Rancher 2.3: Technical Architecture

September 2019

Contents

1 Background

According to 451 Research[1], 76% of enterprises will standardize on Kubernetes within the next 3 years because it premises a consistent set of capabilities across any infrastructure – from datacenter to cloud to the edge.

By unifying their IT operations with Kubernetes, enterprises realize key benefits like increased reliability, improved security and greater efficiencies with standardized automation. However, relying on upstream Kubernetes alone often isn't enough for teams to deploy into production. There is a lack of central visibility, consistent policy application and complex management systems. These capabilities are only achievable on a Kubernetes management platform like Rancher. Rancher has been built from the ground up to deliver:

- **Consistent Cluster Operations** – Simplified Kubernetes upgrades, backups and deployments.
- **Security Policy & User Management** – Consistent RBAC, PSP and user management.
- **Shared Tools & Services** – Out of the box access to tools and services.

2 Rancher 2.3: Built on Kubernetes

Rancher 2.3 is a complete container management platform built on Kubernetes. As illustrated Figure 1, Rancher 2.3 contains three major components - a certified Kubernetes distribution, a Kubernetes management platform and application management.

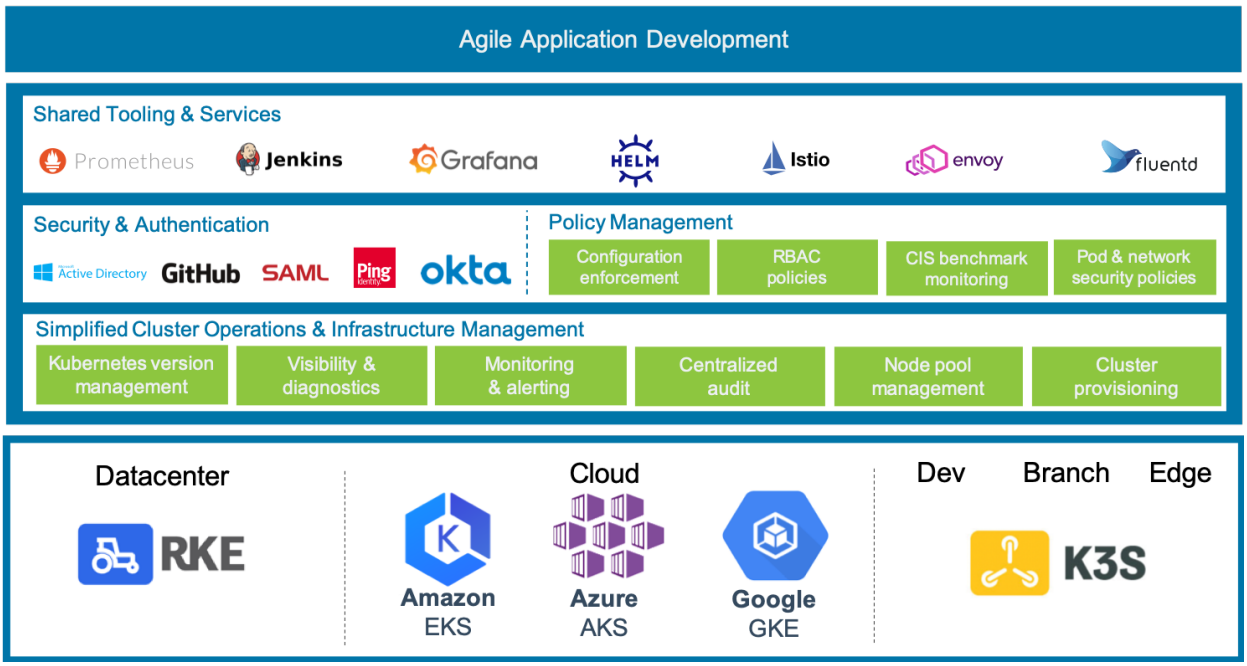

Figure 1 Overview of Rancher

[1] "Kubernetes and Beyond – Effective Implementation of Cloud Native Software in the Enterprise" by Jay Lyman, Principal Analyst 451 Research – Download Whitepaper

2.1 Certified Kubernetes with Rancher Kubernetes Engine (RKE)

RKE is an extremely simple, lightning fast Kubernetes installer that works everywhere. RKE is particularly useful in standing up Kubernetes clusters on VMware clusters, bare metal servers, and VM instances on clouds that do not yet support Kubernetes service. In addition, many people use RKE in cloud providers that already support Kubernetes services so that they have a consistent Kubernetes implementation everywhere. In Rancher 2.3, clusters can be provisioned on Linux x86_64 and Arm64 architectures as well as Windows 19.03 systems.

