Cisco ACI and Rancher Integration with RKE 1.4.9

Revised: November 30, 2023,

New and Changed Information

The following table provides an overview of the significant changes up to this current release. The table does not provide an exhaustive list of all changes or of the new features up to this release.

Cisco ACI CNI Release Version	Feature	Description
6.0.3.1	Rancher support with Cisco ACI Container Network Interface (CNI) for Kubernetes 1.24 or later.	You can integrate Rancher with Cisco ACI to simplify the management of Kubernetes clusters.

Cisco ACI and Rancher Integration

Rancher is a software stack that simplifies the management of containers. Rancher includes the Rancher Kubernetes Engine (RKE), a certified Kubernetes distribution. It operates within platform-independent Docker containers, simplifying installation and providing consistent management of Kubernetes across all platforms. You can install and manage Kubernetes on VMware clusters, bare-metal servers, and on VMs in clouds—regardless of whether the clouds support Kubernetes service.

This document provides an introduction to Rancher and instructions for integrating it with Cisco ACI. It is assumed that you are familiar with the Cisco ACI CNI plug-in and Kubernetes.

This document describes the integration details for Cisco ACI Release 6.0(3), or later, with ACI CNI release 6.0.3.1 using RKE 1.4.9. The document also has information about importing the installed clusters to Rancher 2.7.9 or later.

Rancher Architecture

Rancher contains several key components that enable you to install and maintain Kubernetes clusters. They include Rancher Kubernetes Engine (RKE), Rancher server, and cluster and node agents.

This section describes key components of Rancher. For details, see, *Rancher: Technical Architecture*, and other content on the Rancher website.

Rancher Kubernetes Engine

RKE is Kubernetes distribution certified by the Cloud Native Computing Foundation (CNCF). It runs entirely within Docker containers. You use RKE to install Kubernetes on VMware clusters, bare metal servers, and VMs on clouds, including clouds that do not support Kubernetes.

Rancher server

The Rancher server contains several components that manage the entire Rancher deployment:

- **Rancher API server:** The Rancher API server is the foundation for all Rancher server controllers. All Rancher-specific resources that are created using Rancher API are translated to custom resources definition (CRD) objects. Rancher controllers manage the lifecycle of the objects.
- **Management controller:** Management controllers perform the activities that occur at the Rancher server level that are not specific to an individual cluster. Such activities include configuring access policies to clusters and projects and managing the global-level catalog.

- **User cluster controller:** User cluster controllers perform activities specific to a cluster. For horizontal scaling, user cluster controllers are distributed across the Rancher server pods. Cluster-specific activities include managing workloads and applying roles and bindings that are defined in global policies.
- **Authentication proxy:**

  The authentication proxy proxies all Kubernetes API calls. It integrates with local authentication, Active Directory, and GitHub. For every Kubernetes API call, the proxy authenticates the caller and sets the correct Kubernetes impersonation headers. It forwards the call to the Kubernetes master nodes.

Rancher agents

Rancher deploys agents into Kubernetes nodes and clusters:

- **Cluster agent:** Rancher deploys one cluster agent for each managed Kubernetes cluster. The cluster agent opens a web-socket tunnel to the Rancher server so the user cluster controllers and authentication proxy can communicate with the user cluster Kubernetes API server.
- **Node agent:** RKE uses node agents primarily to deploy components during initial installation and subsequent upgrades. The node agents are installed also on cloud Kubernetes clusters to provide extra functions, such as fallback for cluster agents and proxy for the kubectl shell.

Prerequisites for Integrating Rancher

Complete the following tasks before integrating Rancher, following the referenced procedures on the Rancher and Cisco websites.

1. Configure the following:
   - A Cisco Application Centric Infrastructure (ACI) tenant
   - An attachable entity profile (AEP)
   - A VRF
   - A Layer 3 outside connection (L3Out)
   - A Layer 3 external network for the cluster you plan to provision with RKE

   

   Note The VRF and L3Out in Cisco ACI that you use to provide outside connectivity to Kubernetes external services can be in a tenant. The VRF and L3Out are usually in the common tenant or in a tenant that you dedicate to the Kubernetes cluster. Y can also have separate VRFs, one for the Kubernetes bridge domains and one for the L3Out, and you can configure route leaking between them.

2. Install the Cisco ACI CNI tools from the Cisco Download site for the specific Linux distribution on a tools server that you will use to provision the cluster.

   This server must have SSH access and reachability to all nodes in the cluster that you will provision.

3. Prepare the nodes for Cisco ACI Container Network Interface (CNI) plug-in installation.

   Follow the procedures in *Cisco ACI and Kubernetes Integration* on Cisco.com until the end of the section, *Preparing the Kubernetes Nodes*. Use the flavor `RKE-1.4.9`. Perform the steps for the cluster and every node in the cluster.

