OpenShift Container Platform 4.7

Windows Container Support for OpenShift

Red Hat OpenShift for Windows Containers ガイド

Last Updated: 2021-10-10

OpenShift Container Platform 4.7 Windows Container Support for OpenShift

Red Hat OpenShift for Windows Containers ガイド

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律上の通知

概要

Red Hat OpenShift for Windows コンテナーは、OpenShift Container Platform で Microsoft Windows Server コンテナーを実行するための組み込みサポートを提供します。本書では、すべての詳細情報を提供します。

目次

第1章 WINDOWS CONTAINER SUPPORT FOR RED HAT OPENSHIFT リリースノート

1.1. WINDOWS CONTAINER SUPPORT FOR RED HAT OPENSHIFT について

Windows Container Support for Red Hat OpenShift は、OpenShift Container Platform クラスターで Windows コンピュートノードを実行する機能を提供します。これは、Red Hat Windows Machine Config Operator (WMCO) を使用して Windows ノードをインストールし、管理することで実行できます。Windows ノードが利用可能になると、Windows コンテナーワークロードを OpenShift Container Platform で実行できます。

Red Hat OpenShift for Windows Containers のリースノートでは、OpenShift Container Platform にすべての Windows コンテナーワークロード機能を提供する WMCO の開発の進展を追跡できます。

1.2. サポート

Red Hat WMCO のサポートを受けるには、サブスクリプションが必要です。実稼働クラスターへの Windows コンテナーワークロードのデプロイは、サブスクリプションなしではサポートされません。サブスクリプションをお持ちでない場合は、正式なサポートのないディストリビューションであるコミュニティー版の WMCO 使用できます。Red Hat カスタマーポータル でサポートをリクエストします。

1.3. WINDOWS MACHINE CONFIG OPERATOR の前提条件

以下では、Windows Machine Config Operator のサポート対象のクラウドプロバイダーバージョン、Windows Server バージョン、およびネットワーク設定について詳しく説明します。対象のプラットフォームのみに関連する情報については、vSphere のドキュメントを参照してください。

1.3.1. OpenShift Container Platform および WMCO バージョンをベースとするサポート対象のクラウドプロバイダー

クラウドプロバイダー	サポート対象の Red Hat OpenShift Container Platform バージョン	サポート対象の WMCO バージョン
Amazon Web Services (AWS)	4.6+	WMCO 1.0+
Microsoft Azure	4.6+	WMCO 1.0+
VMware vSphere	4.7+	WMCO 2.0+

1.3.2. サポート対象の Windows Server バージョン

以下の表は、適用可能なクラウドプロバイダーを基にした、サポート対象の Windows Server バージョン の一覧です。リストに含まれない Windows Server バージョンはサポートされず、エラーを引き起こします。これらのエラーを回避するには、使用中のクラウドプロバイダーに合わせて適切なバージョンのみを使用します。

クラウドプロバイダー	サポート対象の Windows Server バージョン
Amazon Web Services (AWS)	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
Microsoft Azure	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
VMware vSphere	Windows Server Semi-Annual Channel(SAC): Windows Server 2004 および 20H2

1.3.3. サポート対象のネットワーク

サポート対象のネットワーク設定は、OVN-Kubernetes を使用したハイブリッドネットワークのみです。この機能の詳細は、以下の追加リソースを参照してください。以下の表は、クラウドプロバイダーで使用するネットワーク設定の種類と Windows Server バージョンの概要を示しています。クラスターのインストール時にネットワーク設定を指定する必要があります。OpenShift SDN ネットワークは OpenShift Container Platform クラスターのデフォルトのネットワークであることに注意してください。ただし、OpenShift SDN は WMCO ではサポートされません。

表1.1 クラウドプロバイダーのネットワークのサポート

クラウドプロバイダー	サポート対象のネットワーク
Amazon Web Services (AWS)	OVN-Kubernetes を使用したハイブリッドネットワーク
Microsoft Azure	OVN-Kubernetes を使用したハイブリッドネットワーク
VMware vSphere	カスタム VXLAN ポートを備えた OVN-Kubernetes でのハイブリッドネットワーク

表1.2 ハイブリッド OVN-Kubernetes Windows Server のサポート

OVN-Kubernetes を使用したハイブリッドネットワーク	サポート対象の Windows Server バージョン
デフォルトの VXLAN ポート	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
カスタム VXLAN ポート	Windows Server Semi-Annual Channel(SAC): Windows Server 2004 および 20H2

1.3.4. サポート対象のインストール方法

サポート対象のインストール方法は、インストーラーでプロビジョニングされるインフラストラクチャーのインストールのみです。これは、サポート対象のクラウドプロバイダーすべてで、一貫しています。ユーザーがプロビジョニングするインフラストラクチャーのインストールはサポート対象外で

す。

1.4. RED HAT WINDOWS MACHINE CONFIG OPERATOR 2.0.3 リリースノート

発行日: 2021-07-28

WMCO 2.0.3 がバグ修正と共に利用できるようになりました。WMCO のコンポーネントは RHBA-2021:2926 でリリースされています。

1.4.1. バグ修正

- この WMCO リリースでは、ユーザーが WMCO 3.0.0 にアップグレードできないバグが修正されました。ユーザーは、WMCO 3.0.0 のみをサポートする OpenShift Container Platform 4.8 にアップグレードする前に、WMCO 2.0.3 にアップグレードする必要があります。(BZ#1985349)

1.5. RED HAT WINDOWS MACHINE CONFIG OPERATOR 2.0.2 リリースノート

発行日： 2021-07-08

WMCO 2.0.2 がバグ修正で利用可能になりました。WMCO のコンポーネントは RHBA-2021:2671 でリリースされています。

重要

2.0.3 より前の WMCO バージョンを実行しているユーザーは、最初に WMCO 2.0.3 にアップグレードしてから WMCO 3.0.0 にアップグレードする必要があります。(BZ#1983153)

1.5.1. バグ修正

- OpenShift Container Platform 4.8 では、デフォルトで **BoundServiceAccountTokenVolume** オプションを有効にします。このオプションは、Projected ボリュームをすべての Pod に割り当てます。さらに、OpenShift Container Platform 4.8 は **RunAsUser** オプションを **SecurityContext** に追加します。この組み合わせにより、Windows Pod が **ContainerCreating** ステータスのままになります。この問題を回避するには、クラスターを OpenShift Container Platform 4.8 にアップグレードする前に、WMCO 2.0.2 にアップグレードする必要があります。(BZ#1975553)

1.6. RED HAT WINDOWS MACHINE CONFIG OPERATOR 2.0.1 リリースノート

発行日： 2021-06-23

WMCO 2.0.1 がバグ修正で利用できるようになりました。WMCO のコンポーネントは RHSA-2021:2130 でリリースされています。

1.6.1. 新機能および改善点

今回のリリースでは、以下の新機能および改善点が追加されました。

1.6.1.1. イメージプルのタイムアウト期間の増加

イメージのプルのタイムアウトが 30 分に増えました。

1.6.2. バグ修正

- 以前のバージョンでは、AWS インストールで Windows の kube-proxy コンポーネントを使用する場合、LoadBalancer サービスを作成すると、パケットは誤ってルーティングされ、意図しない宛先に到達しました。今回のリリースより、パケットは意図しない宛先にルーティングされなくなりました。(BZ#1946538)

- 以前のバージョンでは、Windows ノードは Telemetry モニタリングを使用して一部の主要なノードレベルのメトリクスを報告しませんでした。**windows_exporter** は、node _* と同等の **node_exporter ではなく、さまざまなメトリクスを windows _* として報告します**。Telemetry の結果は予想されるすべてのメトリクスに対応するようになりました。(BZ#1955319)

- 以前のバージョンでは、WMCO が Windows インスタンスを設定している場合、hybrid-overlay または kube-proxy コンポーネントが失敗すると、ノードは自身を **Ready として報告する可能性がありました**。今回のリリースよりエラーが検出され、ノードは自身を **NotReady** として報告するようになりました。(BZ#1956412)

- 以前のバージョンでは、Windows Pod の実行を開始した後にロードバランサーを作成する場合、kube-proxy サービスはロードバランサーの作成後に予期せずに終了していました。kube-proxy サービスは、ロードバランサーサービスを再作成する際にクラッシュしなくなりました。(BZ#1939968)

1.6.3. RHSA-2021:2130 - OpenShift Container Platform のセキュリティー更新における Windows Container のサポート

以前のバグ修正 (BZ#1946538) の一環として、Windows kube-proxy の更新が Red Hat Windows Machine Config Operator 2.0.1 で利用可能になりました。更新の詳細については、RHSA-2021:2130 アドバイザリーに記載されています。

1.7. RED HAT WINDOWS MACHINE CONFIG OPERATOR 2.0.0 リリースノート

本 WMCO リリースノートは、Windows コンピュートノードを OpenShift Container Platform クラスターで実行するためのバグ修正および拡張機能を提供します。WMCO 2.0.0 のコンポーネントは RHBA-2021:0440 でリリースされています。

重要

Windows コンテナーワークロードの実行は、ネットワークが制限された環境または非接続環境のクラスターではサポートされません。

WMCO のバージョン 2.x は OpenShift Container Platform 4.7 とのみ互換性があります。

1.7.1. 新機能および改善点

今回のリリースでは、以下の新機能および改善点が追加されました。

1.7.1.1. VMware vSphere で実行されるクラスターのサポート

VMware vSphere バージョン 6.5、6.7、または 7.0 にインストールされたクラスターで Windows ノードを実行できるようになりました。vSphere で Windows の **MachineSet** オブジェクトを作成して、Windows Server コンピュートノードをホストできます。詳細は、vSphere での Windows の **MachineSet** オブジェクトの作成について参照してください。

1.7.1.2. Windows ノードのモニタリングの強化

Windows ノードは、Web コンソールで提供されるほとんどのモニタリング機能と完全に統合されました。ただし、本リリースでは Windows ノードで実行されている Pod のワークロードグラフを表示することはできません。

