Cisco APIC Container Networking Release Notes, ACI Release 6.1(1)

Introduction

This document describes the features, bugs, and limitations for the Cisco Network Operator (CNO) 1.2 corresponding to ACI release 6.1(1). CNO stitches together Cisco network fabrics, and standard host-based networking technologies to provide differentiated and performant network services to Kubernetes, Red Hat OpenShift, and Rancher RKE clusters on a Cisco ACI fabric. It allows the cluster pods to be treated as fabric end points in the fabric integrated overlay. It also provides the option of IP Address Management (IPAM), security policy enforcement, load balancing services and other network infrastructure services to ensure a frictionless deployment for container workloads.

Release Notes are sometimes updated with new information about restrictions and bugs. See the following website for the most recent version of this document:

https://www.cisco.com/c/en/us/support/cloud-systems-management/application-policy-infrastructure-controller-apic/tsd-products-support-series-home.html

For more information about this product, see " Related Content."

Date	Description
August 28, 2024	Release 6.1(1) became available.

New Software Features

Telco & CNF workloads	
Feature	**Description**
CNF L3 Model	Cisco ACI supports associating particular vlans to map to an ACI L3Out SVI, either pre-existing or generated, so that pods having interfaces in a NAD which uses the given encap, can peer with ACI on the additonal network to redistribute container routes to external routers. Both types of SVI – conventional and floating, are supported.
Webhook based peering environment variables	Annotated pods that come up in additional networks that have been mapped to ACI using CNO L3 Model, can have BGP peering environment like ACI leaf peer addresses, ASN and secret injected into a named container ending with a configurable suffix. This container hosting a router CNF can use these variables to peer with ACI.

Enterprise workloads (with Cisco ACI CNI 6.1.1.1)	
Feature	**Description**
Support for Kubernetes 1.30	Cisco ACI supports Kubernetes 1.30 using the Cisco ACI Container Network Interface (CNI) plug-in and is installed with kubeadm on Ubuntu 22.04 using CRI-O.
OpenShift 4.15 on OpenStack 16.2	Cisco ACI supports Red Hat OpenShift 4.15 nested in Red Hat (OSP) 16.2. To enable this support, Cisco ACI provides customized Ansible modules to complement the upstream OpenShift installer.
OpenShift 4.15 on Bare Metal	Cisco ACI supports Red Hat OpenShift 4.15 on Bare metal with User Provisioned Infrastructure (UPI) method of installation. Cisco ACI provides customized Python

Enterprise workloads (with Cisco ACI CNI 6.1.1.1)	
Feature	**Description**
	script to complement the upstream OpenShift installer for integration with the ACI CNI.
OpenShift 4.15 on VMware vSphere	Cisco ACI supports Red Hat OpenShift 4.15 nested in VMware vSphere 7. Cisco ACI provides customized Ansible modules as reference to complement the upstream OpenShift installer for integration with the Cisco ACI CNI.
OpenShift Service Mesh on OCP 4.15	Cisco ACI supports Red Hat OpenShift Service Mesh on OpenShift 4.15.
OpenShift Virtualization on OCP 4.15	Cisco ACI supports Red Hat OpenShift Virtualization on OpenShift 4.15.
Host Protection Policy (HPP) distrbution optimization	Cisco ACI CNI supports distribution of HPP via Kubernetes control plane for faster convergence and to reduce load on fabric.
Datapath Tracing	Cisco ACI CNI now supports tracing of Open vSwitch flows using the `acikubectl trace` tool.

The following features are being released as a *technology preview* to gather feedback; it is not recommended to use them directly in production.

Feature	Description
Cilium Support	Cisco ACI supports running Cilium CNI with ACI CNI to get the best of both. Cilium handles Network Policy enforcement on the node and ACI CNI continues to provide all other fabric-specific features including policy-based segmentaion, hardware loadbalancing, SNAT, and more. For more details, refer to the ACI and Cilium Integration document.
Rancher Kubernetes Engine (RKE) 1.5.14 and 1.6.3	Cisco ACI supports an RKE 1.5.14 and 1.6.3 installed cluster integrated with Cisco ACI CNI.

Cisco ACI Virtualization Compatibility Matrix

For information about all Cisco ACI-supported (currently-supported) Container products along with the supported ecosystem releases (Kubernetes, OpenShift, OpenStack, Rancher, and vSphere), see the Cisco ACI Virtualization Compatibility Matrix at the following URL:

https://www.cisco.com/c/dam/en/us/td/docs/Website/datacenter/aci/virtualization/matrix/virtmatrix.html

Software

Installers:

OpenShift installer source scripts for OCP 4.13, 4.14, and 4.15 on OpenStack 16.2 are provided as releases artifacts.

For OpenShift on Baremetal, OpenStack and vSphere please refer to the Install Guides.