Note ACI CNI in nested mode is only supported with VMM-integrated VMware (with Distributed Virtual Switch).

4. Prepare the nodes for the Rancher server nodes.

   Fulfill the hardware, software, and networking requirements for the nodes where you will install the Rancher server. See the page *Installation Requirements* for Rancher 2.x on the Rancher website.

5. Prepare the nodes in the user cluster.

   Fulfill the requirements for the nodes where you will install apps and services. See the page *Node Requirements for User Clusters* or Rancher 2.x on the Rancher website.

Integrating Rancher

We recommended that you install Rancher server in a separate cluster, which is called a Rancher server cluster. This cluster creates and manages other clusters, which are called user clusters.

You typically install the Rancher server cluster using Rancher Kubernetes Engine (RKE). The integration with Cisco ACI supports creating a custom cluster on an on-premises deployment and creating one cluster per tenant.

Note The Cisco Application Centric Infrastructure (ACI) Container Network Interface (CNI) does not support Rancher-launched Kubernetes. However, RKE-installed Kubernetes is supported, and RKE-installed clusters can be imported into Rancher. To see the differences, see *Setting up Kubernetes Clusters in Rancher* on the Rancher website.

This section includes prerequisites and instructions for integrating Rancher.

Install Kubernetes Using RKE

Before you begin

Fulfill the requirements in the section Prerequisites for Integrating Rancher, on page 3.

Procedure

Step 1 Install the RKE binary file on the tools server.

Follow the procedure "Download the RKE Binary" on the *RKE Kubernetes Installation* page on the Rancher.com website.

Step 2 Install Docker on all the cluster nodes.

Follow the guidelines and procedure in "Requirements" in the "RKE" section of the Rancher.com website.

Step 3 Run `acc_provision` to generate the network provider section of the `cluster.yml` file that the RKE requires.

See the section "Provisioning Cisco ACI to Work with Kubernetes" in *Cisco ACI and Kubernetes Integration* on Cisco.com.

Example:

```
acc-provision -f RKE-1.2.3 --sample input-file
```

Step 4 Edit the input file that was generated by the previous step to match your cluster:

Example:

```
acc-provision -f RKE-1.4.9 --sample input-file
acc-provision -f RKE-1.4.9 -c input-file -u apic-user-name -p apic-password -o
network_provider_cluster.yml
```

The command generates the network-provider file that is used in step 6.

Step 5 Use RKE to generate a sample `cluster.yml` file and edit the file to add the cluster nodes.

Alternatively, you can use an existing `cluster.yml` file.

Step 6 Replace the network provider section in the `cluster.yml` file that you generated with the **acc_provision** command.

The following text provides an example of a `network-provider` file:

Example:

```
network:
 plugin: "aci"
 aci_network_provider:
    system_id: "rancher"
    apic_hosts: ["\"172.28.184.110\""]
    token: "2f8cc816-21e6-49d3-b18e-00d1bf38f37d"
    apic_user_name: "rancher"
    apic_user_key: "(user-key as generated by acc-provision)"
    apic_user_crt: "(user-crt as generated by acc-provision)" encap_type: "vxlan"
    mcast_range_start: "225.10.1.1"
    mcast_range_end: "225.10.255.255"
    aep: "sauto-aep"
    vrf_name: "sauto_l3out-1_vrf" vrf_tenant: "common"
    l3out: "sauto_l3out-1"
    node_subnet: "10.100.1.1/24"
    l3out_external_networks: ["\"sauto_l3out-1_epg\""]
    extern_dynamic: "10.3.0.1/24"
    extern_static: "10.4.0.1/24"
    node_svc_subnet: "10.5.0.1/24"
    kube_api_vlan: "213"
    service_vlan: "214"
    infra_vlan: "4081"
    service_monitor_interval: "5"
    ovs_memory_request: "256Mi"
    aci_multipod: "true"
    aci_multipod_ubuntu: "true"
    dhcp_renew_max_retry_count: "10"
    dhcp_delay: "10"
    opflex_agent_policy_retry_delay_timer: "60"
    aci_containers_controller_memory_limit: "5Gi"
    aci_containers_controller_memory_request: "256Mi"
    aci_containers_host_memory_limit: "5Gi"
    aci_containers_host_memory_request: "256Mi"
    mcast_daemon_memory_limit: "5Gi"
    mcast_daemon_memory_request: "256Mi"
    opflex_agent_memory_limit: "5Gi"
    opflex_agent_memory_request: "256Mi"
    aci_containers_memory_limit: "20Gi"
    aci_containers_memory_request: "256Mi"
    opflex_device_reconnect_wait_timeout: "15"
services:
```

```
  kube-controller:
    cluster_cidr: "10.2.0.1/16"
system_images:
 aci_cni_deploy_container: noiro/cnideploy:6.0.3.1.81c2369
 aci_host_container: noiro/aci-containers-host:6.0.3.1.81c2369
 aci_opflex_container: noiro/opflex:6.0.3.1.81c2369
 aci_mcast_container: noiro/opflex:6.0.3.1.81c2369
 aci_ovs_container: noiro/openvswitch:6.0.3.1.81c2369
 aci_controller_container: noiro/aci-containers-controller:6.0.3.1.81c2369
```

