OpenShift Container Platform 4.6

OpenShift용 Windows 컨테이너 지원

Windows 컨테이너용 Red Hat OpenShift 가이드

Last Updated: 2023-04-08

OpenShift Container Platform 4.6 OpenShift용 Windows 컨테이너 지원

Windows 컨테이너용 Red Hat OpenShift 가이드

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.

법적 공지

초록

Windows 컨테이너용 Red Hat OpenShift는 OpenShift Container Platform에서 Microsoft Windows Server 컨테이너를 실행하기 위한 기본 지원을 제공합니다. 이 가이드에서는 모든 세부 정보를 제공합니다.

차례

1장. WINDOWS 컨테이너용 RED HAT OPENSHIFT 지원 개요

Windows 컨테이너에 대한 Red Hat OpenShift 지원은 OpenShift Container Platform 클러스터에서 Windows 컴퓨팅 노드를 실행하는 기능을 제공하는 기능입니다. 이는 Red Hat WMCO(Windows Machine Config Operator)를 사용하여 Windows 노드를 설치 및 관리할 수 있습니다. Red Hat 서브스크립션을 통해 OpenShift Container Platform에서 Windows 워크로드를 실행할 수 있습니다. 자세한 내용은 릴리스 노트 를 참조하십시오.

Linux 및 Windows를 비롯한 워크로드의 경우 OpenShift Container Platform을 사용하면 Windows Server 컨테이너에서 실행되는 Windows 워크로드를 배포할 수 있으며 RHCOS(Red Hat Enterprise Linux CoreOS) 또는 RHEL(Red Hat Enterprise Linux)에서 호스팅되는 기존 Linux 워크로드도 제공할 수 있습니다. 자세한 내용은 Windows 컨테이너 워크로드 시작하기 를 참조하십시오.

클러스터에서 Windows 워크로드를 실행하려면 WMCO가 필요합니다. WMCO는 클러스터에서의 Windows 워크로드 배포 및 관리 프로세스를 오케스트레이션합니다. 자세한 내용은 Windows 컨테이너 워크로드를 활성화하는 방법을 참조하십시오.

지원되는 Windows 워크로드를 새 Windows 머신으로 이동할 수 있도록 Windows **MachineSet** 오브젝트를 생성하여 인프라 Windows 머신 세트 및 관련 머신을 생성할 수 있습니다. 여러 플랫폼에서 Windows **MachineSet** 오브젝트를 생성할 수 있습니다.

Windows 워크로드를 Windows 컴퓨팅 노드에 예약할 수 있습니다.

Windows Machine Config Operator 업그레이드를 수행하여 Windows 노드에 최신 업데이트가 있는지 확인할 수 있습니다.

특정 머신을 삭제하여 Windows 노드를 제거할 수 있습니다.

다음을 수행하여 Windows 컨테이너 워크로드를 비활성화할 수 있습니다.

- Windows Machine Config Operator 제거
- Windows Machine Config Operator 네임스페이스 삭제

2장. RED HAT OPENSHIFT 릴리스 노트를 위한 WINDOWS 컨테이너 지원

2.1. RED HAT OPENSHIFT를 위한 WINDOWS 컨테이너 지원 정보

Windows 컨테이너에 대한 Red Hat OpenShift 지원은 OpenShift Container Platform 클러스터에서 Windows 컴퓨팅 노드를 실행하는 기능을 제공하는 기능입니다. 이는 Red Hat WMCO(Windows Machine Config Operator)를 사용하여 Windows 노드를 설치 및 관리할 수 있습니다. 사용할 수 있는 Windows 노드를 통해 OpenShift Container Platform에서 Windows 컨테이너 워크로드를 실행할 수 있습니다.

Windows Containers에 대한 Red Hat OpenShift 지원의 릴리스 노트는 WMCO의 개발을 추적하며, 이는 OpenShift Container Platform의 모든 Windows 컨테이너 워크로드 기능을 제공합니다.

2.2. 지원 요청

Red Hat OpenShift에 대한 Windows 컨테이너 지원은 설치 가능한 선택적 구성 요소로 제공됩니다. Red Hat OpenShift에 대한 Windows 컨테이너 지원은 OpenShift Container Platform 서브스크립션의 일부가 아닙니다. 추가 Red Hat 서브스크립션이 필요하며 적용 범위 및 서비스 수준 계약에 따라 지원됩니다.

Red Hat OpenShift에 대한 Windows 컨테이너 지원에 대한 지원을 받으려면 이 별도의 서브스크립션이 있어야 합니다. 이러한 추가 Red Hat 서브스크립션이 없으면 프로덕션 클러스터에 Windows 컨테이너 워크로드를 배포할 수 없습니다. Red Hat 고객 포털을 통해 지원을 요청할 수 있습니다 .

자세한 내용은 Windows 컨테이너용 Red Hat OpenShift 지원에 대한 Red Hat OpenShift Container Platform 라이프 사이클 정책 문서를 참조하십시오.

이러한 추가 Red Hat 서브스크립션이 없는 경우 공식 지원이 없는 배포판인 Community Windows Machine Config Operator를 사용할 수 있습니다.

2.3. RED HAT WINDOWS MACHINE CONFIG OPERATOR 1.0.6 릴리스 정보

출시 날짜: 2022-02-16

WMCO 1.0.6을 버그 수정으로 사용할 수 있습니다. WMCO의 구성 요소는 RHBA-2022:0240 에서 릴리스되었습니다.

WMCO 1.0.2 릴리스에 설명된 지원 설명 및 알려진 문제는 이번 WMCO 릴리스에도 적용됩니다.

2.4. RED HAT WINDOWS MACHINE CONFIG OPERATOR 1.0.5 릴리스 노트

출시 날짜: 2021-06-14

이제 WMCO 1.0.5를 버그 수정으로 사용할 수 있습니다. WMCO의 구성 요소는 RHBA-2021:2142 에서 릴리스되었습니다.

WMCO 1.0.2 릴리스에 설명된 지원 설명 및 알려진 문제는 이번 WMCO 릴리스에도 적용됩니다.

2.5. RED HAT WINDOWS MACHINE CONFIG OPERATOR 1.0.4 릴리스 노트

출시 날짜: 2021-04-15

이제 버그 수정으로 WMCO 1.0.4를 사용할 수 있습니다. WMCO의 구성 요소는 RHBA-2021:1061 로 릴리스되었습니다.

WMCO 1.0.2 릴리스에 설명된 지원 설명 및 알려진 문제는 이번 WMCO 릴리스에도 적용됩니다.

2.5.1. 버그 수정

- 이전에는 두 개의 Windows 노드가 있는 클러스터가 있고 두 개의 복제본으로 웹 서버 배포를 생성한 경우 Pod는 각각 Windows 컴퓨팅 노드에 배치되었습니다. 이 시나리오에서는 **LoadBalancer** 유형의 **Service** 오브젝트를 생성한 경우 로드 밸런서 끝점과 통신하는 것이 정상이었습니다. 이 문제를 완화하려면 버전 10.0.17763.1457 이하에서 Windows Server 2019를 사용해야 했습니다. 이 문제는 해결되었으며 더 이상 Windows Server 2019 이미지 버전의 하위 집합으로 제한되지 않습니다. (**BZ#1942630**)

2.6. RED HAT WINDOWS MACHINE CONFIG OPERATOR 1.0.3 릴리스 노트

출시 날짜: 2021-02-15

이제 버그 수정 및 보안 업데이트와 함께 WMCO 1.0.3을 사용할 수 있습니다. WMCO의 구성 요소는 RHBA-2021:0410 으로 출시되었습니다.

중요

WMCO 2.x로 업그레이드하기 전에 WMCO 1.0.3+로 업그레이드해야 합니다.

WMCO 1.0.2 릴리스에 설명된 지원 설명 및 알려진 문제는 이번 WMCO 릴리스에도 적용됩니다.