Configuration:

This release uses the acc-provision upstream open source project release 6.1.1.1 for ACI-CNI, Calico, OVN, OVS, MACVLAN, or SR-IOV CNI support and Cisco ACI fabric configuration. Information about acc-provision PyPI package can be found in the upstream release status page.

Installation (needs Python 3.9 or later):

```
pip install acc-provision==6.1.1.1
```

Usage:

For details on how to use acc-provision refer to:

Provisioning Cisco ACI to Work with Kubernetes

Generating an Updated Cisco ACI CNI Configuration

Container Images:

acc-provision generates deployment manifests that reference ACI-CNI container images built in upstream open source projects aci-containers, OpFlex, and acc-provision-operator with tag 6.1.1.1. Package and vulnerability details of these images are documented in the upstream release status page.

Supported Scale

For the verified scalability limits (except for CLI limits), see the Verified Scalability Guide for this release. For Kubernetes-based integrations (including Docker, OpenShift, and Rancher) and OpenStack platform scale limits, see the following table.

Note: The scalability information in the following table applies to Kubernetes or OpenStack resources integrated with OpFlex into the Cisco ACI fabric. It does not apply to Microsoft SCVMM hosts or Cisco ACI Virtual Edge instances.

Limit Type	Maximum Supported
Number of OpFlex hosts per leaf[1]	120
Number of OpFlex hosts per port	20
Number of vPC links per leaf	40
Number of endpoints per leaf	10,000
Number of endpoints per host	400
Number of virtual endpoints per leaf	40,000

[1]- The indicated scale value is for Cisco ACI release 5.0(1) and later. If the Cisco ACI release is earlier than 5.0(1), the number of supported OpFlex hosts are 40.

Notes:

- For containers, an endpoint corresponds to a pod's network interface. The number of pods that can be run on each node is however constrained by other system configuration and Kubernetes distribution specified limits. For kubeadm installed upstream Kubernetes its 110 pods per node, and for OpenShift its 250 pods per node.

- For OpFlex hosts per port — a port is either a physical port or a vPC. One vPC equals one port.The number of member ports in a vPC is inconsequential.
- For the CLI verified scalability limits, see the *Cisco NX-OS Style Command-Line Interface Configuration Guide* for this release.

Known Limitations

- A pod selector has to be always provided in a network policy to map a port name to the port number, and an empty pod selector is not supported in the ingress direction.
- The Cisco ACI CNI Plug-in is not integrated with the Multi-Site Orchestrator. When deploying to a Multi-Site deployment, the Cisco ACI configurations implemented by the plug-in must not be affected by the Multi-Site Orchestrator.
- SNAT policy configuration is not applicable to traffic within the same cluster.
- An SNAT policy which goes into the *Failed* state (for example, on account of reusing an already used SNAT IP), cannot be updated or reused. A failed SNAT policy needs to be deleted and a new one created.
- The SNAT feature was not supported with Cisco APIC release 6.0(3). The support is added from Cisco API release 6.0(4a). Users who have deployed the SNAT feature with the Cisco APIC 5.2 releases should not upgrade to Cisco APIC 6.0(3).
- Due to Python 3 dependencies, the acc-provision tool is supported on RHEL8 operating system.
- The NodePort service statistics exported to Prometheus get accounted under ClusterIp service statistics in on-premises deployments.
- The file openvswitch/db.sock sometimes becomes a directory after node reload due to a race-condition between the openvswitch installed on the node and openvswitch installed by ACI-CNI. The work around is to delete the /var/run/openvswitch/db.sock directory, and restart the aci-containers-openvswitch pod. For more details, see *Red Hat Case 03299085.*
- If the cluster is downgraded from ACI CNI release 6.0.3.2 or higher version to any of the lower versions, there is a chance that opflex-agent container of host-agent pod will restart, and thus the host-agent pod remains in crashloopbackoff. You can resolve this by manually deleting the file /var/lib/opflex-agent-ovs/reboot-conf.d/reset.conf from the node where the hostagent pod in crashloopbackoff is scheduled to.
- After the HPP distribution optimization feature is disabled the old hostprotPols and hostprotRemoteIpContainers CRs will remain, this will not impact the functionality. After the feature is enabled again, the stale CRs will be cleaned, if any.
- When you create or update a namespace after a network policy with egress namespace selectors has already been set up, the new or updated namespace is not selected by the policy, even if it matches the selector criteria. For instance, if you have a network policy that uses namespace selectors for egress and initially no namespaces have the matching labels, adding those labels later to a namespace will not make the egress rules apply to it. The workaround is to delete and recreate the network policy, so that the updated namespace is recognized.

