Red Hat OpenShift Service on AWS 4

클러스터 관리

AWS 클러스터에서 Red Hat OpenShift Service 구성

Last Updated: 2024-09-07

Red Hat OpenShift Service on AWS 4 클러스터 관리

AWS 클러스터에서 Red Hat OpenShift Service 구성

법적 공지

초록

이 문서에서는 ROSA(Red Hat OpenShift Service on AWS) 클러스터 구성에 대한 정보를 제공합니다.

차례

1장. 클러스터 알림

클러스터 알림은 클러스터의 상태, 상태 또는 성능에 대한 메시지입니다.

클러스터 알림은 Red Hat site Reliability Engineering(SRE)이 관리형 클러스터의 상태에 대해 귀하와 통신하는 기본 방법입니다. SRE는 클러스터 알림을 사용하여 클러스터 문제를 해결하거나 방지하기 위해 작업을 수행하도록 요청할 수도 있습니다.

클러스터 소유자 및 관리자는 클러스터가 정상 상태로 유지되고 지원되는지 확인하기 위해 클러스터 알림을 정기적으로 검토하고 조치를 취해야 합니다.

클러스터의 클러스터 **기록 탭에서 Red Hat Hybrid Cloud Console에서 클러스터**알림을 볼 수 있습니다. 기본적으로 클러스터 소유자만 이메일로 클러스터 알림을 수신합니다. 다른 사용자가 클러스터 알림 이메일을 수신해야 하는 경우 각 사용자를 클러스터에 대한 알림 연락처로 추가합니다.

1.1. 추가 리소스

- 고객 담당: 클러스터 알림 검토 및 작업
- 클러스터 알림 이메일
- 문제 해결: 클러스터 알림

1.2. 클러스터 알림에서 예상되는 사항

클러스터 관리자는 클러스터의 상태 및 관리 요구를 효과적으로 이해하기 위해 클러스터 알림과 유형 및 심각도 수준을 전송하는 시기와 이유를 알고 있어야 합니다.

1.2.1. 클러스터 알림 정책

클러스터 알림은 클러스터의 상태와 영향을 미치는 높은 영향을 미치는 이벤트에 대한 정보를 유지하도록 설계되었습니다.

대부분의 클러스터 알림은 자동으로 생성되고 전송되어 즉시 문제에 대한 정보 또는 클러스터 상태에 대한 중요한 변경 사항을 확인할 수 있습니다.

특정 상황에서 Red Hat 사이트 안정성 엔지니어링(SRE)은 클러스터 알림을 생성하고 전송하여 복잡한 문제에 대한 추가 컨텍스트 및 지침을 제공합니다.

영향을 받지 않는 이벤트, 위험이 낮은 보안 업데이트, 일상적인 운영 및 유지 관리 또는 SRE가 신속하게 해결하는 일시적인 문제에 대해서는 클러스터 알림이 전송되지 않습니다.

Red Hat 서비스는 다음과 같은 경우 자동으로 알림을 보냅니다.

- 원격 상태 모니터링 또는 환경 확인 검사에서는 작업자 노드에 디스크 공간이 부족한 경우와 같이 클러스터에서 문제를 감지합니다.
- 예를 들어 예정된 유지 관리 또는 업그레이드가 시작되는 경우 심각한 클러스터 라이프 사이클 이벤트가 발생하거나 클러스터 작업이 이벤트의 영향을 받지만 고객의 개입은 필요하지 않습니다.
- 예를 들어 클러스터 소유권 또는 관리 제어가 한 사용자에서 다른 사용자로 전송되는 경우와 같이 중요한 클러스터 관리 변경이 발생합니다.
- 예를 들어 Red Hat이 클러스터에서 서브스크립션 조건 또는 기능을 업데이트할 때 클러스터 서브스크립션이 변경 또는 업데이트됩니다.

SRE는 다음과 같은 경우 알림을 생성하고 보냅니다.

- 사고로 인해 클러스터의 가용성 또는 성능에 영향을 미치는 성능 저하 또는 중단이 발생합니다(예: 클라우드 공급자의 경우 지역 중단). SRE는 사고 해결 진행 상황을 알려주기 위해 후속 알림을 보냅니다.
- 클러스터에서 보안 취약점, 보안 위반 또는 비정상적인 활동이 감지됩니다.
- Red Hat은 변경 사항이 생성 중이거나 클러스터 불안정성을 초래할 수 있음을 감지합니다.
- Red Hat은 워크로드가 클러스터에서 성능 저하 또는 불안정성을 초래하고 있음을 감지합니다.

1.2.2. 클러스터 알림 심각도 수준

각 클러스터 알림에는 비즈니스에 가장 큰 영향을 미치는 알림을 식별하는 데 도움이 되는 관련 심각도 수준이 있습니다. 클러스터의 클러스터 **기록 탭에서 Red Hat Hybrid Cloud Console의 심각도 수준에 따라 클러스터** 알림을 필터링할 수 있습니다.

Red Hat은 가장 심각한 클러스터 알림에 다음과 같은 심각도 수준을 사용합니다.

심각

즉각적인 조치가 필요합니다. 서비스 또는 클러스터의 하나 이상의 주요 기능이 작동하지 않거나 곧 작동하지 않습니다. 중요한 경고는 직원에게 호출하고 일반 워크플로우를 중단할 수 있을 만큼 중요합니다.

major

즉각적인 조치가 강력히 권장됩니다. 클러스터의 하나 이상의 주요 기능이 곧 작동하지 않습니다. 주요 문제는 적시에 해결되지 않는 경우 중요한 문제로 이어질 수 있습니다.

경고

가능한 한 빨리 조치가 필요합니다. 클러스터의 하나 이상의 주요 기능이 최적으로 작동하지 않고 추가로 성능이 저하될 수 있지만 클러스터 작동에 즉각적인 위험이 발생하지 않습니다.

정보

작업이 필요하지 않습니다. 이 심각도는 처리해야 하는 문제를 설명하지 않으며 의미 있거나 중요한 라이프 사이클, 서비스 또는 클러스터 이벤트에 대한 중요한 정보만 설명합니다.

Debug

작업이 필요하지 않습니다. 디버그 알림은 예기치 않은 동작을 디버깅하는 데 도움이 되도록 덜 중요한 라이프사이클, 서비스 또는 클러스터 이벤트에 대한 낮은 수준의 정보를 제공합니다.

1.2.3. 클러스터 알림 유형

각 클러스터 알림에는 역할 및 역할과 관련된 알림을 식별하는 데 도움이 되는 관련 알림 유형이 있습니다. 클러스터의 클러스터 **기록 탭에서 Red Hat Hybrid Cloud Console에서 이러한 유형에 따라 클러스터**알림을 필터링할 수 있습니다.

Red Hat은 다음 알림 유형을 사용하여 알림을 표시합니다.

용량 관리

노드 풀, 시스템 풀, 컴퓨팅 복제본 또는 할당량(로드 밸런서, 스토리지 등) 업데이트, 생성 또는 삭제와 관련된 이벤트 알림.

클러스터 액세스

STS 인증 정보가 만료되었거나 AWS 역할에 구성 문제가 있는 경우 또는 ID 공급자를 추가하거나 제거할 때와 같은 그룹, 역할 또는 ID 공급자 추가 또는 ID 공급자와 관련된 이벤트에 대한 알림입니다.

클러스터 애드온

애드온의 애드온 관리 또는 업그레이드 유지 관리와 관련된 이벤트의 알림(예: 애드온을 설치, 업그레이드 또는 제거)하거나 미해결 요구 사항으로 인해 설치할 수 없습니다.

클러스터 구성

클러스터 튜닝 이벤트, 워크로드 모니터링 및 진행 중인 검사에 대한 알림입니다.

클러스터 라이프사이클

클러스터 또는 클러스터 리소스 생성, 삭제 및 등록에 대한 알림 또는 클러스터 또는 리소스 상태 변경(예: 준비 또는 누락).

클러스터 네트워킹

HTTP/S 프록시, 라우터 및 수신 상태를 포함한 클러스터 네트워킹과 관련된 알림.

클러스터 소유권

클러스터 소유권과 관련된 알림은 한 사용자의 다른 사용자로 이동합니다.

클러스터 스케일링

노드 풀, 머신 풀, 컴퓨팅 복제본 또는 할당량 업데이트, 생성 또는 삭제와 관련된 알림입니다.

클러스터 보안

예를 들어 클러스터 보안과 관련된 이벤트(예: 실패한 액세스 시도 횟수, 신뢰 번들에 대한 업데이트 또는 보안에 영향을 미치는 소프트웨어 업데이트 등)

클러스터 서브스크립션

클러스터 만료, 평가판 클러스터 알림 또는 무료에서 유료로 전환.

클러스터 업데이트

업그레이드 유지 관리 또는 활성화와 같은 업그레이드 관련 항목입니다.

고객 지원

지원 케이스 상태에 대한 업데이트

일반 알림

기본 알림 유형입니다. 이는 더 구체적인 카테고리가 없는 알림에만 사용됩니다.

1.3. RED HAT HYBRID CLOUD CONSOLE을 사용하여 클러스터 알림 보기

클러스터 알림은 클러스터 상태에 대한 중요한 정보를 제공합니다. Red Hat Hybrid Cloud Console의 클러스터 **기록 탭에서** 클러스터로 전송된 알림을 볼 수 있습니다.

사전 요구 사항

- 하이브리드 클라우드 콘솔에 로그인되어 있습니다.

절차

1. 하이브리드 클라우드 콘솔의 클러스터 페이지로 이동합니다.
2. 클러스터 이름을 클릭하여 클러스터 세부 정보 페이지로 이동합니다.
3. **클러스터 기록** 탭을 클릭합니다.
   클러스터 알림은 클러스터 기록 제목 아래에 표시됩니다.
4. 선택 사항: 관련 클러스터 알림에 대해 필터링
   필터 컨트롤을 사용하여 사용자의 전문 영역에 집중하거나 중요한 문제를 해결할 수 있도록 사용자와 관련이 없는 클러스터 알림을 숨깁니다. 알림 설명의 텍스트, 심각도 수준, 알림 유형, 알림이 수신될 때 및 알림을 트리거한 시스템 또는 사람을 기반으로 알림을 필터링할 수 있습니다.

1.4. 클러스터 알림 이메일

기본적으로 클러스터 알림이 클러스터로 전송되면 클러스터 소유자의 이메일도 전송됩니다. 적절한 모든 사용자가 클러스터 상태에 대한 정보를 유지할 수 있도록 알림 이메일에 대한 추가 수신자를 구성할 수 있습니다.

1.4.1. 클러스터에 알림 연락처 추가

알림 연락처는 클러스터 알림이 클러스터로 전송될 때 이메일을 수신합니다. 기본적으로 클러스터 소유자만 클러스터 알림 이메일을 수신합니다. 클러스터 지원 설정에서 다른 클러스터 사용자를 추가 알림 연락처로 구성할 수 있습니다.

사전 요구 사항

- 클러스터가 Red Hat Hybrid Cloud Console에 배포 및 등록되었습니다.
- 하이브리드 클라우드 콘솔에 로그인되어 있습니다.
- 의도된 알림 수신자의 클러스터에 사용자 계정이 있습니다.

절차

1. 하이브리드 클라우드 콘솔의 클러스터 페이지로 이동합니다.
2. 클러스터 이름을 클릭하여 클러스터 세부 정보 페이지로 이동합니다.
3. **지원** 탭을 클릭합니다.
4. **지원** 탭에서 **알림 연락처** 섹션을 찾습니다.
5. **알림 연락처 추가를** 클릭합니다.
6. **Red Hat 사용자 이름 또는 이메일**필드에 이메일 주소 또는 새 수신자의 사용자 이름을 입력합니다.
7. **연락처 추가를** 클릭합니다.

