VMware vSphere でのエージェントベースの OpenShift 4.15 のインストール

改訂：2025 年 12 月 11 日

新機能および変更された機能に関する情報

次の表は、この最新リリースまでの主な変更点の概要を示したものです。ただし、今リリースまでの変更点や新機能の一部は表に記載されていません。

Cisco ACI CNI プラグインのリリース バージョン	機能
6.0(4)	VMware vSphere 7 でのエージェントベースの Red Hat OpenShift 4.16 のサポート。

VMware vSphere上のエージェントベースの Openshift 4.16

Cisco ACI は、VMware vSphere 7 で Red Hat OpenShift 4.19 をサポートしています。このドキュメントでは、Ansible プレイブックを使用し、コンテナ ネットワーク インターフェイス（CNI）プラグインを使用して VMware vSphereで OpenShift 4.16 をプロビジョニングする手順について説明します。

VMware vSphere に OpenShift 4.16 をインストールするための前提条件

VMware vSphere に OpenShift Container Platform（OCP）4.16 をインストールするには、次の前提条件を満たします。

Cisco ACI

- acc-provision ツール バージョン 6.0.4.1 以降をダウンロードします。

  「--flavor」オプション値を「openshift-4.16-agent-based-esx」と指定し、「-z」オプションを使用します。このツールでは、「-z」オプション値によって指定された .tar アーカイブファイルが作成されます。インストール時にはこのアーカイブ ファイルが必要になります。

  acc-provision ツールへの入力として指定されている Cisco ACI コンテナ イメージがバージョン 6.0.4.1 以降であることを確認してください。

VMware vSphere

仮想対応マシン（VM）を作成する権限を持つユーザーのログイン情報を取得します。

OpenShift

Red Hat のWeb サイトから次の情報を入手します。

- OCP4 クライアント ツール - インストールとクライアント ツールのバージョンがリストされている OpenShift Web サイトのミラー ページに移動し、必要なバージョンを選択します。`openshift-client-linux.tar.gz` および `openshift-install-linux.tar.gz` ファイルをダウンロードします。

- Pull Secret

VMware vSphere での OpenShift 4.16 のインストール

ACI インフラと CNI の構成

この手順を使用して、acc-provision を使用して ACI インフラと CNI を構成します。

始める前に

「前提条件」の項に記載されているタスクを完了します。

vSphere へのクラスタのインストールに関する前提条件およびその他の詳細については、 *RedHat OpenShift* のドキュメントを 参照することを推奨します。

手順

ステップ 1 acc-provision ユーティリティを使用して Cisco ACI ファブリックをプロビジョニングします。要件に応じて、サンプルの acc-provision 入力ファイルをカスタマイズします。次に、 ここ から最新の acc-provision パッケージをインストールし、 **pip install acc-provision** を実行します。

次のように acc-provision を実行します。

```
$ ~/openupi$ pwd
/home/<user>/openupi

$ ~/openupi$ acc-provision -a -c acc_provision_input.yaml  -f openshift-4.16-agent-based-esx -u
<user> -p <password> -o aci_deployment.yaml -z aci deployment.yaml.tar.gz
```

これにより、ACI CNI マニフェストを含む新しい `aci_deployment.yaml.tar.gz` ファイルが生成され、後で OpenShift のインストール中に使用されます。