Usage Guidelines

- To upgrade a Cisco ACI CNI cluster, use acc-provision with the " --upgrade" option.
- In ACI 6.0(x), a new attribute, `serviceBdRoutingDisable`, has been introduced to service BD which should be set to *yes* for the SNAT to work. If the ACI CNI version is less than 6.0.4.2, and APIC is upgraded from 5.2(x) to 6.0(x), then, SNAT might not work till the controller pod is restarted as the `serviceBdRoutingDisable` attribute of the service BD is still on *no*. So, it is recommended to upgrade the ACI CNI version to 6.0.4.2 or higher before upgrading APIC from from 5.2(x) to 6.0(x).
- istio_config configuration is not supported.
- To avoid scheduling of pods before the host agent is running on a node we can use the following configuration:

```
kube_config:
        taint_not_ready_node: True
```

  This will make node unschedulable by adding a taint if it's not in Ready state and taint will be removed when the node becomes Ready and host-agent initialization is complete on that node. Below taint will be added to facilitate this:

```
aci-containers-host/unavailable:NoSchedule
```

- Optimizations to mapping of Kubernetes Network Policy to ACI Host Protection Policies can be turned ON with the following configuration:

```
kube_config:
…
    hpp_optimization: True
```

  This, and all other configuration changes should be performed using the acc-provision tool, and will take effect after the new manifests generated by acc-provision are applied. This configuration may be enabled by default in future releases.

- Distribution of ACI Host Protection Policies via Kubernetes control plane can be turned on with the following configuration:

```
kube_config:
...
  enable_hpp_direct: True
```

  By default, this feature is disabled. When enabled using the above configuration, instead of using the fabric to distribute, the host protection policies which are mapped to kubernetes network policies, the Kubernetes plane itself is used to provide the functionality of Kubernetes network policies.

- The forwarding behavior for unknown-unicast traffic within service Bridge Domain (BD) used in Kubernetes Loadbalancer-type services and SNAT feature is set to *proxy* mode by default to avoid flooding. It can be reverted to the earlier *flood* mode by setting the following configuration:

```
kube_config:
    ….
```

```
unknown_mac_unicast_action: "flood"
```

- The host protection policy functionality can be disabled, effectively disabling the Kubernetes network policy functionality, using the configuration listed below:

```
kube_config:
…
  disable_hpp_rendering: True
```

Default value is False.

- For running more than 250 pods per node, the following configuration needs to be added:

```
kube_config:
 …
      opflex_agent_ovs_asyncjson_enabled: "true"
```

This is a preview feature. Note the configuration value is a string in quotation marks.

- The aci-containers-operator uses the Ansible Operator SDK. If another Kubernetes Operator which uses the Ansible Operator SDK is deployed on the same node, the health-check ports of the two Operators will conflict. There is currently no way to override these default ports either. To overcome this issue, the aci-containers-operator pod has node affinity rule for " preferredDuringSchedulingIgnoredDuringExecutionfor" with key “preferred-node” and value “aci-containers-operator-2577247291”. You can ensure that the aci-containers-operator is scheduled on a particular node by adding the following label to the node:

```
preferred-node=aci-containers-operator-2577247291
```

A similar affinity scheme should be applied to other conflicting pods to ensure that they do not get scheduled on the above node.

Note that if no node with the above label exists, then, the aci-containers-operator will still get scheduled on some node.

- The size of each log file collected in the cluster report can be optionally set using the following acc-provision input configuration (default is 10 MB):

```
logging:
      size <size-in-bytes>
```

Note that the truncation happens at the beginning and the latest content of the log file is collected.

- The aci-containers-operator pod logging level can be set to debug by adding the following configuration:

```
logging:
                operator_log_level: debug
```

- The nodes can be excluded from the SNAT redirect policy by providing the following configuration in the acc-provision input configuration file, the nodes which match all the labels in any group will be excluded from the SNAT redirect policy:

```
kube_config:
      node_snat_redirect_exclude:
      - group: <node_group_name_1>:
        labels:
        - <label_1>
```

```
    - <label_2>
    - <label_n>
  - group: <node_group_name_n>:
    labels:
    - <label_1>
    - <label_2>
    - <label_n>
```

Example:

```
kube_config:
    node_snat_redirect_exclude:
    - group: router
      labels:
      - router
      - infra
    - group: infra
      labels:
      - control-plane
      - master
```

The nodes having labels [router and infra] or [control-plane and master] will be excluded from the SNAT redirect policy, i.e if all the labels in any of the mentioned groups in configuration is matched with the node labels then that node will be excluded.

- When the controller starts and is unable to connect to the first APIC in the apic_hosts list, it will switch to the next APIC in the list after retrying the number of times specified in the configuration shown below:

```
kube_config:
    apic_connection_retry_limit: <retry_count>
```

The default is 5 retries.

