



## ベアメタルサーバーでのエージェントベースの **OpenShift 4.16** のインストール

[新機能および変更された機能に関する情報](#) 2

[ベア メタル上のエージェントベースの OpenShift 4.16](#) 2

[ベア メタルサーバーで OpenShift 4.16 をサポートするための要件](#) 2

[インストールプロセス](#) 4

[ベア メタル オペレータによるエージェント ベースのインストールのスケーリング](#) 12

# 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)	Cisco Application Centric Infrastructure (ACI) は、ベア メタル サーバーで Red Hat エージェント ベースの OpenShift をサポートします。

## ベア メタル上のエージェントベースの OpenShift 4.16

このドキュメントは、ACI CNI を使用した OCP のインストールに関連しています。ただし、ACI CNI に関連しないインフラストラクチャの問題を特定して解決するには、関連するインストール ガイドを参照して、デフォルトの OVN Kubernetes を使用してベア メタル ノードに OCP を最初にインストールしてください。OpenShift 4.16 コンテナ プラットフォームのドキュメントを確認できます。



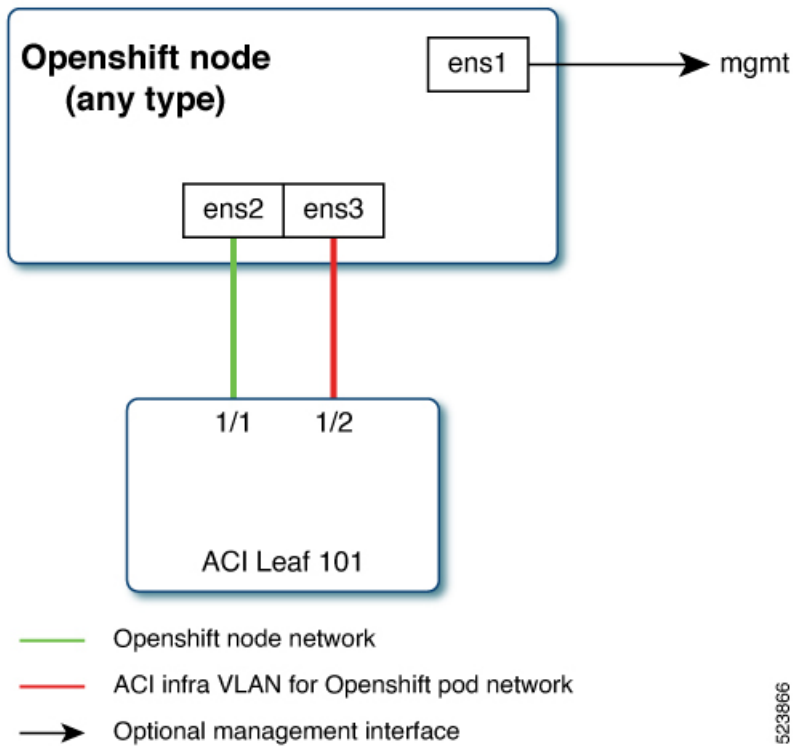
(注) このドキュメントは単独では使用できません。このドキュメントは、OpenShift クラスタのインストールを実行するためのエージェントベースのインストーラドキュメントを使用した Red Hat OpenShift 4.16 オンプレミス クラスタのインストール とともに使用する必要があります。

## ベア メタル サーバーで OpenShift 4.16 をサポートするための要件

ベア メタル ノードには少なくとも 2 つのネットワーク インターフェイスが必要です。1 つはノード ネットワーク用、2 つ目はポッド ネットワーク用です。この設計では、OpenShift ノード トラフィックを異なる Neutron ネットワーク上のポッド トラフィックから分離します。分離を実現するために使用できるオプションは 2 つあります。その結果、制御マシンとコンピューティングマシンにそれぞれ 2 つのネットワーク インターフェイスが割り当てられます。