1.8. 既知の問題

- これらの Pod に関連付けられた Projected ボリュームで **RunAsUserName** が設定された Windows Pod を作成すると、展開されたエンティティーのファイル所有者パーミッションは無視され、所有者のパーミッションが誤って設定されます。

- Windows ノードについては、Web コンソールで利用可能なファイルシステムのグラフは表示されません。この原因は、ファイルシステムのクエリーの変更にあります。これは、WMCO の今後のリリースで修正されます。(BZ#1930347)

- 現時点で、WMCO で使用される Prometheus windows_exporter は HTTP 経由でメトリクスを収集するため、安全でないと見なされます。信頼できるユーザーのみがエンドポイントからメトリクスを取得できるようにする必要があります。window_exporter 機能では、最近 HTTPS 設定のサポートが追加されましたが、この設定は WMCO については実装されていません。WMCO での HTTPS 設定のサポートは今後のリリースで追加される予定です。

第2章 WINDOWS コンテナーワークロードについて

Windows Container Support for Red Hat OpenShift は Microsoft Windows Server コンテナーを OpenShift Container Platform で実行するための組み込みサポートを提供します。Linux と Windows ワークロードの組み合わせを使用して異種環境を管理する場合、OpenShift Container Platform では、Windows Server コンテナーで実行されている Windows ワークロードをデプロイできますが、Red Hat Enterprise Linux CoreOS (RHCOS) または Red Hat Enterprise Linux (RHEL) でホストされる従来の Linux ワークロードも提供できます。

注記

Windows ノードを持つクラスターのマルチテナンシーはサポートされません。マルチテナントの悪意のある使用により、すべての Kubernetes 環境でのセキュリティー上のリスクが導入されます。Pod セキュリティーポリシー、またはノードのより詳細なロールベースアクセス制御 (RBAC) などの追加のセキュリティー機能により、悪用がより困難になります。ただし、悪意のあるマルチテナントワークロードの実行を選択する場合、ハイパーバイザーは使用する必要のある唯一のセキュリティーオプションになります。Kubernetes のセキュリティードメインは、個別のノードではなく、クラスター全体に対応します。これらのタイプの悪意のあるマルチテナントワークロードには、物理的に分離されたクラスターを使用する必要があります。

Windows Server コンテナーは共有カーネルを使用してリソース分離を行いますが、悪意のあるマルチテナンシーのシナリオで使用することは意図されていません。悪意のあるマルチテナンシーを使用するシナリオの場合、テナントを確実に分離するために Hyper-V 分離コンテナーを使用する必要があります。

2.1. WINDOWS MACHINE CONFIG OPERATOR の前提条件

以下では、Windows Machine Config Operator のサポート対象のクラウドプロバイダーバージョン、Windows Server バージョン、およびネットワーク設定について詳しく説明します。対象のプラットフォームのみに関連する情報については、vSphere のドキュメントを参照してください。

2.1.1. OpenShift Container Platform および WMCO バージョンをベースとするサポート対象のクラウドプロバイダー

クラウドプロバイダー	サポート対象の Red Hat OpenShift Container Platform バージョン	サポート対象の WMCO バージョン
Amazon Web Services (AWS)	4.6+	WMCO 1.0+
Microsoft Azure	4.6+	WMCO 1.0+
VMware vSphere	4.7+	WMCO 2.0+

2.1.2. サポート対象の Windows Server バージョン

以下の表は、適用可能なクラウドプロバイダーを基にした、サポート対象の Windows Server バージョン の一覧です。リストに含まれない Windows Server バージョンはサポートされず、エラーを引き起こします。これらのエラーを回避するには、使用中のクラウドプロバイダーに合わせて適切なバージョンのみを使用します。

クラウドプロバイダー	サポート対象の Windows Server バージョン
Amazon Web Services (AWS)	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
Microsoft Azure	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
VMware vSphere	Windows Server Semi-Annual Channel(SAC): Windows Server 2004 および 20H2

2.1.3. サポート対象のネットワーク

サポート対象のネットワーク設定は、OVN-Kubernetes を使用したハイブリッドネットワークのみです。この機能の詳細は、以下の追加リソースを参照してください。以下の表は、クラウドプロバイダーで使用するネットワーク設定の種類と Windows Server バージョンの概要を示しています。クラスターのインストール時にネットワーク設定を指定する必要があります。OpenShift SDN ネットワークは OpenShift Container Platform クラスターのデフォルトのネットワークであることに注意してください。ただし、OpenShift SDN は WMCO ではサポートされません。

表2.1 クラウドプロバイダーのネットワークのサポート

クラウドプロバイダー	サポート対象のネットワーク
Amazon Web Services (AWS)	OVN-Kubernetes を使用したハイブリッドネットワーク
Microsoft Azure	OVN-Kubernetes を使用したハイブリッドネットワーク
VMware vSphere	カスタム VXLAN ポートを備えた OVN-Kubernetes でのハイブリッドネットワーク

表2.2 ハイブリッド OVN-Kubernetes Windows Server のサポート

OVN-Kubernetes を使用したハイブリッドネットワーク	サポート対象の Windows Server バージョン
デフォルトの VXLAN ポート	Windows Server 長期サービスチャネル (Long-Term Servicing Channel、LTSC): Windows Server 2019
カスタム VXLAN ポート	Windows Server Semi-Annual Channel(SAC): Windows Server 2004 および 20H2

2.1.4. サポート対象のインストール方法

サポート対象のインストール方法は、インストーラーでプロビジョニングされるインフラストラクチャーのインストールのみです。これは、サポート対象のクラウドプロバイダーすべてで、一貫しています。ユーザーがプロビジョニングするインフラストラクチャーのインストールはサポート対象外で

す。

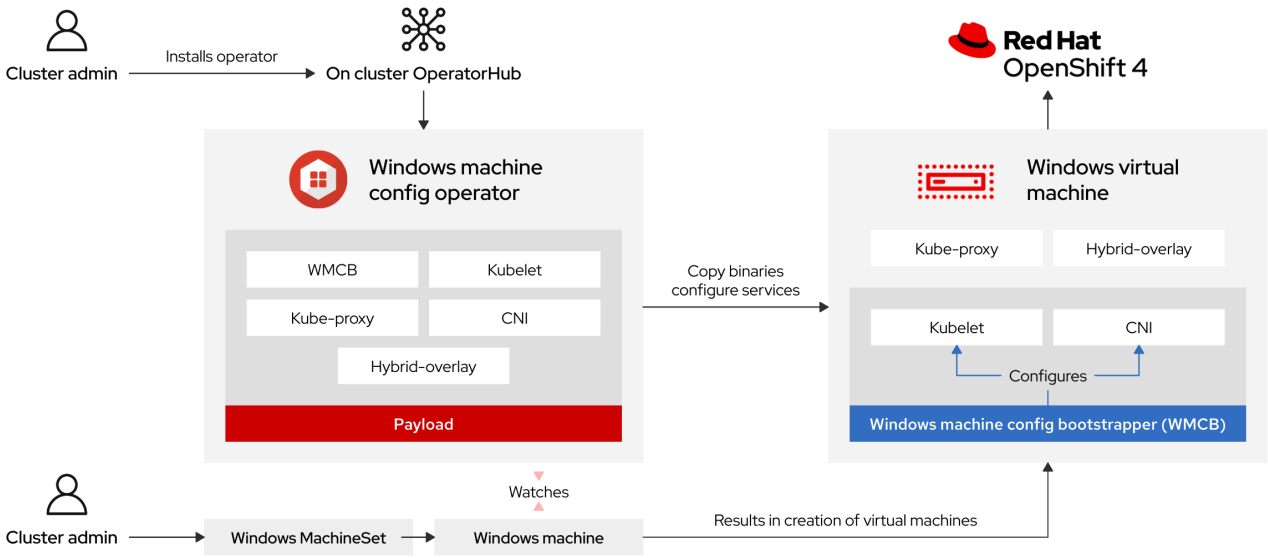
関連情報

- 「OVN-Kubernetes を使用したハイブリッドネットワークの設定」を参照してください。

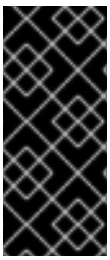
2.2. WINDOWS ワークロード管理

クラスターで Windows ワークロードを実行するには、まず Windows Machine Config Operator (WMCO) をインストールする必要があります。WMCO は Linux ベースのコントロールプレーンおよびコンピュートノードで実行される Linux ベースの Operator です。WMCO は、クラスター上で Windows ワークロードをデプロイし、管理するプロセスをオーケストレーションします。

図2.1 WMCO の設計



Windows ワークロードをデプロイする前に、Windows コンピュートノードを作成し、これをクラスターに参加させる必要があります。Windows ノードは、クラスター内の Windows ワークロードをホストし、他の Linux ベースのコンピュートノードと共に実行できます。Windows Server コンピュートマシンをホストする Windows マシンセットを作成して、Windows コンピュートノードを作成することができます。Docker 形式のコンテナーランタイムアドオンが有効にされた Windows OS イメージを指定する Windows 固有のラベルをマシンセットに適用する必要があります。

