VMware vSphere での OpenShift 4.18 のインストール

改訂：2026 年 2 月 5 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
6.1(1)	VMware vSphere 7 User-Provisioned Infrastructure（UPI）での Red Hat OpenShift 4.19 のサポート。

VMware vSphere上の Openshift 4.19

Cisco ACI は、VMware vSphere 7 User-Provisioned Infrastructure (UPI) 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 4.19 をプロビジョニングする手順について説明します。

Ansibleプレイブックは、必要なインターフェイス構成で仮想マシン（VM）をプロビジョニングし、イグニッション構成ファイルを生成します。ユーザーは、独自の DHCP、DNS、およびロードバランシング インフラストラクチャを高可用性のベストプラクティスに従って展開する必要があります。

Ansible プレイブックは GitHubで入手できます。

以下は Ansible プレイブックです。

- `asserts.yml`：このプレイブックは、 `all.yml` ファイル内の変数宣言について、基本的な検証を行います。
- `setup.yml`：このプレイブックは次のタスクを実行します。
  - オーケストレータ ノードの設定：
    - Terraform、OpenShift クライアント、および OpenShiftインストーラをインストールします。ブートストラップ、マスターノード、およびワーカーノードの Terraform 変数を作成します。マスターとワーカーの machine-config オペレータ、OpenShift インストール構成ファイルを作成します。
    - ロードバランサ ノードの設定：Security-Enhanced Linux（SELinux）を無効にし、HAProxy を設定し、選択されている場合は DHCPと DNS をセットアップします。

      このオプションの手順は、これらの 3 つのコンポーネントを、 `provision_dhcp`、 `provision_dns`、および `provision_lb` 変数が true に設定した場合にのみ、構成します。

- `oshift_prep.yml`：
  - インストールおよびブートストラップ ディレクトリをセットアップします。
  - openshift-install を使用してマニフェストを生成します。
  - 追加の machine-config オペレータ マニフェストを追加します。

- Cisco ACI-CNI マニフェストを追加します。
- マニフェストのバックアップを作成します。
- ブートストラップ、マスター、およびワーカー ノードのイグニッション ファイルを設定します。
- ブートストラップ イグニッション ファイルをロードバランサ ノードにコピーします。

- `create_nodes.yml`：
  - Terraform を使用して、ブートストラップ、マスター、およびワーカー ノードをプロビジョニングします。
  - 選択されている場合、シスコの証明書署名要求（CSR）を承認する cron ジョブをセットアップします。
- `delete_nodes.yml`：すべてのマスター ノードとワーカー ノードを削除します。

VMware vSphere に OpenShift 4.19 をインストールするための前提条件

VMware vSphere に OpenShift Container Platform（OCP）4.19 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 6.1.1.1 以降をダウンロードします。

  「--flavor」オプション値を「openshift-4.19-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 6.1.1.1 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 Open Virtualization Appliance（OVA）：関連するリリースの OVA イメージをダウンロードしていることを確認します。次に *mirror* ページに移動します。これはすべての RHCOS バージョンがリストされている OpenShift Web サイトのページで、ここで必要なバージョンを選択します。次に `rhcos-vmware.x86_64.ova` ファイルをダウンロードします。
- OCP4 クライアント ツール： *mirror* ページに移動します。これは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 サイトのページで、ここで必要なバージョンを選択します。次に `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。

- シークレットのプル

VMware vSphere での OpenShift 4.19 のインストール

（注） 詳細については、 サンプル Ansible group_vars/all.yml ファイル （13 ページ） セクションに指定します。

始める前に

前記の 前提条件 セクションのタスクを完了します。

vSphere へのクラスタのインストールの前提条件と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：

```
acc-provision -a -c acc_provision_input.yaml -u admin -p ### -f openshift-4.19-esx -z manifests.tar.gz
```