RKE within Rancher manages the complete lifecycle of Kubernetes clusters from initial install to on-going maintenance. Rancher users can:

a. Automate VM instance provisioning on many clouds using machine drivers.
b. Install Kubernetes control plane and etcd database nodes.
c. Worker nodes provisioned on Windows and Linux Arm64 and x86_64 nodes.
d. Add or remove nodes in existing Kubernetes clusters.
e. Upgrade Kubernetes clusters to new versions.
f. Monitor the health of Kubernetes clusters.

2.2 Consistent Cluster Operations

With Rancher 2.3, you can choose to manage your own existing Kubernetes clusters provisioned with by existing tools or use Kubernetes clusters managed by a cloud. Kubernetes services like EKS, GKE and AKS can easily be provisioned or imported into your Rancher installation provision and operate RKE Kubernetes clusters on any cloud, virtualized, or bare metal infrastructure.

Rancher can easily be managed with Infrastructure-as-code with the Rancher 2.0 Terraform provider. From this you can easily store your configurations for clusters, namespaces, secrets and catalog apps in Git. Rancher has a powerful API that can be scripted against to perform routine tasks.

2.3 Security Policy and User Management

Unified cluster management starts with centralized authentication. Rancher 2.3 provides an authentication proxy for all Kubernetes clusters under management. This is a crucial capability for enterprise IT to adopt cloud Kubernetes services like GKE.

GKE normally requires its users to authenticate using their Google credentials. With Rancher 2.3, an enterprise developer can sign into a GKE cluster using the Active Directory credentials managed by corporate IT.

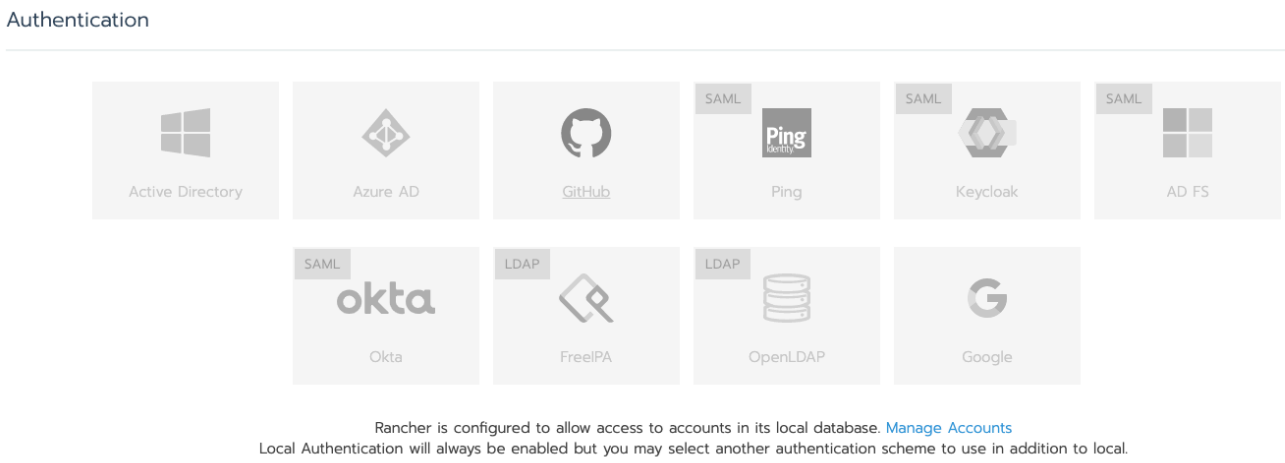

Figure 3 Authentication providers in Rancher 2.3

Rancher 2.3 builds upon centralized authentication with centralized access control policies defined at a global level and applied to clusters under management. Leveraging the native RBAC capabilities of Kubernetes, Rancher 2.3 enables IT administrators to configure and enforce access control and security policies across multiple Kubernetes clusters.