（注）
「サンプル acc-provision」 セクションを参照してください。

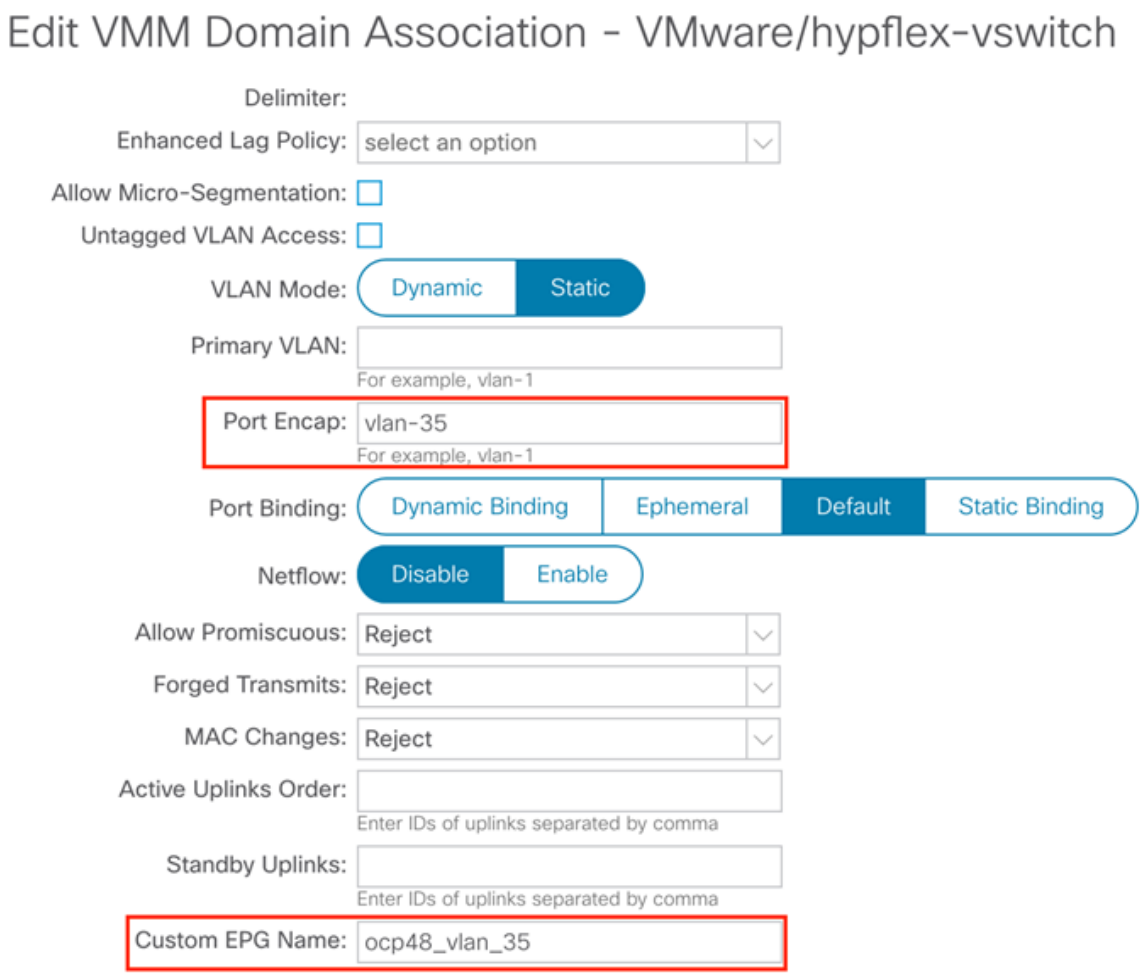
acc-provision ツールは、RHEL8 および RHEL9 オペレーティングシステムをサポートしています。

ステップ 2 Cisco ACI ファブリックがプロビジョニングされたら、 *system_id*_vlan_*kubeapi_vlan* という名前のポート グループが分散スイッチの下に作成されていることを確認します。

このドキュメントでは、このポート グループを次のように参照しています：

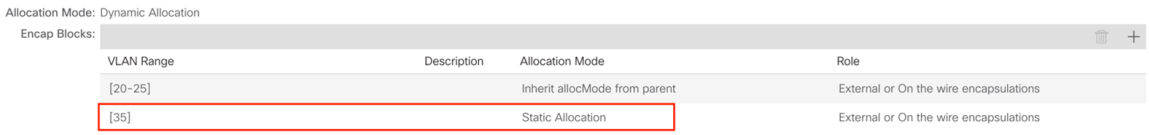
（注）
VMware 分散仮想スイッチの *api-vlan-progroup* ポート グループは、acc_provision_input ファイルに *kubeapi_vlan* として提供されるカスタム VLAN ID を使用して作成されます。

図 *1 : VMM VMware* ドメインの *aci-containers-node EPG* との 関連付け



Kube_api VLAN は、 VMware VMM ドメインに関連付けられたダイナミック VLAN プールに追加されます。割り当てモードは静的として設定されます。

図 *2 : VMM VMware* ドメインに使用される *VLAN* プール



ステップ 3 (オプション) `api-vlan-portgroup` に接続されたネットワーク インターフェースを使用して、Red Hat Enterprise Orchestrator VM をプロビジョニングします。

この VM を OpenShift クラスタの DNS サーバーとして設定します。

OpenShift ノードのカスタム ネットワーク構成の準備

ACI CNI では、追加の VLAN を各 OpenShift ノードに拡張する必要があります。マスター ノードとワーカー ノードには追加のVLANが必要ですが、ブートストラップ ノードには必要ありません。