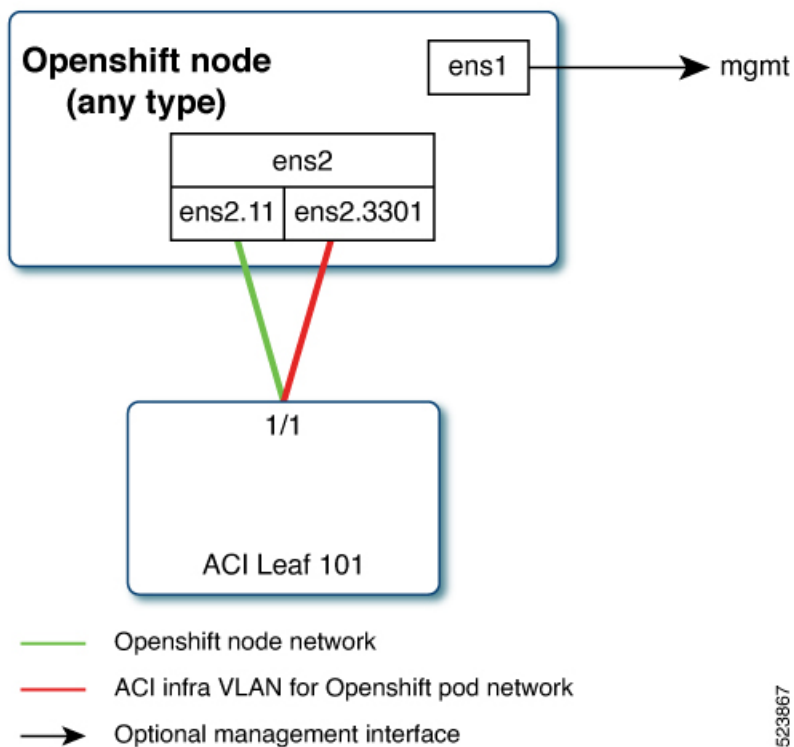
- ノード ネットワークとインフラ ネットワークに分離された物理インターフェイス
- ノード ネットワークとインフラ ネットワークの両方に対応する単一のサブインターフェイス

ノード ネットワークとインフラ ネットワークに分離された物理インターフェイス



1つのインターフェイスはノードネットワーク用で、2つ目はポッドネットワーク用です。2番目のインターフェイスも、Cisco ACI コントロールプレーン トラフィックを伝送します。VLAN タグ付きサブインターフェイスは、ポッドトラフィックと Cisco ACI コントロールプレーン トラフィックを伝送するように2番目のインターフェイスで設定されます。

ノード ネットワークとインフラ ネットワークの両方に対応する単一のサブインターフェイス



ノードネットワークとポッドネットワークは、bond0 または物理 NIC のいずれかの VLAN サブインターフェイスとして設定されています。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。