詳細については、 サンプル acc-provision-input ファイル （12 ページ） セクションに指定します。

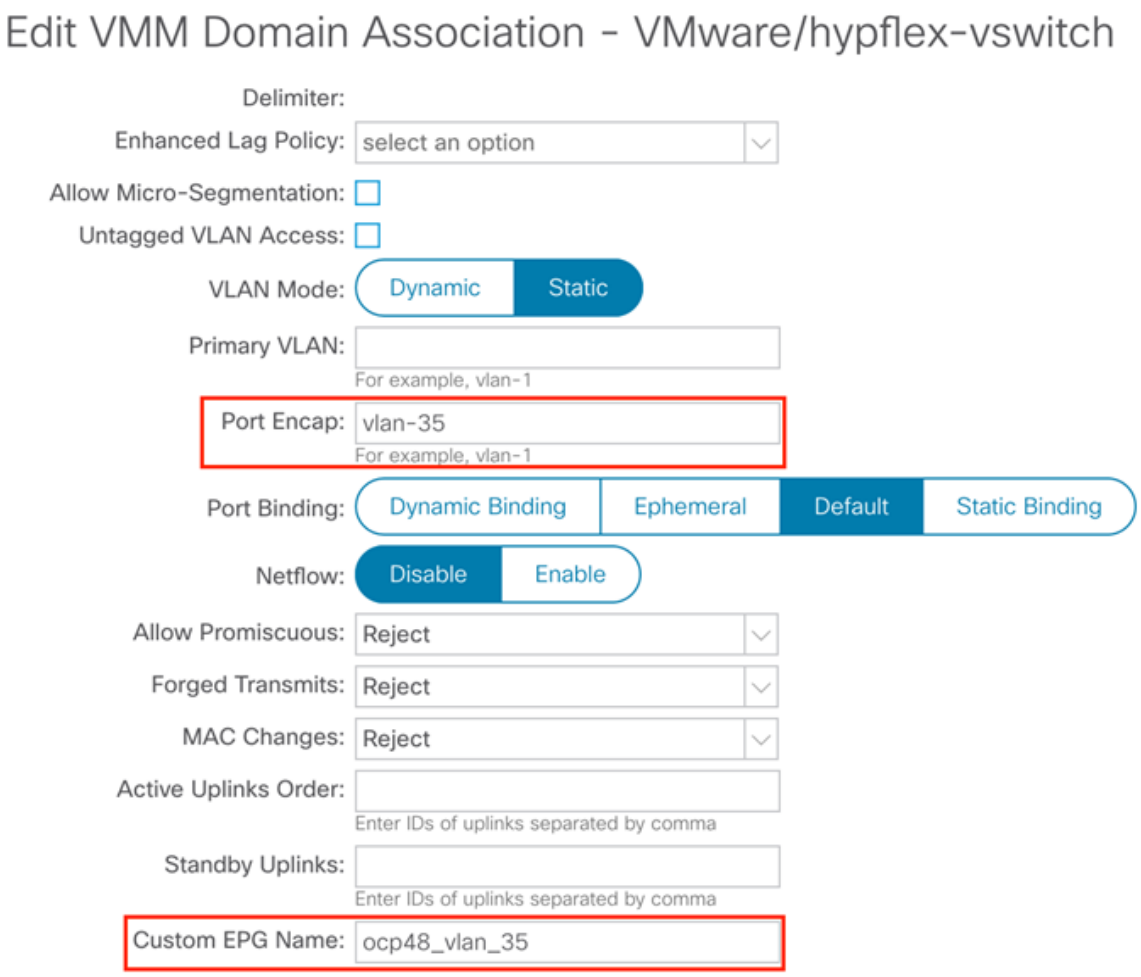
（注）
現在、RHEL8 でのみ満たされる Python 3 の依存関係が存在するため、acc-provision ツールは RHEL8 オペレーティング システムでのみ実行できます。

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、ポート グループが、*system_id*_vlan_*kubeapi_vlan* という名前で、分散スイッチの下に作成されます。

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： `api-vlan-portgroup`。

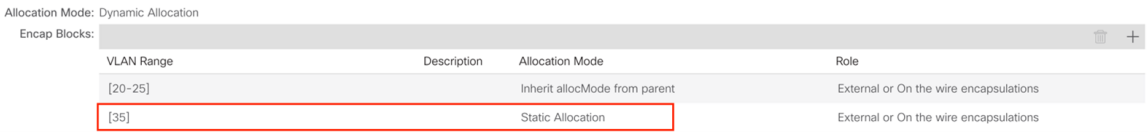
（注）
api-vlan-progroup ポートグループは、VMware 分散仮想スイッチ内に、acc_provision_input ファイル内で *kubeapi_vlan* として指定されたカスタム VLAN ID を使用して作成されます。