ノード ネットワーク サブネットで構成されるインターフェイス上で追加の VLAN を構成することも、ホスト上の追加の物理インターフェイス上で構成することもできます。

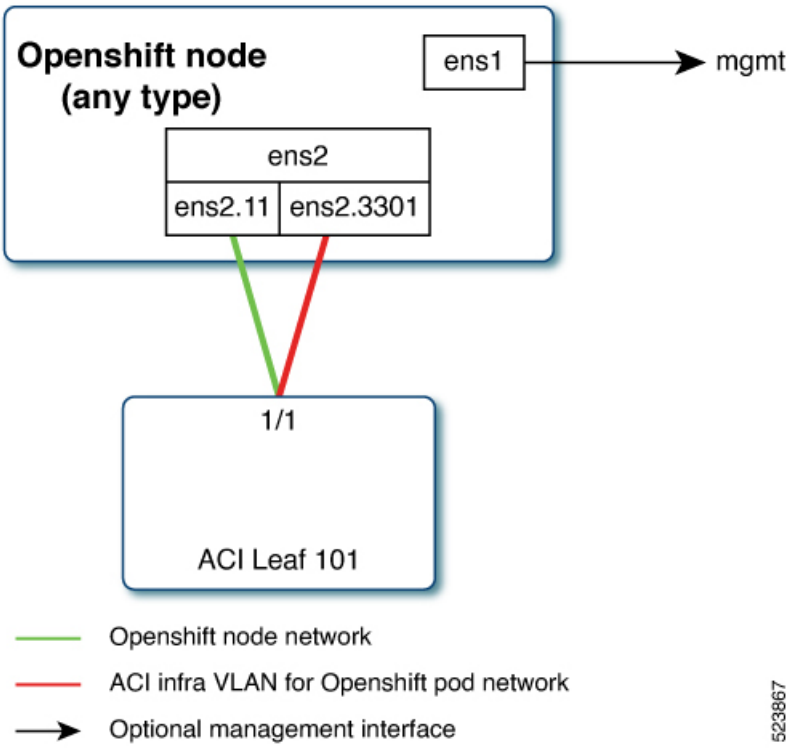
ホストのネットワーク インターフェイスを構成するために使用可能なオプションは、NMState 形式の agent-config.yaml で構成を提供することです。「*agent-config file* ファイルのサンプル」の項を参照してください。

agent-config ファイルの変更

この手順を使用して、 `agent-config.yaml` ファイルを変更します。

始める前に

追加の NIC 設定を含む agent-config ファイルは、Cisco ACI 内部ネットワーク（インフラ VLAN）をサーバー レベルまで拡張する必要があります。このインターフェイスは、ポッドネットワークに適切なタグを使用して、OVS から ACI リーフ スイッチに VxLAN トラフィックを伝送するために使用されます。OpenShift ノード トラフィックとポッド トラフィックとの分離を実現するには、ノードとインフラのネットワーク アプローチの両方にシングル サブ インターフェイス を使用します。



ノード ネットワークは、bond0 または仮想マシン NIC の VLAN サブインターフェイスとして設定されます。管理用に追加の VLAN を使用してサーバーを設定したり、 ノード ネットワークを管理ネットワークに使用したりできます。設計は、サーバーのプロビジョニング方式（PXE または手動 ISO ブート）に依存する場合があります。

以下のサンプルの YAML スニペットは、VMware での OpenShift 展開の AgentConfig の概要を示しています。これには、ランデブー IP、ホスト構成、ネットワーク インターフェイス設定など、合理化された展開のための重要な詳細が含まれています。

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: ocpvmw11
rendezvousIP: 192.168.12.3. -> A
AdditionalNTPSources:
  - time.cisco.com
hosts: -> B
  - hostname: ocpvmw11-master1 -> C
    role: master
    interfaces:
    - name: ens192
      macAddress: 00:50:56:97:2a:d6
    networkConfig: -> D
      interfaces:
        - name: ens192
          mtu: 9000
          ipv4:
            enabled: false
          ipv6:
            enabled: false
        - name: node
          type: vlan
          mtu: 9000
          state: up
          vlan:
            base-iface: ens192
            id: 131
          ipv4:
            enabled: true
            address:
              - ip: 192.168.12.3
                prefix-length: 24
            dhcp: false
          ipv6:
            enabled: false
        - name: infra
          type: vlan
          mtu: 9000
          state: up
          vlan:
            base-iface: ens160
            id: 3301
          ipv4:
            enabled: true
            dhcp: true
          ipv6:
            enabled: false
        - name: infra
          type: vlan
          mtu: 9000
          state: up
          vlan:
```