검증 단계

- "알림 연락처가 성공적으로 추가됨" 메시지가 표시됩니다.

1.4.2. 클러스터에서 알림 연락처 제거

알림 연락처는 클러스터 알림이 클러스터로 전송될 때 이메일을 수신합니다.

클러스터 지원 설정에서 알림 연락처를 제거하여 알림 이메일을 수신하지 못하도록 할 수 있습니다.

사전 요구 사항

- 클러스터가 Red Hat Hybrid Cloud Console에 배포 및 등록되었습니다.
- 하이브리드 클라우드 콘솔에 로그인되어 있습니다.

절차

1. 하이브리드 클라우드 콘솔의 클러스터 페이지로 이동합니다.
2. 클러스터 이름을 클릭하여 클러스터 세부 정보 페이지로 이동합니다.
3. **지원** 탭을 클릭합니다.
4. **지원** 탭에서 **알림 연락처** 섹션을 찾습니다.
5. 제거하려는 수신자 옆에 있는 옵션 메뉴(t)를 클릭합니다.
6. **삭제를 클릭합니다.**

검증 단계

- **"알림 연락처가 성공적으로 삭제됨" 메시지가 표시됩니다.**

1.5. 문제 해결

클러스터 알림 이메일을 수신하지 못하는 경우

- **@redhat.com 주소에서 전송된 이메일이 이메일 수신함에서 필터링되지 않았는지 확인합니다.**
- **올바른 이메일 주소가 클러스터의 알림 연락처로 나열되어 있는지 확인합니다.**
- **클러스터 소유자 또는 관리자에게 알림 연락처로 추가하도록 요청합니다. 클러스터 알림 이메일.**

클러스터가 알림을 수신하지 않는 경우

- **클러스터가 api.openshift.com 에서 리소스에 액세스할 수 있는지 확인합니다.**
- **문서화된 사전 요구 사항에 따라 방화벽이 구성되었는지 확인합니다. AWS 방화벽 사전 요구 사항**

2장. 프라이빗 연결 구성

2.1. 프라이빗 연결 구성

ROSA(Red Hat OpenShift Service on AWS) 환경의 요구 사항에 맞게 프라이빗 클러스터 액세스를 구현할 수 있습니다.

절차

1. **ROSA AWS 계정에 액세스하여 다음 방법 중 하나 이상을 사용하여 클러스터에 대한 개인 연결을 설정합니다.**
   - **AWS VPC 피어링 구성 : VPC 피어링을 활성화하여 두 개인 IP 주소 간에 네트워크 트래픽을 라우팅합니다.**
   - **AWS VPN 구성: 프라이빗 네트워크를 Amazon Virtual Private Cloud에 안전하게 연결하기 위해 가상 사설 네트워크를 구축합니다.**
   - **AWS Direct Connect 구성: AWS Direct Connect를 구성하여 프라이빗 네트워크와 AWS Direct Connect 위치 간의 전용 네트워크 연결을 설정합니다.**
2. **ROSA에서 개인 클러스터를 구성합니다.**

2.2. AWS VPC 피어링 구성

이 샘플 프로세스는 다른 AWS VPC 네트워크와 피어링할 AWS 클러스터의 Red Hat OpenShift Service가 포함된 AWS(Amazon Web Services) VPC를 구성합니다. AWS VPC 피어링 연결 또는 기타 가능한 구성을 생성하는 방법에 대한 자세한 내용은 AWS VPC 피어링 가이드를 참조하십시오.

2.2.1. VPC 피어링 용어

두 개의 별도의 AWS 계정에서 두 VPC 간에 VPC 피어링 연결을 설정할 때 다음 용어가 사용됩니다.

Red Hat OpenShift Service on AWS 계정	AWS 클러스터의 Red Hat OpenShift Service가 포함된 AWS 계정입니다.
Red Hat OpenShift Service on AWS Cluster VPC	AWS 클러스터의 Red Hat OpenShift Service가 포함된 VPC입니다.
고객 AWS 계정	AWS 계정의 Red Hat OpenShift Service가 아닌 경우 피어링할 수 있습니다.
Customer VPC	공유하려는 AWS 계정의 VPC입니다.
고객 VPC 리전	고객의 VPC가 있는 리전입니다.

참고

2018년 7월 현재 AWS는 중국을 제외한 모든 상용 리전 간에 리전 간 VPC 피어링을지원합니다.

2.2.2. VPC 피어 요청 시작

AWS 계정의 Red Hat OpenShift Service에서 고객 AWS 계정으로 VPC 피어링 연결 요청을 보낼 수 있습니다.

사전 요구 사항

- 피어 요청을 시작하는 데 필요한 고객 VPC에 대한 다음 정보를 수집합니다.
  - 고객 AWS 계정 번호
  - Customer VPC ID
  - 고객 VPC 리전
  - 고객 VPC CIDR
- AWS Cluster VPC의 Red Hat OpenShift Service에서 사용하는 CIDR 블록을 확인합니다. 고객 VPC의 CIDR 블록과 겹치거나 일치하는 경우 이 두 VPC 간의 피어링이 불가능합니다. 자세한 내용은 Amazon VPC Unsupported VPC Peering 구성 문서를 참조하십시오. CIDR 블록이 겹치지 않으면 절차를 계속할 수 있습니다.

절차

1. AWS 계정의 Red Hat OpenShift Service에 대한 웹 콘솔에 로그인하고 클러스터가 호스팅되는 리전의 VPC 대시보드 로 이동합니다.
2. Peering Connections 페이지로 이동하여 Create Peering Connection 버튼을 클릭합니다.
3. 로그인한 계정의 세부 정보와 연결 중인 계정 및 VPC의 세부 정보를 확인합니다.
   a. 피어링 연결 이름 태그: VPC 피어링 연결에 대한 설명적 이름을 설정합니다.
   b. VPC(Requester): 드롭다운 목록에서 AWS Cluster VPC ID의 Red Hat OpenShift Service 를 선택합니다.
   c. 계정: 다른 계정을 선택하고 고객 AWS 계정 *(대시 제외)를 입력합니다.
   d. 리전: Customer VPC Region이 현재 리전과다른 경우 다른 리전을 선택하고 드롭다운 목록에서 고객 VPC 리전을 선택합니다.
   e. VPC (Accepter): 고객 VPC ID 설정
4. Create Peering Connection을 클릭합니다.
5. 요청이 Pending (보류 중) 상태가 되고 있는지 확인합니다.Failed 상태가 되면 세부 사항을 확인하고 프로세스를 반복합니다.

2.2.3. VPC 피어 요청 수락

VPC 피어링 연결을 생성한 후 고객 AWS 계정의 요청을 수락해야 합니다.

사전 요구 사항

- VPC 피어 요청을 시작합니다.

절차

1. AWS 웹 콘솔에 로그인합니다.
2. VPC 서비스로 이동합니다.
3. 피어링 연결로 이동합니다.
4. Pending peering 연결을 클릭합니다.
5. 요청이 시작된 AWS 계정 및 VPC ID를 확인합니다. 이는 AWS 계정의 Red Hat OpenShift Service 및 AWS Cluster VPC의 Red Hat OpenShift Service 여야 합니다.
6. 요청 수락 을 클릭합니다.

2.2.4. 라우팅 테이블 구성

VPC 피어링 요청을 수락한 후 두 VPC가 피어 연결 간에 통신하도록 경로를 구성해야 합니다.

사전 요구 사항

- VPC 피어 요청을 시작하고 수락합니다.

절차

1. AWS 계정의 Red Hat OpenShift 서비스에 대해 AWS 웹 콘솔에 로그인합니다.
2. VPC 서비스로 이동한 다음경로 표 로 이동합니다.
3. AWS Cluster VPC에서 Red Hat OpenShift Service의 경로 테이블을 선택합니다.

참고

일부 클러스터에서는 특정 VPC에 대해 둘 이상의 라우팅 테이블이 있을 수 있습니다. 명시적으로 연결된 서브넷이 여러 개인 개인을 선택합니다.

4. 경로 탭을 선택한 다음편집 을 선택합니다.
5. 대상 텍스트 상자에 Customer VPC CIDR 블록을 입력합니다.
6. 대상 텍스트 상자에 피어링 연결 ID를 입력합니다.
7. 저장을 클릭합니다.
8. 다른 VPC의 CIDR 블록과 동일한 프로세스를 완료해야 합니다.
    a. 고객 AWS Web Console → VPC 서비스 → 경로 테이블에 로그인합니다.
    b. VPC의 경로 테이블을 선택합니다.
    c. 경로 탭을 선택한 다음편집 을 선택합니다.

d. 대상 텍스트 상자에 AWS Cluster VPC CIDR 블록의 Red Hat OpenShift Service를 입력합니다.

e. 대상 텍스트 상자에 피어링 연결 ID를 입력합니다.

f. 저장을 클릭합니다.

이제 VPC 피어링 연결이 완료되었습니다. 확인 절차에 따라 피어링 연결의 연결이 작동하는지 확인합니다.

2.2.5. VPC 피어링 확인 및 문제 해결

VPC 피어링 연결을 설정한 후에는 해당 연결이 구성되어 있고 올바르게 작동하는지 확인하는 것이 가장 좋습니다.

사전 요구 사항

- VPC 피어 요청을 시작하고 수락합니다.
- 라우팅 테이블을 구성합니다.

절차

- AWS 콘솔에서 피어링된 클러스터 VPC의 경로 테이블을 확인합니다. 라우팅 테이블 구성 단계를 따르고 VPC CIDR 범위 대상을 피어링 연결 대상을 가리키는 경로 테이블 항목이 있는지 확인합니다.
  AWS Cluster VPC 경로 테이블 및 Customer VPC 경로 테이블의 Red Hat OpenShift Service에서 경로가 올바르게 표시되면 아래 **netcat** 방법을 사용하여 연결을 테스트해야 합니다. 테스트 호출이 성공하면 VPC 피어링이 올바르게 작동합니다.
- 엔드포인트 장치에 대한 네트워크 연결을 테스트하기 위해 **nc** (또는 **netcat**)는 문제 해결에 유용한 도구입니다. 기본 이미지에 포함되어 있으며 연결을 설정할 수 있는 경우 빠르고 명확한 출력을 제공합니다.

  a. 자체 후에 정리되는 **busybox** 이미지를 사용하여 임시 Pod를 생성합니다.

  ```
  $ oc run netcat-test \
      --image=busybox -i -t \
      --restart=Never --rm \
      -- /bin/sh
  ```

  b. **nc** 를 사용하여 연결을 확인합니다.

  - 연결 결과의 예:

  ```
  / nc -zvv 192.168.1.1 8080
  10.181.3.180 (10.181.3.180:8080) open
  sent 0, rcvd 0
  ```

  - 실패한 연결 결과의 예:

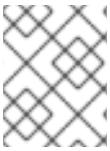
  ```
  / nc -zvv 192.168.1.2 8080
  nc: 10.181.3.180 (10.181.3.180:8081): Connection refused
  sent 0, rcvd 0
  ```

  c. 컨테이너를 종료하면 Pod가 자동으로 삭제됩니다.

```
/ exit
```

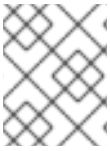
2.3. AWS VPN 구성

이 샘플 프로세스는 고객의 현장 하드웨어 VPN 장치를 사용하도록 AWS 클러스터에서 AWS(Amazon Web Services) Red Hat OpenShift Service를 구성합니다.