2.7. RED HAT WINDOWS MACHINE CONFIG OPERATOR 1.0.2 릴리스 노트

출시 날짜: 2020-12-18

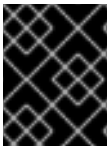
이번 WMCO 릴리스에서는 OpenShift Container Platform 클러스터에서 Windows 컴퓨팅 노드를 실행하기 위한 초기 지원을 제공합니다. WMCO의 구성 요소는 RHBA-2020:5596 으로 출시되었습니다.

WMCO는 다음 클라우드 공급자에서 실행되는 설치 관리자 프로비저닝 인프라를 사용하여 구축된 자체 관리 클러스터를 지원합니다.

- AWS(Amazon Web Services)
- Microsoft Azure

다음 Windows Server 운영 체제는 WMCO 초기 릴리스에서 지원됩니다.

- Windows Server LTSC(Long-Term Servicing Channel): Windows Server 2019



중요

제한된 네트워크 또는 연결이 끊긴 환경의 클러스터에서는 Windows 컨테이너 워크로드 실행이 지원되지 않습니다.

2.8. 알려진 제한 사항

WMCO(Windows 노드)에서 관리하는 Windows 노드로 작업할 때는 다음과 같은 제한 사항이 있습니다.

- 다음 OpenShift Container Platform 기능은 Windows 노드에서 지원되지 않습니다.
  - Red Hat OpenShift Developer CLI (odo)
  - 이미지 빌드
  - OpenShift Pipelines
  - OpenShift Service Mesh
  - 사용자 정의 프로젝트의 OpenShift 모니터링
  - OpenShift Serverless
  - 수평 Pod 자동 스케일링
  - 수직 Pod 자동 확장
- 다음 Red Hat 기능은 Windows 노드에서 지원되지 않습니다.
  - Red Hat 비용 관리
  - Red Hat OpenShift Local
- Windows 노드는 개인 레지스트리에서 컨테이너 이미지 가져오기를 지원하지 않습니다. 공용 레지스트리에서 이미지를 사용하거나 이미지를 사전 가져올 수 있습니다.
- Windows 노드는 배포 구성을 사용하여 생성된 워크로드를 지원하지 않습니다. 배포 또는 기타 방법을 사용하여 워크로드를 배포할 수 있습니다.
- 클러스터 전체 프록시를 사용하는 클러스터에서는 Windows 노드가 지원되지 않습니다. 이는 WMCO가 워크로드에 대한 프록시 연결을 통해 트래픽을 라우팅할 수 없기 때문입니다.
- 연결이 끊긴 환경에 있는 클러스터에서는 Windows 노드가 지원되지 않습니다.
- Windows Containers 용 Red Hat OpenShift 지원은 모든 클라우드 공급자에 대해 In-tree 스토리지 드라이버만 지원합니다.
- Kubernetes에서 다음 노드 기능 제한 사항을 확인했습니다.
  - Windows 컨테이너에서는 대규모 페이지가 지원되지 않습니다.
  - 권한이 있는 컨테이너는 Windows 컨테이너에서 지원되지 않습니다.
  - Pod 종료 유예 기간에는 containerd 컨테이너 런타임을 Windows 노드에 설치해야 합니다.
- Kubernetes는 여러 API 호환성 문제 를 확인했습니다.

3장. WINDOWS 컨테이너 워크로드 이해

Windows 컨테이너에 대한 Red Hat OpenShift 지원은 OpenShift Container Platform에서 Microsoft Windows Server 컨테이너를 실행하기 위한 기본 지원을 제공합니다. Linux 및 Windows 워크로드가 혼합된 동시 환경을 관리하는 경우 OpenShift Container Platform을 사용하면 RHCOS(Red Hat Enterprise Linux CoreOS) 또는 RHEL(Red Hat Enterprise Linux)에서 호스팅되는 기존 Linux 워크로드를 제공하면서 Windows Server 컨테이너에서 실행 중인 Windows 워크로드를 배포할 수 있습니다.

다음 클라우드 공급자에서 실행되는 클러스터의 경우 Windows 컨테이너 워크로드가 지원됩니다.

- AWS(Amazon Web Services)
- Microsoft Azure

OpenShift Container Platform 4.6에서 지원되는 Windows Server 운영 체제는 다음과 같습니다.

- Windows Server LTSC(Long-Term Servicing Channel): Windows Server 2019

자세한 내용은 Windows Server 채널에 대한 Microsoft 문서를 참조하십시오.

참고

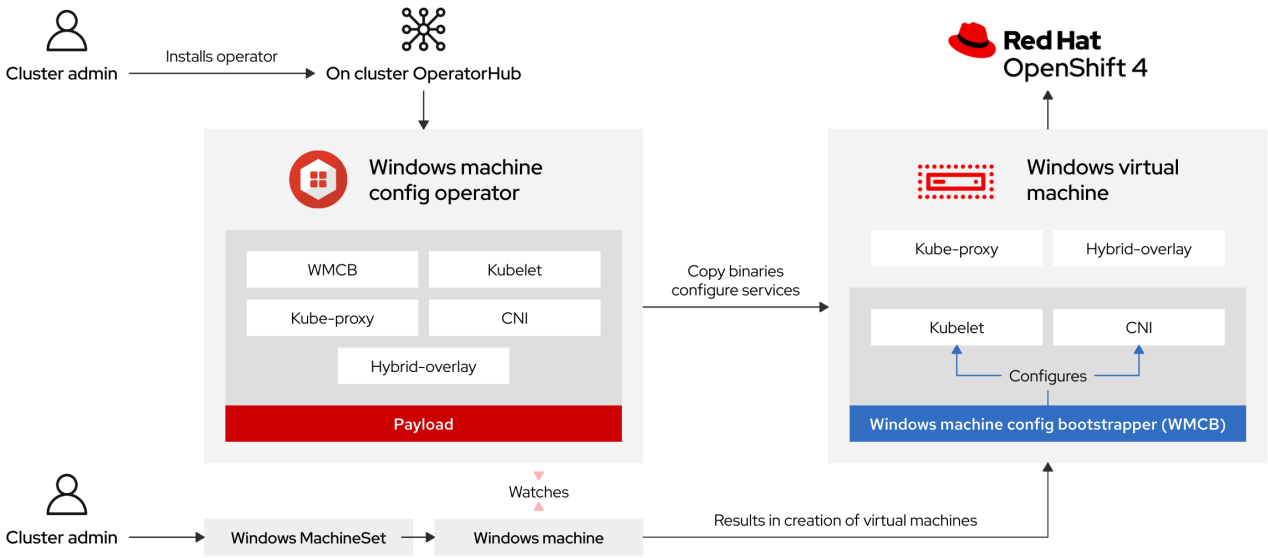
Windows 노드가 배치된 클러스터에 대한 멀티 테넌시는 지원되지 않습니다. 유해한 멀티 테넌트 사용으로 모든 Kubernetes 환경에서 보안 문제가 발생할 수 있습니다. Pod 보안 정책 또는 노드에서 보다 세밀하게 조정된 역할 기반 액세스 제어(RBAC)와 같은 추가 보안 기능으로 인해 노드를 악용하기가 더 어렵습니다. 그러나 유해한 멀티 테넌트 워크로드를 실행하도록 선택하는 경우 하이퍼바이저가 사용할 수 있는 유일한 보안 옵션입니다. Kubernetes용 보안 도메인에는 개별 노드가 아닌 전체 클러스터가 포함됩니다. 이러한 유형의 유해한 멀티 테넌트 워크로드의 경우 물리적으로 분리된 클러스터를 사용해야 합니다.

