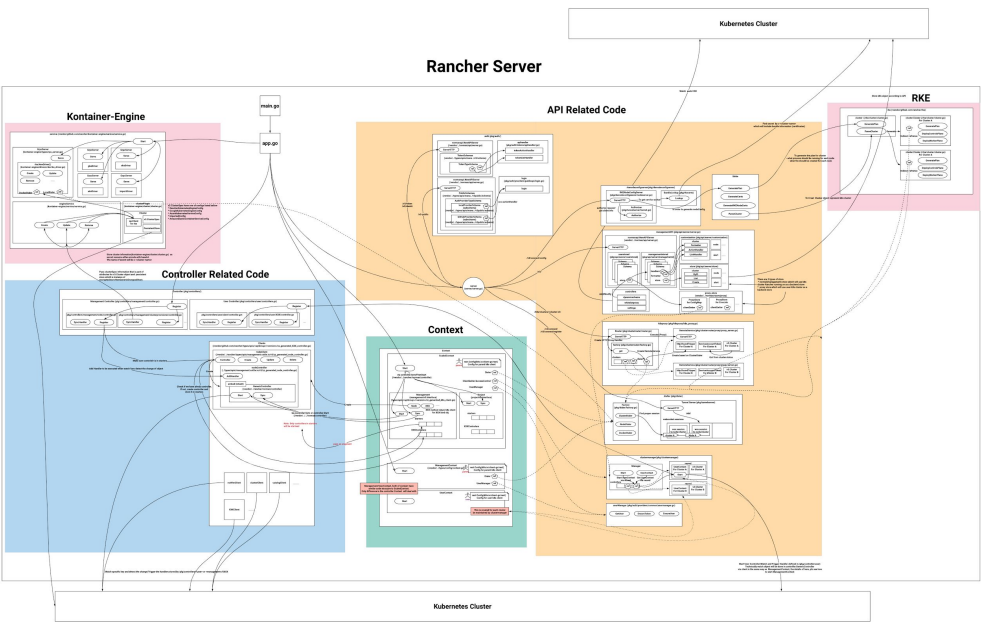
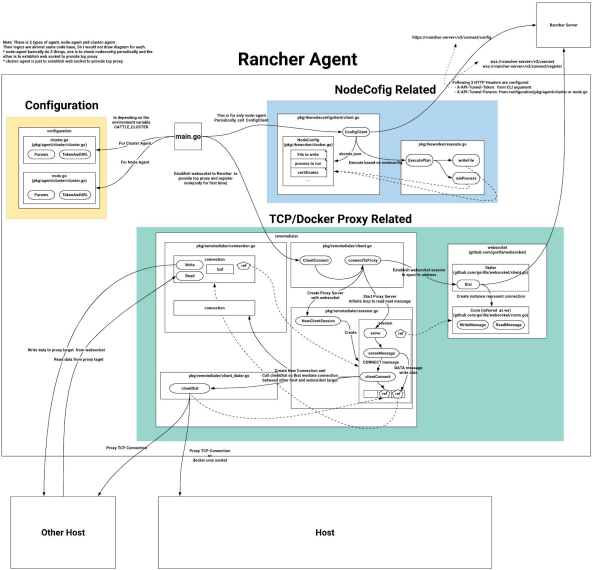
- Love to understand/customize OSS at source code level
  - Kubernetes
  - Etcd
  - OpenStack
  - Rancher
  - Ceph...