Rancher 2.3 introduces a new concept called Projects. A project is one or more Kubernetes namespaces and their associated policies such as RBAC rules, resource quota, etc. Users can create projects and assign other users or groups to the projects he owns. Users can be assigned different roles across multiple projects. For example, a developer can be given full create/read/update/delete privilege in a "dev" project but just read-only access in the staging and production projects. Users can only create, modify, or delete namespaces in the projects they are a member of. This greatly enhances the multi-tenant self-service functionality of Kubernetes.

In a future release, Rancher will provide visibility into capacity and cost of underlying resources consumed by Kubernetes clusters.

2.4 Application Workload Management

Rancher 2.3 UI does not attempt to hide the underlying Kubernetes concepts and introduce an application deployment framework that is different from Kubernetes. Rancher provides an easy-to-use UI for native Kubernetes resources like pods and deployments.

The app catalog experience in Rancher 1.0 has been adapted to support Helm charts. Helm is a powerful templating mechanism for deploying applications on Kubernetes. But users still need to read through lengthy documentation to understand exactly what variables to set and the right values for these variables. This is an error-prone process. Rancher simplifies Helm chart deployment by exposing just the right set of variables and by guiding the user through the process. Rancher catalog guides the user by asking the right questions and presenting sensible defaults and multiple-choice values.

Rancher 2.3 works with any CI/CD systems that integrate with Kubernetes. For example, Jenkins, Drone, and GitLab will continue to work with Rancher 2.3 as they did with Rancher 1.0. Rancher 2.3 additionally

includes a managed CI/CD service built on Jenkins. The Rancher 2.3 CI/CD service seamless integrate with Rancher UI.

Rancher 2.3 works with any monitoring and logging systems that integrates with Kubernetes. For an out of the box experience, users can use the built in Prometheus functionality. If existing systems like DataDog, Sysdig, or ELK are in place, they will continue to work with Rancher 2.3. For log aggregation, Rancher has a built in Fluentd service that will ship logs from the hosts.

3 High-level Architecture

Rancher 2.3 software consists of two parts. The Rancher server components manages the entire Rancher deployment. Rancher also deploys agent components into Kubernetes clusters and nodes.

Figure 2 illustrates the high-level architecture of Rancher 2.3. 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.