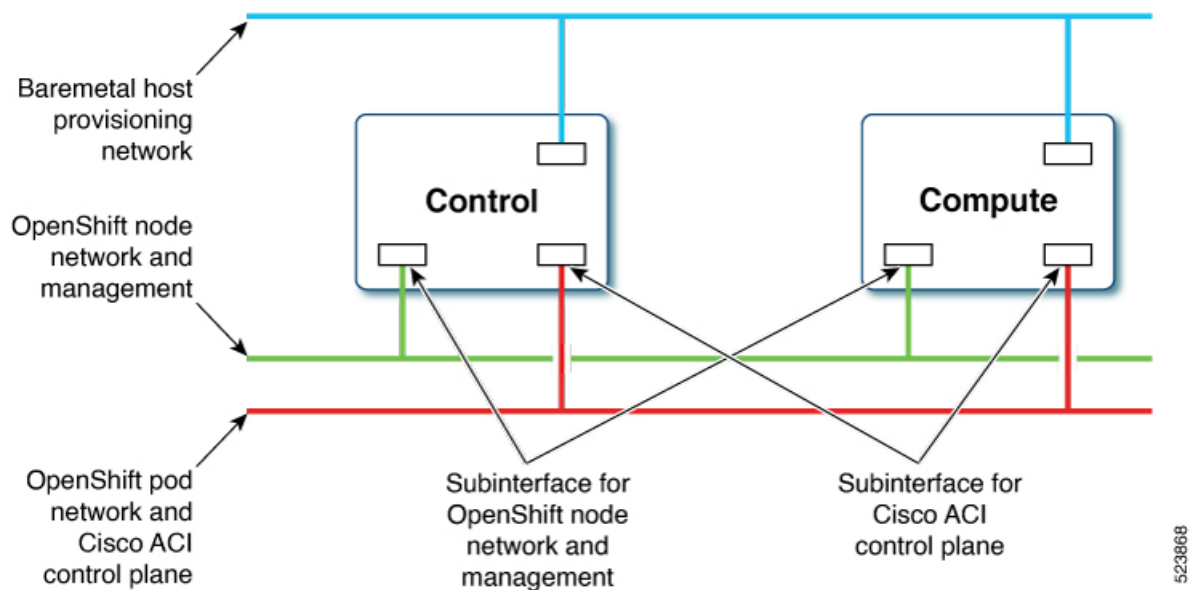
## インストール プロセス

以下のセクションでは、ACI CNI を使用して OpenShift クラスタをインストールするために必要な手順について詳しく説明します。

- [OpenShift インストーラの構成（5 ページ）](#)
- [ACI インフラと CNI の構成（8 ページ）](#)
- [OpenShift ノードのカスタム ネットワーク構成の準備（10 ページ）](#)

次の図は、インストール プロセスで使用されるさまざまなタイプのネットワークを示しています。

ベアメタルノードには少なくとも2つのネットワーク インターフェイスが必要です。1つはノードネットワーク用、もう1つはポッドネットワーク用です。この設計では、OpenShift ノードトラフィックをポッドトラフィックから分離します。3 番目のインターフェイスは、ベアメタルホストのプロビジョニングに不可欠なプライベートネットワーク用に構成されます。



## OpenShift インストーラの構成

OpenShift インストーラを設定するには、次の手順を実行します。インストールでは、3 ノードクラスタを使用します（制御にはスケジューリングが有効になります）。インストール後のノードのスケーリングについては、「ベア メタル オペレータによるエージェントベースのインストールのスケーリング」セクションを参照してください。

### 始める前に

OpenShift インストーラと OC クライアントをダウンロードします。

インストーラをダウンロードできる場所の詳細については、「エージェントベースのインストーラを使用したオンプレミス クラスタのインストール」というタイトルの *OpenShift 4.16* のドキュメントを参照してください。

### 手順

**ステップ 1** `install-config.yaml` ファイルを作成します。

```
apiVersion: v1
baseDomain: noiro.local
proxy:
  httpsProxy: <http-proxy>
  httpProxy: <https-proxy>
  noProxy: <no-proxy>
compute:
- name: worker
  replicas: 0
controlPlane:
  name: master
  replicas: 3
metadata:
  name: ocpbml
networking:
  machineNetwork:
```

```

- cidr: 192.168.1.0/24
clusterNetwork:
- cidr: 10.2.0.0/16
  hostPrefix: 23
networkType: CiscoACI
serviceNetwork:
- 172.30.0.0/16
platform:
  baremetal:
    apiVIPs:
      - 192.168.1.30
    ingressVIPs:
      - 192.168.1.29
  fips: false
pullSecret: <RH-account-pull-secret>
sshKey: <host-ssh-key>