<https://linecorp.com/ja/career/position/827>

<https://linecorp.com/ja/career/position/564>

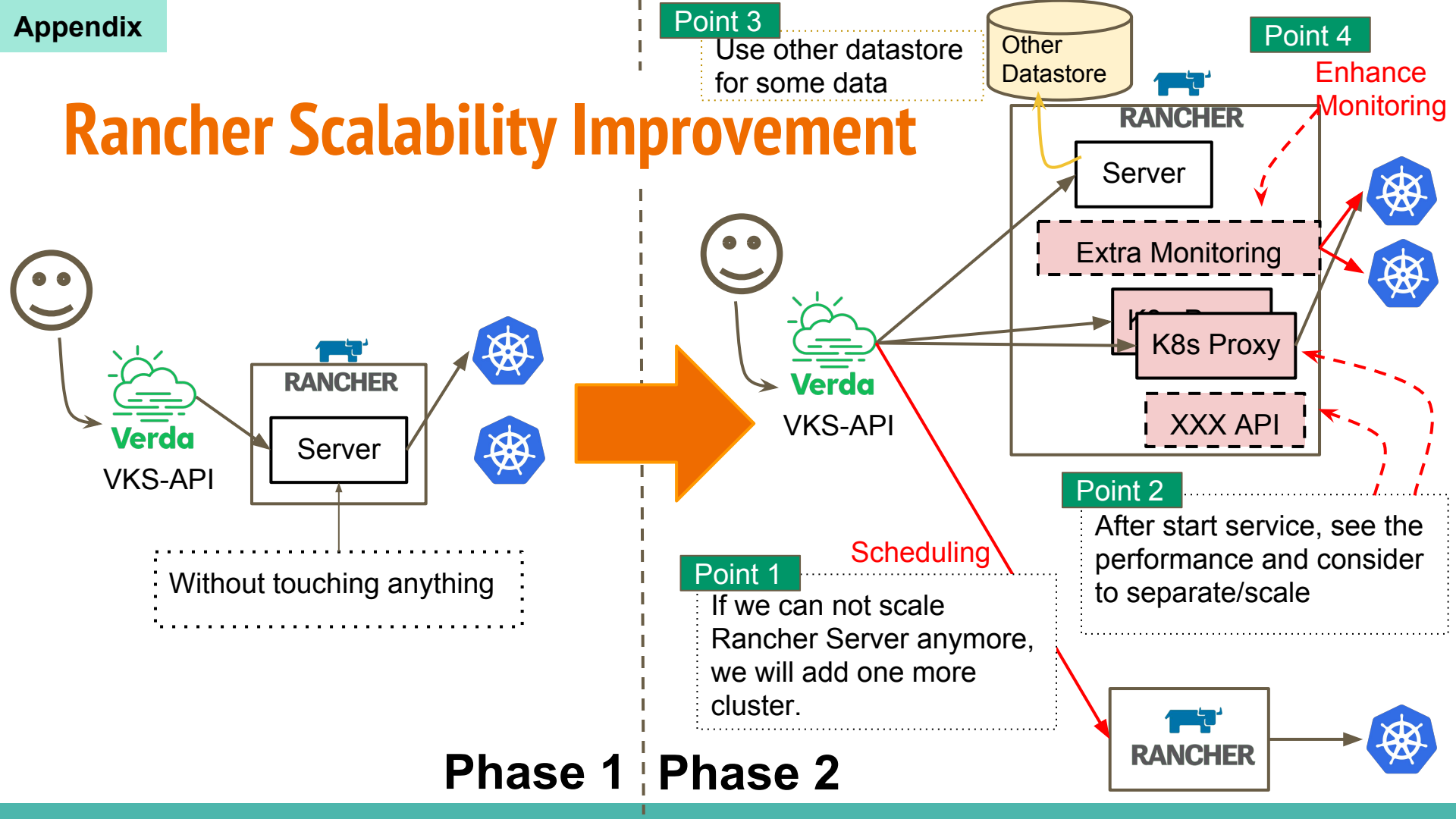


I straighten my understandings as a diagram.  
It's available in (<https://github.com/ukinau/rancher-analyse>)