Windows Server 컨테이너는 공유 커널을 사용하여 리소스 격리를 제공하지만 다중 테넌시 시나리오에서는 사용되지 않습니다. 멀티 테넌시와 관련된 시나리오에서는 Hyper-V 격리 컨테이너를 사용하여 테넌트를 강력하게 격리해야 합니다.

3.1. WINDOWS 워크로드 관리

클러스터에서 Windows 워크로드를 실행하려면 먼저 WMCO(Windows Machine Config Operator)를 설치해야 합니다. WMCO는 Linux 기반 Operator로, Linux 기반 컨트롤 플레인 및 컴퓨팅 노드에서 실행됩니다. WMCO는 클러스터에서의 Windows 워크로드 배포 및 관리 프로세스를 오케스트레이션합니다.

그림 3.1. WMCO 설계



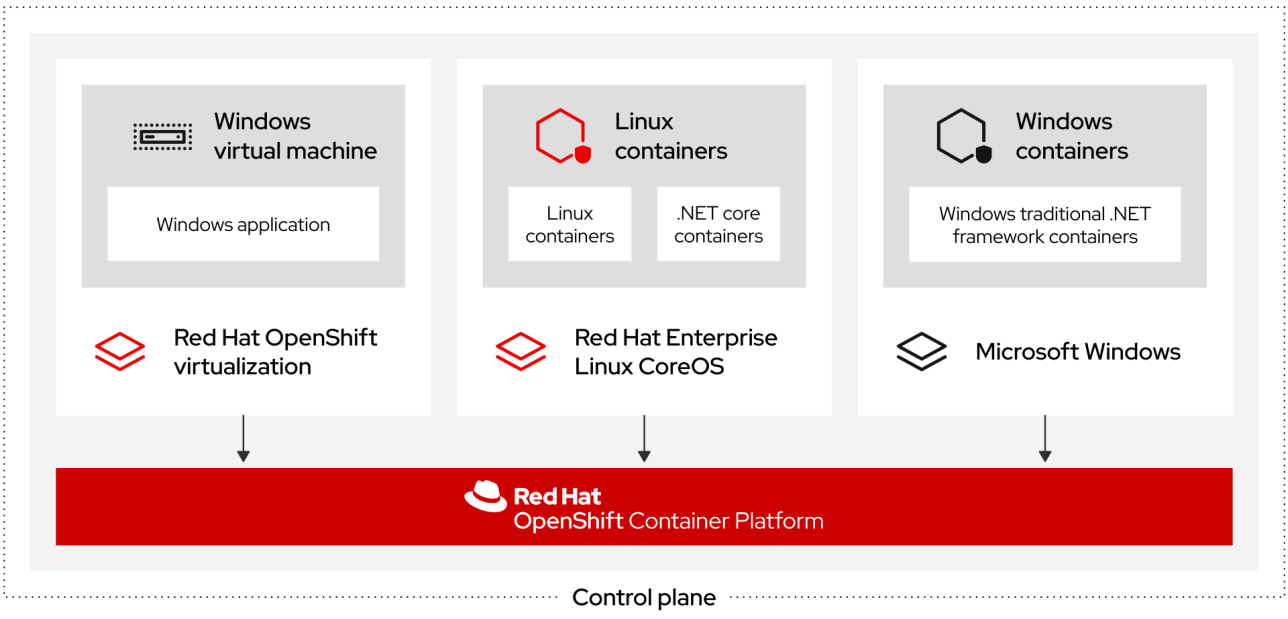
Windows 워크로드를 배포하기 전, Windows 컴퓨팅 노드를 생성한 후 클러스터에 참여하도록 해야 합니다. Windows 노드는 클러스터에서 Windows 워크로드를 호스팅하고 다른 Linux 기반 컴퓨팅 노드와 함께 실행할 수 있습니다. 호스트 Windows Server 컴퓨팅 머신으로 설정된 Windows 머신을 생성하여 Windows 컴퓨팅 노드를 생성할 수 있습니다. Docker 형식의 컨테이너 런타임 애드온이 활성화된 Windows OS 이미지를 지정하는 머신 세트에 Windows별 레이블을 적용해야 합니다.

중요

현재는 Docker 형식의 컨테이너 런타임이 Windows 노드에서 사용됩니다. Kubernetes는 더 이상 Docker를 컨테이너 런타임으로 사용하지 않습니다. Docker 사용 중지에 대한 자세한 내용은 Kubernetes 문서를 참조하십시오. 향후 Kubernetes 릴리스에서는 Containerd가 Windows 노드에서 지원되는 새로운 컨테이너 런타임이 됩니다.

WMCO는 Windows 레이블이 있는 머신을 감시합니다. Windows 머신 세트가 감지되고 각 머신이 프로비저닝되면 WMCO는 기본 Windows 가상 머신(VM)을 구성하여 클러스터에 컴퓨팅 노드로 참여할 수 있습니다.

그림 3.2. Windows 및 Linux 워크로드



WMCO는 Windows 인스턴스와 상호 작용하는 데 사용되는 개인 키가 포함된 네임스페이스에 사전 결정된 시크릿이 있을 것으로 예상합니다. WMCO는 부팅 시 이 시크릿을 확인하고 사용자가 생성한 Windows **MachineSet** 오브젝트에서 참조해야 하는 사용자 데이터 시크릿을 생성합니다. 그런 다음 WMCO는 사용자 데이터 시크릿을 개인 키에 해당하는 공개 키로 채웁니다. 이 데이터를 사용하여 SSH 연결을 통해 클러스터가 Windows VM에 연결할 수 있습니다.

클러스터가 Windows VM과 관련된 연결을 설정한 후 Linux 기반 노드와 비슷한 방법을 사용하여 Windows 노드를 관리할 수 있습니다.

참고

OpenShift Container Platform 웹 콘솔은 Windows 노드에 대한 노드 그래프 및 워크로드 그래프를 제공하지 않습니다. 현재 Windows 노드에 사용할 수 있는 지표가 없습니다.

Windows 노드에 Windows 워크로드 예약은 테인트, 허용 오차 및 노트 선택기와 같은 일반적인 Pod 예약 방법으로 수행할 수 있습니다. 아니면, **RuntimeClass** 오브젝트를 사용하여 Windows 워크로드를 Linux 워크로드 및 기타 Windows 버전 워크로드와 차별화할 수 있습니다.

3.2. WINDOWS 노드 서비스

다음 Windows별 서비스가 각 Windows 노드에 설치됩니다.

Service	설명
kubelet	Windows 노드를 등록하고 상태를 관리합니다.
CNI(컨테이너 네트워크 인터페이스) 플러그인	Windows 노드에 대한 네트워킹을 노출합니다.
WMCB(Windows Machine Config Bootstrapper)	kubelet 및 CNI 플러그인을 구성합니다.
hybrid-overlay	OpenShift Container Platform HNS(Host Network Service)를 생성합니다.
kube-proxy	외부 통신을 허용하는 노드에서 네트워크 규칙을 유지합니다.

3.3. 알려진 제한 사항

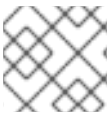
WMCO(Windows 노드)에서 관리하는 Windows 노드로 작업할 때는 다음과 같은 제한 사항이 있습니다.