```

**ステップ2** agent-config.yaml ファイルを作成します。

```

apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: ocpbml
rendezvousIP: 192.168.1.3
AdditionalNTPSources:
- time.cisco.com
hosts:
- hostname: ocpbml-master1
  role: master
  interfaces:
  - name: ens160
    macAddress: 00:50:56:97:16:db
  networkConfig:
    interfaces:
    - name: ens160
      mtu: 9000
      ipv4:
        enabled: false
      ipv6:
        enabled: false
    - name: node
      type: vlan
      mtu: 9000
      state: up
      vlan:
        base-iface: ens160
        id: 11
      ipv4:
        enabled: true
        address:
          - ip: 192.168.1.3
            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
    dns-resolver:
        config:
            server:
                - 192.168.1.2
    routes:
        config:
            - destination: 0.0.0.0/0
              next-hop-address: 192.168.1.1
              next-hop-interface: node
            - destination: 224.0.0.0/4
              next-hop-interface: infra
- hostname: ocpbml-master2
  role: master
  interfaces:
  - name: ens160
    macAddress: 00:50:56:97:63:de
  networkConfig:
    interfaces:
      - name: ens160
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens160
          id: 11
        ipv4:
          enabled: true
          address:
            - ip: 192.168.1.4
              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
    dns-resolver:
        config:
            server:
                - 192.168.1.2
    routes:
        config:
            - destination: 0.0.0.0/0
              next-hop-address: 192.168.1.1
              next-hop-interface: node

```

```

        - destination: 224.0.0.0/4
          next-hop-interface: infra
- hostname: ocpbml-master3
  role: master
  interfaces:
- name: ens160
  macAddress: 00:50:56:97:00:e5
networkConfig:
  interfaces:
    - name: ens160
      mtu: 9000
      ipv4:
        enabled: false
      ipv6:
        enabled: false
    - name: node
      type: vlan
      mtu: 9000
      state: up
      vlan:
        base-iface: ens160
        id: 11
      ipv4:
        enabled: true
        address:
          - ip: 192.168.1.5
            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
  dns-resolver:
    config:
      server:
        - 192.168.1.2
  routes:
    config:
      - destination: 0.0.0.0/0
        next-hop-address: 192.168.1.1
        next-hop-interface: node
      - destination: 224.0.0.0/4
        next-hop-interface: infra

```

---

## 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。



## 手順

### ACI 構成サンプル

```
# Configuration for ACI Fabric
#
aci_config:
  system_id: openupi                                # Every opflex cluster on the same fabric must have a distinct
  ID                                                ID
  apic_hosts:                                       # List of APIC hosts to connect to for APIC API access
    - <APIC-IP>
  apic_login:
    username: <username>
    password: <password>
  vmm_domain:                                     # Kubernetes VMM domain configuration
    encap_type: vxlan                             # Encap mode: vxlan or vlan
    mcast_range:                                  # Every vxlan VMM on the same fabric must use a distinct range
      start: 225.115.1.1
      end: 225.115.255.255
  # The following resources must already exist on the APIC,
  # this is a reference to use them
  aep: <AAEP_NAME>                                # The attachment profile for ports/VPCs connected to this cluster
  vrf:                                             # VRF used to create all subnets used by this Kubernetes cluster
    name: <VRF_NAME>                               # This should exist, the provisioning tool does not create it
    tenant: <TENANT_WITH_VRF_DEFINITION>           # This can be tenant for this cluster (system-id)
  or common
  l3out:                                           # L3out to use for this kubernetes cluster (in the VRF above)
    name:<L3OUT_NAME>                               # This is used to provision external service IPs/LB
    external_networks:
      <EXTERNAL_EPG_NAME>                         # This should also exist, the provisioning tool does not create it
  agent based installer:
    enable: true
  # Networks used by Kubernetes
  #
net_config:
  node_subnet: 192.168.1.1/24                     # Subnet to use for nodes
  pod_subnet: 10.2.0.1/16                         # Subnet to use for Kubernetes Pods
  extern_dynamic: 10.3.0.1/16                     # Subnet to use for dynamically allocated external services
  extern_static: 10.4.0.1/16                     # Subnet to use for statically allocated external services
  node_svc_subnet: 10.5.0.1/16                   # Subnet to use for service graph
  kubeapi_vlan: 11                               # The VLAN used by the internal physdom for nodes
  service_vlan: 21                               # The VLAN used for external LoadBalancer services
  infra_vlan: 3301
```