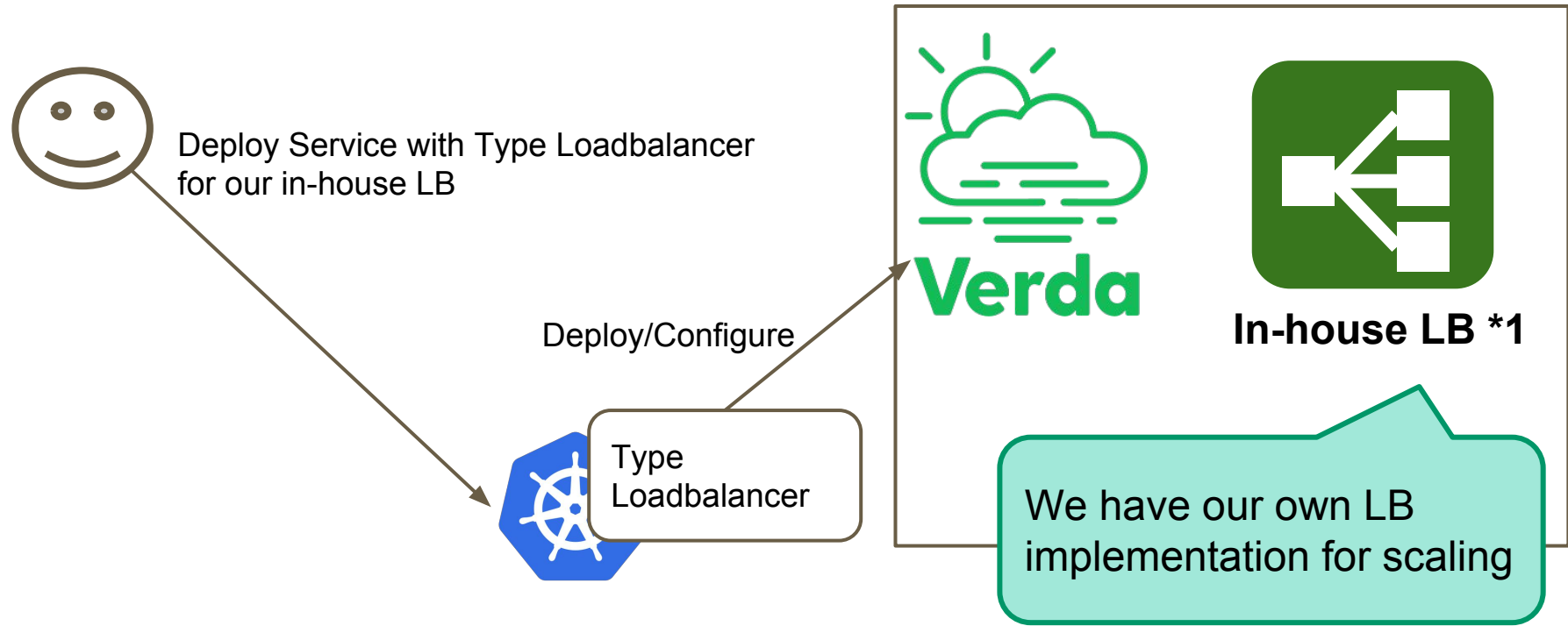




# Rancher Scalability Improvement

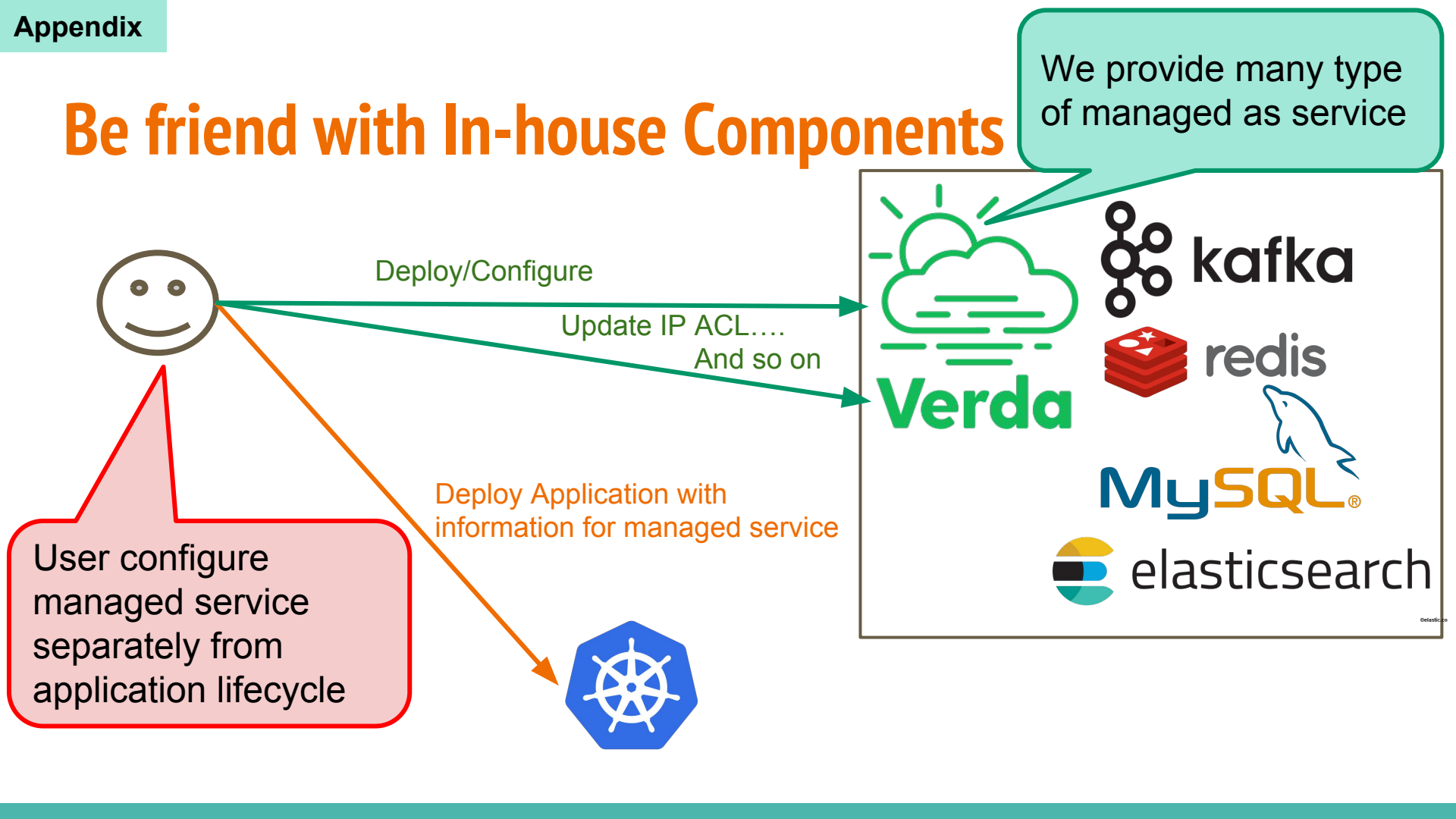


# Support Type Loadbalancer for in-house LB



\*1 [https://www.janog.gr.jp/meeting/janog40/application/files/6115/0105/4928/janog40\\_sp6lb.pdf](https://www.janog.gr.jp/meeting/janog40/application/files/6115/0105/4928/janog40_sp6lb.pdf)

# Be friend with In-house Components



# Be friend with In-house Components