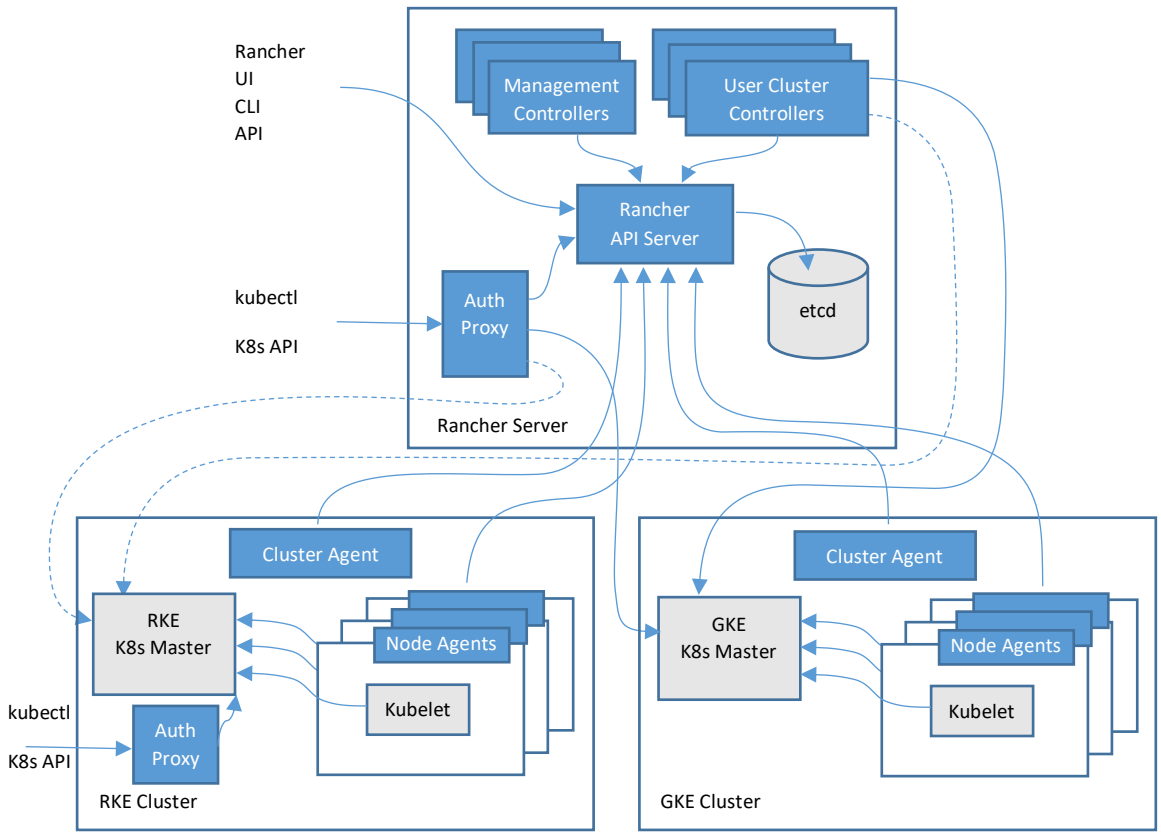

Figure 2 Rancher 2.3 High Level Architecture

4 Rancher Server Components

In this section we describe the functionalities of each Rancher server components.

4.1 Rancher API Server

Rancher API server is built on top of an embedded Kubernetes API server and etcd database. All Rancher specific resources that are being created using Rancher API, get translated to CRD (Custom Resource Definition) objects, with their lifecycle being managed by one or several Rancher controllers.

Rancher API Server is the foundational layer for all controllers in the Rancher server. It includes the following functionalities:

- User facing API schema generation with an ability to plug custom formatters and validators.
- Controller interfaces generation for CRDs and native Kubernetes objects types.
- Object lifecycle management framework.
- Conditions management framework.
- Simplified generic controller implementation by encapsulating TaskQueue and SharedInformer logic into a single interface.

4.2 Management Controllers

The management controllers perform the activities that happen at the Rancher server level, not specific to an individual cluster. The activities include:

a. Configuring access control policies to clusters and projects.
b. Managing pod security policy templates.
c. Provisioning clusters by invoking the necessary Docker machine drivers and invoking Kubernetes engines like RKE and GKE.
d. Managing users – CRUD operations on users.
e. Managing global-level catalog, fetch content of the upstream Helm repo, etc.
f. Managing cluster and project-level catalogs.
g. Aggregating and displaying cluster stats and events.
h. Managing of node drivers, node templates, and node pools.
i. Managing cluster cleanup when cluster is removed from Rancher.

4.3 User Cluster Controllers

User cluster controllers perform activities specific to a cluster. User cluster controllers are spread out across the running Rancher server pods for horizontal scaling. Activities include:

a. Managing workloads, which includes, for example, creating pods and deployments in each cluster.
b. Applying roles and bindings that are defined in global policies into every cluster.
c. Propagating information from cluster to rancher server: events, stats, node info, and health.
d. Managing network policies.
e. Managing alerts, monitoring, log aggregation, and CI/CD pipelines.

f. Managing resource quota.
g. Propagating secrets down from Rancher server to individual clusters.

User cluster controllers connect to API servers in GKE clusters directly, but tunnel through the cluster agent to connect to API servers in RKE clusters.

4.4 Authentication Proxy

The authentication proxy proxies all Kubernetes API calls. It integrates with authentication services like local authentication, Active Directory, and GitHub. On every Kubernetes API call, the authentication proxy authenticates the caller and sets the proper Kubernetes impersonation headers before forwarding the call to Kubernetes masters. Rancher communicates with Kubernetes clusters using a service account.