- The OpFlex Agent reconnect after VM Migration can be enabled by providing the following configuration in the acc-provision input configuration file:

```
kube_config:
  enable_opflex_agent_reconnect: True
```

- Set tolerations seconds for aci controller pod by providing the following configuration in the kube_config section of acc-provision input file:

```
toleration_seconds: 100
```

default value is 60 and 600 for non rke and rke flavors respectively.

- Sometimes it takes longer for service endpoints to be ready but since they are configured are successfully configured as endpoints of that service, traffic will start get loadbalanced to these endpoints and may get temporarily blackholed. To avoid this, a delay along with the details of the services of type Loadbalancer can be specified in the acc-provision input file, such that the Cisco

ACI service graph will be programmed with a delay. The following example shows a delay of 30 seconds being introduced for ingress-service (belonging to openshift-ingress) and a delay pf 60 seconds for monitoring-service (belonging to openshift-monitoring):

```
kube_config:
…
       service_graph_endpoint_add_delay:
        delay: 30
        services:
        - name: ingress-service
          namespace: openshift-ingress
        - name: monitoring-service
          namespace: openshift-monitoring
          delay:60           #override delay of 30
```

Note that endpoints are added to the service graph only after the pod goes into Ready state.

- To enable drop logging, perform the following configuration in the acc-provision input file:

```
drop_log_config:
   enable: True
```

For more information, see Enabling the OpFlex Drop Log Feature.

To disable recording packet drop event under the pod add following configuration in the acc provision input file:

```
drop_log_config:
      disable_events: True
```

- The scope of the SNAT service graph contract can be configured by the user in the acc-provision input file as follows:

```
kube_config:
  snat_operator:
    contract_scope: <scope name>
```

Valid values (as allowed by Cisco APIC) are "global", "tenant" and "context". The default is set to "global".

- The subnets listed under extern_static and extern_dynamic can be automatically added to rdconfig usersubnets by setting the following configuration in acc-provision input file:

```
kube_config:
  add_external_subnets_to_rdconfig: True
```

Note that if the initial value of add_external_subnets_to_rdconfig is *true* but later modified to *false*, the usersubnets automatically will not be removed and the rdconfig CR will have to be updated manually to remove them. Each entry in the rdconfig results in a new OVS flow regardless of whether the subnets overlap or not.

- The aci-containers-controller pod subscribes for notifications on certain objects to the Cisco APIC. There is a timeout associated with this subscription. A shorter timeout requires more frequent

subscription renewals. The timeout is set to 900 seconds, and can be changed by configuring the acc-provision input file:

```
aci_config:
  apic_refreshtime: 1200
```

Note: The subscription timeout is configurable only in Cisco APIC 4.x or later.

- To ensure that the subscription renewal happens in time before the subscription timeout expires on the APIC side, the aci-containers-controller pod starts the renewal process a little earlier. By default, it starts 150 seconds before the subscription expiry. If the system is heavily loaded and you notice subscriptions are not renewed in time (this requires examining the aci-containers-controller and Nginx APIC logs), this period can be altered by adjusting the following configuration in the acc-provision input file:

```
aci_config:
  apic_refreshticker_adjust: 150
```

- The memory request and limit for the Open vSwitch container is set to 128Mi and 1Gi respectively. It can be changed by configuring the acc-provision input file as follows:

```
kube_config:
  ovs_memory_limit: "5Gi"
  ovs_memory_request: "512Mi"
```

- The Multus CNI deployment can be enabled in the OpenShift installation by performing the following configuration in the acc-provision input file:

```
multus:
  disable: False
```

- The memory request and limit for the Open vSwitch container is set to 128Mi and 1Gi respectively. It can be changed by configuring the acc-provision input file as follows:

```
kube_config:
  ovs_memory_limit: "5Gi"
  ovs_memory_request: "512Mi"
```

- Default memory request and limit for aci-containers-system namespace pods is set to 128Mi and 3Gi respectively and can be changed by configuring the acc-provision input file as follows.

```
kube_config:
  aci_containers_memory_request: "512Mi"
  aci_containers_memory_limit: "5Gi"
```

Note: This namespace wide memory resource setting is not applied to openvswitch container. As stated earlier, please use ovs_memory_request, ovs_memory_limit to change openvswitch container memory request and limit.

Apart from the above option to set namespace wide memory request and limit, you can choose to configure container specific memory request and limit through acc-provision input file as follows:

```
kube_config:
  aci_containers_controller_memory_request: "256Mi"
```

```
aci_containers_controller_memory_limit: "5Gi"
aci_containers_host_memory_request: "256Mi"
aci_containers_host_memory_limit: "5Gi"
mcast_daemon_memory_request: "256Mi"
mcast_daemon_memory_limit: "5Gi"
opflex_agent_memory_request: "256Mi"
opflex_agent_memory_request: "5Gi"
acc_provision_operator_memory_request: "256Mi"
acc_provision_operator_memory_limit: "5Gi"
aci_containers_operator_memory_request: "256Mi"
aci_containers_operator_memory_limit: "5Gi"
```

This container specific configuration takes priority over the namespace wide configuration.