#### (注)

ユーザーがプロビジョニングした DNS 内の \*.apps.<cluster\_name>.<base\_domain> レコードは、install-config.yaml ファイルの ingressVIP で使用されているものと同じ IP アドレスを参照する必要があります。

上記のサンプル acc-provision 入力ファイルを要件に応じてカスタマイズします。次に、[こちら](#) 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-baremetal
-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。

## 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。すべてのマスター ノードとワーカー ノードに追加の VLAN が必要です。

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.yaml` の作成の詳細については、「*OpenShift* インストーラの構成」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。関連する詳細については、「要件」の項で説明しています。

次の YAML スニペットは、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、展開を合理化するための重要な詳細が含まれています。

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: ocpbml
rendezvousIP: 192.168.1.3. -> A
AdditionalNTPSources:
  - time.cisco.com
hosts: -> B
  - hostname: ocpbml-master1 -> C
    role: master
    interfaces:
      - name: ens160
        macAddress: 00:50:56:97:16:db
    networkConfig: -> D
      interfaces:
        - name: ens160
          mtu: 9000
          ipv4:
            enabled: false
          ipv6:
            enabled: false
        - name: node
          type: vlan
          mtu: 9000
```

```

state: up
vlan:
  base-iface: ens160
  id: 11
ipv4:
  enabled: true
  address:
    - ip: 192.168.1.3
      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
dns-resolver:
  config:
    server:
      - 192.168.1.2
routes:
  config:
    - destination: 0.0.0.0/0
      next-hop-address: 192.168.1.1
      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。

**ステップ3** openshift ディレクトリを作成します。

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

**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。

---

## 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、『*Ingress Operator in OpenShift Container Platform Red Hat*』ガイドの「*Configuring the Default Ingress Controller for your Cluster to be Internal*」セクションを参照してください。

## ベアメタルオペレータによるエージェントベースのインストールのスケールリング

この手順を使用して、クラスタ内のワーカーまたはスケールノードを追加します。

手順

---

- ステップ1** ベースボード管理コントローラ（BMC）を使用してベアメタルノードの電源をオフにし、オフになっていることを確認します。
- ステップ2** ベアメタルノードの設定ファイルを適用し、次の例のいずれかの `bmh.yaml` ファイルを使用して、環境に合わせて `YAML` の値を置き換えます。

```
apiVersion: metal3.io/v1alpha1
kind: Provisioning
metadata:
  finalizers:
  - provisioning.metal3.io
  name: provisioning-configuration
spec:
  preProvisioningOSDownloadURLs: {}
  provisioningMacAddresses:
  - <control-node01 mac address>
  - <control-node02 mac address>
  - <control-node03 mac address>
  provisioningNetwork: Managed
  provisioningIP: 192.168.254.30
  provisioningNetworkCIDR: 192.168.254.0/24
  provisioningDHCPRange: 192.168.254.3,192.168.254.10
  provisioningInterface: ens70s0f1
---
apiVersion: v1
kind: Secret
metadata:
  name: bmc-credentials
  namespace: openshift-machine-api
data:
  username: <base64_of_uid>
  password: <base64_of_pwd>
---
apiVersion: v1
kind: Secret
metadata:
  name: bm-compute-0-netconfig
  namespace: openshift-machine-api
type: Opaque
stringData:
  nmstate: |
    interfaces:
      - name: ens160
        mtu: 9000
        ipv4:
          enabled: false
        ipv6:
          enabled: false
      - name: node
        type: vlan
        mtu: 9000
        state: up
        vlan:
          base-iface: ens160
          id: 11
        ipv4:
          enabled: true
          address:
            - ip: 192.168.1.6
              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
  dns-resolver:
    config:
      server:
        - 192.168.1.2
  routes:
    config:
      - destination: 0.0.0.0/0
        next-hop-address: 192.168.1.1
        next-hop-interface: node
      - destination: 224.0.0.0/4
        next-hop-interface: infra
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: compute-0
  namespace: openshift-machine-api
spec:
  automatedCleaningMode: metadata
  online: true
  bootMACAddress: <nic1_mac_address>
  bmc:
    address: <protocol>://<bmc_url>
    credentialsName: bmc-credentials
    disableCertificateVerification: True
  preprovisioningNetworkDataName: bm-compute-0-netconfig