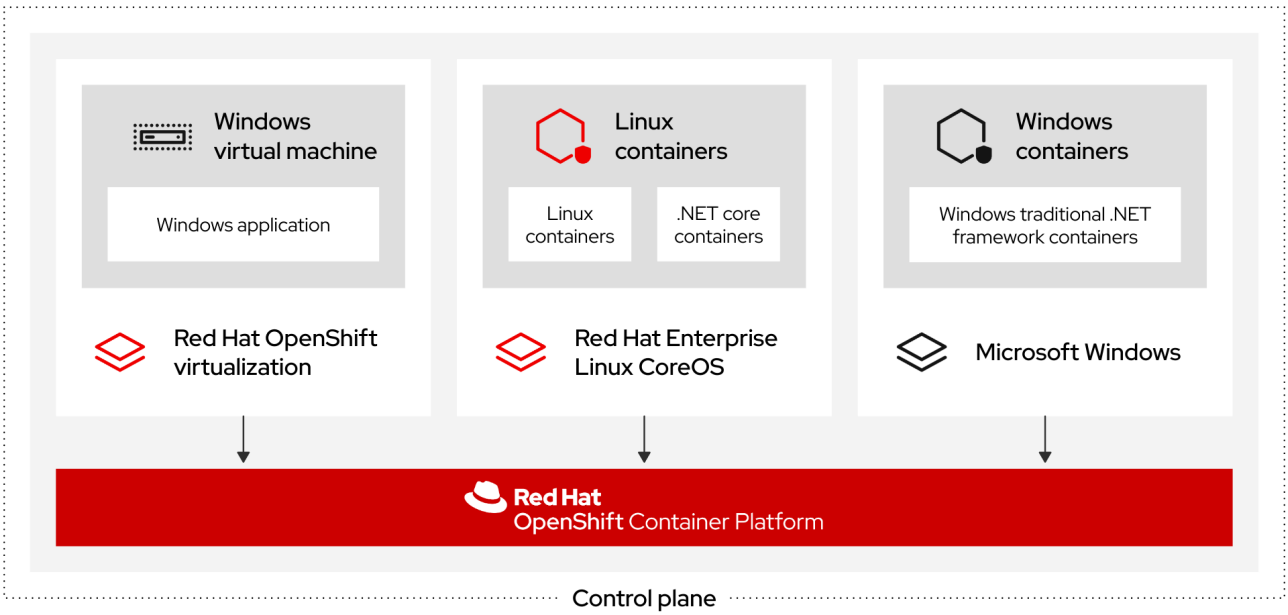


重要

現時点で、Docker 形式のコンテナーランタイムは Windows ノードで使用されます。Kubernetes では、コンテナーランタイムとしての Docker を非推奨としています。詳細は、Kubernetes ドキュメントの Docker の非推奨について参照してください。Containerd は、今後の Kubernetes リリースで Windows ノードについてサポートされる新しいコンテナーランタイムになります。

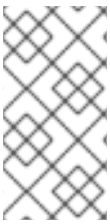
WMCO は Windows ラベルの付いたマシンを監視します。Windows マシンセットを検出し、そのそれぞれのマシンがプロビジョニングされると、WMCO は基礎となる Windows 仮想マシン (VM) を設定し、それがコンピュートノードとしてクラスターに参加できるようにします。

図2.2 Windows および Linux ワークロードの組み合わせ



WMCO は、Windows インスタンスとの対話に使用されるプライベートキーが含まれる namespace に事前に決定されたシークレットがあることを想定します。WMCO は起動時にこのシークレットの有無を確認し、作成した Windows **MachineSet** オブジェクトで参照する必要のあるユーザーデータのシークレットを作成します。次に WMCO は、プライベートキーに対応するパブリックキーでユーザーデータのシークレットを設定します。このデータが適用されると、クラスターは SSH 接続を使用して Windows 仮想マシンに接続できます。

クラスターが Windows 仮想マシンとの接続を確立した後に、Linux ベースのノードの場合と同様の手法を使用して Windows ノードを管理できます。



注記

OpenShift Container Platform Web コンソールは、Linux ノードで利用可能な機能と同じモニタリング機能のほとんどを Windows ノードについて提供します。ただし、現時点で Windows ノードで実行されている Pod のワークロードグラフを監視する機能は利用できません。

Windows ワークロードの Windows ノードへのスケジュールは、テイント、容認、ノードセレクターなどの一般的な Pod スケジュールの手法を使用して実行できますが、**RuntimeClass** オブジェクトを使用して Windows ワークロードを Linux ワークロードおよび他の Windows 版のワークロードから区別できます。

2.3. WINDOWS ノードサービス

以下の Windows 固有のサービスは、各 Windows ノードにインストールされます。

サービス	説明
kubelet	Windows ノードを登録し、そのステータスを管理します。
Container Network Interface (CNI) プラグイン	Windows ノードのネットワークを公開します。

サービス	説明
Windows Machine Config Bootstrapper (WMCB)	kubelet および CNI プラグインを設定します。
hybrid-overlay	OpenShift Container Platform Host Network Service (HNS) を作成します。
kube-proxy	外部との通信を許可するノードでネットワークルールを維持します。

第3章 WINDOWS コンテナーワークロードの有効化

Windows ワークロードをクラスターに追加する前に、OpenShift Container Platform OperatorHub で利用可能な Windows Machine Config Operator (WMCO) をインストールする必要があります。WMCO は、クラスター上で Windows ワークロードをデプロイし、管理するプロセスをオーケストレーションします。

前提条件

- **cluster-admin** パーミッションを持つアカウントを使用して OpenShift Container Platform クラスターにアクセスできること。
- OpenShift CLI (**oc**) がインストールされている。
- インストーラーでプロビジョニングされるインフラストラクチャーを使用した vSphere にクラスターをインストールしている。ユーザーによってプロビジョニングされるインフラストラクチャーでインストールされるクラスターは、Windows コンテナーのワークロードではサポートされません。
- クラスターに OVN-Kubernetes を使用してハイブリッドネットワークを設定している。これは、クラスターのインストール時に完了する必要があります。詳細は、「ハイブリッドネットワークの設定」を参照してください。
- OpenShift Container Platform クラスター (バージョン 4.6.8 以降) を実行している。

関連情報

- Windows Machine Config Operator の包括的な前提条件については、「Windows コンテナーワークロードについて 」を参照してください。

3.1. WINDOWS MACHINE CONFIG OPERATOR のインストール

Web コンソールまたは OpenShift CLI (**oc**) のいずれかを使用して Windows Machine Config Operator をインストールできます。

3.1.1. Web コンソールを使用した Windows Machine Config Operator のインストール

OpenShift Container Platform Web コンソールを使って Windows Machine Config Operator (WMCO) をインストールすることができます。

手順

1. OpenShift Container Platform Web コンソールの **Administrator** パースペクティブから、**Operators → OperatorHub** ページに移動します。
2. **Filter by keyword** ボックスを使用して、カタログで **Windows Machine Config Operator** を検索します。**Windows Machine Config Operator** タイルをクリックします。
3. Operator についての情報を確認してから、**Install** をクリックします。
4. **Install Operator** ページで以下を行います。
    a. **Update Channel** として **stable** チャネルを選択します。**stable** チャネルを使用すると、WMCO の最新の安定したリリースをインストールできます。

b. WMCO は単一の namespace でのみ利用できるようにする必要があるため、**インストールモード** は事前に設定されます。

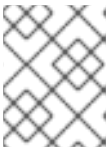
c. WMCO の **Installed Namespace** を選択します。デフォルトの Operator の推奨される namespace は **openshift-windows-machine-config-operator** です。

d. **Enable Operator recommended cluster monitoring on the Namespace**チェックボックスをクリックし、WMCO についてクラスターのモニタリングを有効にします。

e. **Approval Strategy** を選択します。

- **Automatic** ストラテジーにより、Operator Lifecycle Manager (OLM) は新規バージョンが利用可能になると Operator を自動的に更新できます。
- **Manual** ストラテジーには、Operator の更新を承認するための適切な認証情報を持つユーザーが必要です。

1. **Install** をクリックします。WMCO が **Installed Operators** ページに一覧表示されます。



注記

WMCO は、**openshift-windows-machine-config-operator** などのように、定義した namespace に自動的にインストールされます。

2. **Status** に **Succeeded** が表示されていることを検証し、WMCO のインストールが正常に行われたことを確認します。

3.1.2. CLI を使用した Windows Machine Config Operator のインストール

OpenShift CLI (**oc**) を使用して、Windows Machine Config Operator (WMCO) をインストールできます。

手順

1. WMCO の namespace を作成します。

a. WMCO の **Namespace** オブジェクト YAML ファイルを作成します。例: **wmco-namespace.yaml**

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-windows-machine-config-operator 1
  labels:
    openshift.io/cluster-monitoring: "true" 2
```

1 WMCO を **openshift-windows-machine-config-operator** namespace にデプロイすることが推奨されます。

2 このラベルは、WMCO についてクラスターモニタリングを有効にするために必要です。

b. namespace を作成します。

```
$ oc create -f <file-name>.yaml
```

以下は例になります。

```
$ oc create -f wmco-namespace.yaml
```

2. WMCO の Operator グループを作成します。

a. **OperatorGroup** オブジェクト YAML ファイルを作成します。例: **wmco-og.yaml**

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: windows-machine-config-operator
  namespace: openshift-windows-machine-config-operator
spec:
  targetNamespaces:
  - openshift-windows-machine-config-operator
```

b. Operator グループを作成します。

```
$ oc create -f <file-name>.yaml
```

以下は例になります。

```
$ oc create -f wmco-og.yaml
```

3. namespace を WMCO にサブスクライブします。

a. **Subscription** オブジェクト YAML ファイルを作成します。例: **wmco-sub.yaml**

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: windows-machine-config-operator
  namespace: openshift-windows-machine-config-operator
spec:
  channel: "stable" (1)
  installPlanApproval: "Automatic" (2)
  name: "windows-machine-config-operator"
  source: "redhat-operators" (3)
  sourceNamespace: "openshift-marketplace" (4)
```

(1) **stable** をチャネルとして指定します。

(2) 承認ストラテジーを設定します。**Automatic** または **Manual** を設定できます。

(3) **windows-machine-config-operator** パッケージマニフェストが含まれる、**redhat-operators** カタログソースを指定します。OpenShift Container Platform が、非接続クラスターとも呼ばれる制限されたネットワークにインストールされている場合、Operator LifeCycle Manager (OLM) の設定時に作成した **CatalogSource** オブジェクトの名前を指定します。

4 カタログソースの namespace。デフォルトの OperatorHub カタログソースには **openshift-marketplace** を使用します。

b. サブスクリプションを作成します。

```
$ oc create -f <file-name>.yaml
```

以下は例になります。

```
$ oc create -f wmco-sub.yaml
```

WMCO が **openshift-windows-machine-config-operator** にインストールされるようになりました。

4. WMCO インストールを確認します。