- ACI CNI pods are cirtical and to mark them node critical, PriorityClass " system-node-critical" can be set by configuring the acc-provision input file as follows:

```
kube_config:
  use_system_node_priority_class: True
```

- Running Rancher Kubernetes clusters with ACI CNI on VMware ESX-based virtual machines is supported in a Cisco ACI Multi-Pod environment. The following configuration should be provided in the acc-provision input file to enable this feature:

```
kube_config:
  aci_multipod: True
```

For clusters running on the Ubuntu host operating system, the following additional configuration should be set in the acc-provision input file:

```
kube_config:
  aci_multipod_ubuntu: True
```

- If moving an ESX-based VM across Cisco ACI pods, the DHCP needs to be renewed on the infra-vlan after the VM move. ACI-CNI's host-agent initiates this DHCP release and request. To reliably obtain the DHCP, sometimes multiple attempts may be required. By default, this is attempted five times each after a delay of five seconds. These defaults can be tuned using the following configuration in the acc-provision input file:

```
kube_config:
  dhcp_renew_max_retry_count: 4
  dhcp_delay: 6
```

To avoid redundant DHCP renews on opflex-agent disconnects when not performing VM migration, a timeout interval in seconds can be configured. The host-agent waits for this interval before triggerring a

DHCP release and renew. If the OpFlex agent reconnects before this interval expires, no DHCP renewal is initiated.

This timeout interval can be configured as follows:

```
kube_config:
  opflex_device_reconnect_wait_timeout: 10
```

- Dual-stack can be configured by providing the IPv4 and IPv6 in the acc-provision input configuration file as follows:

```
net_config:
  node_subnet:
    - <ipv4-subnet>
    - <ipv6-subnet>
  pod_subnet:
    - <ipv4-subnet>
    - <ipv6-subnet>
  extern_dynamic:
    - <ipv4-subnet>
    - <ipv6-subnet>
  extern_static:
    - <ipv4-subnet>
    - <ipv6-subnet >
```

Note that the dual-stack feature is only supported for OpenShift on Bare Metal.

- Static loadbalancer IP can be provided to the loadbalancer service via annotation in the below format:

```
"opflex.cisco.com/lb-ipam-ips": "<ipv4>,<ipv6>"
```

- In cases of heavy load, the opflex-agent requests to the leaf switch may fail and the opflex-agent needs to retry after a randomized backoff. The upper bound on this backoff can be configured to adapt specific load conditions to avoid frequent retries:

```
kube_config:
  opflex_agent_policy_retry_delay_timer: 60  # default is 10 seconds
```

- Control opflex agent statistics by adding following configuration in the acc provision input file:

```
kube_config:
   opflex_agent_statistics: False # default is True
```

Set this to `False` if statistics load is very high.

- The opflex agent builds with the latest version of openssl (3.x) which causes compatibility issues when FIPS is enabled on the host running opflex-agent. This happens because the leaf filters out the ciphers needed for openssl 3.x to work properly. Until this is fixed on the leaf side the workaround is to run opflex-agent in openssl 1.1 compatibility mode so the connectivity to leaf is not broken when FIPS is enabled on the host running opflex-agent. This can be done by adding following configuration in acc provision input file:

```
kube_config:
```

```
opflex_openssl_compat: True # default is False
```

- The ingress contract created by ACI CNI plugin every time a service is exposed outside as type LoadBalancer needs to be consumed under aci-containers-default EPG to allow PBR redirect for E/W communication towards external IP. The contract assignmet was done manually earlier but now can be automated by adding following configuration in acc provision input file:

```
kube_config:
   add_external_contract_to_default_epg: True # default is False
```

Prerequisites:

1. Flag `add_external_subnets_to_rdconfig: true` should be cofigured in acc provision input file:

```
kube_config:
   add_external_subnets_to_rdconfig: True
```

2. SnatPolicy should be created.

- For the VMware VDS integration, you can refer to the Enhanced Link Aggregation Group (eLAG) configured through the Cisco APIC by using the following configuration in the acc-provision input file:

```
nested_inside:
 type: vmware
…
 elag_name: <eLAG-name-used>
```

- User update to include a firewall node in the loadbalancer service graph template are now supported. The following annotation should be added to the corresponding Kubernetes loadbalancer type service:

```
opflex.cisco.com/service-graph-name: <some-value>
```

- Policy Based Routing (PBR) tracking can be enabled for the Cisco APIC service graph created for supporting the SNAT feature. More details on PBR tracking can be found in the chapter " Configuring Policy-Based Redirect" in the Cisco APIC Layer 4 to Layer 7 Services Deployment Guide.