- 다음 OpenShift Container Platform 기능은 Windows 노드에서 지원되지 않습니다.
  - Red Hat OpenShift Developer CLI (odo)
  - 이미지 빌드
  - OpenShift Pipelines
  - OpenShift Service Mesh
  - 사용자 정의 프로젝트의 OpenShift 모니터링

  - OpenShift Serverless
  - 수평 Pod 자동 스케일링
  - 수직 Pod 자동 확장
- 다음 Red Hat 기능은 Windows 노드에서 지원되지 않습니다.
  - Red Hat 비용 관리
  - Red Hat OpenShift Local
- Windows 노드는 개인 레지스트리에서 컨테이너 이미지 가져오기를 지원하지 않습니다. 공용 레지스트리에서 이미지를 사용하거나 이미지를 사전 가져올 수 있습니다.
- Windows 노드는 배포 구성을 사용하여 생성된 워크로드를 지원하지 않습니다. 배포 또는 기타 방법을 사용하여 워크로드를 배포할 수 있습니다.
- 클러스터 전체 프록시를 사용하는 클러스터에서는 Windows 노드가 지원되지 않습니다. 이는 WMCO가 워크로드에 대한 프록시 연결을 통해 트래픽을 라우팅할 수 없기 때문입니다.
- 연결이 끊긴 환경에 있는 클러스터에서는 Windows 노드가 지원되지 않습니다.
- Windows Containers 용 Red Hat OpenShift 지원은 모든 클라우드 공급자에 대해 In-tree 스토리지 드라이버만 지원합니다.
- Kubernetes에서 다음 노드 기능 제한 사항을 확인했습니다.
  - Windows 컨테이너에서는 대규모 페이지가 지원되지 않습니다.
  - 권한이 있는 컨테이너는 Windows 컨테이너에서 지원되지 않습니다.
  - Pod 종료 유예 기간에는 containerd 컨테이너 런타임을 Windows 노드에 설치해야 합니다.
- Kubernetes는 여러 API 호환성 문제 를 확인했습니다.

4장. WINDOWS 컨테이너 워크로드 활성화

Windows 워크로드를 클러스터에 추가하기 전에 OpenShift Container Platform OperatorHub에서 사용 가능한 WMCO(Windows Machine Config Operator)를 설치해야 합니다. WMCO는 클러스터에서의 Windows 워크로드 배포 및 관리 프로세스를 오케스트레이션합니다.



참고

WMCO에서 관리하는 Windows 인스턴스에서 듀얼 NIC가 지원되지 않습니다.

사전 요구 사항

- **cluster-admin** 권한이 있는 계정을 사용하여 OpenShift Container Platform 클러스터에 액세스할 수 있습니다.
- OpenShift CLI(**oc**)가 설치되어 있습니다.
- 설치 관리자 프로비저닝 인프라를 사용하여 클러스터가 설치되어 있습니다. 사용자 프로비저닝 인프라로 설치된 클러스터는 Windows 컨테이너 워크로드에서 지원되지 않습니다.
- 클러스터를 위한 OVN-Kubernetes를 사용하여 하이브리드 네트워킹이 구성되었습니다. 클러스터를 설치하는 동안 완료해야 합니다. 자세한 내용은 하이브리드 네트워킹 구성 을 참조하십시오.
- OpenShift Container Platform 클러스터 버전 4.6.8 이상이 실행 중입니다.

참고

WMCO는 워크로드에 대한 프록시 연결을 통해 트래픽을 라우팅할 수 없기 때문에 클러스터 전체 프록시 를 사용하는 클러스터에서는 WMCO가 지원되지 않습니다.

4.1. WINDOWS MACHINE CONFIG OPERATOR 설치

웹 콘솔 또는 OpenShift CLI(**oc**)를 사용하여 Windows Machine Config Operator를 설치할 수 있습니다.

4.1.1. 웹 콘솔을 사용하여 Windows Machine Config Operator 설치

OpenShift Container Platform 웹 콘솔을 사용하여 WMCO(Windows Machine Config Operator)를 설치할 수 있습니다.

참고

WMCO에서 관리하는 Windows 인스턴스에서 듀얼 NIC가 지원되지 않습니다.

절차

1. OpenShift Container Platform 웹 콘솔에서 **관리자**로 **Operator → OperatorHub** 페이지로 이동합니다.
2. **키워드로 필터링** 상자를 사용하여 카탈로그에서 **Windows Machine Config Operator**를 검색합니다. **Windows Machine Config Operator** 타일을 클릭합니다.
3. Operator에 대한 정보를 확인하고 **설치**를 클릭합니다.
4. **Operator 설치** 페이지에서 다음을 수행합니다.

a. **stable** 채널을 **업데이트 채널**로 선택합니다. **stable** 채널을 사용하면 WMCO의 안정적인 최신 릴리스를 설치할 수 있습니다.

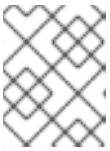
b. WMCO는 단일 네임스페이스에서만 사용 가능해야 하므로 **설치 모드**가 사전 구성됩니다.

c. WMCO용으로 **설치된 네임스페이스**를 선택합니다. 기본 Operator 권장 네임스페이스는 **openshift-windows-machine-config-operator**입니다.

d. **승인 전략**을 선택합니다.

- **자동** 전략을 사용하면 Operator 새 버전이 준비될 때 OLM(Operator Lifecycle Manager)이 자동으로 Operator를 업데이트할 수 있습니다.
- **수동** 전략을 사용하려면 적절한 자격 증명을 가진 사용자가 Operator 업데이트를 승인해야 합니다.

1. **설치**를 클릭합니다. **설치된 Operator** 페이지에 이제 WMCO가 나열됩니다.



참고

WMCO는 사용자가 정의한 네임스페이스에 **openshift-windows-machine-config-operator**와 같이 자동으로 설치됩니다.

2. WMCO가 성공적으로 설치되었는지 확인하려면 **상태**가 **성공**으로 표시되는지 확인합니다.

4.1.2. CLI를 사용하여 Windows Machine Config Operator 설치

OpenShift CLI(**oc**)를 사용하여 WMCO(Windows Machine Config Operator)를 설치할 수 있습니다.

참고

WMCO에서 관리하는 Windows 인스턴스에서 듀얼 NIC가 지원되지 않습니다.

절차

1. WMCO를 위한 네임스페이스를 생성합니다.

a. WMCO를 위한 **네임스페이스** 오브젝트 YAML 파일을 생성합니다. 예를 들어, **wmco-namespace.yaml**은 다음과 같습니다.

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-windows-machine-config-operator 1
```

1 **openshift-windows-machine-config-operator** 네임스페이스에 WMCO를 배포하는 것이 좋습니다.

b. 네임스페이스를 생성합니다.

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

예를 들면 다음과 같습니다.

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

2. WMCO를 위한 Operator 그룹을 생성합니다.

   a. **OperatorGroup** 오브젝트 YAML 파일을 생성합니다. 예를 들어, **wmco-og.yaml**은 다음과 같습니다.

```
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 그룹을 생성합니다.

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

   예를 들면 다음과 같습니다.

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

3. 네임스페이스가 WMCO를 구독하도록 합니다.

   a. **서브스크립션** 오브젝트 YAML 파일을 생성합니다. 예를 들어, **wmco-sub.yaml**은 다음과 같습니다.

```
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**을 채널로 지정합니다.

   (2) 승인 전략을 설정합니다. **자동** 또는 **수동**을 설정할 수 있습니다.

   (3) **windows-machine-config-operator** 패키지 매니페스트가 포함된 **redhat-operators** 카탈로그 소스를 지정합니다. OpenShift Container Platform이 제한된 네트워크(연결이 끊긴 클러스터)에 설치된 경우 OLM(Operator Lifecycle Manager)을 구성할 때 생성된 **CatalogSource** 오브젝트의 이름을 지정합니다.

   (4) 카탈로그 소스의 네임스페이스입니다. 기본 OperatorHub 카탈로그 소스에는 **openshift-marketplace**를 사용합니다.

b. 서브스크립션을 생성합니다.

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

예를 들면 다음과 같습니다.

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

이제 WMCO가 **openshift-windows-machine-config-operator**에 설치됩니다.

4. WMCO 설치를 확인합니다.

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

출력 예

```
NAME                                 DISPLAY                          VERSION   REPLACES   PHASE
windows-machine-config-operator.1.0.0   Windows Machine Config Operator   1.0.0
Succeeded
```

4.2. WINDOWS MACHINE CONFIG OPERATOR에 대한 시크릿 구성

WMCO(Windows Machine Config Operator)를 실행하려면 개인 키가 포함된 WMCO 네임스페이스에 시크릿을 생성해야 합니다. 이를 위해서는 WMCO가 Windows VM(가상 머신)과 통신할 수 있도록 허용되어야 합니다.

사전 요구 사항

- OLM(Operator Lifecycle Manager)을 사용하여 WMCO(Windows Machine Config Operator)를 설치했습니다.
- RSA 키가 포함된 PEM 인코딩 파일이 생성되었습니다.

절차

- Windows VM에 액세스하기 위해 필요한 시크릿을 정의합니다.