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 VMware vSphereで、OpenShift Container Platform 4（OCP4）Open Virtual Appliance（OVA）イメージをインポートします。

ドメインの `api-vlan-portgroup` を、ネットワーク インターフェイスのポート グループとして指定します。

ステップ 4 Red Hat Enterprise ロードバランサ仮想マシン（VM）を、`api-vlan-portgroup` に接続されたネットワーク インターフェイスを使用して、プロビジョニングします。。

Ansible プレイブックでは、必要に応じて、この VM を OpenShift クラスターのロードバランサ、DNS サーバー、および DHCP サーバーとして設定できます。

ステップ 5 Red Hat Enterprise オーケストレータ VM を、`api-vlan-portgroup` に接続されたネットワーク インターフェイスを使用して、プロビジョニングします。。

Ansible プレイブックはオーケストレータ VM から動作します。

ステップ 6 オーケストレータ VM で次のタスクを実行します：

a) GitHub からリポジトリをクローンします。 https://github.com/noironetworks/openshift_vsphere_upi。

b) 「ocp419」ブランチをチェックアウトします。

c) セキュア シェル（SSH）キーを生成し、それらをロードバランサ VM にコピーします。

d) 有効化 `ansible-2.9-for-rhel-8-x86_64-rpms` リポジトリを有効化します。 リポジトリにイメージを追加します。

e) Ansible パッケージを最新バージョンに更新します。

f) ディレクトリを Git のクローン先ディレクトリに変更します。

g) Ansible モジュールのインストールの要件：

```
ansible-galaxy install -r requirements.yaml
```

h) よく利用するテキストエディタで `groups/all.yml` および `hosts.ini` ファイルを編集し、サイトに必要な値を設定します。

詳細については、 サンプルの hosts.ini ファイル （15 ページ） セクションに指定します。

i) 変数値の基本的な検証を `asserts.yml` プレイブックを使用して実行します。

```
ansible-playbook asserts.yml
```

j) acc-provision ユーティリティで作成したアーカイブ ファイルをファイル ディレクトリにコピーし、`aci_manifests.tar.gz` という名前を付けます。。

k) Ansible setup プレイブックを実行します。

```
ansible-playbook setup.yml
```

オーケストレータとロードバランサの VM を構成するために `setup` プレイブックを実行します。

l) Ansible `oshift_prep` プレイブックを実行します。

```
ansible-playbook oshift_prep.yml
```

OpenShift マニフェストおよびイグニッション ファイルを生成するために、`oshift_prep` を実行します。

m) Ansible `create_nodes` プレイブックを実行します。

```
ansible-playbook create_nodes.yml
```

値は、`create_nodes` プレイブックは VM を作成します。VM が作成されると、OCP4 のインストールプロセスがバックグラウンドで開始されます。この段階では、インストーラが作成した `kubeconfig` ファイルによって、クラスター API にアクセスできるはずです。

次の kubeconfig ファイルは、 *base_dir*/bootstrap/auth ディレクトリにあります。値 base_dir は group_vars/all.yml ファイルで設定されます。そのデフォルト値は /root/ocpinstallです。

ステップ 7 （オプション）次のコマンド openshift-install wait-for bootstrap-complete および openshift-install wait-for install-complete は、インストールの進行状況をチェックするために使用できます。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。

デフォルトの入力コントローラの更新

ACI ロードバランサを使用するようにデフォルトの Ingress コントローラの公開戦略を更新するには、cluster-admin 権限を持つユーザーとしてログインし、次を実行します：

```
oc replace --force --wait --filename - <<EOF apiVersion: operator.openshift.io/v1 kind: IngressController
metadata:   namespace: openshift-ingress-operator   name: default spec:   endpointPublishingStrategy:     type:
 LoadBalancerService     loadBalancer:       scope: External EOF >
```

詳細については、 クラスターのデフォルト *Ingress* コントローラを内部に構成する セクションを参照してください。 *Ingress Operator in OpenShift Container Platform* Red Hat ガイドに記載されています。