The configuration parameters are the same as in the `acc_provision` input file. The pod subnet is a main configuration option, called cluster_cidr, in the `cluster.yml` file and is generated under the kube controller section by acc-provision.

The installation defaults to 6.0.3.1.81c2369 image tag for the ACI-CNI images for the supported kubernetes versions. You can pick up any other tag by specifying the tags using the version parameters under the registry section of the acc_provision input file and adding the generated system_images configuration to the `cluster.yaml` file:

Note The use_digest parameter allows images to be specified using their digest values. If this is set, it requires the digest value of each image to be specified using the version parameters under the registry section. You can copy the generated system_images configuration to the `cluster.yaml` file.

Example:

```
system_images:
  aci_cni_deploy_container:
noiro/cnideploy@sha256:96f1df66843660905fa2cb07b058d8ecc5cb956c4799d661459cc0bfdfd291d2
  aci_host_container:
noiro/aci-containers-host@sha256:d37de5ac9093dff471c0602a79064a7cbac85f6513785ed86eb037ef8740ceed
  aci_opflex_container:
noiro/opflex@sha256:6ae620eb8ba66a627a9c96b0e34b5d31b05aa22c9196b1885362c6273b0b76e4
  aci_mcast_container:
noiro/opflex@sha256:6ae620eb8ba66a627a9c96b0e34b5d31b05aa22c9196b1885362c6273b0b76e4
  aci_ovs_container:
noiro/openvswitch@sha256:71d04aa713ff90ce26382bc234941beff9e51a365b3f85c76666a524b7384766
  aci_controller_container:
noiro/aci-containers-controller@sha256:375c61f113c207b152f70b4c1abc8390f23dedb73245e3d67b99c6a00dbd6bca
```

Note The `aci_gbp_server_container` and `aci_opflex_server_container` images are not required for on-premises deployment, which is the only profile that this release supports.

Step 7 Enter the following command: **`rke up`**.

The command internally generates the Cisco ACI Container Network Interface (CNI) `.yaml` file and applies it. The resulting `.yml` file is stored as `configmap` in `kube-system/rke-network-plugin`.

What to do next

User can import the RKE cluster into the Rancher server. Follow the procedure *Importing Existing Clusters into Rancher* for Rancher 2.x on the Rancher website.

Upgrade Kubernetes Using RKE

You can use Rancher Kubernetes Engine (RKE) to upgrade Kubernetes.

Before you begin

- Ensure that the version of Kubernetes that you want to upgrade to is supported for your environment. See the *Cisco ACI Virtualization Compatibility Matrix* and *All Supported Versions* on the Rancher website.
- Take a backup of `cluster.rkestate` and `cluster.yml` and a snapshot of `etcd`.

  Follow the instructions in *One-time Snapshots* for RKE on the Rancher website.

Note If upgrade fails for any reason, you can use the snapshot and backed-up files to roll back to the previous version of Kubernetes. See *Restoring from Backup* for RKE on the Rancher website.

Procedure

Step 1 Update the `cluster.yml` file with the desired version of Kubernetes by revising the line `kubernetes_version: (rancher-image-version)` as needed.

Step 2 Enter the following command: **`rke up`**.

The command internally generates the Cisco ACI Container Network Interface (CNI) `.yaml` file and applies it. The resulting `.yml` file is stored as `configmap` in `kube-system/rke-network-plugin`.

Related Documentation

In addition to the Rancher integration information in this document, you may want to consult the following documentation:

- **Rancher**

  The Rancher.com website provides documentation, videos, infographics, and white papers, including *Rancher: Technical Architecture.*

- **Kubernetes**

  The Kubernetes.io website provides conceptual information, tutorials, task-based information, and reference material.

- **Cisco ACI CNI**

  See the following on Cisco.com:

  - *Cisco ACI and Kubernetes Integration*
  - *Cisco ACI CNI plug-in for Red Hat OpenShift Container Platform Architecture and Design Guide*

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA 95134-1706
USA

Asia Pacific Headquarters
CiscoSystems(USA)Pte.Ltd.
Singapore

Europe Headquarters
CiscoSystemsInternationalBV
Amsterdam,TheNetherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.