```

(注)

複数のワーカーノードを有効にするには、ノードごとに個別の **netconfig** シークレットを生成する必要があります。さらに、**BaremetalHost** オブジェクトを削除すると、関連付けられたシークレットも削除されることに注意してください。そのため、複数の **BaremetalHost** オブジェクトを利用する場合は、削除されていない **BaremetalHost** インスタンスのログイン情報のシークレットが保持され、適切な機能が維持されるようにします。

**ステップ 3** 作成されたそれぞれのオブジェクトを確認します（必要なコマンドが各オブジェクトに示されています）。

- **Provisioning Network** : PXE ブートに使用されるプライベート ネットワーク。  
`. oc describe provisioning provisioning-configuration`
- **Secret bmc-credentials** : bmc アクセス用のログイン情報。  
`. oc describe secret -n openshift-machine-api bmc-credentials`
- **Secret bm-compute-0-netconfig** : ワーカー ノードのカスタム ネットワーク設定。  
`. oc describe secret -n openshift-machine-api bm-compute-0-netconfig`
- **BareMetalHost compute-0** : ベアメタル ノードを管理する構成。

```
. oc describe baremetalhost compute-0 -n openshift-machine-api
```

**ステップ 4** 使用可能なベア メタル ホストの数に一致するように、レプリカ数をスケールアップします。

```
oc scale machineset -n openshift-machine-api <worker-machineset> --replicas=1
```

---

## 次のタスク

クラスタのインストール進行状況の追跡と確認を続行します。 *Redhat OpenShift 4.16* のドキュメントを参照してください（本章で前述しています）。

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS REFERENCED IN THIS DOCUMENTATION ARE SUBJECT TO CHANGE WITHOUT NOTICE. EXCEPT AS MAY OTHERWISE BE AGREED BY CISCO IN WRITING, ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS DOCUMENTATION ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.

The Cisco End User License Agreement and any supplemental license terms govern your use of any Cisco software, including this product documentation, and are located at: <https://www.cisco.com/c/en/us/about/legal/cloud-and-software/software-terms.html>. Cisco product warranty information is available at <https://www.cisco.com/c/en/us/products/warranty-listing.html>. US Federal Communications Commission Notices are found here <https://www.cisco.com/c/en/us/products/us-fcc-notice.html>.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any products and features described herein as in development or available at a future date remain in varying stages of development and will be offered on a when-and if-available basis. Any such product or feature roadmaps are subject to change at the sole discretion of Cisco and Cisco will have no liability for delay in the delivery or failure to deliver any products or feature roadmap items that may be set forth in this document.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

The documentation set for this product strives to use bias-free language. For the purposes of this documentation set, bias-free is defined as language that does not imply discrimination based on age, disability, gender, racial identity, ethnic identity, sexual orientation, socioeconomic status, and intersectionality. Exceptions may be present in the documentation due to language that is hardcoded in the user interfaces of the product software, language used based on RFP documentation, or language that is used by a referenced third-party product.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: <https://www.cisco.com/c/en/us/about/legal/trademarks.html>. Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1721R)

© 2025 Cisco Systems, Inc. All rights reserved.



【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/](http://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。

©2008 Cisco Systems, Inc. All rights reserved.

Cisco, Cisco Systems, およびCisco Systemsロゴは、Cisco Systems, Inc.またはその関連会社の米国およびその他の一定の国における登録商標または商標です。

本書類またはウェブサイトに掲載されているその他の商標はそれぞれの権利者の財産です。

「パートナー」または「partner」という用語の使用はCiscoと他社との間のパートナーシップ関係を意味するものではありません。(0809R)

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。