ACI CNI を使用した MachineSet の設定

マシン API は、アップストリームのクラスタ API プロジェクトとカスタムの OpenShift Container Platform リソースに基づくプライマリリソースの組み合わせです。

OpenShift Container Platform 4.19 クラスタの場合、クラスタのインストールが完了した後、マシン API はすべてのノード ホストのプロビジョニング管理アクションを実行します。OpenShift Container Platform 4.19 は、マシン API により、パブリックまたはプライベート クラウド インフラストラクチャ上で柔軟でダイナミックなプロビジョニング方式を提供します。

2 つの主要なリソースは次のとおりです。

- マシン：ノードのホストを記述する基本単位。マシンには、さまざまなクラウド プラットフォームに提供されるコンピューティング ノードのタイプを説明する providerSpec 仕様があります。たとえば、Amazon Web Services（AWS）のワーカーノードのマシン タイプは、特定のマシン タイプと必要なメタデータを定義できます。
- MachineSet：MachineSet リソースはマシンのグループです。マシン セットはマシンに対応し、レプリカセットはポッドに対応します。マシンの追加が必要な場合や、それらのマシンをスケールダウンする必要がある場合は、マシンセットのレプリカ フィールドを変更して、コンピューティングのニーズに合わせます。

OpenShift Container Platform ノード上のオペレーティング システムは、Machine Config Operator によって管理される MachineConfig オブジェクトを作成することで変更することができます。

MachineConfiguration ファイルの作成

次の手順に基づき、新しいノードのネットワーク インターフェイスを設定する MachineConfig ファイルを作成します。

- ACI インフラ インターフェイス（ens224）
- ACI インフラ サブインターフェイス（ens224.{InfraVlanID}）
- BUM トラフィック レプリケーションの Opflex-route（224.0.0.0/4）

必要な構成（サンプル）は、次の手順に示されていますが、特定の環境に合わせてカスタマイズする必要があります。一般に、次の変更が必要です。

- 現れている {InfraVLAN} をすべてファブリックの ACI インフラ VLAN で置き換えます。
- 現れているすべての {MTU} をすべてクラスターに選択した MTU で置き換えます。

（注） ネットワーク インターフェイスが *ens224* であることを確認します（別の名前ではありません）。

手順

ステップ 1 80-opflex-route を作成します。

```
#!/bin/bash
 if [ "$1" == "ens224.{InfraVLAN}" ] && [ "$2" == "up" ]; then
 route add -net 224.0.0.0 netmask 240.0.0.0 dev ens224.{InfraVLAN}
 fi
```

ステップ 2 次に **ens224.nmconnection**を作成します。

```
[connection]
id=ens224
type=ethernet
interface-name=ens224
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=
mtu={MTU}

[ipv4]
dns-search=
method=disabled

[ipv6]
addr-gen-mode=eui64
dns-search=
method=disabled

[proxy]
```

ステップ 3 次に **ens224.{InfraVLAN}.nmconnection**を作成します。

```
[connection]
```

```
id=ens224.{InfraVLAN}
type=vlan
interface-name=ens224.{InfraVLAN}
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[vlan]
egress-priority-map=
flags=1
id={InfraVLAN}
ingress-priority-map=
parent=ens224

[ipv4]
dns-search=
method=auto

[ipv6]
addr-gen-mode=eui64
dns-search=
method=auto

[proxy]
```

ステップ 4 上記の 3 つの構成（ステップ 1、2、3）を、 *base64* エンコード文字列で置き換え、次に示すように、machineconfigテンプレートで使用します（base64 の後）。