The authentication proxy connects to API servers in GKE clusters directly, but tunnels through the cluster agent to connect to API servers in RKE clusters.

In Rancher 2.2, the authentication cluster endpoint was introduced into RKE based clusters to bring centralized auth to the local cluster. This provides increased availability by removing the Rancher 2.3 management server from the authentication path. Allowing disconnected management and operations of your Kubernetes clusters.

5 Rancher Agent Components

In this section, we describe software components deployed in Kubernetes clusters managed by Rancher.

5.1 Cluster Agents

Rancher deploys one cluster agent for each Kubernetes cluster under management. The cluster agent opens a WebSocket tunnel back to Rancher server so that the user cluster controllers and authentication proxy can communicate with user cluster Kubernetes API server. Note that only RKE clusters and imported clusters utilize the cluster agent to tunnel Kubernetes API. Cloud Kubernetes services like GKE already exposes API endpoint on the public Internet and therefore does not require the cluster agent to function as a tunnel.

Cluster agents serve two additional functions:

a. They serve as a proxy for other services in the cluster, like Rancher's built-in alert, log aggregation, and CI/CD pipelines. In fact, any services running in user clusters can be exposed through the cluster agents. This capability is sometimes called "the magic proxy."
b. During registration, cluster agents get service account credentials from the Kubernetes cluster and send the service account credentials to the Rancher server.

5.2 Node Agents

Node agents are primarily used by RKE to deploy the components during initial install and follow-on upgrades. Node agents, however, are deployed on cloud Kubernetes clusters like GKE even though they

are not needed for Kubernetes install and upgrade. Node agents serve several additional functions for all clusters:

a. Fallback for cluster agents: if the cluster agent is not available for any reason, Rancher server will use node agent to connect to the Kubernetes API server.
b. Proxy for `kubectl` shell. Rancher server connects through node agents to tunnel the `kubectl` shell in the UI. Node agent runs with more privileges than cluster agent, and that additional privilege is required to tunnel the `kubectl` shell.

6 Upgrade

Users can upgrade to new versions Rancher 2.3 by upgrading the Rancher server. Rancher 2.3 handles RKE cluster upgrades. Rancher 2.3 integrates with cloud providers like GKE to upgrade GKE clusters. Rancher 2.3 does not attempt to upgrade imported Kubernetes clusters. As a part of Rancher 2.3, Kubernetes releases are now able to be done out of a Rancher release cycle. This allows faster delivery time to end users and minimizes the scope of change.

Rancher 1.0 was built on Docker and cannot be upgraded to a Kubernetes cluster without disrupting the workload. Rancher 1.0 users must setup a separate Rancher 2.3 cluster, migrate their workload, and decommission the Rancher 1.0 cluster.

7 High Availability

Users may use a dedicated RKE cluster to run the Rancher server. The standard Rancher 2.3 installation guide, for example, creates an RKE deployment with 3 nodes, each running one instance of the API server and the etcd database. Rancher server automatically imports the Kubernetes cluster it runs on. It is called “the local cluster.” Rancher will leverage the Kubernetes API and indirectly use that clusters etcd as the primary datastore.

8 Scalability

8.1 Scalability of Kubernetes Clusters

As of Kubernetes version 1.6, A Kubernetes cluster can scale to 5,000 nodes and 150,000 pods. User can expect Rancher 2.3 to manage and provision RKE clusters up to that scale as well.

8.2 Scalability of Rancher Server

There is no inherent limit on how many Kubernetes clusters each Rancher server can manage. We do not expect an issue for Rancher 2.3 to manage up to 1,000 clusters.

The real scalability limits of Rancher server are:

a. Total nodes across all clusters
b. Users and groups

c. Events collected from all clusters

Rancher server stores all the above entities in the underlying Kubernetes etcd database. We will improve scalability along these dimensions over time to meet user needs.

To connect with the Rancher team, visit rancher.com, email us at info@rancher.com or follow us on Twitter - @rancher_labs