```
$ 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**와 같이 WMCO 네임스페이스에 개인 키를 생성해야 합니다.

클러스터를 설치할 때 사용한 것과 다른 개인 키를 사용하는 것이 좋습니다.

추가 리소스

- SSH 개인 키 생성 및 에이전트에 추가
- 클러스터에 Operator 추가.

5장. WINDOWS MACHINESET 오브젝트 생성

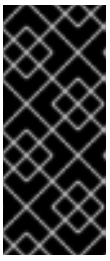
5.1. AWS에서 WINDOWS MACHINESET 오브젝트 생성

AWS(Amazon Web Services)의 OpenShift Container Platform 클러스터에서 특정 목적을 수행하기 위해 Windows **MachineSet** 오브젝트를 생성할 수 있습니다. 예를 들어, 지원되는 워크로드를 새 Windows 머신으로 이동할 수 있도록 인프라 Windows MachineSet 및 관련 머신을 생성할 수 있습니다.

사전 요구 사항

- OLM(Operator Lifecycle Manager)을 사용하여 WMCO(Windows Machine Config Operator)를 설치했습니다.
- 지원되는 Windows Server를 Docker 형식 컨테이너 런타임 애드온이 활성화된 운영 체제 이미지로 사용하고 있습니다.
  다음 **aws** 명령을 사용하여 유효한 AMI 이미지를 쿼리합니다.

```
$ aws ec2 describe-images --region <aws region name> --filters
"Name=name,Values=Windows_Server-2019*English*Full*Containers*" "Name=is-
public,Values=true" --query "reverse(sort_by(Images, &CreationDate))[*].{name: Name, id:
ImageId}" --output table
```



중요

현재는 Docker 형식의 컨테이너 런타임이 Windows 노드에서 사용됩니다. Kubernetes는 더 이상 Docker를 컨테이너 런타임으로 사용하지 않습니다. Docker 사용 중지에 대한 자세한 내용은 Kubernetes 문서를 참조하십시오. 향후 Kubernetes 릴리스에서는 Containerd가 Windows 노드에서 지원되는 새로운 컨테이너 런타임이 됩니다.

5.1.1. Machine API 개요

Machine API는 업스트림 Cluster API 프로젝트 및 사용자 정의 OpenShift Container Platform 리소스를 기반으로하는 주요 리소스의 조합입니다.

OpenShift Container Platform 4.6 클러스터의 경우 Machine API는 클러스터 설치가 완료된 후 모든 노드 호스트 프로비저닝 관리 작업을 수행합니다. 이 시스템으로 인해 OpenShift Container Platform 4.6은 퍼블릭 또는 프라이빗 클라우드 인프라에 더하여 탄력적이고 동적인 프로비저닝 방법을 제공합니다.

두 가지 주요 리소스는 다음과 같습니다.

Machine

노드의 호스트를 설명하는 기본 단위입니다. 머신에는 **providerSpec** 사양이 있으며 이는 다른 클라우드 플랫폼에 제공되는 컴퓨팅 노드 유형을 설명합니다. 예를 들어 AWS(Amazon Web Services)의 작업자 노드에 대한 머신 유형은 특정 머신 유형과 필요한 메타 데이터를 정의할 수 있습니다.

머신 세트

MachineSet 리소스는 머신 그룹입니다. 머신 세트는 머신에 연관되어 있고 복제본 세트는 pod에 연관되어 있습니다. 더 많은 머신이 필요하거나 규모를 줄여야 하는 경우 컴퓨터 요구 사항에 맞게 머신 세트의 **replicas** 필드를 변경합니다.

주의

컨트롤 플레인 시스템은 머신 세트에서 관리할 수 없습니다.

다음 사용자 지정 리소스는 클러스터에 더 많은 기능을 추가할 수 있습니다.

머신 자동 스케일러

MachineAutoscaler 리소스는 클라우드에서 머신을 자동으로 확장합니다. 지정된 머신 세트에서 노드의 최소 및 최대 스케일링 경계를 설정할 수 있으며 머신 자동 스케일러는 해당 노드 범위를 유지합니다. **MachineAutoscaler** 객체는 **ClusterAutoscaler** 객체를 설정한 후에 사용할 수 있습니다. **ClusterAutoscaler** 및 **MachineAutoscaler**리소스는 모두 **ClusterAutoscalerOperator** 오브젝트에서 사용 가능합니다.

Cluster autoscaler

이 리소스는 업스트림 클러스터 자동 스케일러 프로젝트를 기반으로 합니다. OpenShift Container Platform 구현에서는 머신 세트 API를 확장하여 Machine API와 통합됩니다. 코어, 노드, 메모리, GPU 등과 같은 리소스의 클러스터 전체에서 확장 제한을 설정할 수 있습니다. 중요도가 낮은 Pod에 대해 새 노드가 온라인 상태가 되지 않도록 클러스터가 Pod에 우선 순위를 설정할 수 있습니다. 노드를 확장할 수는 있지만 축소할 수 없도록 확장 정책을 설정할 수도 있습니다.

머신 상태 점검

MachineHealthCheck 리소스는 머신의 비정상적인 상태를 감지하여 삭제한 후 지원되는 플랫폼에서 새 머신을 생성합니다.

OpenShift Container Platform 버전 3.11에서는 클러스터가 머신 프로비저닝을 관리하지 않았기 때문에 다중 영역 아키텍처를 쉽게 롤아웃할 수 없었습니다. OpenShift Container Platform 버전 4.1부터 이러한 프로세스가 더 쉬워졌습니다. 각 머신 세트의 범위는 단일 영역에서 지정되므로 설치 프로그램은 사용자를 대신하여 가용성 영역 전체에 머신 세트를 보냅니다. 또한 계산이 동적이고 영역 장애가 발생하여 머신을 재조정해야하는 경우 처리할 수 있는 영역을 확보할 수 있습니다. Autoscaler는 클러스터의 수명 기간 동안 최적의 균형을 유지합니다.

5.1.2. AWS에서 Windows MachineSet 오브젝트를 위한 샘플 YAML

이 샘플 YAML은 WMCO(Windows Machine Config Operator)에서 응답할 수 있는 AWS(Amazon Web Services)에서 실행 중인 Windows **MachineSet** 오브젝트를 정의합니다.

```
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 클러스터를 프로비저닝할 때 설정한 클러스터 ID를 기반으로 하는 인프라 ID를 지정합니다. 다음 명령을 실행하여 인프라 ID를 가져올 수 있습니다.

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

2 4 6 인프라 ID, 작업자 레이블 및 영역을 지정합니다.

7 머신 세트를 Windows 머신으로 구성합니다.

8 Windows 노드를 컴퓨팅 머신으로 구성합니다.

9 컨테이너 런타임이 설치된 Windows 이미지의 AMI ID를 지정합니다. 버전 10.0.17763.1457 또는 이전 버전의 Windows Server 2019를 사용해야 합니다.

11 **us-east-1a**와 같이 AWS 영역을 지정합니다.

12 **us-gov-east-1**과 같이 AWS 리전을 지정합니다.

16 첫 번째 Windows 머신을 구성할 때 WMCO에 의해 생성되었습니다. 이후에는, 모든 후속 머신 세트에서 **Windows-user-data**를 사용할 수 있습니다.

5.1.3. 머신 세트 만들기

설치 프로그램에 의해 생성되는 것 이외에도 고유한 머신 세트를 만들어 선택한 특정 워크로드의 머신 컴퓨팅 리소스를 동적으로 관리할 수 있습니다.

사전 요구 사항

- OpenShift Container Platform 클러스터를 배포합니다.
- OpenShift CLI(**oc**)를 설치합니다.
- **cluster-admin** 권한이 있는 사용자로 **oc**에 로그인합니다.

절차

1. 머신 세트 CR(사용자 지정 리소스) 샘플이 포함된 이름이 **<file_name>.yaml**인 새 YAML 파일을 만듭니다.
   <clusterID> 및 **<role>** 매개 변수 값을 설정해야 합니다.

   a. 특정 필드에 설정할 값이 확실하지 않은 경우 클러스터에서 기존 머신 세트를 확인할 수 있습니다.

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

   출력 예

   ```
   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. 특정 머신 세트의 값을 확인합니다.