```
$ oc get csv -n openshift-windows-machine-config-operator
```

出力例

```
NAME                                 DISPLAY                          VERSION   REPLACES   PHASE
windows-machine-config-operator.2.0.0   Windows Machine Config Operator   2.0.0
Succeeded
```

3.2. WINDOWS MACHINE CONFIG OPERATOR のシークレットの設定

Windows Machine Config Operator (WMCO) を実行するには、プライベートキーを含む WMCO namespace でシークレットを作成する必要があります。これは、WMCO が Windows 仮想マシン (VM) と通信できるようにするために必要です。

前提条件

- Operator Lifecycle Manager (OLM) を使用して Windows Machine Config Operator (WMCO) をインストールしている。
- RSA キーが含まれる PEM でエンコードされたファイルを作成します。

手順

- Windows 仮想マシンへのアクセスに必要なシークレットを定義します。

```
$ oc create secret generic cloud-private-key --from-file=private-
key.pem=${HOME}/.ssh/<key> \
    -n openshift-windows-machine-config-operator 1
```

1 **openshift-windows-machine-config-operator** などの、WMCO namespace のプライベートキーを作成する必要があります。

クラスターのインストール時に使用されるものとは異なるプライベートキーを使用することが推奨されます。

3.3. 関連情報

- SSH プライベートキーの生成およびエージェントへの追加
- Operator のクラスターへの追加。

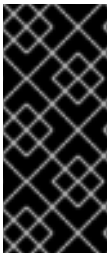
第4章 WINDOWS MACHINESET オブジェクトの作成

4.1. AWS での WINDOWS MACHINESET オブジェクトの作成

Amazon Web Services (AWS) で OpenShift Container Platform クラスターの特定の機能を果たすように **MachineSet** オブジェクトを作成することができます。たとえば、インフラストラクチャー Windows マシンセットおよび関連マシンを作成して、サポートする Windows ワークロードを新規の Windows マシンに移動できます。

前提条件

- Operator Lifecycle Manager (OLM) を使用して Windows Machine Config Operator (WMCO) をインストールしている。
- Docker フォーマットのコンテナーランタイムアドオンアドオンが有効な状態で、サポートされている Windows Server をオペレーティングシステムのイメージとして使用している。



重要

現時点で、Docker 形式のコンテナーランタイムは Windows ノードで使用されます。Kubernetes では、コンテナーランタイムとしての Docker を非推奨としています。詳細は、Kubernetes ドキュメントの Docker の非推奨について参照してください。Containerd は、今後の Kubernetes リリースで Windows ノードについてサポートされる新しいコンテナーランタイムになります。

4.1.1. マシン API の概要

マシン API は、アップストリームのクラスター API プロジェクトおよびカスタム OpenShift Container Platform リソースに基づく重要なリソースの組み合わせです。

OpenShift Container Platform 4.7 クラスターの場合、マシン API はクラスターインストールの終了後にすべてのノードホストのプロビジョニングの管理アクションを実行します。このシステムにより、OpenShift Container Platform 4.7 はパブリックまたはプライベートのクラウドインフラストラクチャーに加えて弾力性があり、動的なプロビジョニング方法を提供します。

以下の 2 つのリソースは重要なリソースになります。

Machines

ノードのホストを記述する基本的なユニットです。マシンには、複数の異なるクラウドプラットフォーム用に提供されるコンピュートノードのタイプを記述する **providerSpec** 仕様があります。たとえば、Amazon Web Services (AWS) 上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および必要なメタデータを定義する場合があります。

マシンセット

MachineSet リソースはマシンのグループです。マシンセットとマシンの関係は、レプリカセットと Pod の関係と同様です。マシンを追加する必要がある場合や、マシンの数を縮小したりする必要がある場合、コンピューティングのニーズに応じてマシンセットの **replicas** フィールドを変更します。

以下のカスタムリソースは、クラスターに機能を追加します。

Machine Autoscaler

MachineAutoscaler リソースはマシンをクラウドで自動的にスケーリングします。ノードに対する最小および最大のスケーリングの境界を、指定されるマシンセットに設定でき、Machine Autoscaler

はノードの該当範囲を維持します。**MachineAutoscaler** オブジェクトは **ClusterAutoscaler** オブジェクトの設定後に有効になります。**ClusterAutoscaler** および **MachineAutoscaler** リソースは、どちらも **ClusterAutoscalerOperator** オブジェクトによって利用可能にされます。

Cluster Autoscaler

このリソースはアップストリームの Cluster Autoscalerプロジェクトに基づいています。OpenShift Container Platform の実装では、これはマシンセット API を拡張することによってクラスター API に統合されます。コア、ノード、メモリー、および GPU などのリソースのクラスター全体でのスケーリング制限を設定できます。優先順位を設定することにより、重要度の低い Pod のために新規ノードがオンラインにならないようにクラスターで Pod の優先順位付けを実行できます。また、スケーリングポリシーを設定してノードをスケールダウンせずにスケールアップできるようにすることもできます。

マシンのヘルスチェック

MachineHealthCheck リソースはマシンの正常でない状態を検知し、マシンを削除し、サポートされているプラットフォームでは新規マシンを作成します。

OpenShift Container Platform バージョン 3.11 では、クラスターでマシンのプロビジョニングが管理されないためにマルチゾーンアーキテクチャーを容易に展開することができませんでした。しかし、OpenShift Container Platform バージョン 4.1 以降、このプロセスはより簡単になりました。それぞれのマシンセットのスコープが単一ゾーンに設定されるため、インストールプログラムはユーザーに代わって、アベイラビリティーゾーン全体にマシンセットを送信します。さらに、コンピューティングは動的に展開されるため、ゾーンに障害が発生した場合の、マシンのリバランスが必要な場合に使用するゾーンを常に確保できます。Autoscaler はクラスターの有効期間中にベストエフォートでバランシングを提供します。

4.1.2. AWS の Windows MachineSet オブジェクトのサンプル YAML

このサンプル YAML は、Windows Machine Config Operator (WMCO) が応答する Amazon Web Services (AWS) で実行される Windows **MachineSet** オブジェクトを定義します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <infrastructure_id>-windows-worker-<zone> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <infrastructure_id>-windows-worker-<zone> 4
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: <infrastructure_id>-windows-worker-<zone> 6
        machine.openshift.io/os-id: Windows 7
    spec:
      metadata:
        labels:
```

```
      node-role.kubernetes.io/worker: "" (8)
  providerSpec:
    value:
      ami:
        id: <windows_container_ami> (9)
      apiVersion: awsproviderconfig.openshift.io/v1beta1
      blockDevices:
        - ebs:
            iops: 0
            volumeSize: 120
            volumeType: gp2
      credentialsSecret:
        name: aws-cloud-credentials
      deviceIndex: 0
      iamInstanceProfile:
        id: <infrastructure_id>-worker-profile (10)
      instanceType: m5a.large
      kind: AWSMachineProviderConfig
      placement:
        availabilityZone: <zone> (11)
        region: <region> (12)
      securityGroups:
        - filters:
            - name: tag:Name
              values:
                - <infrastructure_id>-worker-sg (13)
      subnet:
        filters:
          - name: tag:Name
            values:
              - <infrastructure_id>-private-<zone> (14)
      tags:
        - name: kubernetes.io/cluster/<infrastructure_id> (15)
          value: owned
      userDataSecret:
        name: windows-user-data (16)
        namespace: openshift-machine-api
```

(1)(3)(5)(10)(13)(14)(15) クラスターのプロビジョニング時に設定したクラスター ID を基にするインフラストラクチャー ID を指定します。以下のコマンドを実行してインフラストラクチャー ID を取得できます。

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

(2)(4)(6) インフラストラクチャー ID、ワーカーラベル、およびゾーンを指定します。

(7) Windows マシンとしてマシンセットを設定します。

(8) Windows ノードをコンピュートマシンとして設定します。

(9) コンテナーランタイムがインストールされている Windows イメージの AMI ID を指定します。Windows Server 2019 を使用する必要があります。

(11) **us-east-1a** などの AWS ゾーンを指定します。

12 **us-east-1** などの AWS リージョンを指定します。

16 最初の Windows マシンの設定時に WMCO によって作成されます。その後、後続のすべてのマシンセットで **windows-user-data** を消費できるようになります。

4.1.3. マシンセットの作成

インストールプログラムによって作成されるものに加え、独自のマシンセットを作成して、選択する特定のワークロードに対するマシンのコンピュートリソースを動的に管理することができます。

前提条件

- OpenShift Container Platform クラスターをデプロイすること。
- OpenShift CLI (**oc**) をインストールすること。
- **cluster-admin** パーミッションを持つユーザーとして、**oc** にログインすること。

手順

1. 説明されているようにマシンセット カスタムリソース (CR) サンプルを含む新規 YAML ファイルを作成し、そのファイルに **<file_name>.yaml** という名前を付けます。
   <clusterID> および **<role>** パラメーターの値を設定していることを確認します。

   a. 特定のフィールドに設定する値が不明な場合は、クラスターから既存のマシンセットを確認できます。

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   出力例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1d   0         0                             55m
   agl030519-vplxk-worker-us-east-1e   0         0                             55m
   agl030519-vplxk-worker-us-east-1f   0         0                             55m
   ```

   b. 特定のマシンセットの値を確認します。

   ```
   $ oc get machineset <machineset_name> -n \
     openshift-machine-api -o yaml
   ```

   出力例

   ```
   ...
   template:
     metadata:
       labels:
         machine.openshift.io/cluster-api-cluster: agl030519-vplxk 1
   ```

```
      machine.openshift.io/cluster-api-machine-role: worker 2
      machine.openshift.io/cluster-api-machine-type: worker
      machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
```

1 クラスター ID。

2 デフォルトのノードラベル。

2. 新規 **MachineSet** CR を作成します。