```
      base-iface: ens192
      id: 3301
     ipv4:
      enabled: true
      dhcp: true
    ipv6:
      enabled: false
dns-resolver:
  config:
    server:
      - 192.168.12.2
routes:
  config:
    - destination: 0.0.0.0/0
      next-hop-address: 192.168.12.1
      next-hop-interface: node
    - destination: 224.0.0.0/4
      next-hop-interface: infra
```

上記のサンプルでは、セクションは A、B、C、D とマークされています。よりよく理解するために詳細を以下に示します。

- **A**：この IP アドレスは、ブートストラップ プロセスを実行するノードと、アシスト型サービス コンポーネントを実行するノードを決定するために使用されます。少なくとも 1 つのホストの IP アドレスを `networkConfig` パラメータで指定しない場合は、ランデブー IP アドレスを指定する必要があります。このアドレスが指定されていない場合は、指定されたホストの `networkConfig`から 1 つの IP アドレスが選択されます。
- **B**：ホスト構成定義されたホストの数は、 `install-config.yaml` ファイルで定義されたホストの総数（ `compute.replicas` パラメータと `controlPlane.replicas` パラメータの値の合計）を超えないようにする必要があります。
- **C**：Dynamic Host Configuration Protocol（DHCP）または逆引き DNS ルックアップから取得したホスト名をオーバーライドします。各ホストには、次のいずれかの方法で指定する一意のホスト名が必要です。
- **D**：ホストのネットワーク インターフェイスを NMSte 形式で構成します。

手順

ステップ 1 クラスタのルート フォルダを作成します。

```
cd /home/<user>/openupi
mkdir upi
```

ステップ 2 `install-config.yaml`、 `agent-config.yaml` を新しく作成した upi フォルダにコピーします。

サンプルの *install-config* および *agent-config* セクションを参照してください。

ステップ 3 openshift ディレクトリを作成します。

```
mkdir -p /home/<user>/openupi/upi/opensfhit
```

ステップ 4 upi/openshift/ にあるすべての ACI マニフェスト ファイルを抽出します。

```
Tar -xvf aci_deployment.yaml.tar.gz -C upi/openshift/
```

ステップ 5 .iso イメージを作成します。

```
openshift-install agent create image -dir=upi -log-level debug
```

ステップ 6 ベアメタル マシンで agent.x86_64.iso イメージを起動します。

`agent.x86_64.iso` の準備が整いましたので、HTTP サーバーにコピーして ノードで使用できるようになりました。`agent.x86_64.iso` ファイルはすべてのノードによって使用され、各ノードのネットワーク構成は、各ノードの NMSate 構成に記載されている MAC アドレスに基づいて認識されます。

ステップ 7 VM を作成します（命名リファレンスについては、 サンプルの *agent-config* ファイルを参照してください）。

- `agent-config.yaml` ホスト名 フィールドに記載されているホスト（マスター/ワーカー）の名前を指定します。
- ネットワークとして **system_id_vlan_kubeapi_vlan** を選択します。`agent-config.yaml` で VM 用に記載されている MAC アドレスと一致するように MAC アドレスを編集します。
- UUID を有効にするには、次の手順に従います。
  1. **[VM オプション（VM Options）]** タブをクリックします。
  2. **[詳細オプション（Advanced Options）]** をクリックします。
  3. [構成パラメータ（Configuration Parameters）] の下で **[構成の編集（Edit Configuration）]** 、 **[パラメータの追加（Add Parameter）] の** 順にクリックします。
  4. [キー（Key）] 列に、 `disk.EnableUUID`と入力します。
  5. [値（Value）] 列に、 *TRUE*と入力します。
  6. **[OK]** をクリックしてから、 **[保存（Save）]** をクリックします。
- 関連するデータストアでアップロードしたイメージ `agent.x86_64.iso` を選択します。

次のタスク

openshift-install agent wait-for Bootstrap-complete および **openshift-install agent wait-for install-complete** コマンドを使用して、インストールの進行状況を確認できます。ブートストラップ ディレクトリからコマンドを実行します。

VMware vSphere にエージェントベースの OpenShift 4.16 をインストールするためのサンプル ファイル

このセクションには、 VMware vSphereにエージェント ベースの OpenShift 4.16 をインストールするために必要なサンプル ファイルが含まれています。

サンプル acc-provision-input ファイル

以下は、acc-provision-input.yaml のサンプルです。

```
#
# Configuration for ACI Fabric
#
aci_config:
  system_id: ocp4aci
  #apic-refreshtime: 1200
  apic_hosts:
  - 1.1.1.1
  vmm_domain:
    encap_type: vxlan
    mcast_range:                # Every opflex VMM must use a distinct range
      start: 225.28.1.1
      end: 225.28.255.255
    nested_inside:
      type: vmware
      name: my-vswitch
 elag_name: <eLAG_name>   # Beginning Cisco APIC 5.0(1), you can configure VMware teaming policy
                              # when link aggregation groups (LAGs) are used.