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

출력 예

```
...
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. 클러스터 ID입니다.
2. 기본 노드 레이블입니다.

2. 새 **MachineSet** CR을 만듭니다.

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

3. 머신 세트 목록을 표시합니다.

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

출력 예

```
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
```

새 머신 세트가 사용 가능한 경우**DESIRED** 및 **CURRENT** 값이 일치합니다. 머신 세트를 사용할 수 없는 경우 몇 분 후에 명령을 다시 실행합니다.

5.1.4. 추가 리소스

- 머신 세트 관리에 대한 자세한 내용은 *머신 관리* 섹션을 참조하십시오.

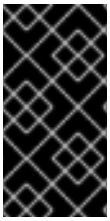
5.2. AZURE에서 WINDOWS MACHINESET 오브젝트 만들기

Microsoft Azure의 OpenShift Container Platform 클러스터에서 특정 목적을 수행하기 위해 Windows **MachineSet** 오브젝트를 생성할 수 있습니다. 예를 들어, 지원되는 워크로드를 새 Windows 머신으로 이동할 수 있도록 인프라 Windows MachineSet 및 관련 머신을 생성할 수 있습니다.

사전 요구 사항

- OLM(Operator Lifecycle Manager)을 사용하여 WMCO(Windows Machine Config Operator)를 설치했습니다.

- 지원되는 Windows Server를 Docker 형식 컨테이너 런타임 애드온이 활성화된 운영 체제 이미지로 사용하고 있습니다.



중요

현재는 Docker 형식의 컨테이너 런타임이 Windows 노드에서 사용됩니다. Kubernetes는 더 이상 Docker를 컨테이너 런타임으로 사용하지 않습니다. Docker 사용 중지에 대한 자세한 내용은 Kubernetes 문서를 참조하십시오. 향후 Kubernetes 릴리스에서는 Containerd가 Windows 노드에서 지원되는 새로운 컨테이너 런타임이 됩니다.

5.2.1. Machine API 개요

Machine API는 업스트림 Cluster API 프로젝트 및 사용자 정의 OpenShift Container Platform 리소스를 기반으로하는 주요 리소스의 조합입니다.

OpenShift Container Platform 4.6 클러스터의 경우 Machine API는 클러스터 설치가 완료된 후 모든 노드 호스트 프로비저닝 관리 작업을 수행합니다. 이 시스템으로 인해 OpenShift Container Platform 4.6은 퍼블릭 또는 프라이빗 클라우드 인프라에 더하여 탄력적이고 동적인 프로비저닝 방법을 제공합니다.

두 가지 주요 리소스는 다음과 같습니다.

Machine

노드의 호스트를 설명하는 기본 단위입니다. 머신에는 **providerSpec** 사양이 있으며 이는 다른 클라우드 플랫폼에 제공되는 컴퓨팅 노드 유형을 설명합니다. 예를 들어 AWS(Amazon Web Services)의 작업자 노드에 대한 머신 유형은 특정 머신 유형과 필요한 메타 데이터를 정의할 수 있습니다.

머신 세트

MachineSet 리소스는 머신 그룹입니다. 머신 세트는 머신에 연관되어 있고 복제본 세트는 pod에 연관되어 있습니다. 더 많은 머신이 필요하거나 규모를 줄여야 하는 경우 컴퓨터 요구 사항에 맞게 머신 세트의 **replicas** 필드를 변경합니다.

주의

컨트롤 플레인 시스템은 머신 세트에서 관리할 수 없습니다.

다음 사용자 지정 리소스는 클러스터에 더 많은 기능을 추가할 수 있습니다.

머신 자동 스케일러

MachineAutoscaler 리소스는 클라우드에서 머신을 자동으로 확장합니다. 지정된 머신 세트에서 노드의 최소 및 최대 스케일링 경계를 설정할 수 있으며 머신 자동 스케일러는 해당 노드 범위를 유지합니다. **MachineAutoscaler** 객체는 **ClusterAutoscaler** 객체를 설정한 후에 사용할 수 있습니다. **ClusterAutoscaler** 및 **MachineAutoscaler**리소스는 모두 **ClusterAutoscalerOperator** 오브젝트에서 사용 가능합니다.

Cluster autoscaler

이 리소스는 업스트림 클러스터 자동 스케일러 프로젝트를 기반으로 합니다. OpenShift Container Platform 구현에서는 머신 세트 API를 확장하여 Machine API와 통합됩니다. 코어, 노드, 메모리, GPU 등과 같은 리소스의 클러스터 전체에서 확장 제한을 설정할 수 있습니다. 중요도가 낮은 Pod에 대해 새

노드가 온라인 상태가 되지 않도록 클러스터가 Pod에 우선 순위를 설정할 수 있습니다. 노드를 확장할 수는 있지만 축소할 수 없도록 확장 정책을 설정할 수도 있습니다.

머신 상태 점검

MachineHealthCheck 리소스는 머신의 비정상적인 상태를 감지하여 삭제한 후 지원되는 플랫폼에서 새 머신을 생성합니다.

OpenShift Container Platform 버전 3.11에서는 클러스터가 머신 프로비저닝을 관리하지 않았기 때문에 다중 영역 아키텍처를 쉽게 롤아웃할 수 없었습니다. OpenShift Container Platform 버전 4.1부터 이러한 프로세스가 더 쉬워졌습니다. 각 머신 세트의 범위는 단일 영역에서 지정되므로 설치 프로그램은 사용자를 대신하여 가용성 영역 전체에 머신 세트를 보냅니다. 또한 계산이 동적이고 영역 장애가 발생하여 머신을 재조정해야하는 경우 처리할 수 있는 영역을 확보할 수 있습니다. Autoscaler는 클러스터의 수명 기간 동안 최적의 균형을 유지합니다.

5.2.2. Azure에서 Windows MachineSet 오브젝트를 위한 샘플 YAML

이 샘플 YAML은 Microsoft Azure에서 WMCO(Windows Machine Config Operator)가 응답할 수 있는 Windows **MachineSet** 오브젝트를 정의합니다.

```
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 클러스터를 프로비저닝할 때 설정한 클러스터 ID를 기반으로 하는 인프라 ID를 지정합니다. 다음 명령을 실행하여 인프라 ID를 가져올 수 있습니다.

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

2 4 6 Windows 머신 세트의 이름을 지정합니다. Azure의 Windows 머신 이름은 15자 이하여야 합니다. 따라서 머신 세트 이름은 머신 이름이 생성되는 방식으로 인해 9자를 초과할 수 없습니다.

7 머신 세트를 Windows 머신으로 구성합니다.

8 Windows 노드를 컴퓨팅 머신으로 구성합니다.

9 **17763.1457.2009030514** 이상 버전의 **2019-Datacenter-with-Containers** SKU를 정의하는 **WindowsServer** 이미지 제공을 지정합니다.