참고

AWS VPN은 현재 VPN 트래픽에 NAT를 적용하는 관리형 옵션을 제공하지 않습니다. 자세한 내용은 AWS Knowledge Center 에서 참조하십시오.



참고

개인 연결을 통해 모든 트래픽(예: **0.0.0.0/0**)을 라우팅하는 것은 지원되지 않습니다. 이를 위해서는 SRE 관리 트래픽을 비활성화하는 인터넷 게이트웨이를 삭제해야 합니다.

하드웨어 VPN 장치를 사용하여 AWS VPC를 원격 네트워크에 연결하는 방법에 대한 자세한 내용은 Amazon VPC VPN 연결 설명서를 참조하십시오.

2.3.1. VPN 연결 생성

다음 절차에 따라 고객의 현장 하드웨어 VPN 장치를 사용하도록 AWS 클러스터에서 AWS(Amazon Web Services) Red Hat OpenShift Service를 구성할 수 있습니다.

사전 요구 사항

- 하드웨어 VPN 게이트웨이 장치 모델 및 소프트웨어 버전 (예: Cisco ASA 버전 8.3을 실행함) AWS에서 게이트웨이 장치를 지원하는지 확인하려면 Amazon VPC 네트워크 관리자 가이드를 참조하십시오.
- VPN 게이트웨이 장치의 공용 고정 IP 주소입니다.
- BGP 또는 정적 라우팅: BGP인 경우 ASN이 필요합니다. 정적 라우팅인 경우 하나 이상의 정적 경로를 구성해야 합니다.
- 선택 사항: VPN 연결을 테스트하기 위한 연결 서비스의 IP 및 포트/프로토어입니다.

2.3.1.1. VPN 연결 구성

절차

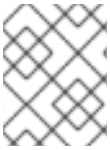
1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 VPC 대시보드로 이동합니다.
2. VPC를 클릭하고 AWS 클러스터의 Red Hat OpenShift Service가 포함된 VPC의 이름과 VPC ID를 식별합니다.
3. VPC 대시보드에서 Customer Gateway 를 클릭합니다.
4. Create Customer Gateway 를 클릭하고 의미 있는 이름을 지정합니다.
5. 라우팅 방법(Dynamic 또는 Static)을 선택합니다.

6. Dynamic인 경우 표시되는 필드에 BGP ASN을 입력합니다.
7. VPN 게이트웨이 끝점 IP 주소에 붙여넣습니다.
8. 생성을 클릭합니다.
9. 의도한 VPC에 가상 프라이빗 게이트웨이가 아직 연결되어 있지 않은 경우:
    a. VPC 대시보드에서 Virtual Private Gateway를 클릭합니다.
    b. 가상 프라이빗 게이트웨이 만들기를 클릭하고 의미 있는 이름을 지정한 다음생성을 클릭합니다.
    c. 기본 Amazon 기본값 ASN을 그대로 둡니다.
    d. 새로 생성된 게이트웨이를 선택하고 VPC에 연결한 다음 이전에 확인한 클러스터 VPC에 연결합니다.

2.3.1.2. VPN 연결 설정

절차

1. VPC 대시보드에서 사이트 간 VPN 연결을클릭합니다.
2. VPN 연결 생성을클릭합니다.
    a. 의미 있는 이름 태그를 지정합니다.
    b. 이전에 만든 가상 개인 게이트웨이를 선택합니다.
    c. Customer Gateway에 대해 Existing 을 선택합니다.
    d. 이름으로 고객 게이트웨이 장치를 선택합니다.
    e. VPN에서 BGP를 사용하는 경우 Dynamic 을 선택하고, 그렇지 않으면Static 을 선택합니다. 고정 IP CIDR을 입력합니다. CIDR이 여러 개인 경우 각 CIDR을 Another Rule 으로 추가합니다.
    f. 생성을 클릭합니다.
    g. VPN 상태가 사용 가능 으로 변경될 때까지 약 5분에서 10분 정도 기다립니다.
3. 방금 만든 VPN을 선택하고 구성 다운로드를 클릭합니다.
    a. 드롭다운 목록에서 고객 게이트웨이 장치의 공급 업체, 플랫폼 및 버전을 선택한 다음 다운로드를 클릭합니다.
    b. 일반 공급 업체 구성은 일반 텍스트 형식으로 정보를 검색하는 데도 사용할 수 있습니다.



참고

VPN 연결이 설정된 후 Route Propagation을 설정하거나 VPN이 예상대로 작동하지 않을 수 있습니다.

참고

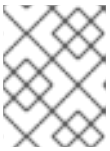
구성에 추가해야 하는 VPC 서브넷 정보를 원격 네트워크로 기록해 둡니다.

2.3.1.3. VPN 경로 전파 활성화

VPN 연결을 설정한 후에는 필요한 경로가 VPC의 경로 테이블에 추가되도록 경로 전파가 활성화되어 있는지 확인해야 합니다.

절차

1. VPC 대시보드에서 경로 테이블을 클릭합니다.
2. AWS 클러스터에서 Red Hat OpenShift Service가 포함된 VPC와 연결된 프라이빗 경로 테이블을 선택합니다.



참고

일부 클러스터에서는 특정 VPC에 대해 둘 이상의 라우팅 테이블이 있을 수 있습니다. 명시적으로 연결된 서브넷이 여러 개인 개인을 선택합니다.

3. Route Propagation 탭을 클릭합니다.
4. 표시되는 표에는 이전에 만든 가상 개인 게이트웨이가 표시되어야 합니다. Propagate 열의 값을 확인합니다.
   a. Propagate가 No 로 설정된 경우경로 전파 편집 을 클릭하고 가상 개인 게이트웨이 이름 옆에 있는 Propagate 확인란을 선택한 다음 저장 을 클릭합니다.

VPN 터널을 구성하고 AWS가 이를 Up 으로 탐지하면 static 또는 BGP 경로가 경로 테이블에 자동으로 추가됩니다.

2.3.2. VPN 연결 확인

VPN 터널을 설정한 후에는 터널이 AWS 콘솔에 있고 터널이 작동하는지 확인할 수 있습니다.

사전 요구 사항

- VPN 연결을 생성했습니다.

절차

1. 터널이 AWS에 있는지 확인합니다.
   a. VPC 대시보드에서 VPN 연결을 클릭합니다.
   b. 이전에 만든 VPN 연결을 선택하고 Details 탭을 클릭합니다.
   c. VPN 터널 중 하나 이상이 Up 임을 확인할 수 있습니다.
2. 연결을 확인합니다.
   엔드포인트 장치에 대한 네트워크 연결을 테스트하기 위해 **nc** (또는 **netcat**)는 문제 해결에 유용한 도구입니다. 기본 이미지에 포함되어 있으며 연결을 설정할 수 있는 경우 빠르고 명확한 출력을 제공합니다.

a. 자체 후에 정리되는 **busybox** 이미지를 사용하여 임시 Pod를 생성합니다.

```
$ oc run netcat-test \
    --image=busybox -i -t \
    --restart=Never --rm \
    -- /bin/sh
```

b. **nc** 를 사용하여 연결을 확인합니다.

- 연결 결과의 예:

```
/ nc -zvv 192.168.1.1 8080
10.181.3.180 (10.181.3.180:8080) open
sent 0, rcvd 0
```

- 실패한 연결 결과의 예:

```
/ nc -zvv 192.168.1.2 8080
nc: 10.181.3.180 (10.181.3.180:8081): Connection refused
sent 0, rcvd 0
```

c. 컨테이너를 종료하면 Pod가 자동으로 삭제됩니다.

```
/ exit
```

2.3.3. VPN 연결 문제 해결

터널이 연결되지 않음
터널 연결이 여전히 Down 이면 다음과 같은 몇 가지 사항을 확인할 수 있습니다.

- AWS 터널은 VPN 연결을 시작하지 않습니다. 연결 시도는 고객 게이트웨이에서 시작되어야 합니다.
- 소스 트래픽이 구성된 고객 게이트웨이와 동일한 IP에서 제공되는지 확인합니다. AWS는 소스 IP 주소가 일치하지 않는 게이트웨이에 대한 모든 트래픽을 자동으로 삭제합니다.
- 구성이 AWS에서 지원하는 값과 일치하는지 확인합니다. 여기에는 IKE 버전, DH 그룹, IKE 수명 등이 포함됩니다.
- VPC의 경로 테이블을 다시 확인합니다. 전파가 활성화되어 있고 이전에 대상으로 만든 가상 개인 게이트웨이가 있는 경로 테이블에 항목이 있는지 확인합니다.
- 중단을 일으킬 수 있는 방화벽 규칙이 없는지 확인합니다.
- 정책 기반 VPN을 사용하고 있는지 확인합니다. 이로 인해 구성된 방법에 따라 복잡성이 발생할 수 있습니다.
- 추가 문제 해결 단계는 AWS Knowledge Center 에서 확인할 수 있습니다.

터널은 연결되어 있지 않습니다.
터널 연결에 지속적으로 유지되는 데 문제가 있는 경우 모든 AWS 터널 연결을 게이트웨이에서 시작해야 합니다. AWS 터널 은 터널링을 시작하지 않습니다.

Red Hat은 터널 옆에 SLA 모니터(Cisco ASA) 또는 일부 장치를 IP CIDR 범위 내에 구성된 모든 IP 주소로 "중요" 트래픽(예: **ping**,**nc** 또는 **telnet**)을 지속적으로 전송하는 것을 권장합니다. 연결이 성공하든 트래픽이 터널로 전달되는지 여부는 중요하지 않습니다.

Down 상태의 보조 터널
VPN 터널이 생성되면 AWS는 추가 페일오버 터널을 생성합니다. 게이트웨이 장치에 따라 보조 터널이 Down 상태로 표시되는 경우가 있습니다.

AWS 알림은 다음과 같습니다.

> You have new non-redundant VPN connections
>
> One or more of your vpn connections are not using both tunnels. This mode of operation is not highly available and we strongly recommend you configure your second tunnel. View your non-redundant VPN connections.

2.4. AWS DIRECT CONNECT 구성

이 프로세스는 AWS에서 Red Hat OpenShift Service를 사용하여 AWS Direct Connect 가상 인터페이스를 수락하는 방법을 설명합니다. AWS Direct Connect 유형 및 구성에 대한 자세한 내용은 AWS Direct Connect 구성 요소 설명서를 참조하십시오.

2.4.1. AWS Direct Connect 방법

Direct Connect 연결을 위해서는 Direct Connect Gateway(DXGateway)에 연결된 호스팅 가상 인터페이스(VIF)가 필요합니다. 이 인터페이스는 동일한 계정 또는 원격 VPC에 액세스하기 위해 VGW(Virtual Gateway) 또는 Transit Gateway에 연결됩니다.

기존 CryostatGateway가 없는 경우 일반적인 프로세스에는 호스팅 VIF를 생성해야 하며, AWS 계정의 Red Hat OpenShift Service에서 CryostatGateway 및 VGW가 생성됩니다.

기존의 CryostatGateway가 하나 이상의 기존 VGW에 연결되어 있는 경우 이 프로세스에는 AWS 계정의 Red Hat OpenShift Service가 AWS 계정의 Red Hat OpenShift Service를 involves sending an Association Proposal to the CryostatGateway 소유자입니다. DXGateway 소유자는 제안된 CIDR이 연결된 다른 VGW와 충돌하지 않는지 확인해야 합니다.

자세한 내용은 다음 AWS 설명서를 참조하십시오.

- 가상 인터페이스
- Direct Connect Gateways
- 계정 간에 VGW 연결

중요

기존 DXGateway에 연결할 때 비용에 대한 책임이 있습니다.