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 00-worker-cni
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
      - contents:
          source: data:text/plain;charset=UTF-8;base64,
                  IyEvYmluL2Jhc2gKIGlmIFsgIiQxIiA9PSAiZW5zMjI0LjM0NTYiIF0gJiYgWyAi
                  JDIiID09ICJ1cCIgXTsgdGhlbgogcm91dGUgYWRkIC1uZXQgMjI0LjAuMC4wIG5ld
                  G1hc2sgMjQwLjAuMC4wIGRldiBlbnMyMjQuMzQ1NgogZmk=
        mode: 0755
        overwrite: true
        path: /etc/NetworkManager/dispatcher.d/80-opflex-route
      - contents:
          source: data:text/plain;charset=UTF-8;base64,
                  W2Nvbm5lY3Rpb25dCmlkPWVuczIyNAp0eXBlPWV0aGVybmV0CmludGVyZmFjZS1uYW1lPWVuczIy
                  NAptdWx0aS1jb25uZWN0PTEKcGVybWlzc2lvbnM9CgpbZXRoZXJuZXRdCm1hYy1hZGRyZXNzLWJs
                  YWNrbGlzdD0KbXR1PTkwMDAKCltpcHY0XQpkbnMtc2VhcmNoPQptZXRob2Q9ZGlzYWJsZWQKCltpc
                HY2XQphZGRyLWdlbi1tb2RlPWV1aTY0CmRucy1zZWFyY2g9Cm1ldGhvZD1kaXNhYmxlZAoKW3Byb3h5XQ==

        mode: 0600
        overwrite: true
        path: /etc/NetworkManager/system-connections/ens224.nmconnection
      - contents:
          source: data:text/plain;charset=UTF-8;base64,
                W2Nvbm5lY3Rpb25dCmlkPWVuczIyNC4zNDU2CnR5cGU9dmxhbgppbnRlcmZhY2UtbmFtZT1lbnMyMjQuMz
```

```
            Q1NgptdWx0aS1jb25uZWN0PTEKcGVybWlzc2lvbnM9CgpbZXRoZXJuZXRdCm1hYy1hZGRyZXNzLWJsYWNrb
            GlzdD0KClt2bGFuXQplZ3Jlc3MtcHJpb3JpdHktbWFwPQpmbGFncz0xCmlkPTM0NTYKaW5ncmVzcy1wcmlv
            cml0eS1tYXA9CnBhcmVudD1lbnMyMjQKCltpcHY0XQpkbnMtc2VhcmNoPQptZXRob2Q9YXV0bwoKW2lwdjZd
               CmFkZHItZ2VuLW1vZGU9ZXVpNjQKZG5zLXNlYXJjaD0KbWV0aG9kPWF1dG8KCltwcm94eV0=
      mode: 0600
      overwrite: true
      path: /etc/NetworkManager/system-connections/ens224.{InfraVLAN}.nmconnection
```

（注）
次のパス内の {InfraVLAN} をACI InfraVLAN ID で置き換えます。
`/etc/NetworkManager/system-connections/ens224.{InfraVLAN}.nmconnection` （上記の例の最終行）。
MachineConfig の名前をカスタマイズすることもできます。上記の例では、名前は `00-worker-cni`です。

ステップ 5 次の **oc create -f** コマンドを使用して、クラスターの MachineConfig を作成します。

machineconfig は既存のワーカーノード（2 つ）に適用され、すでに存在する 3 つのファイルを同一のコピーで置き換えます。マシン構成に md5 構成を含めると、ノードに 3 つのファイルが作成されます。既存のワーカーノードは、一度に 1 つずつ再起動します。

ステップ 6 クラスターID を、 **oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster** コマンドを使用して取得します

ステップ 7 マシンセットを作成します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  name: ocp4aci-jlff9-worker
  namespace: openshift-machine-api
  labels:
    machine.openshift.io/cluster-api-cluster: ocp4aci-jlff9
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: ocp4aci-jlff9
      machine.openshift.io/cluster-api-machineset: ocp4aci-jlff9-worker
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: ocp4aci-jlff9
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: ocp4aci-jlff9-worker
    spec:
      metadata: {}
      providerSpec:
        value:
          numCoresPerSocket: 1
          diskGiB: 120
          snapshot: ''
          userDataSecret:
            name: worker-user-data
          memoryMiB: 8192
          credentialsSecret:
            name: vsphere-cloud-credentials
```

```
network:
  devices:
    - networkName: ocp4aci_vlan_35
    - networkName: ocp4aci
metadata:
  creationTimestamp: null
numCPUs: 2
kind: VSphereMachineProviderSpec
workspace:
  datacenter: my-dc
  datastore: mydatastore
  folder: /mydatastore/vm/ocp4aci
  server: myvsphere.local.lab
template: RHCOS418
apiVersion: vsphereprovider.openshift.io/v1beta1
```