10 **centralus**와 같이 Azure 리전을 지정합니다.

14 첫 번째 Windows 머신을 구성할 때 WMCO에 의해 생성되었습니다. 이후에는, 모든 후속 머신 세트에서 **Windows-user-data**를 사용할 수 있습니다.

16 머신을 배치할 리전 내 영역을 지정합니다. 해당 리전이 지정한 영역을 지원하는지 확인합니다.

5.2.3. 머신 세트 만들기

설치 프로그램에 의해 생성되는 것 이외에도 고유한 머신 세트를 만들어 선택한 특정 워크로드의 머신 컴퓨팅 리소스를 동적으로 관리할 수 있습니다.

사전 요구 사항

- OpenShift Container Platform 클러스터를 배포합니다.
- OpenShift CLI(**oc**)를 설치합니다.
- **cluster-admin** 권한이 있는 사용자로 **oc**에 로그인합니다.

절차

1. 머신 세트 CR(사용자 지정 리소스) 샘플이 포함된 이름이 **<file_name>.yaml**인 새 YAML 파일을 만듭니다.
   <clusterID> 및 **<role>** 매개 변수 값을 설정해야 합니다.

   a. 특정 필드에 설정할 값이 확실하지 않은 경우 클러스터에서 기존 머신 세트를 확인할 수 있습니다.

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

   출력 예

   ```
   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. 특정 머신 세트의 값을 확인합니다.

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

   출력 예

   ```
   ...
   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) 클러스터 ID입니다.

   (2) 기본 노드 레이블입니다.

2. 새 **MachineSet** CR을 만듭니다.

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

3. 머신 세트 목록을 표시합니다.

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

   출력 예

   ```
   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
```

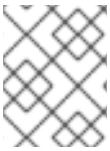
새 머신 세트가 사용 가능한 경우**DESIRED** 및 **CURRENT** 값이 일치합니다. 머신 세트를 사용할 수 없는 경우 몇 분 후에 명령을 다시 실행합니다.

5.2.4. 추가 리소스

- 머신 세트 관리에 대한 자세한 내용은 *머신 관리* 섹션을 참조하십시오.

6장. WINDOWS 컨테이너 워크로드 예약

Windows 워크로드를 Windows 컴퓨팅 노드에 예약할 수 있습니다.



참고

WMCO는 워크로드에 대한 프록시 연결을 통해 트래픽을 라우팅할 수 없기 때문에 클러스터 전체 프록시 를 사용하는 클러스터에서는 WMCO가 지원되지 않습니다.

사전 요구 사항

- OLM(Operator Lifecycle Manager)을 사용하여 WMCO(Windows Machine Config Operator)를 설치했습니다.
- Docker 형식의 컨테이너 런타임 애드온이 활성화된 OS 이미지로 Windows 컨테이너를 사용하고 있습니다.
- Windows 머신 세트를 생성했습니다.

중요

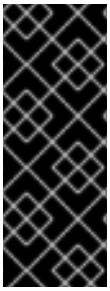
현재는 Docker 형식의 컨테이너 런타임이 Windows 노드에서 사용됩니다. Kubernetes는 더 이상 Docker를 컨테이너 런타임으로 사용하지 않습니다. Docker 사용 중지 에 대한 자세한 내용은 Kubernetes 문서를 참조하십시오. 향후 Kubernetes 릴리스에서는 Containerd가 Windows 노드에서 지원되는 새로운 컨테이너 런타임이 됩니다.

6.1. WINDOWS POD 배치

Windows 워크로드를 클러스터에 배포하기 전에 Pod가 올바르게 할당되도록 Windows 노드 스케줄링을 구성해야 합니다. Windows 노드를 호스팅하는 머신이 있으므로 Linux 기반 노드와 동일하게 관리됩니다. 유사하게, 테인트, 허용 오차, 노드 선택기와 같은 방법을 사용하여 적절한 Windows 노드에 Windows Pod도 예약되어야 합니다.

동일한 클러스터에서 여러 Windows OS 변형을 실행하는 기능과 동일한 클러스터에서 여러 Windows OS 변형을 실행할 수 있는 경우 **RuntimeClass** 오브젝트를 사용하여 Windows Pod를 기본 Windows OS 변형에 매핑해야 합니다. 예를 들어, 다른 Windows Server 컨테이너 버전에서 여러 Windows 노드가 있는 경우 클러스터는 호환되지 않는 Windows OS 변형에 Windows Pod를 예약할 수 있습니다. 클러스터의 각 Windows OS 변형에 대해 **RuntimeClass** 오브젝트가 구성되어 있어야 합니다. 클러스터에서 사용 가능한 Windows OS 변형만 있는 경우에는 **RuntimeClass** 오브젝트를 사용하는 것이 좋습니다.

자세한 내용은 호스트 및 컨테이너 버전 호환성 에 대한 Microsoft 문서를 참조하십시오.



중요

컨테이너 기본 이미지는 사용자가 예약해야 하는 노드에서 실행 중인 것과 동일한 Windows OS 버전이어야 합니다.

또한 Windows 노드를 한 버전에서 다른 버전으로 업그레이드하는 경우(예: 20H2에서 2022로 이동) 컨테이너 기본 이미지를 새 버전과 일치시켜야 합니다. 자세한 내용은 Windows 컨테이너 버전 호환성 을 참조하십시오.

추가 리소스

- 스케줄러를 사용하여 Pod 배치 제어

- 노드 테인트를 사용하여 Pod 배치 제어
- 노드 선택기를 사용하여 특정 노드에 Pod 배치

6.2. 스케줄링 메커니즘을 캡슐화하기 위해 RUNTIMECLASS 오브젝트 생성

RuntimeClass 오브젝트를 사용하면 테인트 및 허용 오차와 같은 스케줄링 방식을 편리하게 사용할 수 있습니다. 사용자는 테인트 및 허용 오차를 캡슐화는 런타임 클래스를 배포한 후 이를 Pod에 적용하여 적절한 노드에 예약할 수 있습니다. 여러 운영 체제 변형을 지원하는 클러스터에도 런타임 클래스를 생성해야 합니다.

절차

1. **RuntimeClass** 오브젝트 YAML 파일을 생성합니다. 예를 들어, **runtime-class.yaml**은 다음과 같습니다.

```
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. **RuntimeClass** 오브젝트 이름을 지정하며, 이는 이 런타임 클래스로 관리할 Pod에 정의됩니다.
2. 이 런타임 클래스를 지원하는 노드에 존재해야 하는 레이블을 지정합니다. 이 런타임 클래스를 사용하는 Pod는 이 선택기와 일치하는 노드에만 예약할 수 있습니다. 런타임 클래스의 노드 선택기는 Pod의 기존 노드 선택기와 병합됩니다. 충돌이 발생하면 Pod를 노드에 예약할 수 없습니다.
3. 허용 중에 이 런타임 클래스와 함께 실행 중인 pod(중복 제외)에 추가하려면 허용 오차를 지정합니다. 이 작업을 수행하면 Pod 및 런타임 클래스에서 허용되는 노드 집합이 결합됩니다.

2. **RuntimeClass** 오브젝트를 생성합니다.

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

예를 들면 다음과 같습니다.

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

3. Pod에 **RuntimeClass** 오브젝트를 적용하여 적절한 운영 체제 변형에 예약되어 있는지 확인합니다.