  # The following resources must already exist on the APIC.
  # They are used, but not created, by the provisioning tool.
  aep: my-aep
  vrf:                          # This VRF used to create all kubernetes EPs
    name: myl3out_vrf
    tenant: common
  l3out:
    name: myl3out
    external_networks:
    - myl3out_net
agent_based_installer:
  enable: true
#
# Networks used by ACI containers
#
net_config:
  node_subnet: 192.168.18.1/24
  pod_subnet: 10.128.0.1/16     # Subnet to use for Kubernetes # Pods/CloudFoundry containers
  extern_dynamic: 10.3.0.1/24   # Subnet to use for dynamic external IPs
  extern_static: 10.4.0.1/24    # Subnet to use for static external IPs
  node_svc_subnet: 10.5.0.1/24  # Subnet to use for service graph
  kubeapi_vlan: 131
  service_vlan: 132
  infra_vlan: 3301
#interface_mtu: 1600
#service_monitor_interval: 5  # IPSLA interval probe time for PBR tracking
                              # default is 0, set to > 0 to enable, max: 65535
#pbr_tracking_non_snat: true  # Default is false, set to true for IPSLA to
                              # be effective with non-snat services

#
# Configuration for container registry
# Update if a custom container registry has been setup
#
kube-config:
  image_pull_policy: Always
  ovs_memory_limit: 1Gi
registry:
  image_prefix: quay.io/noiro
```

agent-config ファイルのサンプル

以下は、 agent-config.yaml のサンプルです。

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: ocpvmw11
rendezvousIP: 192.168.12.3
AdditionalNTPSources:
  - time.cisco.com
hosts:
  - hostname: ocpvmw11-master1
    role: master
    interfaces:
    - name: ens192
      macAddress: 00:50:56:97:2a:d6
    networkConfig:
      interfaces:
        - name: ens192
          mtu: 9000
          ipv4:
            enabled: false
          ipv6:
            enabled: false
        - name: node
          type: vlan
          mtu: 9000
          state: up
          vlan:
            base-iface: ens192
            id: 131
          ipv4:
            enabled: true
            address:
              - ip: 192.168.12.3
                prefix-length: 24
            dhcp: false
          ipv6:
            enabled: false
        - name: infra
          type: vlan
          mtu: 9000
          state: up
          vlan:
            base-iface: ens192
            id: 3301
          ipv4:
            enabled: true
            dhcp: true
          ipv6:
            enabled: false
      dns-resolver:
        config:
          server:
            - 192.168.12.2
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.12.1
            next-hop-interface: node
          - destination: 224.0.0.0/4
            next-hop-interface: infra
```

```
- hostname: ocpvmw11-master2
  role: master
  interfaces:
  - name: ens192
    macAddress: 00:50:56:97:f6:65
  networkConfig:
    interfaces:
      - name: ens192
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 131
        ipv4:
          enabled: true
          address:
            - ip: 192.168.12.4
              prefix-length: 24
          dhcp: false
        ipv6:
          enabled: false
      - name: infra
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 3301
        ipv4:
          enabled: true
          dhcp: true
        ipv6:
          enabled: false
    dns-resolver:
      config:
        server:
          - 192.168.12.2
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.12.1
          next-hop-interface: node
        - destination: 224.0.0.0/4
          next-hop-interface: infra
- hostname: ocpvmw11-master3
  role: master
  interfaces:
  - name: ens192
    macAddress: 00:50:56:97:07:42
  networkConfig:
    interfaces:
      - name: ens192
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
```

```
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 131
        ipv4:
          enabled: true
          address:
            - ip: 192.168.12.5
              prefix-length: 24
          dhcp: false
        ipv6:
          enabled: false
      - name: infra
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 3301
        ipv4:
          enabled: true
          dhcp: true
        ipv6:
          enabled: false
    dns-resolver:
      config:
        server:
          - 192.168.12.2
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.12.1
          next-hop-interface: node
        - destination: 224.0.0.0/4
          next-hop-interface: infra
- hostname: ocpvmw11-worker1
  role: worker
  interfaces:
  - name: ens192
    macAddress: 00:50:56:97:b5:07
  networkConfig:
    interfaces:
      - name: ens192
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 131
        ipv4:
          enabled: true
          address:
            - ip: 192.168.12.6
              prefix-length: 24
          dhcp: false
```

```
        ipv6:
          enabled: false
      - name: infra
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 3301
        ipv4:
          enabled: true
          dhcp: true
        ipv6:
          enabled: false
    dns-resolver:
      config:
        server:
          - 192.168.12.2
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.12.1
          next-hop-interface: node
        - destination: 224.0.0.0/4
          next-hop-interface: infra
- hostname: ocpvmw11-worker2
  role: worker
  interfaces:
  - name: ens192
    macAddress: 00:50:56:97:44:9b
  networkConfig:
    interfaces:
      - name: ens192
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 131
        ipv4:
          enabled: true
          address:
            - ip: 192.168.12.7
              prefix-length: 24
          dhcp: false
        ipv6:
          enabled: false
      - name: infra
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens192
          id: 3301
        ipv4:
          enabled: true
          dhcp: true
        ipv6:
```

```
          enabled: false
      dns-resolver:
        config:
          server:
            - 192.168.12.2
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.12.1
            next-hop-interface: node
          - destination: 224.0.0.0/4
            next-hop-interface: infra
```