上記は設定例であり、次のように変更が必要になる場合があります。

- クラスタ ID は `ocp4aci-jlff9` です。必要なクラスター ID に置き換えます。
- 次の `ocp4aci_vlan_35` および `ocp4aci` をポート グループ名で置き換えます（順序は重要です。順番を入れ替えないでください）。
- 次の `replicas` の値を設定します。クラスターの起動中に作成されるワーカー上に、新しいワーカーをいくつ作成するかを決めます。
- すべての `workspace` パラメータ（datacenter、datastore）を編集し、vSphere設定と一致させます。
- 必要に応じて、VM の仕様（memoryMiB、numCPUs、および diskGiB）を編集します。

（注）
元のワーカーは MachineSet によって管理されていないため、削除することができます。ワーカーを削除する場合は、OCP ルータが新しいノードで起動するまで待ちます。

MachineSet を使用したコンピューティング ノードの拡張

この手順に従って、MachineSet を使用してコンピューティング ノードを拡張します。

手順

ステップ 1 VMフォルダを作成します： */DC name*/vm/*cluster name* コマンドを vCenter内 で実行します。 *cluster name* は OpenShift クラスターID です。この例では、フォルダの名前は次のようになります。
`/mydatastore/vm/ocp4aci`。

ステップ 2 vCenterでテンプレートを作成します。

次に、 *template name* が が RHCOS419 テンプレートの *cluster name* と一致していることを確認します 。

ステップ 3 vCenter で *id* という TAG カテゴリを作成します。

ステップ 4 DHCP サーバを設定して、ノードに IP アドレスを割り当てます。

VMware vSphereに OpenShift 4.19 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 4.19 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

次に`acc-provision-input.yaml`の例を示します。強調表示または太字で示されている値は、設置場所の要件に合わせて変更する必要がある値です。

```
#
# 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.
    installer_provisioned_lb_ip: 10.213.0.201

  # 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

#
# 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: 35
  service_vlan: 36
  infra_vlan: 3901
  #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
```