다음 두 가지 구성 옵션을 사용할 수 있습니다.

방법 1	호스팅된 VIF를 만든 다음 DXGateway 및 VGW를 만듭니다.
방법 2	보유 중인 기존 Direct Connect Gateway를 통해 연결을 요청합니다.

2.4.2. 호스팅된 가상 인터페이스 생성

사전 요구 사항

- AWS 계정 ID에서 Red Hat OpenShift Service를 수집합니다.

2.4.2.1. Direct Connect 연결 유형 확인

Direct Connect 가상 인터페이스 세부 정보를 보고 연결 유형을 결정합니다.

절차

1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 올바른 리전을 선택합니다.
2. 서비스 메뉴에서 Direct Connect 를 선택합니다.
3. 하나 이상의 가상 인터페이스가 수락되기를 기다리고 있으며 그 중 하나를 선택하여 요약 을 확인합니다.
4. 가상 인터페이스 유형(공개 또는 공용) 보기.
5. Amazon 사이드 ASN 값을 기록합니다.

Direct Connect 가상 인터페이스 유형이 프라이빗인 경우 가상 프라이빗 게이트웨이가 생성됩니다. Direct Connect Virtual Interface가 공용인 경우 Direct Connect Gateway가 생성됩니다.

2.4.2.2. 개인 직접 연결 생성

Direct Connect 가상 인터페이스 유형이 프라이빗인 경우 개인 Direct Connect가 생성됩니다.

절차

1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 올바른 리전을 선택합니다.
2. AWS 리전의 서비스 메뉴에서 VPC 를 선택합니다.
3. VPN 연결에서 가상 개인 게이트웨이를 선택합니다.
4. 가상 개인 게이트웨이 생성을 클릭합니다.
5. 가상 개인 게이트웨이에 적절한 이름을 지정합니다.
6. 사용자 지정 ASN 을 선택하고 이전에 수집한Amazon 사이드 ASN 값을 입력합니다.
7. 가상 개인 게이트웨이 만들기.
8. 새로 생성된 가상 프라이빗 게이트웨이를 클릭하고 Actions 탭에서 VPC에 연결을 선택합니다.
9. 목록에서 Red Hat OpenShift Service on AWS Cluster VPC를 선택하고 Virtual Private Gateway를 VPC에 연결합니다.
10. Services 메뉴에서 Direct Connect 를 클릭합니다. 목록에서 직접 연결 가상 인터페이스 중 하나를 선택합니다.
11. I understand that Direct Connect port charges apply once I click Accept Connection message, select Accept Connection 을 선택합니다.

12. 가상 개인 게이트웨이 연결을 수락하고 이전 단계에서 만든 가상 개인 게이트웨이를 선택합니다.
13. 연결을 수락 하려면 수락을 선택합니다.
14. 가상 인터페이스가 두 개 이상인 경우 이전 단계를 반복합니다.

2.4.2.3. 공용 직접 연결 생성

Direct Connect 가상 인터페이스 유형이 공용인 경우 공용 Direct Connect가 생성됩니다.

절차

1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 올바른 리전을 선택합니다.
2. AWS 계정의 Red Hat OpenShift Service의 서비스 메뉴에서 Direct Connect 를 선택합니다.
3. Direct Connect Gateways 및 Direct Connect Gateway 만들기를 선택합니다.
4. Direct Connect Gateway에 적절한 이름을 지정합니다.
5. Amazon 측 ASN 에서 이전에 수집된 Amazon 사이드 ASN 값을 입력합니다.
6. Direct Connect Gateway를 만듭니다.
7. 서비스 메뉴에서 Direct Connect 를 선택합니다.
8. 목록에서 Direct Connect Virtual Interfaces 중 하나를 선택합니다.
9. I understand that Direct Connect port charges apply once I click Accept Connection message, select Accept Connection 을 선택합니다.
10. Direct Connect Gateway 연결을 수락하고 이전 단계에서 만든 Direct Connect Gateway를 선택합니다.
11. 수락 을 클릭하여 연결을 수락합니다.
12. 가상 인터페이스가 두 개 이상인 경우 이전 단계를 반복합니다.

2.4.2.4. 가상 인터페이스 확인

Direct Connect 가상 인터페이스가 허용되면 단기간에 기다렸다가 인터페이스 상태를 확인합니다.

절차

1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 올바른 리전을 선택합니다.
2. AWS 계정의 Red Hat OpenShift Service의 서비스 메뉴에서 Direct Connect 를 선택합니다.
3. 목록에서 Direct Connect Virtual Interfaces 중 하나를 선택합니다.
4. 인터페이스 상태가 사용가능한지 확인
5. Interface BGP 상태가 Up 인지 확인합니다.
6. 나머지 Direct Connect 인터페이스에 대해 이 확인을 반복합니다.

직접 연결 가상 인터페이스를 사용할 수 있는 후 AWS AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 구성을 위해 직접 연결 구성 파일을 다운로드할 수 있습니다.

2.4.3. 기존 Direct Connect 게이트웨이에 연결

사전 요구 사항

- AWS VPC에서 Red Hat OpenShift Service의 CIDR 범위가 연결된 다른 VGW와 충돌하지 않는지 확인합니다.
- 다음 정보를 수집합니다.
  - Direct Connect Gateway ID입니다.
  - 가상 인터페이스와 연결된 AWS 계정 ID입니다.
  - DXGateway에 할당된 BGP ASN입니다. 선택사항: Amazon 기본 ASN도 사용할 수 있습니다.

절차

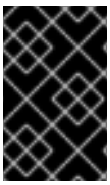
1. AWS 계정 대시보드에서 Red Hat OpenShift Service에 로그인하고 올바른 리전을 선택합니다.
2. AWS 계정의 Red Hat OpenShift Service에서 Services 메뉴에서 VPC 를 선택합니다.
3. VPN 연결에서 가상 개인 게이트웨이를 선택합니다.
4. 가상 개인 게이트웨이 생성을 선택합니다.
5. 가상 개인 게이트웨이에 적절한 이름을 지정합니다.
6. Custom ASN 을 클릭하고 이전에 수집된Amazon 사이드 ASN 값을 입력하거나 Amazon Provided ASN을 사용합니다.
7. 가상 개인 게이트웨이 만들기.
8. AWS 계정 대시보드의 Red Hat OpenShift Service 탐색 창에서 가상 개인 게이트웨이를 선택하고 가상 개인 게이트웨이 를 선택합니다. 세부 정보 보기 를 선택합니다.
9. Direct Connect 게이트웨이 연결을 선택하고 Direct Connect 게이트웨이 연결을 클릭합니다.
10. terminate account type에서 계정 소유자의 경우Another account를 선택합니다.
11. Direct Connect 게이트웨이 소유자의 경우 Direct Connect 게이트웨이를소유한 AWS 계정의 ID 를 입력합니다.
12. association 설정에서 Direct Connect 게이트웨이 ID에 대해 Direct Connect 게이트웨이의 ID를 입력합니다.
13. association 설정에서 Virtual interface owner에 대해 연결을 위한 가상 인터페이스를 소유한 AWS 계정의 ID를 입력합니다.
14. 선택 사항: 허용 접두사에 접두사를 추가하고 쉼표를 사용하여 분리합니다.
15. Associate Direct Connect Gateway를 선택합니다.
16. association 제안서를 보낸 후, 귀하의 수락을 기다리고 있을 것입니다. 수행해야 하는 최종 단계는 AWS 설명서에서 확인할 수 있습니다.

2.4.4. Direct Connect 문제 해결

추가 문제 해결은 AWS Direct Connect 문제 해결 설명서에서 확인할 수 있습니다.

3장. 클러스터 자동 스케일링

AWS 클러스터에서 Red Hat OpenShift Service에 자동 스케일링을 적용하려면 클러스터 자동 스케일러를 구성한 다음 클러스터에서 하나 이상의 머신 풀에 대한 머신 자동 스케일러를 구성해야 합니다.



중요

머신 API가 작동하는 클러스터에서만 클러스터 자동 스케일러를 구성할 수 있습니다.

클러스터당 하나의 클러스터 자동 스케일러만 생성할 수 있습니다.

3.1. 클러스터 자동 스케일러 정보

클러스터 자동 스케일러는 현재 배포 요구 사항에 맞게 AWS 클러스터의 Red Hat OpenShift Service 크기를 조정합니다. 이는 Kubernetes 형식의 선언적 인수를 사용하여 특정 클라우드 공급자의 개체에 의존하지 않는 인프라 관리를 제공합니다. 클러스터 자동 스케일러에는 클러스터 범위가 있으며 특정 네임 스페이스와 연결되어 있지 않습니다.

리소스가 부족하여 현재 작업자 노드에서 Pod를 예약할 수 없거나 배포 요구를 충족하기 위해 다른 노드가 필요한 경우 클러스터 자동 스케일러는 클러스터 크기를 늘립니다. 클러스터 자동 스케일러는 사용자가 지정한 제한을 초과하여 클러스터 리소스를 늘리지 않습니다.

클러스터 자동 스케일러는 컨트롤 플레인 노드를 관리하지 않더라도 클러스터의 모든 노드에서 총 메모리 및 CPU를 계산합니다. 이러한 값은 단일 시스템 지향이 아닙니다. 이는 전체 클러스터에 있는 모든 리소스를 집계한 것입니다. 예를 들어 최대 메모리 리소스 제한을 설정하면 현재 메모리 사용량을 계산할 때 클러스터 자동 스케일러에 클러스터의 모든 노드가 포함됩니다. 그런 다음 이 계산은 클러스터 자동 스케일러에 더 많은 작업자 리소스를 추가할 수 있는 용량이 있는지 확인하는 데 사용됩니다.

중요

작성한 **ClusterAutoscaler** 리솟스 정의의 **maxNodesTotal** 값이 클러스터에서 예상되는 총 머신 수를 대응하기에 충분한 크기의 값인지 확인합니다. 이 값에는 컨트롤 플레인 머신 수 및 확장 가능한 컴퓨팅 머신 수가 포함되어야 합니다.

10초마다 클러스터 자동 스케일러는 클러스터에서 불필요한 노드를 확인하고 제거합니다. 다음 조건이 적용되는 경우 클러스터 자동 스케일러는 노드가 제거되도록 간주합니다.

- 노드 사용률은 클러스터의 노드 *사용률 수준*임계값보다 적습니다. 노드 사용률 수준은 요청된 리소스의 합계를 노드에 할당된 리소스로 나눈 값입니다. **ClusterAutoscaler** 사용자 지정 리소스에서 값을 지정하지 않으면 클러스터 자동 스케일러는 기본값 **0.5** 를 사용하며 이는 사용률 50%에 해당합니다.
- 클러스터 자동 스케일러는 노드에서 실행 중인 모든 Pod를 다른 노드로 이동할 수 있습니다. Kubernetes 스케줄러는 노드에 Pod를 예약해야 합니다.
- 클러스터 자동 스케일러에는 축소 비활성화 주석이 없습니다.

노드에 다음 유형의 pod가 있는 경우 클러스터 자동 스케일러는 해당 노드를 제거하지 않습니다.