```
$ oc create -f <file_name>.yaml
```

3. マシンセットの一覧を表示します。

```
$ oc get machineset -n openshift-machine-api
```

出力例

```
NAME                                  DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
agl030519-vplxk-worker-us-east-1a           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d           0         0                             55m
agl030519-vplxk-worker-us-east-1e           0         0                             55m
agl030519-vplxk-worker-us-east-1f           0         0                             55m
```

新規のマシンセットが利用可能な場合、 **DESIRED** および **CURRENT** の値は一致します。マシンセットが利用可能でない場合、数分待機してからコマンドを再度実行します。

4.1.4. 関連情報

- マシンセットの管理に関する詳細は、「**マシン管理**」セクションを参照してください。

4.2. AZURE での WINDOWS MACHINESET オブジェクトの作成

Microsoft Azure 上の OpenShift Container Platform クラスターで特定の機能を果たすように Windows **MachineSet** オブジェクトを作成できます。たとえば、インフラストラクチャー Windows マシンセットおよび関連マシンを作成して、サポートする Windows ワークロードを新規の Windows マシンに移動できます。

前提条件

- Operator Lifecycle Manager (OLM) を使用して Windows Machine Config Operator (WMCO) をインストールしている。
- Docker フォーマットのコンテナーランタイムアドオンアドオンが有効な状態で、サポートされている Windows Server をオペレーティングシステムのイメージとして使用している。

重要

現時点で、Docker 形式のコンテナーランタイムは Windows ノードで使用されます。Kubernetes では、コンテナーランタイムとしての Docker を非推奨としています。詳細は、Kubernetes ドキュメントの Docker の非推奨について参照してください。Containerd は、今後の Kubernetes リリースで Windows ノードについてサポートされる新しいコンテナーランタイムになります。

4.2.1. マシン API の概要

マシン API は、アップストリームのクラスター API プロジェクトおよびカスタム OpenShift Container Platform リソースに基づく重要なリソースの組み合わせです。

OpenShift Container Platform 4.7 クラスターの場合、マシン API はクラスターインストールの終了後にすべてのノードホストのプロビジョニングの管理アクションを実行します。このシステムにより、OpenShift Container Platform 4.7 はパブリックまたはプライベートのクラウドインフラストラクチャーに加えて弾力性があり、動的なプロビジョニング方法を提供します。

以下の 2 つのリソースは重要なリソースになります。

Machines

ノードのホストを記述する基本的なユニットです。マシンには、複数の異なるクラウドプラットフォーム用に提供されるコンピュートノードのタイプを記述する **providerSpec** 仕様があります。たとえば、Amazon Web Services (AWS) 上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および必要なメタデータを定義する場合があります。

マシンセット

MachineSet リソースはマシンのグループです。マシンセットとマシンの関係は、レプリカセットと Pod の関係と同様です。マシンを追加する必要がある場合や、マシンの数を縮小したりする必要がある場合、コンピューティングのニーズに応じてマシンセットの **replicas** フィールドを変更します。

以下のカスタムリソースは、クラスターに機能を追加します。

Machine Autoscaler

MachineAutoscaler リソースはマシンをクラウドで自動的にスケーリングします。ノードに対する最小および最大のスケーリングの境界を、指定されるマシンセットに設定でき、Machine Autoscaler はノードの該当範囲を維持します。**MachineAutoscaler** オブジェクトは **ClusterAutoscaler** オブジェクトの設定後に有効になります。**ClusterAutoscaler** および **MachineAutoscaler** リソースは、どちらも **ClusterAutoscalerOperator** オブジェクトによって利用可能にされます。

Cluster Autoscaler

このリソースはアップストリームの Cluster Autoscalerプロジェクトに基づいています。OpenShift Container Platform の実装では、これはマシンセット API を拡張することによってクラスター API に統合されます。コア、ノード、メモリー、および GPU などのリソースのクラスター全体でのスケーリング制限を設定できます。優先順位を設定することにより、重要度の低い Pod のために新規ノードがオンラインにならないようにクラスターで Pod の優先順位付けを実行できます。また、スケーリングポリシーを設定してノードをスケールダウンせずにスケールアップできるようにすることもできます。

マシンのヘルスチェック

MachineHealthCheck リソースはマシンの正常でない状態を検知し、マシンを削除し、サポートされているプラットフォームでは新規マシンを作成します。

OpenShift Container Platform バージョン 3.11 では、クラスターでマシンのプロビジョニングが管理されないためにマルチゾーンアーキテクチャーを容易に展開することができませんでした。しかし、

OpenShift Container Platform バージョン 4.1 以降、このプロセスはより簡単になりました。それぞれのマシンセットのスコープが単一ゾーンに設定されるため、インストールプログラムはユーザーに代わって、アベイラビリティーゾーン全体にマシンセットを送信します。さらに、コンピューティングは動的に展開されるため、ゾーンに障害が発生した場合の、マシンのリバランスが必要な場合に使用するゾーンを常に確保できます。Autoscaler はクラスターの有効期間中にベストエフォートでバランシングを提供します。

4.2.2. Azure の Windows MachineSet オブジェクトのサンプル YAML

このサンプル YAML は、Windows Machine Config Operator (WMCO) が応答する Microsoft Azure で実行される Windows **MachineSet** オブジェクトを定義します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <windows_machine_set_name> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 4
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 6
        machine.openshift.io/os-id: Windows 7
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/worker: "" 8
      providerSpec:
        value:
          apiVersion: azureproviderconfig.openshift.io/v1beta1
          credentialsSecret:
            name: azure-cloud-credentials
            namespace: openshift-machine-api
          image: 9
            offer: WindowsServer
            publisher: MicrosoftWindowsServer
            resourceID: ""
            sku: 2019-Datacenter-with-Containers
            version: latest
          kind: AzureMachineProviderSpec
          location: <location> 10
          managedIdentity: <infrastructure_id>-identity 11
          networkResourceGroup: <infrastructure_id>-rg 12
          osDisk:
```

```
        diskSizeGB: 128
        managedDisk:
          storageAccountType: Premium_LRS
        osType: Windows
      publicIP: false
      resourceGroup: <infrastructure_id>-rg (13)
      subnet: <infrastructure_id>-worker-subnet
      userDataSecret:
        name: windows-user-data (14)
        namespace: openshift-machine-api
      vmSize: Standard_D2s_v3
      vnet: <infrastructure_id>-vnet (15)
      zone: "<zone>" (16)
```

(1)(3)(5)(11)(12)(13)(15) クラスターのプロビジョニング時に設定したクラスター ID を基にするインフラストラクチャー ID を指定します。以下のコマンドを実行してインフラストラクチャー ID を取得できます。

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

(2)(4)(6) Windows マシンセット名を指定します。Azure の Windows マシン名は 15 文字を超えることができません。そのため、マシン名の生成方法により、マシンセット名は 9 文字を超えることができません。

(7) Windows マシンとしてマシンセットを設定します。

(8) Windows ノードをコンピュートマシンとして設定します。

(9) **2019-Datacenter-with-Containers** SKU を定義する **WindowsServer** イメージオファリングを指定します。

(10) **centralus** などの Azure リージョンを指定します。

(14) 最初の Windows マシンの設定時に WMCO によって作成されます。その後、後続のすべてのマシンセットで **windows-user-data** を消費できるようになります。

(16) マシンを配置するリージョン内のゾーンを指定します。リージョンがゾーンをサポートすることを確認してください。

4.2.3. マシンセットの作成

インストールプログラムによって作成されるものに加え、独自のマシンセットを作成して、選択する特定のワークロードに対するマシンのコンピュートリソースを動的に管理することができます。

前提条件

- OpenShift Container Platform クラスターをデプロイすること。
- OpenShift CLI (**oc**) をインストールすること。
- **cluster-admin** パーミッションを持つユーザーとして、**oc** にログインすること。

手順

1. 説明されているようにマシンセット カスタムリソース (CR) サンプルを含む新規 YAML ファイルを作成し、そのファイルに **<file_name>.yaml** という名前を付けます。
<clusterID> および **<role>** パラメーターの値を設定していることを確認します。

   a. 特定のフィールドに設定する値が不明な場合は、クラスターから既存のマシンセットを確認できます。

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   出力例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1d   0         0                             55m
   agl030519-vplxk-worker-us-east-1e   0         0                             55m
   agl030519-vplxk-worker-us-east-1f   0         0                             55m
   ```

   b. 特定のマシンセットの値を確認します。

   ```
   $ oc get machineset <machineset_name> -n \
       openshift-machine-api -o yaml
   ```

   出力例

   ```
   ...
   template:
     metadata:
       labels:
         machine.openshift.io/cluster-api-cluster: agl030519-vplxk 1
         machine.openshift.io/cluster-api-machine-role: worker 2
         machine.openshift.io/cluster-api-machine-type: worker
         machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
   ```

   1 クラスター ID。

   2 デフォルトのノードラベル。

2. 新規 **MachineSet** CR を作成します。

   ```
   $ oc create -f <file_name>.yaml
   ```

3. マシンセットの一覧を表示します。

   ```
   $ oc get machineset -n openshift-machine-api
   ```

   出力例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
   agl030519-vplxk-worker-us-east-1a           1         1         1       1           55m
   ```

```
agl030519-vplxk-worker-us-east-1b   1   1   1   1   55m
agl030519-vplxk-worker-us-east-1c   1   1   1   1   55m
agl030519-vplxk-worker-us-east-1d   0   0               55m
agl030519-vplxk-worker-us-east-1e   0   0               55m
agl030519-vplxk-worker-us-east-1f   0   0               55m
```

新規のマシンセットが利用可能な場合、 **DESIRED** および **CURRENT** の値は一致します。マシンセットが利用可能でない場合、数分待機してからコマンドを再度実行します。

4.2.4. 関連情報

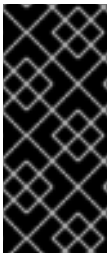
- マシンセットの管理に関する詳細は、「**マシン管理**」セクションを参照してください。

4.3. VSPHERE での WINDOWS MACHINESET オブジェクトの作成

VMware vSphere 上の OpenShift Container Platform クラスターで特定の機能を果たすように Windows **MachineSet** オブジェクトを作成できます。たとえば、インフラストラクチャー Windows マシンセットおよび関連マシンを作成して、サポートする Windows ワークロードを新規の Windows マシンに移動できます。

前提条件

- Operator Lifecycle Manager (OLM) を使用して Windows Machine Config Operator (WMCO) をインストールしている。
- Docker フォーマットのコンテナーランタイムアドオンアドオンが有効な状態で、サポートされている Windows Server をオペレーティングシステムのイメージとして使用している。