サンプル Ansible `group_vars/all.yml` ファイル

次はサンプルのgroup_vars/all.ymlです。強調表示または太字で示されている値は、設置場所の要件に合わせて変更する必要がある値です。

```
#domainname
#   type: string, base dns domain name, cluster metadata name is added as subdomain to this
#   required: yes
domainname: ocplab.local

#provision_dns
#   type: boolean, True or False
#   required: yes
#   notes: If set to true, load balancer is configured as dns server.
#          If false, it is assumed that the dns server pre-exists.
provision_dns: True

#dns_forwarder:
#   type: ip address
#   required: yes
#   notes: This value is used when setting up a dhcp service and also for 'forwarders' value in dns configuration.
dns_forwarder: 172.28.184.18

#loadbalancer_ip:
#   type: ip address or resolvable hostname
#   required: yes
#   notes: This host is configured as load balancer for cluster and also as dhcp and dns server if required .
 This IP address is the same as the one that you configure in installer_provisioned_lb_ip in the acc-provision
 config.
loadbalancer_ip: 192.168.18.201. This IP address is the same as the one that you configure in
installer_provisioned_lb_ip in the acc-provision config.

#auto_approve_csr:
#   type: boolean
#   required: yes
#   notes: when set to true, sets up a cron job to auto approve openshift csr
auto_approve_csr: True

#proxy_env
#
proxy_env:
   #donot remove dummy field, irrespective of whether setup needs a proxy or not.
   dummy: dummy
   #set the http/https proxy server, if setup does not need proxy, comment the below values.
   #these values are used for ansible tasks and also passed on to openshift installer
   http_proxy: http://1.1.1.1:80
```

```
   https_proxy: http://1.1.1.1:80
   no_proxy: 1.2.1.1,1.2.1.2

#packages
#  defines the urls to download terraform, openshift client and openshift-install tools from.
packages:
   validate_certs: False
   terraform_url: https://releases.hashicorp.com/terraform/1.0.0/terraform_0.13.6_linux_amd64.zip
   openshift_client_linux_url:
https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.19.9/openshift-client-linux-4.19.9.tar.gz
   openshift_install_linux_url:
https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.19.9/openshift-install-linux-4.19.9.tar.gz

#default_aci_manifests_archive:
#  default filename that is searched under files directory.
#  this can be overridden by passing extra parameter aci_manifests_archive on ansible command line
default_aci_manifests_archive: aci_manifests.tar.gz

#opflex_interface_mtu:
# required: yes
# MTU size for interface connected to fabric, must be greater than 1564
opflex_interface_mtu: 1800

#vsphere
platform
   vsphere:
      vcenters:
      - server:  myvsphere.local.lab
        user: administrator@vsphere.local
        passwd: xxxx
        datacenters:
        - my-dc

      failureDomains:
      - name: fd-1
        region: region-1
        zone: zone-1
        server: myvsphere.local.lab
        topology:
          computeCluster:  /my-dc/host/my-cluster
          networks:
          - VM Network
          datacenter: my-dc
          datastore:  /my-dc/datastore/my-datastore
      RHCOS_template_name:  RHCOS419
      folder:  ocp-419

#base_dir
#  type: directory path
#  required: yes
#  notes: All install files and directories are created under this directory
base_dir: /root/ocpinstall

#node network details. This is common for bootstrap, master and worker nodes.
node_network_cidr: 192.168.53.0/24
node_network_gateway: 192.168.53.1
node_network_netmask: 255.255.255.0

service_network_cidr: 172.30.0.0/16
#bootstrap node variables
bootstrap_vars:
   node_ip: 192.168.18.210       #required
   cpu_count: 8                  #optional: defaults to 4
   memory_KB: 16384              #optional: defaults to 8192
```

```
   disk_size_MB: 40              #optional: defaults to 40

masters_vars:
   cpu_count: 8                  #optional: defaults to 4
   memory_KB: 16384              #optional: defaults to 16384
   disk_size_MB: 40              #optional: defaults to 40
   nodes:
      #mac address and ip address for each node is required
      - master-1:
         ip: 192.168.18.211
      - master-2:
         ip: 192.168.18.212
      - master-3:
         ip: 192.168.18.213

workers_vars:
   cpu_count: 8                #optional: defaults to 4
   memory_KB: 16384            #optional: defaults to 16384
   disk_size_MB: 40            #optional: defaults to 40
   nodes:
      #mac address and ip address for each node is required
      - worker-1:
          ip: 192.168.18.214
      - worker-2:
          ip: 192.168.18.215

#user_ssh_key:
#  required: no
#  notes: if specified this key is setup on nodes, else ssh key of current
#         user is used.
user_ssh_key: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQD...

#additional_trust_bundle:
#  required: no
#  notes: use this field to add a certificate for private repository
#
# example:
#additional_trust_bundle: |
#   -----BEGIN CERTIFICATE-----
#   MIIDDDCCAfQCCQDuOnV7XBjpODANBgkqhkiG9w0BAQsFADBIMQswCQYDVQQGEwJV
#   UzELMAkGA1UECAwCQ0ExDDAKBgNVBAcMA1NKQzEOMAwGA1UECgwFQ2lzY28xDjAM
#  -----END CERTIFICATE-----

#openshift_pullsecret:
#  required: yes
#  notes: refer to https://cloud.redhat.com/openshift/install/pull-secret
#  example:
#    openshift_pullsecret: {"auths":{"cloud.openshift.com":{"auth":.........}
openshift_pullsecret: xxx
```

サンプルの hosts.ini ファイル

次はサンプルの hosts.ini です強調表示または太字で示されている値は、設置場所の要件に合わせて変更する必要がある値です。

```
[orchestrator]
192.168.18.200

[lb]
192.168.18.201
```

ワーカーノードで IP 転送を有効にする

ノードポート構成を使用してサービスをテストするには、ワーカーノードで IP 転送を有効にする必要があります。

この手順を使用して、ワーカーノードで IP 転送を有効にします。

手順

ステップ 1 ワーカーノードに SSH で接続します。

ステップ 2 次のコマンドを実行して、IP 転送を有効にします。

```
sudo sysctl -w net.ipv4.ip_forward=1
```

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-4.19-esx -c *acc-input-file* **-u** *user* **-p** *password*

例：

```
acc-provision -d -f openshift-4.19-esx -c acc-input-config.yaml -u admin -p password
```

【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。

この資料の記載内容は2008年10月現在のものです。
この資料に記載された仕様は予告なく変更する場合があります。

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。