- 제한적인 PDB (Pod Disruption Budgets)가 있는 pod
- 기본적으로 노드에서 실행되지 않는 Kube 시스템 pod
- PDB가 없거나 제한적인 PDB가있는 Kube 시스템 pod

- deployment, replica set 또는 stateful set와 같은 컨트롤러 객체가 지원하지 않는 pod
- 로컬 스토리지가 있는 pod
- 리소스 부족, 호환되지 않는 노드 선택기 또는 어피니티(affinity), 안티-어피니티(anti-affinity) 일치 등으로 인해 다른 위치로 이동할 수 없는 pod
- **"cluster-autoscaler.kubernetes.io/safe-to-evict": "true"** 주석이없는 경우**"cluster-autoscaler.kubernetes.io/safe-to-evict": "false"** 주석이 있는 pod

예를 들어 최대 CPU 제한을 64코어로 설정하고 각각 코어가 8개인 머신만 생성하도록 클러스터 자동 스케일러를 구성합니다. 클러스터가 30개의 코어로 시작하는 경우 클러스터 자동 스케일러는 총 62개에 대해 32개의 코어가 있는 노드를 최대 4개까지 추가할 수 있습니다.

클러스터 자동 스케일러를 구성하면 추가 사용 제한이 적용됩니다.

- 자동 스케일링된 노드 그룹에 있는 노드를 직접 변경하지 마십시오. 동일한 노드 그룹 내의 모든 노드는 동일한 용량 및 레이블을 가지며 동일한 시스템 pod를 실행합니다.
- pod 요청을 지정합니다.
- pod가 너무 빨리 삭제되지 않도록 해야 하는 경우 적절한 PDB를 구성합니다.
- 클라우드 제공자 할당량이 구성하는 최대 노드 풀을 지원할 수 있는 충분한 크기인지를 확인합니다.
- 추가 노드 그룹 Autoscaler, 특히 클라우드 제공자가 제공하는 Autoscaler를 실행하지 마십시오.

HPA (Horizond Pod Autoscaler) 및 클러스터 자동 스케일러는 다른 방식으로 클러스터 리소스를 변경합니다. HPA는 현재 CPU 로드를 기준으로 배포 또는 복제 세트의 복제 수를 변경합니다. 로드가 증가하면 HPA는 클러스터에 사용 가능한 리소스 양에 관계없이 새 복제본을 만듭니다. 리소스가 충분하지 않은 경우 클러스터 자동 스케일러는 리소스를 추가하고 HPA가 생성한 pod를 실행할 수 있도록 합니다. 로드가 감소하면 HPA는 일부 복제를 중지합니다. 이 동작으로 일부 노드가 충분히 활용되지 않거나 완전히 비어 있을 경우 클러스터 자동 스케일러가 불필요한 노드를 삭제합니다.

클러스터 자동 스케일러는 pod 우선 순위를 고려합니다. Pod 우선 순위 및 선점 기능을 사용하면 클러스터에 충분한 리소스가 없는 경우 우선 순위에 따라 pod를 예약할 수 있지만 클러스터 자동 스케일러는 클러스터에 모든 pod를 실행하는 데 필요한 리소스가 있는지 확인합니다. 두 기능을 충족하기 위해 클러스터 자동 스케일러에는 우선 순위 컷오프 기능이 포함되어 있습니다. 이 컷오프 기능을 사용하여 "best-effort" pod를 예약하면 클러스터 자동 스케일러가 리소스를 늘리지 않고 사용 가능한 예비 리소스가 있을 때만 실행됩니다.

컷오프 값보다 우선 순위가 낮은 pod는 클러스터가 확장되지 않거나 클러스터가 축소되지 않도록합니다. pod를 실행하기 위해 추가된 새 노드가 없으며 이러한 pod를 실행하는 노드는 리소스를 확보하기 위해 삭제될 수 있습니다.

머신 API를 사용할 수 있는 플랫폼에서 클러스터 자동 스케일링이 지원됩니다.

3.2. OPENSHIFT CLUSTER MANAGER를 사용하여 클러스터 생성 중 자동 스케일링 활성화

OpenShift Cluster Manager를 사용하여 클러스터 생성 중에 자동 스케일링할 수 있습니다.

절차

1. 클러스터 생성 중에 자동 스케일링 활성화 확인란을 선택합니다. Edit cluster autoscaling settings 버튼을 선택합니다.

a. 자동 스케일링할 최소 또는 최대 노드 양을 선택할 수도 있습니다.

2. 클러스터 자동 스케일링 설정 편집을클릭합니다.

3. 원하는 설정을 편집한 다음 닫기 를 클릭합니다.

3.3. OPENSHIFT CLUSTER MANAGER를 사용하여 클러스터 생성 후 자동 스케일링 활성화

OpenShift Cluster Manager를 사용하여 클러스터 생성 후 자동 스케일링할 수 있습니다.

절차

1. OpenShift Cluster Manager에서 자동 스케일링할 클러스터의 이름을 클릭합니다. 클러스터의 개요 페이지에는 자동 확장 항목이 있으며 활성화되어 있는지 여부를 나타냅니다.

2. 머신 풀 탭을 클릭합니다.

3. Edit cluster autoscaling 버튼을 클릭합니다. Edit cluster autoscalingsettings 창이 표시됩니다.

4. 창 상단에 있는 자동 스케일링 클러스터 토글을 클릭합니다. 이제 모든 설정을 편집할 수 있습니다.

5. 원하는 설정을 편집한 다음 저장을 클릭합니다.

6. 화면 오른쪽 상단에 있는 x 를 클릭하여 설정 창을 닫습니다.

모든 자동 스케일링 설정을 기본값으로 되돌리려면 Revert all to defaults 버튼을 클릭합니다.

3.4. OPENSHIFT CLUSTER MANAGER를 사용한 클러스터 자동 스케일링 설정

이 표에서는 OpenShift Cluster Manager에서 클러스터 자동 스케일링을 사용할 때 구성 가능한 모든 UI 설정을 설명합니다.

3.4.1. 일반 설정

표 3.1. OpenShift Cluster Manager를 사용할 때 클러스터 자동 스케일링을 위한 구성 가능한 일반 설정

설정	설명	유형 또는 범위	Default
log-verbosity	자동 스케일러 로그 수준을 설정합니다. 기본값은 1입니다. 수준 4는 디버깅에 권장됩니다. 레벨 6은 거의 모든 것을 가능하게 합니다.	**integer**	1
skip-nodes-with-local-storage	**true** 인 경우 클러스터 자동 스케일러는 로컬 스토리지가 있는 Pod(예: EmptyDir 또는 HostPath)가 있는 노드를 삭제하지 않습니다.	**boolean**	true

설정	설명	유형 또는 범위	Default
max-pod-grace-period	축소하기 전에 Pod의 정상 종료 시간(초)을 제공합니다.	**integer**	600
max-node-provision-time	클러스터 자동 스케일러가 노드를 프로비저닝할 때까지 대기하는 최대 시간입니다.	**string**	15m
pod-priority-threshold	사용자가 "best-effort" Pod를 예약할 수 있습니다. 이 Pod는 클러스터 자동 스케일러 작업을 트리거할 수 없습니다. 이러한 Pod는 예비 리소스를 사용할 수 있는 경우에만 실행됩니다.	**integer**	-10
ignore-daemonsets-utilization	축소할 리소스 사용률을 계산할 때 클러스터 자동 스케일러가 데몬 세트 Pod를 무시하는지 여부를 결정합니다.	**boolean**	false
balance-similar-node-groups	**true** 인 경우 이 설정은 동일한 인스턴스 유형 및 동일한 레이블 세트로 노드 그룹을 자동으로 식별하고 해당 노드 그룹의 각 크기를 균형 있게 유지하려고 합니다.	**boolean**	false
balancing-ignored-labels	이 옵션은 노드 그룹 유사성을 고려할 때 클러스터 자동 스케일러가 무시해야 하는 레이블을 지정합니다. 이 옵션에는 공백을 포함할 수 없습니다.	**배열(문자열)**	형식은 쉼표로 구분된 레이블 목록이어야 합니다.

3.4.2. 리소스 제한

표 3.2. OpenShift Cluster Manager를 사용할 때 클러스터 자동 스케일링에 대해 구성 가능한 리소스 제한 설정

설정	설명	유형 또는 범위	Default

설정	설명	유형 또는 범위	Default
cores-total-min	클러스터의 최소 코어 수입니다. 클러스터 자동 스케일러는 이 수보다 클러스터를 확장하지 않습니다.	**object**	0
cores-total-max	클러스터의 최대 코어 수입니다. 클러스터 자동 스케일러는 이 수보다 클러스터를 확장하지 않습니다.	**object**	180 * 64 (11520)
memory-total-min	클러스터의 최소 메모리 수입니다. 클러스터 자동 스케일러는 이 수보다 클러스터를 확장하지 않습니다.	**object**	0
memory-total-max	클러스터의 최대 메모리 수입니다. 클러스터 자동 스케일러는 이 수보다 클러스터를 확장하지 않습니다.	**object**	180 * 64 * 20 (230400)
max-nodes-total	모든 노드 그룹의 최대 노드 수입니다. 자동으로 확장되는 노드가 아닌 모든 노드를 포함합니다. 클러스터 자동 스케일러는 이 수보다 클러스터를 늘리지 않습니다.	**integer**	180
GPU	클러스터에 있는 다른 GPU의 최소 및 최대 수입니다. 클러스터 자동 스케일러는 이러한 숫자보다 작거나 큰 클러스터를 확장하지 않습니다.	**array**	형식은 쉼표로 구분된 "<min>:<max>"{p> 목록이어야 합니다.

3.4.3. 구성 축소

표 3.3. OpenShift Cluster Manager를 사용할 때 클러스터 자동 스케일링을 위한 구성 가능한 축소 설정

설정	설명	유형 또는 범위	Default
scale-down-enabled	클러스터 자동 스케일러가 클러스터를 축소해야 합니까.	**boolean**	true

설정	설명	유형 또는 범위	Default
scale-down-utilization-threshold	요청된 리소스를 용량으로 나눈 총합으로 정의된 노드 사용률 수준보다 노드를 축소할 수 있습니다.	**플로트**	0.5
scale-down-unneeded-time	노드를 축소할 수 있기 전에 필요하지 않은 노드 수입니다.	**string**	10m
scale-down-delay-after-add	확장 후 축소 평가가 재개되는 시간입니다.	**string**	10m
scale-down-delay-after-delete	노드를 삭제한 후 축소 평가가 재개되는 기간입니다.	**string**	0s
scale-down-delay-after-failure	스케일 다운 평가가 다시 시작되는 실패 후의 시간입니다.	**string**	3m

3.5. ROSA CLI와 함께 대화형 모드를 사용하여 클러스터 생성 중 자동 스케일링 활성화

사용 가능한 경우 터미널의 대화형 모드를 사용하여 클러스터 생성 중에 클러스터 전체 자동 스케일링 동작을 설정할 수 있습니다.

대화형 모드는 사용 가능한 구성 가능한 매개변수에 대한 자세한 정보를 제공합니다. 대화형 모드는 기본 검사 및 preflight 검증을 수행합니다. 즉, 제공된 값이 유효하지 않으면 터미널에서 유효한 입력을 요청하는 프롬프트가 출력됩니다.

절차

- **클러스터 생성 중에 --enable-autoscaling 및 --interactive 매개변수를 사용하여 클러스터 자동 스케일링을 활성화합니다.**

  예제

  ```
  $ rosa create cluster --cluster-name <cluster_name> --enable-autoscaling --interactive
  ```

참고

클러스터 이름이 15자보다 긴 경우 *.openshiftapps.com 에서 프로비저닝된 클러스터의 하위 도메인으로 자동 생성된 도메인 접두사가 포함됩니다.

하위 도메인을 사용자 지정하려면 --domain-prefix 플래그를 사용합니다. 도메인 접두사는 15자를 초과할 수 없으며 고유해야 하며 클러스터 생성 후에는 변경할 수 없습니다.

다음 프롬프트가 표시되면 y 를 입력하여 사용 가능한 모든 자동 스케일링 옵션을 통과합니다.