重要

現時点で、Docker 形式のコンテナーランタイムは Windows ノードで使用されます。Kubernetes では、コンテナーランタイムとしての Docker を非推奨としています。詳細は、Kubernetes ドキュメントの Docker の非推奨について参照してください。
Containerd は、今後の Kubernetes リリースで Windows ノードについてサポートされる新しいコンテナーランタイムになります。

4.3.1. マシン API の概要

マシン API は、アップストリームのクラスター API プロジェクトおよびカスタム OpenShift Container Platform リソースに基づく重要なリソースの組み合わせです。

OpenShift Container Platform 4.7 クラスターの場合、マシン API はクラスターインストールの終了後にすべてのノードホストのプロビジョニングの管理アクションを実行します。このシステムにより、OpenShift Container Platform 4.7 はパブリックまたはプライベートのクラウドインフラストラクチャーに加えて弾力性があり、動的なプロビジョニング方法を提供します。

以下の 2 つのリソースは重要なリソースになります。

Machines

ノードのホストを記述する基本的なユニットです。マシンには、複数の異なるクラウドプラットフォーム用に提供されるコンピュートノードのタイプを記述する **providerSpec** 仕様があります。たとえば、Amazon Web Services (AWS) 上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および必要なメタデータを定義する場合があります。

マシンセット

MachineSet リソースはマシンのグループです。マシンセットとマシンの関係は、レプリカセットと Pod の関係と同様です。マシンを追加する必要がある場合や、マシンの数を縮小したりする必要がある場合、コンピューティングのニーズに応じてマシンセットの **replicas** フィールドを変更します。

以下のカスタムリソースは、クラスターに機能を追加します。

Machine Autoscaler

MachineAutoscaler リソースはマシンをクラウドで自動的にスケーリングします。ノードに対する最小および最大のスケーリングの境界を、指定されるマシンセットに設定でき、Machine Autoscaler はノードの該当範囲を維持します。**MachineAutoscaler** オブジェクトは **ClusterAutoscaler** オブジェクトの設定後に有効になります。**ClusterAutoscaler** および **MachineAutoscaler** リソースは、どちらも **ClusterAutoscalerOperator** オブジェクトによって利用可能にされます。

Cluster Autoscaler

このリソースはアップストリームの Cluster Autoscalerプロジェクトに基づいています。OpenShift Container Platform の実装では、これはマシンセット API を拡張することによってクラスター API に統合されます。コア、ノード、メモリー、および GPU などのリソースのクラスター全体でのスケーリング制限を設定できます。優先順位を設定することにより、重要度の低い Pod のために新規ノードがオンラインにならないようにクラスターで Pod の優先順位付けを実行できます。また、スケーリングポリシーを設定してノードをスケールダウンせずにスケールアップできるようにすることもできます。

マシンのヘルスチェック

MachineHealthCheck リソースはマシンの正常でない状態を検知し、マシンを削除し、サポートされているプラットフォームでは新規マシンを作成します。

OpenShift Container Platform バージョン 3.11 では、クラスターでマシンのプロビジョニングが管理されないためにマルチゾーンアーキテクチャーを容易に展開することができませんでした。しかし、OpenShift Container Platform バージョン 4.1 以降、このプロセスはより簡単になりました。それぞれのマシンセットのスコープが単一ゾーンに設定されるため、インストールプログラムはユーザーに代わって、アベイラビリティーゾーン全体にマシンセットを送信します。さらに、コンピューティングは動的に展開されるため、ゾーンに障害が発生した場合の、マシンのリバランスが必要な場合に使用するゾーンを常に確保できます。Autoscaler はクラスターの有効期間中にベストエフォートでバランシングを提供します。

4.3.2. Windows コンテナーワークロード用の vSphere 環境の準備

vSphere Windows 仮想マシンのゴールドイメージを作成し、WMCO の内部 API サーバーとの通信を有効にして、Windows コンテナーのワークロード用に vSphere 環境を準備する必要があります。

4.3.2.1. vSphere Windows 仮想マシンのゴールドイメージの作成

vSphere Windows 仮想マシン (VM) のゴールドイメージを作成します。

前提条件

- OVN-Kubernetes を使用してハイブリッドネットワークで設定された vSphere にクラスターをインストールしている。
- ホスト間での Pod 間の接続の問題に対応するためにハイブリッドネットワーク設定でカスタム VXLAN ポートを定義している。

手順

1. Microsoft パッチ KB4565351を含む Windows Server 1909 仮想マシンイメージの更新された

バージョンで仮想マシンを作成します。このパッチは、vSphere にインストールされたクラスターに必要な VXLAN UDP ポートの設定に必要です。このパッチは、**Windows Server 2019** 仮想マシンイメージでは利用できません。

2. **openshift-windows-machine-config-operator** namespace で作成したシークレットにあるプライベートキーに対応するパブリックキーが含まれる Windows 仮想マシンに **C:\Users\Administrator\.ssh\authorized_keys** ファイルを作成します。シークレットのプライベートキーは、OpenShift Container Platform に Windows 仮想マシンへのアクセスを付与するために Windows Machine Config Operator (WMCO) を最初にインストールした際に作成されました。**authorized_keys** ファイルは、Windows 仮想マシンで SSH を設定するために使用されます。

3. 以下の Powershell スクリプトを実行して、Windows 仮想マシンで SSH を設定します。

```
# Powershell script to configure SSH on vSphere Windows VMs
Add-WindowsCapability -Online -Name OpenSSH.Server~~~~0.0.1.0
$firewallRuleName = "ContainerLogsPort"
$containerLogsPort = "10250"
New-NetFirewallRule -DisplayName $firewallRuleName -Direction Inbound -Action Allow -
Protocol TCP -LocalPort $containerLogsPort -EdgeTraversalPolicy Allow
Install-PackageProvider -Name NuGet -MinimumVersion 2.8.5.201 -Force
Install-Module -Force OpenSSHUtils
Set-Service -Name ssh-agent -StartupType 'Automatic'
Set-Service -Name sshd -StartupType 'Automatic'
Start-Service ssh-agent
Start-Service sshd
$pubKeyConf = (Get-Content -path C:\ProgramData\ssh\sshd_config) -replace
'#PubkeyAuthentication yes','PubkeyAuthentication yes'
$pubKeyConf | Set-Content -Path C:\ProgramData\ssh\sshd_config
$passwordConf = (Get-Content -path C:\ProgramData\ssh\sshd_config) -replace
'#PasswordAuthentication yes','PasswordAuthentication yes'
$passwordConf | Set-Content -Path C:\ProgramData\ssh\sshd_config
$authFileConf = (Get-Content -path C:\ProgramData\ssh\sshd_config) -replace
'AuthorizedKeysFile
__PROGRAMDATA__/ssh/administrators_authorized_keys','#AuthorizedKeysFile
__PROGRAMDATA__/ssh/administrators_authorized_keys'
$authFileConf | Set-Content -Path C:\ProgramData\ssh\sshd_config
$pubKeyLocationConf = (Get-Content -path C:\ProgramData\ssh\sshd_config) -replace
'Match Group administrators','#Match Group administrators'
$pubKeyLocationConf | Set-Content -Path C:\ProgramData\ssh\sshd_config
Restart-Service sshd
New-item -Path $env:USERPROFILE -Name .ssh -ItemType Directory -force
```

4. Windows 仮想マシンに VMware Tools バージョン 11.0.6 以降をインストールし、設定します。詳細は、VMware Tools のドキュメントを参照してください。

5. Windows 仮想マシンに VMware Tools をインストールした後、以下を確認します。

   a. **C:\ProgramData\VMware\VMware Tools\tools.conf** ファイルには、以下のエントリーが含まれます。

   ```
   exclude-nics=
   ```

   このエントリーにより、以下が確認されます。

- hybrid-overlay により Windows 仮想マシンで生成され、クローン作成された vNIC は無視されない。
- 仮想マシンには vCenter の IP アドレスがある。

b. VMTools Windows サービスが実行中である。

6. アプリケーションに必要な Windows コンテナーベースイメージをすべてプルします。プルするイメージは、使用している Windows カーネルによって異なります。詳細は、Microsoft のドキュメントでWindows コンテナーベースイメージのプル について参照してください。

7. Windows 仮想マシンで Windows Sysprep ツールを実行します。

```
C:\> sysprep.exe /generalize /oobe /shutdown /unattend:<path_to_unattend.xml>
```

サンプルの **unattend.xml** が提供され、これは WMCO で必要なすべての変更を維持します。たとえば、**unattend.xml** ファイルでは、管理者のホームディレクトリーが SSH パブリックキーでそのままの状態であることを確認する必要があります。必要に応じてサンプルをカスタマイズする必要があります。

例4.1 サンプル unattend.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!--A sample unattend.xml which helps in setting admin password and running scripts on first boot-->
<unattend xmlns="urn:schemas-microsoft-com:unattend">
  <settings pass="specialize">
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http:// www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-International-Core" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
      <InputLocale>0409:00000409</InputLocale>
      <SystemLocale>en-US</SystemLocale>
      <UILanguage>en-US</UILanguage>
      <UILanguageFallback>en-US</UILanguageFallback>
      <UserLocale>en-US</UserLocale>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-Security-SPP-UX" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
      <SkipAutoActivation>true</SkipAutoActivation>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-SQMApi" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
      <CEIPEnabled>0</CEIPEnabled>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
      <ComputerName>windows-host</ComputerName>
      <ProductKey>My_Product_key</ProductKey>
    </component>
```