  One HealthGroup for each node is created, and it is associated with the redirect policy of the SNAT service graph with the internet protocol service level agreement (IP SLA) interval set to 5 seconds. This interval is configurable through the acc- provision input file:

```
net_config:
  service_monitor_interval: 10
```

  If the service_monitor_interval is set to zero, PBR tracking is disabled.

  PBR tracking can be also be enabled for other Cisco APIC service graphs created for each Kubernetes external service, setting the following configuration in the acc-provision input file:

```
net_config:
  pbr_tracking_non_snat: true
```

If enabled, the service_monitoring_interval described earlier applies here as well.

Note: In a Cisco ACI CNI-based cluster, the same worker node is used to provide both the external Layer 4 load balancer and SNAT services. So if PBR tracking is enabled, and if the worker node reports unhealthy status for SNAT, a fault appears in the redirect policies associated with all other (non-SNAT) service graphs that have this node. However, this fault does not actually affect those other services and traffic from those services is still distributed to that node. The fault manifests for those other services only in the Cisco APIC GUI.

- OpenShift's default OVN-Kubernetes CNI can be used as the primary CNI for the primary interface using acc-provision flavor `openshift-sdn-ovn-baremetal`. The following input configuration is required for acc-provision to be able to configure the node network:

```
aci_config:
  system_id: mykube             # Every cluster must have a distinct ID
  apic_hosts:                   # List of APIC hosts to connect for APIC API
  - 10.1.1.101
  physical_domain:              # Non mandatory field
    domain: kube-physdom        # If physical domain provided, then mention
name. Otherwise one will be created with name <system_id>-physdom
  aep: kube-cluster             # The AEP for ports/VPCs used by this cluster
  vrf:                          # This VRF used to create the BDs
    name: mykube-vrf
tenant: common              # This can be system-id or common
net_config:
  node_subnet: 10.1.0.1/16      # Subnet to use for nodes
  kubeapi_vlan: 4001            # The VLAN used by the physdom for nodes
```

  - Creation of additional networks is supported when using OpenShift's default OVN-Kubernetes CNI as the primary CNI for the primary interface and the acc-provision flavor `openshift-sdn-ovn-baremetal`.
  - A new AEP is required for the additional network and can be configured as follows:

```
aci_config:
       secondary_aep: kube-cluster-2 # The AEP for additional networks
```

  - Cisco CNI can be automatically chained at the end of every Network-Attachment-Definition(NAD) or only those NADs with the annotation netop-cni.cisco.com/auto-chain-cni:" true" by enabling the following options. A webhook server will be deployed as a result. This server can use the cert-manager provided by the platform (eg RedHat OCP) or Cisco CNI can deploy a local cert-manager instance if the platform does not have one, which has an option below:

```
chained_cni_config:
  require_annotation_for_nad_mutation_webhook: True
  local_cert_manager_enabled: True #Default False
```

    These additional networks are isolated using VLAN configuration on the Cisco ACI fabric.Following CNIs are supported as the primary CNI on secondary networks: MACVLAN, SRIOV, IPVLAN, Linux-bridge and OVS(OpenVswitch). Where both L2 and L3 modes are possible with a CNI – as in bridge and IPVLAN, only L2 mode is currently supported.

OVS support requires the configuration as follows: .

```
chained_cni_config:
   enable_ovs_cni_support: True
```

Furthermore OpenVswitch needs to be installed on all the nodes, OVS-CNI plugin must be copied over to the local CNI binary directory, required OVS bridge should be configured with the uplink port added to the bridge. Openvswitch support enablement will change the image tags to ones with the prefix –ovscni.

- NetworkFabricConfiguration CR supports renaming auto-created epg/bd,adding subnets, changing vrf, tenant and associating contracts with the epgs. An example is shown below:

```
apiVersion: aci.fabricattachment/v1
kind: NetworkFabricConfiguration
      metadata:
         name: networkfabricconfiguration
         namespace: aci-containers-system
      spec:
        nadVlanRefs:
          vlans:
          - vlans: "101"  # Match EPG based on explicit VLAN ID
             aeps:
             - "router-aaep"
             epg:
             name: custom-epg1
               bd:
                 name: custom-bd1
                 common-tenant: false
                 subnets:
                 - "10.2.3.0/24"
                 vrf:
                   name: common-vrf1
                   common-tenant: true
               contracts:
                 consumer:
                 - ctrct1
                 provider:
                 - ctrct2
```

Automating the configuration of these VLANs is achieved by leveraging a Chained CNI configuration using the following configuration:

```
chained_cni_config:
  secondary_interface_chaining: True
  secondary_vlans:              # VLANs for additional networks
  - 402
  - 403-406
```

```
        - 410, 425
```