대화형 프롬프트 예

```
? Configure cluster-autoscaler (optional): [? for help] (y/N) y <enter>
```

3.5.1. ROSA CLI와 함께 대화형 모드를 사용하여 클러스터 생성 후 자동 스케일링 활성화

사용 가능한 경우 터미널의 대화형 모드를 사용하여 클러스터 생성 후 클러스터 전체 자동 스케일링 동작을 설정할 수 있습니다.

절차

- 클러스터를 생성한 후 다음 명령을 입력합니다.

  예제

  ```
  $ rosa create autoscaler --cluster=<mycluster> --interactive
  ```

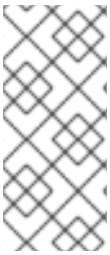
  그런 다음 사용 가능한 모든 자동 스케일링 매개변수를 설정할 수 있습니다.

3.6. ROSA CLI를 사용하여 클러스터 생성 중 자동 스케일링 활성화

ROSA CLI(**rosa**)를 사용하여 클러스터 생성 중에 클러스터 전체 자동 스케일링 동작을 설정할 수 있습니다. 전체 머신 또는 클러스터에서만 자동 스케일러를 활성화할 수 있습니다.

절차

- 클러스터 생성 중에 클러스터 이름 뒤에 **--enable** 자동 스케일링을 입력하여 머신 자동 스케일링을 활성화합니다.



참고

클러스터 이름이 15자보다 긴 경우 ***.openshiftapps.com** 에서 프로비저닝된 클러스터의 하위 도메인으로 자동 생성된 도메인 접두사가 포함됩니다.

하위 도메인을 사용자 지정하려면 **--domain-prefix** 플래그를 사용합니다. 도메인 접두사는 15자를 초과할 수 없으며 고유해야 하며 클러스터 생성 후에는 변경할 수 없습니다.

예제

```
$ rosa create cluster --cluster-name <cluster_name> --enable-autoscaling
```

다음 명령을 실행하여 클러스터 자동 스케일링을 활성화하려면 하나 이상의 매개변수를 설정합니다.

예제

```
$ rosa create cluster --cluster-name <cluster_name> --enable-autoscaling <parameter>
```

3.6.1. ROSA CLI를 사용하여 클러스터 생성 후 자동 스케일링 활성화

ROSA CLI(**rosa**)를 사용하여 클러스터 생성 후 클러스터 전체 자동 스케일링을 설정할 수 있습니다.

절차

- 클러스터를 생성한 후 자동 스케일러를 생성합니다.

  예제

  ```
  $ rosa create autoscaler --cluster=<mycluster>
  ```

  a. 다음 명령을 사용하여 특정 매개변수로 자동 스케일러를 생성할 수도 있습니다.

  예제

  ```
  $ rosa create autoscaler --cluster=<mycluster> <parameter>
  ```

3.6.2. ROSA CLI를 사용하여 클러스터 생성 후 자동 스케일링 편집

자동 스케일러를 생성한 후 클러스터 자동 스케일러의 특정 매개변수를 편집할 수 있습니다.

- 클러스터 자동 스케일러를 편집하려면 다음 명령을 실행합니다.

  예제

  ```
  $ rosa edit autoscaler --cluster=<mycluster>
  ```

  a. 특정 매개변수를 편집하려면 다음 명령을 실행합니다.

  예제

  ```
  $ rosa edit autoscaler --cluster=<mycluster> <parameter>
  ```

3.6.3. ROSA CLI를 사용하여 자동 스케일링 삭제

더 이상 사용하지 않으려면 클러스터 자동 스케일러를 삭제할 수 있습니다.

- 클러스터 자동 스케일러를 삭제하려면 다음 명령을 실행합니다.

  예제

  ```
  $ rosa delete autoscaler --cluster=<mycluster>
  ```

3.7. ROSA CLI를 사용한 클러스터 자동 스케일링 매개변수

ROSA CLI(**rosa**)를 사용할 때 자동 스케일러 매개변수를 구성하려면 클러스터 생성 명령에 다음 매개변수를 추가할 수 있습니다.

표 3.4. ROSA CLI에서 사용할 수 있는 구성 가능한 자동 스케일러 매개변수 (**osa**)

설정	설명	유형 또는 범위	예/디렉션
--autoscaler-balance-similar-node-groups	동일한 인스턴스 유형 및 레이블이 설정된 노드 그룹을 식별하고 해당 노드 그룹의 각 크기를 조정하십시오.	**boolean**	true로 설정하려면 옵션을 생략하여 false로 설정합니다.
--autoscaler-skip-nodes-with-local-storage	설정된 경우 클러스터 자동 스케일러는 로컬 스토리지가 있는 Pod(예: EmptyDir 또는 HostPath)가 있는 노드를 삭제하지 않습니다.	**boolean**	true로 설정하려면 옵션을 생략하여 false로 설정합니다.
--autoscaler-log-verbosity *int*	자동 스케일러 로그 수준. 명령에서 *int* 를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-log-verbosity 4**
--autoscaler-max-pod-grace-period *int*	축소 전 Pod의 정상 종료 시간을 초 단위로 측정합니다. 명령에서 *int* 를 사용하려는 초 수로 바꿉니다.	**integer**	**--autoscaler-max-pod-grace-period 0**
--autoscaler-pod-priority-threshold *int*	클러스터 자동 스케일러가 추가 노드를 배포하도록 하려면 Pod가 초과해야하는 우선 순위입니다. 명령에서 *int* 를 사용할 번호로 바꿉니다. 음수가 될 수 있습니다.	**integer**	**--autoscaler-pod-priority-threshold -10**
--autoscaler-gpu-limit *stringArray*	클러스터에 있는 다른 GPU의 최소 및 최대 수입니다. 클러스터 자동 스케일러는 이러한 숫자보다 작거나 큰 클러스터를 확장하지 않습니다. 형식은 쉼표로 구분된 ",<min>, <max>"{p> 목록이어야 합니다.	**array**	**--autoscaler-gpu-limit nvidia.com/gpu,0,10 --autoscaler-gpu-limit amd.com/gpu,1,5**
--autoscaler-ignore-daemonsets-utilization	설정된 경우 cluster-autoscaler는 축소를 위해 리소스 사용률을 계산할 때 데몬 세트 Pod를 무시합니다.	**boolean**	true로 설정하려면 옵션을 생략하여 false로 설정합니다.

설정	설명	유형 또는 범위	예/디렉션
--autoscaler-max-node-provision-time *string*	클러스터 자동 스케일러가 노드를 프로비저닝할 때까지 대기하는 최대 시간입니다. 명령의 *문자열*을 정수 및 시간 단위(ns,us, Cryostats,ms,s,m,h)로 바꿉니다.	**string**	**--autoscaler-max-node-provision-time 35m**
--autoscaler-balancing-ignored-labels *strings*	노드 그룹을 유사성과 비교할 때 클러스터 자동 스케일러가 무시해야 하는 쉼표로 구분된 레이블 키 목록입니다. 명령의 *문자열*을 관련 레이블로 바꿉니다.	**string**	**--autoscaler-balancing-ignored-labels topology.ebs.csi.aws.com/zone,alpha.eksctl.io/instance-id**
--autoscaler-max-nodes-total *int*	자동 스케일링된 노드를 포함하여 클러스터의 최대 노드 양입니다. 명령에서 *int*를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-max-nodes-total 180**
--autoscaler-min-cores *int*	클러스터에 배포할 최소 코어 수입니다. 명령에서 *int*를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-min-cores 0**
--autoscaler-max-cores *int*	클러스터에 배포할 최대 코어 수입니다. 명령에서 *int*를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-max-cores 100**
--autoscaler-min-memory *int*	클러스터에서 최소 메모리 양(GiB)입니다. 명령에서 *int*를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-min-memory 0**
--autoscaler-max-memory *int*	클러스터의 최대 메모리 양(GiB)입니다. 명령에서 *int*를 사용할 번호로 바꿉니다.	**integer**	**--autoscaler-max-memory 4096**
--autoscaler-scale-down-enabled	설정된 경우 클러스터 자동 스케일러는 클러스터를 축소해야 합니다.	**boolean**	true로 설정하려면 옵션을 생략하여 false로 설정합니다.

설정	설명	유형 또는 범위	예/디렉션
--autoscaler-scale-down-unneeded-time *string*	노드를 축소할 수 있기 전에 필요하지 않은 노드 수입니다. 명령의 *문자열*을 정수 및 시간 단위(ns,us, Cryostats,ms,s,m,h)로 바꿉니다.	**string**	**--autoscaler-scale-down-unneeded-time 1h**
--autoscaler-scale-down-utilization-threshold *float*	요청된 리소스의 합계로 구성된 노드 사용률 수준으로, 용량으로 나눈 후 노드를 축소할 수 있습니다. 값은 0에서 1 사이여야 합니다.	**플로트**	**--autoscaler-scale-down-utilization-threshold 0.5**
--autoscaler-scale-down-delay-after-add *string*	평가 규모를 축소한 후 평가가 재개되는 기간이 얼마나 됩니까. 명령의 *문자열*을 정수 및 시간 단위(ns,us, Cryostats,ms,s,m,h)로 바꿉니다.	**string**	**--autoscaler-scale-down-delay-after-add 1h**
--autoscaler-scale-down-delay-after-delete *string*	평가를 축소하는 노드 삭제 후의 기간이 다시 시작되는 기간입니다. 명령의 *문자열*을 정수 및 시간 단위(ns,us, Cryostats,ms,s,m,h)로 바꿉니다.	**string**	**--autoscaler-scale-down-delay-after-delete 1h**
--autoscaler-scale-down-delay-after-failure *string*	평가를 축소하는 실패 후의 시간입니다. 명령의 *문자열*을 정수 및 시간 단위(ns,us, Cryostats,ms,s,m,h)로 바꿉니다.	**string**	**--autoscaler-scale-down-delay-after-failure 1h**

4장. 머신 풀을 사용하여 노드 관리

4.1. 머신 풀 정보

AWS의 Red Hat OpenShift Service는 클라우드 인프라에 더하여 탄력적이고 동적인 프로비저닝 방법으로 머신 풀을 사용합니다.

기본 리소스는 시스템, 컴퓨팅 머신 세트, 시스템 풀입니다.

4.1.1. Machine

머신은 작업자 노드의 호스트를 설명하는 기본 단위입니다.

4.1.2. 머신 세트

MachineSet 리소스는 컴퓨팅 머신 그룹입니다. 더 많은 시스템이 필요하거나 규모를 줄여야 하는 경우 컴퓨팅 시스템 세트가 속하는 시스템 풀의 복제본 수를 변경합니다.

머신 세트는 ROSA에서 직접 수정할 수 없습니다.

4.1.3. 머신 풀

머신 풀은 컴퓨팅 머신 세트에 대한 더 높은 수준의 구성 요소입니다.

시스템 풀은 모두 가용성 영역에서 동일한 구성을 복제하는 컴퓨팅 머신 세트를 생성합니다. 머신 풀은 작업자 노드에서 모든 호스트 노드 프로비저닝 관리 작업을 수행합니다. 더 많은 머신이 필요하거나 규모를 줄여야하는 경우 컴퓨팅 요구 사항에 맞게 시스템 풀의 복제본 수를 변경합니다. 스케일링을 수동으로 구성하거나 자동 스케일링을 설정할 수 있습니다.

단일 클러스터에 여러 머신 풀이 존재할 수 있으며 각 머신 풀에는 고유한 노드 유형 및 노드 크기 구성이 포함될 수 있습니다.