```
  </settings>
  <settings pass="oobeSystem">
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS">
      <AutoLogon>
        <Password>
          <Value>MyPassword</Value>
          <PlainText>true</PlainText>
        </Password>
        <Enabled>true</Enabled>
        <Username>Administrator</Username>
      </AutoLogon>
      <OOBE>
        <HideEULAPage>true</HideEULAPage>
        <HideLocalAccountScreen>true</HideLocalAccountScreen>
        <HideOEMRegistrationScreen>true</HideOEMRegistrationScreen>
        <HideOnlineAccountScreens>true</HideOnlineAccountScreens>
        <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
        <NetworkLocation>Work</NetworkLocation>
        <ProtectYourPC>1</ProtectYourPC>
        <SkipMachineOOBE>true</SkipMachineOOBE>
        <SkipUserOOBE>true</SkipUserOOBE>
      </OOBE>
      <RegisteredOrganization>Organization</RegisteredOrganization>
      <RegisteredOwner>Owner</RegisteredOwner>
      <DisableAutoDaylightTimeSet>false</DisableAutoDaylightTimeSet>
      <TimeZone>Eastern Standard Time</TimeZone>
      <UserAccounts>
        <AdministratorPassword>
          <Value>MyPassword</Value>
          <PlainText>true</PlainText>
        </AdministratorPassword>
        <LocalAccounts>
          <LocalAccount wcm:action="add">
            <Description>Administrator</Description>
            <DisplayName>Administrator</DisplayName>
            <Group>Administrators</Group>
            <Name>Administrator</Name>
          </LocalAccount>
        </LocalAccounts>
      </UserAccounts>
    </component>
  </settings>
</unattend>
```

4.3.2.2. vSphere での WMCO についての内部 API サーバーとの通信の有効化

Windows Machine Config Operator (WMCO) は Ignition 設定ファイルを内部 API サーバーエンドポイントからダウンロードします。Windows 仮想マシン (VM) が Ignition 設定ファイルをダウンロードできるように、また設定された仮想マシンがkubelet が内部 API サーバーとのみ通信できるように内部 API サーバーとの通信を有効にする必要があります。

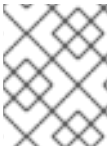
前提条件

前提条件

- クラスターを vSphere にインストールしている。

手順

- 外部 API サーバー URL **api.<cluster_name>.<base_domain>** を参照する **api-int.<cluster_name>.<base_domain>** の新規 DNS エントリー追加します。これには、CNAME または追加の A レコードを指定できます。



注記

外部 API エンドポイントは、vSphere への初期クラスターインストールの一部としてすでに作成されています。

4.3.3. vSphere での Windows の MachineSet オブジェクトのサンプル YAML

このサンプル YAML は、Windows Machine Config Operator (WMCO) が応答する VMware vSphere で実行される Windows **MachineSet** オブジェクトを定義します。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <windows_machine_set_name> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 4
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 6
        machine.openshift.io/os-id: Windows 7
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/worker: "" 8
      providerSpec:
        value:
          apiVersion: vsphereprovider.openshift.io/v1beta1
          credentialsSecret:
            name: vsphere-cloud-credentials
          diskGiB: 128
          kind: VSphereMachineProviderSpec
          memoryMiB: 16384
          network:
```

```
      devices:
      - networkName: "<vm_network_name>" 9
    numCPUs: 4
    numCoresPerSocket: 1
    snapshot: ""
    template: <windows_vm_template_name> 10
    userDataSecret:
      name: windows-user-data 11
    workspace:
      datacenter: <vcenter_datacenter_name> 12
      datastore: <vcenter_datastore_name> 13
      folder: <vcenter_vm_folder_path> 14
      resourcePool: <vsphere_resource_pool> 15
      server: <vcenter_server_ip> 16
```

1 3 5 クラスターのプロビジョニング時に設定したクラスター ID を基にするインフラストラクチャー ID を指定します。以下のコマンドを実行してインフラストラクチャー ID を取得できます。

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

2 4 6 Windows マシンセット名を指定します。マシンセットの名前は、マシン名が vSphere で生成される方法により、9 文字を超えることができません。

7 Windows マシンとしてマシンセットを設定します。

8 Windows ノードをコンピュートマシンとして設定します。

9 マシンセットをデプロイする vSphere 仮想マシンネットワークを指定します。

10 使用する Windows vSphere 仮想マシンテンプレートの完全パス (例: **/Datacenter/vm/ocp4-llplx/windows-golden-image**) を指定します。名前は一意である必要があります。

重要

元の仮想マシンテンプレートは指定しないでください。仮想マシンテンプレートは停止した状態でなければなりません。また、新規 RHCOS マシン用にクローン作成する必要があります。仮想マシンテンプレートを起動すると、仮想マシンテンプレートがプラットフォームの仮想マシンとして設定されるので、これをマシンセットで設定を適用できるテンプレートとして使用できなくなります。

11 **windows-user-data** は、最初の Windows マシンの設定時に WMCO によって作成されます。その後、後続のすべてのマシンセットで **windows-user-data** を消費できるようになります。

12 マシンセットをデプロイする vCenter Datacenter を指定します。

13 マシンセットをデプロイする vCenter Datastore を指定します。

14 **/dc1/vm/user-inst-5ddjd** などの vCenter の vSphere 仮想マシンフォルダーへのパスを指定します。

15 オプション: Windows 仮想マシンの vSphere リソースプールを指定します。

16 vCenter サーバーの IP または完全修飾ドメイン名を指定します。

4.3.4. マシンセットの作成

インストールプログラムによって作成されるものに加え、独自のマシンセットを作成して、選択する特定のワークロードに対するマシンのコンピュートリソースを動的に管理することができます。

前提条件

- OpenShift Container Platform クラスターをデプロイすること。
- OpenShift CLI (**oc**) をインストールすること。
- **cluster-admin** パーミッションを持つユーザーとして、**oc** にログインすること。

手順

1. 説明されているようにマシンセット カスタムリソース (CR) サンプルを含む新規 YAML ファイルを作成し、そのファイルに **<file_name>.yaml** という名前を付けます。
<clusterID> および **<role>** パラメーターの値を設定していることを確認します。

   a. 特定のフィールドに設定する値が不明な場合は、クラスターから既存のマシンセットを確認できます。

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   出力例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1d   0         0                             55m
   agl030519-vplxk-worker-us-east-1e   0         0                             55m
   agl030519-vplxk-worker-us-east-1f   0         0                             55m
   ```

   b. 特定のマシンセットの値を確認します。

   ```
   $ oc get machineset <machineset_name> -n \
     openshift-machine-api -o yaml
   ```

   出力例

   ```
   ...
   template:
     metadata:
       labels:
         machine.openshift.io/cluster-api-cluster: agl030519-vplxk 1
         machine.openshift.io/cluster-api-machine-role: worker 2
         machine.openshift.io/cluster-api-machine-type: worker
         machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
   ```

   1 クラスター ID。

   2 デフォルトのノードラベル。

2. 新規 **MachineSet** CR を作成します。

```
$ oc create -f <file_name>.yaml
```

3. マシンセットの一覧を表示します。

```
$ oc get machineset -n openshift-machine-api
```

出力例

```
NAME                                  DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
agl030519-vplxk-worker-us-east-1a           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d           0         0                             55m
agl030519-vplxk-worker-us-east-1e           0         0                             55m
agl030519-vplxk-worker-us-east-1f           0         0                             55m
```

新規のマシンセットが利用可能な場合、 **DESIRED** および **CURRENT** の値は一致します。マシンセットが利用可能でない場合、数分待機してからコマンドを再度実行します。

4.3.5. 関連情報

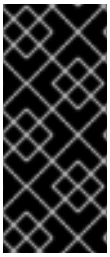
- マシンセットの管理に関する詳細は、「**マシン管理**」セクションを参照してください。

第5章 WINDOWS コンテナーワークロードのスケジューリング

Windows ワークロードを Windows コンピュートノードにスケジュールすることができます。

前提条件

- Operator Lifecycle Manager (OLM) を使用して Windows Machine Config Operator (WMCO) をインストールしている。
- Docker 形式のコンテナーランタイムびアドオンが有効な状態で Windows コンテナーを OS イメージとして使用している。
- Windows マシンセットを作成している。



重要

現時点で、Docker 形式のコンテナーランタイムは Windows ノードで使用されます。Kubernetes では、コンテナーランタイムとしての Docker を非推奨としています。詳細は、Kubernetes ドキュメントの Docker の非推奨について参照してください。Containerd は、今後の Kubernetes リリースで Windows ノードについてサポートされる新しいコンテナーランタイムになります。

5.1. WINDOWS POD の配置

Windows ワークロードをクラスターにデプロイする前に、Pod が適切に割り当てられるように Windows ノードのスケジューリングを設定する必要があります。Windows ノードをホストするマシンがあるので、これは Linux ベースのノードと同じように管理できます。同様に、テイント、容認およびノードセレクターなどのメカニズムを使用して、Windows Pod の適切な Windows ノードへのスケジュールも同様に実行されます。

複数のオペレーティングシステム、および同じクラスターで複数の Windows OS バリアントを実行する機能で、**RuntimeClass** を使用して Windows Pod をベース Windows OS バリアントにマップする必要があります。たとえば、複数の Windows ノードが複数の Windows Server コンテナーのバージョンで実行されている場合、クラスターは Windows Pod を互換性のない Windows OS バリアントにスケジュールする可能性があります。クラスター上の Windows OS バリアントごとに **RuntimeClass** オブジェクトを設定する必要があります。クラスターで 1 つの Windows OS バリアントのみが利用可能である場合、**RuntimeClass** オブジェクトを使用することも推奨されます。

詳細は、ホストとコンテナーのバージョンの互換性について参照してください。

関連情報

- スケジューラーによる Pod 配置の制御
- ノードテイントを使用した Pod 配置の制御
- ノードセレクターの使用による特定ノードへの Pod の配置

5.2. スケジューリングメカニズムをカプセル化するための RUNTIMECLASS オブジェクトの作成

RuntimeClass オブジェクトを使用することにより、テイントおよび容認などのスケジュールの仕組みの使用を単純化できます。テイントおよび容認をカプセル化するランタイムクラスをデプロイしてから、これを Pod に適用して Pod を適切なノードにスケジュールできるようにします。ランタイムクラ

スの作成は、複数のオペレーティングシステムのバリアントをサポートするクラスターでも必要になります。

手順

1. **RuntimeClass** オブジェクト YAML ファイルを作成します。例: **runtime-class.yaml**