`install-config` ファイルの例

以下は、 `install-config.yaml` の例です。

```
apiVersion: v1
baseDomain: ocplab.local
proxy:
  httpsProxy: <http-proxy>
  httpProxy: <https-proxy>
  noProxy: <no-proxy>
compute:
- name: worker
  replicas: 2
controlPlane:
  name: master
  replicas: 3
metadata:
  name: ocpvmw11
networking:
  machineNetwork:
  - cidr: 192.168.12.0/24
  clusterNetwork:
  - cidr: 10.2.0.0/16
    hostPrefix: 23
  networkType: CiscoACI
  serviceNetwork:
  - 172.30.0.0/16
platform:
  vsphere:
    apiVIPs:
    - 192.168.12.30
    ingressVIPs:
    - 192.168.12.29
fips: false
pullSecret: <RH-account-pull-secret>
sshKey: <host-ssh-key>
```

OpenShift の廃止

この手順に従って、OpenShift を廃止し、ACI からプロビジョニングされた設定を ACI から削除します。

（注） Cisco APIC リリース 5.2 以降、OpenShift の VMM ドメインを APIC GUI から削除することはできません。これは REST API を使用する場合にのみ可能であるため、acc-provision ツールを使用して VMM ドメイン、および廃止された OpenShift クラスターで使用されているその他の関連オブジェクトを削除するのが便利です。APIC にアクセスするために acc-provision ツー—ルが使用する `acc-input-config.yaml` ファイルと証明書があることを確認します。

始める前に

Openshift クラスターを廃止または削除する場合は、そのクラスターにプロビジョニングされた ACI 設定を ACI から削除する必要があります。acc-provision ツールを使用して、その構成を削除できます。

手順

次のコマンドを使用して、ACI インフラストラクチャのプロビジョニングに使用されたマシンとフォルダから、事前にプロビジョニングされた設定と VMM ドメインを削除します。

acc-provision -d -f openshift-4.16-agent-based-esx -c *acc-input-file* **-u** *user* **-p** *password*

例：

```
acc-provision -d -f openshift-4.16-agent-based-esx -c acc-input-config.yaml -u admin -p password
```

既知の問題

インストール プロセスに影響を与える可能性がある既知の問題は次のとおりです。

- ノードの汚染が原因でインストールが妨げられる。RedHat サポート ケースの *Web* サイトでケース番号 03682671 を参照してください。
- 「Storage Cluster Operator Degraded - Solution in progress」 : *RedHat* ソリューション ケースの *Web* サイトでケース番号 5926951 を参照してください。
- OCP クラスタの vSphere 構成の変更：プラットフォーム統合が有効になっている状態で Assisted Installer を使用する場合、インストールされているクラスタの vSphere 構成の更新は手動で行う必要があります。このアクションは、インストールが完全に完了し、クラスタが console.redhat.com にリンクされた後にのみ実行される必要があります。 *RedHat* ソリューションの *Web* サイトでソリューション番号 6677901 を参照してください。

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255（フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

翻訳について
このドキュメントは、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。