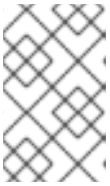
4.1.3.1. 클러스터 설치 중 머신 풀

기본적으로 클러스터에는 하나의 시스템 풀이 있습니다. 클러스터 설치 중에 인스턴스 유형 또는 크기를 정의하고 이 머신 풀에 레이블을 추가할 수 있습니다.

4.1.3.2. 클러스터 설치 후 머신 풀 구성

클러스터 설치 후 다음을 수행합니다.

- 머신 풀에 레이블을 제거하거나 추가할 수 있습니다.
- 기존 클러스터에 머신 풀을 추가할 수 있습니다.
- 테인트 없이 머신 풀이 한 개 있는 경우 머신 풀에 테인트를 추가할 수 있습니다.
- 테인트가 없는 머신 풀과 Single-AZ 클러스터에 대해 두 개 이상의 복제본이 있거나 Multi-AZ 클러스터의 복제본 3개가 있는 경우 머신 풀을 생성하거나 삭제할 수 있습니다.



참고

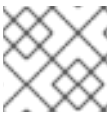
머신 풀 노드 유형 또는 크기는 변경할 수 없습니다. 머신 풀 노드 유형 또는 크기는 생성 중에만 지정됩니다. 다른 노드 유형 또는 크기가 필요한 경우 머신 풀을 다시 생성하고 필요한 노드 유형 또는 크기 값을 지정해야 합니다.

- 추가된 각 머신 풀에 레이블을 추가할 수 있습니다.

4.1.4. 여러 영역 클러스터의 머신 풀

여러 가용성 영역(AZ)에서 생성된 클러스터에서 머신 풀은 세 개의 AZ 또는 선택한 단일 AZ 모두에서 생성할 수 있습니다. 클러스터 생성 시 기본적으로 생성된 머신 풀은 3개의 AZ 모두에 있는 시스템과 3개의 배수로 스케일링됩니다.

새 Multi-AZ 클러스터를 생성하면 머신 풀이 해당 영역에 자동으로 복제됩니다. 기본적으로 기존 Multi-AZ 클러스터에 머신 풀을 추가하면 모든 영역에서 새 머신 풀이 자동으로 생성됩니다.



참고

이 기본 설정을 재정의하고 선택한 Single-AZ에서 머신 풀을 생성할 수 있습니다.

마찬가지로 머신 풀을 삭제하면 모든 영역에서 삭제됩니다. 이러한 다기능 효과 때문에 Multi-AZ 클러스터에서 머신 풀을 사용하면 머신 풀을 생성할 때 특정 리전에 더 많은 프로젝트의 할당량을 사용할 수 있습니다.

4.1.5. 추가 리소스

- 컴퓨팅 노드 관리
- 자동 스케일링 정보
- PID 제한 구성

4.2. 컴퓨팅 노드 관리

이 문서에서는 ROSA(Red Hat OpenShift Service on AWS)를 사용하여 컴퓨팅(작업자) 노드를 관리하는 방법을 설명합니다.

컴퓨팅 노드의 대부분의 변경 사항은 시스템 풀에 구성됩니다. 머신 풀은 동일한 구성이 있어 쉽게 관리할 수 있는 클러스터의 컴퓨팅 노드 그룹입니다.

스케일링, 노드 레이블 추가, 테인트 추가와 같은 머신 풀 구성 옵션을 편집할 수 있습니다.

4.2.1. 머신 풀 생성

ROSA(Red Hat OpenShift Service) 클러스터를 설치할 때 시스템 풀이 생성됩니다. 설치 후 OpenShift Cluster Manager 또는 ROSA CLI(**rosa**)를 사용하여 클러스터에 대한 추가 머신 풀을 생성할 수 있습니다.

참고

ROSA CLI **rosa** 버전 1.2.25 및 이전 버전의 사용자의 경우 클러스터와 함께 생성된 머신 풀은 **Default** 로 식별됩니다. ROSA CLI 버전 **1.2.26** 이상 사용자의 경우 클러스터와 함께 생성된 머신 풀은 **worker** 로 식별됩니다.

4.2.1.1. OpenShift Cluster Manager를 사용하여 머신 풀 생성

OpenShift Cluster Manager를 사용하여 AWS(ROSA) 클러스터에서 Red Hat OpenShift Service를 위한 추가 머신 풀을 생성할 수 있습니다.

사전 요구 사항

- ROSA 클러스터를 생성하셨습니다.

절차

1. OpenShift Cluster Manager 로 이동하여 클러스터를 선택합니다.
2. 머신 풀 탭에서 머신 풀 추가 를 클릭합니다.
3. 머신 풀 이름을 추가합니다.
4. 드롭다운 메뉴에서 컴퓨팅 노드 인스턴스 유형을 선택합니다. 인스턴스 유형은 시스템 풀의 각 계산 노드에 대한 vCPU 및 메모리 할당을 정의합니다.

참고

풀을 생성한 후에는 머신 풀의 인스턴스 유형을 변경할 수 없습니다.

5. 선택 사항: 머신 풀의 자동 스케일링을 구성합니다.
   a. 배포 요구에 맞게 자동 스케일링 활성화를 선택하여 머신 풀의 머신 수를 자동으로 스케일링합니다.
   b. 자동 스케일링에 대한 최소 및 최대 노드 수 제한을 설정합니다. 클러스터 자동 스케일러는 지정한 제한을 초과하여 머신 풀 노드 수를 줄이거나 늘리지 않습니다.
      - 단일 가용성 영역을 사용하여 클러스터를 배포한 경우 최소 및 최대 노드 수를 설정합니다. 이는 가용성 영역에 최소 및 최대 컴퓨팅 노드 제한을 정의합니다.
      - 여러 가용성 영역을 사용하여 클러스터를 배포한 경우 영역당 최소 노드 및 영역당최대 노드를 설정합니다. 이는 영역당 최소 및 최대 컴퓨팅 노드 제한을 정의합니다.

참고

또는 머신 풀을 생성한 후 머신 풀에 대한 자동 스케일링 기본 설정을 설정할 수 있습니다.

6. 자동 스케일링을 활성화하지 않은 경우 컴퓨팅 노드 수를 선택합니다.
   - 단일 가용성 영역을 사용하여 클러스터를 배포한 경우 드롭다운 메뉴에서 컴퓨팅 노드 수를 선택합니다. 이 명령은 영역의 시스템 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다.
   - 여러 가용성 영역을 사용하여 클러스터를 배포한 경우 드롭다운 메뉴에서 컴퓨팅 노드 수(영역당) 를 선택합니다. 이는 영역당 시스템 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다.
7. 선택 사항: 루트 디스크 크기를 구성합니다.
8. 선택사항: 머신 풀의 노드 레이블 및 테인트를 추가합니다.
   a. Edit node labels and taints 메뉴를 확장합니다.

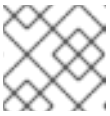
b. 노드 라벨 에서 노드 레이블에 대한Key 및 Value 항목을 추가합니다.

c. 테인트에서 테인트의 키 및 값 항목을 추가합니다.

참고

테인트를 사용하여 머신 풀을 생성하는 것은 클러스터에 테인트 없이 하나 이상의 머신 풀이 이미 있는 경우에만 가능합니다.

d. 각 테인트에 대해 드롭다운 메뉴에서 Effect 를 선택합니다. 사용 가능한 옵션에는 **NoSchedule**,**PreferNoSchedule** 및 **NoExecute** 가 포함됩니다.



참고

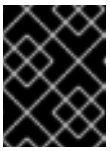
또는 머신 풀을 생성한 후 노드 레이블 및 테인트를 추가할 수 있습니다.

9. 선택 사항: 이 머신 풀의 노드에 사용할 추가 사용자 지정 보안 그룹을 선택합니다. 보안 그룹을 이미 생성하여 이 클러스터에 대해 선택한 VPC와 연결되어야 합니다. 머신 풀을 생성한 후에는 보안 그룹을 추가하거나 편집할 수 없습니다. 자세한 내용은 "추가 리소스" 섹션의 보안 그룹에 대한 요구 사항을 참조하십시오.

10. 선택 사항: 머신 풀을 구성하여 보장되지 않는 AWS Spot 인스턴스로 머신을 구성하려면 Amazon EC2 Spot 인스턴스를 사용합니다.

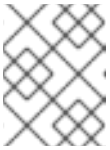
a. Amazon EC2 Spot 인스턴스 사용을선택합니다.

b. 온 디맨드 인스턴스 가격을사용하도록 선택한 온 디맨드 인스턴스 가격은 그대로 둡니다. 또는 최대 가격 설정을 선택하여 Spot 인스턴스의 최대 시간당 가격을 정의합니다.
Amazon EC2 Spot 인스턴스에 대한 자세한 내용은 AWS 설명서 를 참조하십시오.



중요

언제든지 Amazon EC2 Spot 인스턴스가 중단될 수 있습니다. 중단을 허용할 수 있는 워크로드에만 Amazon EC2 Spot 인스턴스를 사용합니다.



참고

머신 풀에 Amazon EC2 Spot 인스턴스 사용을선택하는 경우 머신 풀을 생성한 후에는 옵션을 비활성화할 수 없습니다.

11. 머신 풀 추가를 클릭하여 머신 풀을 생성합니다.

검증

- 시스템 풀이 머신 풀 페이지에 표시되고 구성이 예상대로 표시되는지 확인합니다.

추가 리소스

- 추가 사용자 정의 보안 그룹

4.2.1.2. ROSA CLI를 사용하여 머신 풀 생성

ROSA CLI(**rosa**)를 사용하여 AWS(ROSA) 클러스터에서 Red Hat OpenShift Service를 위한 추가 머신 풀을 생성할 수 있습니다.

사전 요구 사항

- 워크스테이션에 최신 Red Hat OpenShift Service on AWS(ROSA) CLI 를 설치하고 구성하셨습니다.
- ROSA CLI(**rosa**)를 사용하여 Red Hat 계정에 로그인했습니다.
- ROSA 클러스터를 생성하셨습니다.

절차

- 자동 스케일링을 사용하지 않는 머신 풀을 추가하려면 머신 풀을 생성하고 인스턴스 유형, 컴퓨팅 (작업자) 노드 수 및 노드 레이블을 정의합니다.

```
$ rosa create machinepool --cluster=<cluster-name> \
                          --name=<machine_pool_id> \1
                          --replicas=<replica_count> \2
                          --instance-type=<instance_type> \3
                          --labels=<key>=<value>,<key>=<value> \4
                          --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \5
                          --use-spot-instances \6
                          --spot-max-price=0.5 \7
                          --disk-size=<disk_size> \8
                          --availability-zone=<availability_zone_name> \9
                          --additional-security-group-ids <sec_group_id> \10
                          --subnet string 11
```

1. 시스템 풀의 이름을 지정합니다. & lt;machine_pool_id >를 머신 풀 이름으로 바꿉니다.
2. 프로비저닝할 컴퓨팅 노드 수를 지정합니다. 단일 가용성 영역을 사용하여 ROSA를 배포한 경우 영역의 머신 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 모든 영역에서 총 프로비저닝할 컴퓨팅 노드 수를 정의하고 개수는 3개 중 일부여야 합니다. 자동 스케일링이 구성되지 않은 경우 **--replicas** 인수가 필요합니다.
3. 선택 사항: 시스템 풀의 컴퓨팅 노드의 인스턴스 유형을 설정합니다. 인스턴스 유형은 풀의 각 계산 노드에 대한 vCPU 및 메모리 할당을 정의합니다. &lt ;instance_type>을 인스턴스 유형으로 바꿉니다. 기본값은 **m5.xlarge** 입니다. 풀을 생성한 후에는 머신 풀의 인스턴스 유형을 변경할 수 없습니다.
4. 선택 사항: 머신 풀의 레이블을 정의합니다. < key>=<value>,<key>=<value >를 쉼표로 구분된 키-값 쌍 목록(예: **--labels=key1=value1,key2=value** 2)로 바꿉니다.
5. 선택 사항: 머신 풀의 테인트를 정의합니다. < key>=<value>:<effect>,<key>=<value>:<effect >:<effect>를 각 테인트의 키, 값 및 효과(예:--taints=key1=value1:NoSchedule,key2=value2:NoSchedule,key2=value2:NoExecute)로 바꿉니다. 사용 가능한 효과에는 **NoSchedule**,**PreferNoSchedule** 및 **NoExecute** 가 포함됩니다.
6. 선택 사항: 머신 풀을 구성하여 머신을 보장되지 않는 AWS Spot 인스턴스로 배포합니다. 자세한 내용은 AWS 문서의 Amazon EC2 Spot 인스턴스를 참조하십시오. 머신 풀에 Amazon EC2 Spot 인스턴스 사용을 선택하는 경우 머신 풀을 생성한 후에는 옵션을 비활성화할 수 없

습니다.