```
apiVersion: node.k8s.io/v1beta1
kind: RuntimeClass
metadata:
  name: <runtime_class_name> 1
handler: 'docker'
scheduling:
  nodeSelector: 2
    kubernetes.io/os: 'windows'
    kubernetes.io/arch: 'amd64'
    node.kubernetes.io/windows-build: '10.0.17763'
  tolerations: 3
  - effect: NoSchedule
    key: os
    operator: Equal
    value: "Windows"
```

1. このランタイムクラスで管理する必要のある Pod で定義される **RuntimeClass** オブジェクト名を指定します。
2. このランタイムクラスをサポートするノードに存在する必要があるラベルを指定します。このランタイムクラスを使用する Pod は、このセレクターに一致するノードにのみスケジュールできます。ランタイムクラスのノードセレクターは Pod の既存のノードセレクターとマージされます。競合が発生した場合は、Pod をノードにスケジュールできなくなります。
3. Pod に追加する容認を指定します。ただし、受付時にこのランタイムクラスで実行される重複を除きます。これによって、Pod によって許容されるノードのセットとランタイムクラスが組み合わされます。

2. **RuntimeClass** オブジェクトを作成します。

```
$ oc create -f <file-name>.yaml
```

以下は例になります。

```
$ oc create -f runtime-class.yaml
```

3. **RuntimeClass** オブジェクトを Pod に適用し、これが適切なオペレーティングシステムバリアントにスケジュールされていることを確認します。

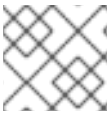
```
apiVersion: v1
kind: Pod
metadata:
  name: my-windows-pod
```

```
spec:
  runtimeClassName: <runtime_class_name> 1
...
```

1 Pod のスケジュールを管理するためにランタイムクラスを指定します。

5.3. WINDOWS コンテナーワークロードのデプロイメント例

Windows コンピュートノードが利用可能になる時点で、Windows コンテナーワークロードをクラスターにデプロイできます。



注記

このサンプルデプロイメントは参照用にのみ提供されます。

Service オブジェクトの例

```
apiVersion: v1
kind: Service
metadata:
  name: win-webserver
  labels:
    app: win-webserver
spec:
  ports:
    # the port that this service should serve on
  - port: 80
    targetPort: 80
  selector:
    app: win-webserver
  type: LoadBalancer
```

Deployment オブジェクトの例

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: win-webserver
  name: win-webserver
spec:
  selector:
    matchLabels:
      app: win-webserver
  replicas: 1
  template:
    metadata:
      labels:
        app: win-webserver
      name: win-webserver
    spec:
      tolerations:
```

```
      - key: "os"
        value: "Windows"
        Effect: "NoSchedule"
      containers:
      - name: windowswebserver
        image: mcr.microsoft.com/windows/servercore:ltsc2019
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - $listener = New-Object System.Net.HttpListener; $listener.Prefixes.Add('http://*:80/');
$listener.Start();Write-Host('Listening at http://*:80/'); while ($listener.IsListening) { $context =
$listener.GetContext(); $response = $context.Response; $content='<html><body><H1>Red Hat
OpenShift + Windows Container Workloads</H1></body></html>'; $buffer =
[System.Text.Encoding]::UTF8.GetBytes($content); $response.ContentLength64 = $buffer.Length;
$response.OutputStream.Write($buffer, 0, $buffer.Length); $response.Close(); };
        securityContext:
          windowsOptions:
            runAsUserName: "ContainerAdministrator"
      nodeSelector:
        beta.kubernetes.io/os: windows
```

注記

mcr.microsoft.com/powershell:<tag> コンテナーイメージを使用する場合、コマンドを **pwsh.exe** として定義する必要があります。**mcr.microsoft.com/windows/servercore:<tag>** コンテナーイメージを使用している場合、コマンドを **powershell.exe** として定義する必要があります。詳細は、Microsoft のドキュメントを参照してください。

5.4. マシンセットの手動によるスケーリング

マシンセットのマシンのインスタンスを追加したり、削除したりする必要がある場合、マシンセットを手動でスケーリングできます。

本書のガイダンスは、完全に自動化されたインストーラーでプロビジョニングされるインフラストラクチャーのインストールに関連します。ユーザーによってプロビジョニングされるインフラストラクチャーのカスタマイズされたインストールにはマシンセットがありません。

前提条件

- OpenShift Container Platform クラスターおよび **oc** コマンドラインをインストールすること。
- **cluster-admin** パーミッションを持つユーザーとして、**oc** にログインすること。

手順

1. クラスターにあるマシンセットを表示します。

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   マシンセットは **<clusterid>-worker-<aws-region-az>** の形式で一覧表示されます。

2. マシンセットをスケーリングします。

```
$ oc scale --replicas=2 machineset <machineset> -n openshift-machine-api
```

または、以下を実行します。

```
$ oc edit machineset <machineset> -n openshift-machine-api
```

マシンセットをスケールアップまたはスケールダウンできます。新規マシンが利用可能になるまで数分の時間がかかります。

第6章 WINDOWS ノードのアップグレード

Windows Machine Config Operator (WMCO) をアップグレードすることで、Windows ノードに最新の更新が含まれることを確認できます。

6.1. WINDOWS MACHINE CONFIG OPERATOR のアップグレード

現在のクラスターバージョンと互換性のある Windows Machine Config Operator (WMCO) の新規バージョンがリリースされると、Operator はアップグレードチャネル、および Operator Lifecycle Manager (OLM) を使用する際にインストールに使用されたサブスクリプションの承認ストラテジーに基づいてアップグレードされます。WMCO のアップグレードにより、Windows マシンの Kubernetes コンポーネントがアップグレードされます。

注記

WMCO の新規バージョンにアップグレードする場合でクラスターモニタリングを使用する必要がある場合は、WMCO namespace に **openshift.io/cluster-monitoring=true** ラベルが必要です。ラベルを既存の WMCO namespace に追加し、すでに Windows ノードが設定されている場合は、WMCO Pod を再起動してモニタリンググラフを表示できるようにします。

中断なしのアップグレードの場合、WMCO は以前のバージョンの WMCO で設定された Windows マシンを中止し、現行バージョンを使用してそれらを再作成します。これは、**Machine** オブジェクトを削除して実行されます。これにより、Windows ノードのドレイン（解放）および削除が実行されます。アップグレードを容易にするために、WMCO は設定されたすべてのノードにバージョンのアノテーションを追加します。アップグレード時に、バージョンのアノテーションで不一致があると、Windows マシンが削除され、再作成されます。アップグレード時のサービスの中断を最小限にするために、WMCO は一度に 1 つの Windows マシンのみを更新します。

重要

WMCO は Kubernetes コンポーネントの更新のみを行い、Windows オペレーティングシステムの更新は行いません。仮想マシンの作成時に Windows イメージを指定できるため、更新されたイメージを指定できます。**MachineSet** 仕様でイメージ設定を変更して、更新された Windows イメージを指定できます。

Operator Lifecycle Manager (OLM) を使用した Operator のアップグレードについての詳細は、 インストールされた Operator のアップグレードについて参照してください。

第7章 WINDOWS ノードの削除

ホスト Windows マシンを削除して、Windows ノードを削除できます。

7.1. 特定マシンの削除

特定のマシンを削除できます。

前提条件

- OpenShift Container Platform クラスターをインストールします。
- OpenShift CLI (**oc**) をインストールすること。
- **cluster-admin** パーミッションを持つユーザーとして、**oc** にログインします。

手順

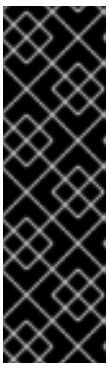
1. クラスターにあるマシンを表示し、削除するマシンを特定します。

```
$ oc get machine -n openshift-machine-api
```

   コマンド出力には、**<clusterid>-worker-<cloud_region>** 形式のマシンの一覧が含まれます。

2. マシンを削除します。

```
$ oc delete machine <machine> -n openshift-machine-api
```



重要

デフォルトでは、マシンコントローラーは、成功するまでマシンによってサポートされるノードをドレイン (解放) しようとします。Pod の Disruption Budget（停止状態の予算）が正しく設定されていない場合などには、ドレイン (解放) の操作を実行してもマシンの削除を防ぐことができない場合があります。特定のマシンの "machine.openshift.io/exclude-node-draining" にアノテーションを付けると、ノードのドレイン（解放）を省略できます。削除中のマシンがマシンセットに属する場合、指定されたレプリカ数に対応するために新規マシンが即時に作成されます。

第8章 WINDOWS コンテナーワークロードの無効化

Windows コンテナーワークロードを実行する機能を無効にするには、Windows Machine Config Operator (WMCO) をアンインストールし、WMCO のインストール時にデフォルトで追加された namespace を削除します。

8.1. WINDOWS MACHINE CONFIG OPERATOR のアンインストール

クラスターから Windows Machine Config Operator (WMCO) をアンインストールできます。

前提条件

- Windows ワークロードをホストする Windows **Machine** オブジェクトを削除します。

手順

1. **Operators → OperatorHub** ページから、**Filter by keyword** ボックスを使用して、**Red Hat Windows Machine Config Operator** を検索します。
2. **Red Hat Windows Machine Config Operator**タイルをクリックします。Operator タイルはこれがインストールされていることを示します。
3. **Windows Machine Config Operator**記述子ページで、**Uninstall** をクリックします。

8.2. WINDOWS MACHINE CONFIG OPERATOR NAMESPACE の削除

デフォルトで Windows Machine Config Operator (WMCO) 用に生成された namespace を削除できます。

前提条件

- WMCO がクラスターから削除されます。

手順

1. **openshift-windows-machine-config-operator** namespace で作成されたすべての Windows ワークロードを削除します。

```
$ oc delete --all pods --namespace=openshift-windows-machine-config-operator
```

2. **openshift-windows-machine-config-operator** namespace のすべての Pod が削除されているか、または終了状態を報告していることを確認します。

```
$ oc get pods --namespace openshift-windows-machine-config-operator
```

3. **openshift-windows-machine-config-operator** namespace を削除します。

```
$ oc delete namespace openshift-windows-machine-config-operator
```

関連情報

- クラスターからの Operator の削除

- Windows ノードの削除