```
apiVersion: v1
kind: Pod
metadata:
  name: my-windows-pod
spec:
  runtimeClassName: <runtime_class_name> 1
...
```

1 Pod 예약을 관리할 런타임 클래스를 지정합니다.

6.3. 샘플 WINDOWS 컨테이너 워크로드 배포

Windows 컴퓨팅 노드를 사용 가능한 경우 Windows 컨테이너 워크로드를 클러스터에 배포할 수 있습니다.

참고

이 샘플 배포는 참조용으로만 제공됩니다.

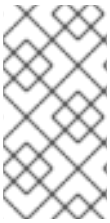
예시 **Service** 오브젝트

```
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
```

예시 **Deployment** 오브젝트

```
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:
        runAsNonRoot: false
        windowsOptions:
          runAsUserName: "ContainerAdministrator"
    nodeSelector:
      kubernetes.io/os: windows
```



참고

mcr.microsoft.com/powershell:<tag> 컨테이너 이미지를 사용하는 경우 이 명령을 **pwsh.exe**로 정의해야 합니다. **mcr.microsoft.com/windows/servercore:<tag>** 컨테이너 이미지를 사용하는 경우 해당 명령을 **powershell.exe**로 정의해야 합니다. 자세한 내용은 Microsoft 문서를 참조하십시오.

6.4. 머신 세트 수동 스케일링

머신 세트에서 머신 인스턴스를 추가하거나 제거하려면 머신 세트를 수동으로 스케일링할 수 있습니다.

이는 완전히 자동화된 설치 프로그램에 의해 프로비저닝된 인프라 설치와 관련이 있습니다. 사용자 지정된 사용자 프로비저닝 인프라 설치에는 머신 세트가 없습니다.

사전 요구 사항

- OpenShift Container Platform 클러스터 및 **oc** 명령행을 설치합니다.
- **cluster-admin** 권한이 있는 사용자로 **oc**에 로그인합니다.

절차

1. 클러스터에 있는 머신 세트를 확인합니다.

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

머신 세트는 **<clusterid>-worker-<aws-region-az>** 형식으로 나열됩니다.

2. 클러스터에 있는 머신을 확인합니다.

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

3. 삭제하려는 머신에 주석을 설정합니다.

```
$ oc annotate machine/<machine_name> -n openshift-machine-api
machine.openshift.io/cluster-api-delete-machine="true"
```

4. 삭제하려는 노드를 비우고 제외합니다.

```
$ oc adm cordon <node_name>
$ oc adm drain <node_name>
```

5. 머신 세트를 스케일링합니다.

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

또는 다음을 수행합니다.

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

머신 세트를 확장 또는 축소할 수 있습니다. 새 머신을 사용할 수 있을 때 까지 몇 분 정도 소요됩니다.

검증

- 원하는 머신 삭제를 확인합니다.

```
$ oc get machines
```

7장. WINDOWS 노드 업그레이드

WMCO(Windows Machine Config Operator)를 업그레이드하여 Windows 노드에 최신 업데이트가 있는지 확인할 수 있습니다.

7.1. WINDOWS MACHINE CONFIG OPERATOR 업그레이드

현재 클러스터 버전과 호환되는 새로운 버전의 WMCO(Windows Machine Config Operator)가 릴리스되면 Operator는 OLM(Operator Lifecycle Manager)을 사용할 때 함께 설치된 업그레이드 채널과 서브스크립션 승인 전략을 기반으로 업그레이드됩니다. WMCO 업그레이드로 인해 Windows 머신의 Kubernetes 구성 요소가 업그레이드됩니다.

장치를 중단할 필요가 없는 업그레이드의 경우 WMCO는 이전 버전의 WMCO에 의해 구성된 Windows 머신을 종료하고 현재 버전을 사용하여 이를 다시 생성합니다. 이는 **머신** 오브젝트를 삭제하여 수행되며, 이로 인해 Windows 노드가 드레이닝 및 삭제됩니다. 편리한 업그레이드를 위해, WMCO는 모든 구성된 노드에 버전 주석을 추가합니다. 업그레이드 중에 버전 주석이 일치하지 않으면 Windows 머신이 삭제되고 다시 생성됩니다. 업그레이드하는 동안 서비스 중단을 최소화하기 위해 WMCO는 한 번에 하나의 Windows 머신만 업데이트합니다.

중요

WMCO는 Windows 운영 체제 업데이트가 아닌 Kubernetes 구성 요소를 업데이트해야 합니다. VM을 생성할 때 Windows 이미지를 제공하므로 업데이트된 이미지를 제공해야 합니다. **MachineSet** 사양에서 이미지 구성을 변경하여 업데이트된 Windows 이미지를 제공할 수 있습니다.

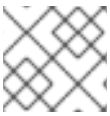
OLM(Operator Lifecycle Manager)을 사용한 Operator 업그레이드에 대한 자세한 내용은 설치된 Operator 업그레이드를 참조하십시오.

8장. WINDOWS 노드 제거

호스트 Windows 머신을 삭제하여 Windows 노드를 제거할 수 있습니다.

8.1. 특정 머신 삭제

특정 머신을 삭제할 수 있습니다.



참고

컨트롤 플레인 머신을 삭제할 수 없습니다.

사전 요구 사항

- OpenShift Container Platform 클러스터를 설치합니다.
- OpenShift CLI(**oc**)를 설치합니다.
- **cluster-admin** 권한이 있는 사용자로 **oc**에 로그인합니다.

절차

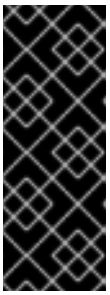
1. 클러스터에있는 머신을 확인하고 삭제할 머신을 식별합니다.

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

   명령 출력에는 **<clusterid>-worker-<cloud_region>** 형식의 머신 목록이 포함되어 있습니다.

2. 머신을 삭제합니다.

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



중요

기본적으로 머신 컨트롤러는 성공할 때까지 머신이 지원하는 노드를 드레이닝하려고 합니다. 일부 경우, Pod의 중단 예산을 잘못 구성하는 등의 경우와 같이 노드 드레이닝 작업으로 인해 머신이 삭제되지 않을 수 있습니다. 특정 머신에서 "machine.openshift.io/exclude-node-draining"에 주석을 사용하여 노드 드레이닝 프로세스를 건너 뛸 수 있습니다. 삭제 중인 머신이 머신 세트에 속하는 경우 지정된 복제본 수를 충족하기 위해 새 머신이 즉시 생성됩니다.

9장. WINDOWS 컨테이너 워크로드 비활성화

WMCO(Windows Machine Config Operator)의 설치를 제거하고 WMCO를 설치할 때 기본적으로 추가된 네임스페이스를 삭제하여 Windows 컨테이너 워크로드를 실행하는 기능을 비활성화할 수 있습니다.

9.1. WINDOWS MACHINE CONFIG OPERATOR 제거

클러스터에서 WMCO(Windows Machine Config Operator)의 설치를 제거할 수 있습니다.

사전 요구 사항

- Windows 워크로드를 호스팅하는 Windows **머신** 오브젝트를 삭제합니다.

절차

1. **Operators → OperatorHub** 페이지에서 **키워드로 필터링** 상자를 사용하여 **Red Hat Windows Machine Config Operator**를 검색합니다.
2. **Red Hat Windows Machine Config Operator**타일을 클릭합니다. Operator 타일은 Operator가 설치되었음을 나타냅니다.
3. **Windows Machine Config Operator** 설명자 페이지에서 **제거**를 클릭합니다.

9.2. WINDOWS MACHINE CONFIG OPERATOR 네임스페이스 삭제

기본적으로 WMCO(Windows Machine Config Operator)에 대해 생성된 네임스페이스를 삭제할 수 있습니다.

사전 요구 사항

- WMCO가 클러스터에서 제거됩니다.

절차

1. **openshift-windows-machine-config-operator** 네임스페이스에서 생성된 모든 Windows 워크로드를 제거합니다.

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

2. **openshift-windows-machine-config-operator** 네임스페이스의 모든 Pod가 삭제되거나 종료 상태를 보고하는지 확인합니다.

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

3. **openshift-windows-machine-config-operator** 네임스페이스를 삭제합니다.

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

추가 리소스

- 클러스터에서 Operator 삭제
- Windows 노드 제거.