- With the usage of NetworkFabricL3Configuration CR, user can map the uplink vlan used by an additional network onto a Cisco ACI L3Out SVI. Both conventional and floating L3Out SVI are supported. The L3out chosen can be pre-existing or be generated. An example of a configuration for mapping a vlan 102 to a pre-existing floating L3Out SVI is shown below. Status in the CR reflects the configuration effectively applied to ACI. These fields can be copied into the spec and changed if needed. Refer the opensource CNO document for more details.

```
apiVersion: aci.fabricattachment/v1
           kind: NetworkFabricL3Configuration
           metadata:
             name: networkfabricl3configuration
           spec:
             vrfs:
             - directlyConnectedNetworks:
               - bgpPeerPolicy:
                   enabled: true
                   peerASN: 64514
                 encap: 102
                 l3OutName: pre-exist-l3out
                 l3OutOnCommonTenant: true
                 primarySubnet: 192.168.100.0/24
                 sviType: floating_svi
                 useExistingL3Out: true
               vrf:
                 common-tenant: true
                 name: pre-exist-vrf
         status:
           vrfs:
           - directlyConnectedNetworks:
             - bgpPeerPolicy:
                 enabled: true
                 peerASN: 64514
                 secret:
                   name: ""
                   namespace: ""
               encap: 102
               l3OutName: pre-exist-l3out
               l3OutOnCommonTenant: true
               nodes:
               - nodeRef:
                   nodeId: 101
                   podId: 1
```

```
      primaryAddress: 192.168.100.247/24
    - nodeRef:
        nodeId: 102
        podId: 1
      primaryAddress: 192.168.100.248/24
    primarySubnet: 192.168.100.0/24
    subnets:
    - connectedSubnet: 192.168.100.0/24
      floatingAddress: 192.168.100.254/24
      secondaryAddress: 192.168.100.253/24
    sviType: floating_svi
    useExistingL3Out: true
  tenants:
  - commonTenant: true
    l3OutInstances:
    - name: pre-exist-l3out
      podRef:
        podId: 1
      rtrNodes:
      - nodeRef:
          nodeId: 101
          podId: 1
        rtrId: 101.101.0.101
      - nodeRef:
          nodeId: 102
          podId: 1
        rtrId: 102.102.0.102
  vrf:
    common-tenant: true
    name: pre-exist-vrf
```

- CNO's webhook will insert the following BGP peering environment variables into a router pod's container, which is named with the suffix fabric-peer when the annotation `netop-cni.cisco.com/fabric-l3peer-inject: <list of comma-separated-NAD-names-for-which-insertion-is-needed>` is present on the pod.

```
BGP_PEERING_ENDPOINTS_<NAD_NAME>=peer-address-list
eg:192.168.120.244/24,192.168.120.243/24
BGP_ASN_<NAD_NAME>=ASN-to-be-used-by-container eg:64515
BGP_SECRET_PATH_<NAD_NAME>=namespace/name of BGP secret
eg: default/bgp-secret-macvlan-net1
```

  By default, the router pod's container is named with suffix 'fabric-peer', but can be changed using the following configuration:

```
chained_cni_config:
```

```
            named_container_for_fabric_bgp_peer_insertion: <prefix name> # Default
is fabric-peer
```

- The aci-containers-operator deployment contains an acc-provision-operator container. The container is not essential for the functioning of this solution and is provided as an experimental feature for ease of configuration. It can be excluded from the deployment by using the following configuration:

```
acc_provision_operator:
  exclude: True
```

- The `acikubectl trace` command can be used to track the OVS datapath for the packet. Following options are available for tracking packet:

    1. Pod to Pod:

```
acikubectl trace_pod_to_pod src_ns:src_pod dest_ns:dest_pod --tcp --tcp_src
<source_port> --tcp_dst <destination_port>
```

    2. Pod to Service:

```
acikubectl trace_pod_to_svc src_ns:src_pod dest_ns:dest_svc --tcp --tcp_src
<source_port> --tcp_dst <destination_port>
```

    3. Pod to External World:

```
acikubectl trace_pod_to_ext src_ns:src_pod dest_ip --tcp --tcp_src <source_port>
--tcp_dst <destination_port>
```

- A fault (vmmClusterFaultInfo) is generated in Cisco ACI, if a Kubernetes namespace, deployment, or pod is annotated with an EPG name that does not resolve to an existing EPG. A log statement is added in the aci-containers-controller log to alert the user. The fault will be cleared upon the next correct annotation, or when the aci-containers-controller restarts, or when the annotated namespace, deployment, or pod is deleted.
- Out-of-band management IP of APICs can be used to perform day-zero provision ACI fabric and in-band management IP for aci controller pod to communicate with the APIC. If there are multiple APICs in the aci-input file, the first APIC is always taken by the aci controller pod to establish websocket connection. This can be specified through command line with the `-apic-oobm-ip` parameter:

```
acc-provision -a -c <input.yaml> -f <flavor> -u <user> -p <pass> -o
<deployment.yaml> --apic-oobm-ip <ip>
```

- By default, tar.gz will not be generated for cluster upgrade. User can pass “-z <tar-file-name>” option to generate it:

```
acc-provision --upgrade -c <input.yaml> -f <flavor> -u <user> -p <pass> -o
<upgrade_deployment.yaml> -z <upgrade_deployment.tar.gz>
```

- While deleting APIC resources with “-d” option of acc-provision, can pass “--skip-app-profile-check" flag to directly delete tenant without checking the presence of application profiles:

```
acc-provision -d -u <user> -p <pass> -c <input.yaml>  -f <flavor> -o
<deployment.yaml> --skip-app-profile-check
```

- Refer to the following documents for details on installation, upgrades, and configuration of specific features:
    - *Cisco ACI and Kubernetes Integration*

- *Cisco ACI CNI Plugin for Red Hat OpenShift Container Platform Architecture and Design Guide*
- *OpenShift Install Guides*
- *Rancher Install Guides*
- *Configuration Options for Features using acc-provision (in-tree document)*
- *In-tree Configuration Documentation*
- *Upgrading the Cisco ACI CNI Plug-in*
- *Cisco Network Operator for Additional Interfaces*
- *Cisco ACI and Calico 3.26.3 Integration*

- OpenShift has a tighter security model by default, and many off-the-shelf Kubernetes applications, such as guestbook, may not run on OpenShift (if, for example, they run as root or open privileged ports like 80).
- Refer to the article *Getting any Docker image running in your own OpenShift cluster* on the Red Hat OpenShift website for details. The Cisco ACI CNI Plug-in is not aware of any configuration on OpenShift cluster or pods when it comes to working behind a proxy. Running OpenShift " oc new-app", for instance, may require access to Git Hub, and if the proxy settings on the OpenShift cluster are not correctly set, this access may fail. Ensure your proxy settings are correctly set.
- In this release, the maximum supported number of PBR based external services is 250 virtual IP addresses (VIPs). Scalability is expected to increase in upcoming releases.

  Note: With OpenShift, master nodes and router nodes are tainted by default, and you might see lower scale than an upstream Kubernetes installation on the same hardware.
- Some deployments require installation of an " allow" entry in IP tables for IGMP. This must be added to all hosts running an OpFlex agent and using VXLAN encapsulation to the leaf. The rule must be added using the following command:

  ```
  $ iptables -A INPUT -p igmp -j ACCEPT
  ```

  In order to make this change persistent across reboots, add the command either to `/etc/rc.d/rc.local` or to a cron job that runs after reboot.
- Both RHEL and Ubuntu distributions set net.ipv4.igmp_max_memberships set to 20 by default. This limits the number of end point groups (EPGs) that can be used in addition to the kube-default EPG for pod networking. If you anticipate using more than 20 EPGs, set the value to the desired number of EPGs on each node as follows:

  ```
  $ sysctl net.ipv4.igmp_max_memberships=desired_number_of_epgs
  ```

Open Issues

There are no open issues targeted for this release.

Resolved Issues

Click the bug ID to access the Bug Search tool and see additional information about the bug.

Bug ID	Description
CSCwk86076	HPP object getting deleted when the network policies are re-applied (without change in spec).
CSCwk59142	Rancher cluster fails to commision after decommision, due to vmmDomP object absent on leaves.
CSCwj36224	ACI CNI: External Service with dynamic allocation takes IP out of static pool.

Known Issues

Click the bug ID to access the Bug Search tool and see additional information about the bug.

Bug ID	Description
CSCwa36696	Temporary loss of K8s Pods cause 30-60 seconds traffic drops due to Cisco ACI objects re-deploy.

Related Content

See the Cisco Application Policy Infrastructure Controller (APIC) page for the documentation.

The documentation includes installation, upgrade, configuration, programming, and troubleshooting guides, technical references, release notes, and knowledge base (KB) articles, as well as other documentation. KB articles provide information about a specific use case or a specific topic.

By using the " Choose a topic" and " Choose a document type" fields of the Cisco APIC documentation website, you can narrow down the displayed documentation list to make it easier to find the desired document.

You can watch videos that demonstrate how to perform specific tasks in the Cisco APIC on the Cisco Data Center Networking YouTube channel.

Documentation Feedback

To provide technical feedback on this document, or to report an error or omission, send your comments to apic-docfeedback@cisco.com. We appreciate your feedback.