7 선택 사항: Spot 인스턴스를 사용하도록 선택하는 경우 이 인수를 지정하여 Spot 인스턴스의 최대 시간당 가격을 정의할 수 있습니다. 이 인수를 지정하지 않으면 온 디맨드 가격이 사용됩니다.

중요

언제든지 Amazon EC2 Spot 인스턴스가 중단될 수 있습니다. 중단을 허용할 수 있는 워크로드에만 Amazon EC2 Spot 인스턴스를 사용합니다.

8 선택 사항: 작업자 노드 디스크 크기를 지정합니다. 값은 GB, GiB, TB 또는 TiB일 수 있습니다. &lt ;disk_size >를 숫자 값 및 단위(예:--disk-size=200GiB)로 바꿉니다.

9 선택 사항: 다중 AZ 클러스터의 경우 선택한 Single-AZ에서 머신 풀을 생성할 수 있습니다. &lt ;az& gt;를 Single-AZ 이름으로 바꿉니다.

참고

다중 AZ 클러스터는 Multi-AZ 컨트롤 플레인을 유지하며 Single-AZ 또는 Multi-AZ에 작업자 머신 풀을 가질 수 있습니다. 머신 풀은 가용성 영역에서 머신(노드)을 균등하게 배포합니다.

주의

Single-AZ가 있는 작업자 머신 풀을 선택하는 경우 머신 복제본 수에 관계없이 해당 머신 풀에 대한 내결함성이 없습니다. 내결함성 작업자 시스템 풀의 경우 Multi-AZ 머신 풀을 선택하면 가용성 영역 간에 3개의 시스템이 분산됩니다.

- 세 개의 가용성 영역이 있는 Multi-AZ 머신 풀은 3개, 6, 9 등과 같이 3개의 머신 수만 머신 수를 가질 수 있습니다.
- 가용성 영역이 하나의 단일 AZ 머신 풀은 1, 2, 3, 4, 등과 같은 1의 배수에 머신 수를 가질 수 있습니다.

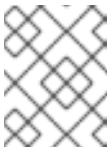
10 선택 사항: Red Hat 관리 VPC가 없는 클러스터의 머신 풀의 경우 머신 풀에서 사용할 추가 사용자 지정 보안 그룹을 선택할 수 있습니다. 보안 그룹을 이미 생성하여 이 클러스터에 대해 선택한 VPC와 연결되어야 합니다. 머신 풀을 생성한 후에는 보안 그룹을 추가하거나 편집할 수 없습니다. 자세한 내용은 "추가 리소스" 섹션의 보안 그룹에 대한 요구 사항을 참조하십시오.

11 선택 사항: BYO VPC 클러스터의 경우 서브넷을 선택하여 Single-AZ 머신 풀을 생성할 수 있습니다. 서브넷이 클러스터 생성 서브넷에서 벗어나는 경우 키 **kubernetes.io/cluster/<infra-id >** 및 값이**shared** 인 태그가 있어야 합니다. 고객은 다음 명령을 사용하여 Infra ID를 얻을 수 있습니다.

```
$ rosa describe cluster -c <cluster name>|grep "Infra ID:"
```

출력 예

```
Infra ID:                   mycluster-xqvj7
```



참고

--subnet 및 **--availability-zone** 을 동시에 설정할 수 없으며 Single-AZ 머신 풀 생성에는 1개만 허용됩니다.

다음 예제에서는 **m5.xlarge** 인스턴스 유형을 사용하고 컴퓨팅 노드 복제본이 2개인 **mymachinepool** 이라는 시스템 풀을 생성합니다. 이 예제에서는 두 개의 워크로드별 라벨도 추가합니다.

```
$ rosa create machinepool --cluster=mycluster --name=mymachinepool --replicas=2 --
instance-type=m5.xlarge --labels=app=db,tier=backend
```

출력 예

```
I: Machine pool 'mymachinepool' created successfully on cluster 'mycluster'
I: To view all machine pools, run 'rosa list machinepools -c mycluster'
```

- 자동 스케일링을 사용하는 머신 풀을 추가하려면 머신 풀을 생성하고 자동 스케일링 구성, 인스턴스 유형 및 노드 레이블을 정의합니다.

```
$ rosa create machinepool --cluster=<cluster-name> \
                          --name=<machine_pool_id> \ 1
                          --enable-autoscaling \ 2
                          --min-replicas=<minimum_replica_count> \ 3
                          --max-replicas=<maximum_replica_count> \ 4
                          --instance-type=<instance_type> \ 5
                          --labels=<key>=<value>,<key>=<value> \ 6
                          --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 7
                          --availability-zone=<availability_zone_name> \ 8
                          --use-spot-instances \ 9
                          --spot-max-price=0.5 10
```

1. 시스템 풀의 이름을 지정합니다. & lt;machine_pool_id >를 머신 풀 이름으로 바꿉니다.
2. 머신 풀에서 자동 스케일링을 활성화하여 배포 요구 사항을 충족할 수 있습니다.
3. 4 최소 및 최대 컴퓨팅 노드 제한을 정의합니다. 클러스터 자동 스케일러는 지정한 제한을 초과하여 머신 풀 노드 수를 줄이거나 늘리지 않습니다. 단일 가용성 영역을 사용하여 ROSA를 배포한 경우 **--min-replicas** 및 **--max-replicas** 인수는 영역의 시스템 풀에 자동 스케일링 제한을 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 인수는 모든 영역에서 총 자동 스케일링 제한을 정의하고 개수는 3개여야 합니다.
5. 선택 사항: 시스템 풀의 컴퓨팅 노드의 인스턴스 유형을 설정합니다. 인스턴스 유형은 풀의 각 계산 노드에 대한 vCPU 및 메모리 할당을 정의합니다. &lt ;instance_type>을 인스턴스 유형으로 바꿉니다. 기본값은 **m5.xlarge** 입니다. 풀을 생성한 후에는 머신 풀의 인스턴스 유형을 변경할 수 없습니다.
6. 선택 사항: 머신 풀의 레이블을 정의합니다. < key>=<value>,<key>=<value >를 쉼표로 구분된 키-값 쌍 목록(예: **--labels=key1=value1,key2=value** 2)로 바꿉니다.

7 선택 사항: 머신 풀의 테인트를 정의합니다. < key>=<value>:<effect>,<key>=<value>:<effect >:<effect>를 각 테인트의 키, 값 및 효과(예:--

8 선택 사항: 다중 AZ 클러스터의 경우 선택한 Single-AZ에서 머신 풀을 생성할 수 있습니다. <az& gt;를 Single-AZ 이름으로 바꿉니다.

9 선택 사항: 머신 풀을 구성하여 머신을 보장되지 않는 AWS Spot 인스턴스로 배포합니다. 자세한 내용은 AWS 문서의 Amazon EC2 Spot 인스턴스를 참조하십시오. 머신 풀에 Amazon EC2 Spot 인스턴스 사용을 선택하는 경우 머신 풀을 생성한 후에는 옵션을 비활성화할 수 없습니다.

중요

언제든지 Amazon EC2 Spot 인스턴스가 중단될 수 있습니다. 중단을 허용할 수 있는 워크로드에만 Amazon EC2 Spot 인스턴스를 사용합니다.

10 선택 사항: Spot 인스턴스를 사용하도록 선택하는 경우 이 인수를 지정하여 Spot 인스턴스의 최대 시간당 가격을 정의할 수 있습니다. 이 인수를 지정하지 않으면 온 디맨드 가격이 사용됩니다.

다음 예제에서는 **m5.xlarge** 인스턴스 유형을 사용하고 자동 스케일링이 활성화된 **mymachinepool** 이라는 시스템 풀을 생성합니다. 최소 컴퓨팅 노드 제한은 3이며 최대값은 총 6개입니다. 이 예제에서는 두 개의 워크로드별 라벨도 추가합니다.

```
$ rosa create machinepool --cluster=mycluster --name=mymachinepool --enable-autoscaling --min-replicas=3 --max-replicas=6 --instance-type=m5.xlarge --labels=app=db,tier=backend
```

출력 예

```
I: Machine pool 'mymachinepool' created successfully on cluster 'mycluster'
I: To view all machine pools, run 'rosa list machinepools -c mycluster'
```

검증

클러스터의 모든 머신 풀을 나열하거나 개별 머신 풀을 설명할 수 있습니다.

1. 클러스터에서 사용 가능한 머신 풀을 나열합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

출력 예

```
ID           AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS              TAINTS  AVAILABILITY ZONES              SPOT INSTANCES
Default      No           3         m5.xlarge                                  us-east-1a, us-east-1b, us-east-1c  N/A
mymachinepool  Yes        3-6       m5.xlarge      app=db, tier=backend        us-east-1a, us-east-1b, us-east-1c  No
```

2. 클러스터의 특정 머신 풀 정보를 설명합니다.

```
$ rosa describe machinepool --cluster=<cluster_name> --
machinepool=mymachinepool
```

출력 예

```
ID:                  mymachinepool
Cluster ID:          27iimopsg1mge0m81l0sqivkne2qu6dr
Autoscaling:           Yes
Replicas:            3-6
Instance type:         m5.xlarge
Labels:              app=db, tier=backend
Taints:
Availability zones:      us-east-1a, us-east-1b, us-east-1c
Subnets:
Spot instances:        No
Disk size:           300 GiB
Security Group IDs:
```

3. 시스템 풀이 출력에 포함되어 있고 구성이 예상대로 구성되어 있는지 확인합니다.

추가 리소스

- 추가 사용자 정의 보안 그룹

4.2.2. 머신 풀 디스크 볼륨 구성

유연성을 높이기 위해 머신 풀 디스크 볼륨 크기를 구성할 수 있습니다. 기본 디스크 크기는 300GiB입니다. 클러스터 버전 4.13 이하의 경우 디스크 크기를 최소 128GiB에서 최대 1TiB로 구성할 수 있습니다. 클러스터 버전 4.14 이상의 경우 디스크 크기를 최소 128GiB에서 최대 16TiB로 구성할 수 있습니다.

OpenShift Cluster Manager 또는 ROSA CLI(**rosa**)를 사용하여 클러스터의 머신 풀 디스크 크기를 구성할 수 있습니다.

참고

기존 클러스터 및 머신 풀 노드 볼륨의 크기를 조정할 수 없습니다.

중요

기본 디스크 크기는 300GiB입니다. 클러스터 버전 4.13 이하의 경우 디스크 크기를 최소 128GiB에서 최대 1TiB로 구성할 수 있습니다. 클러스터 버전 4.14 이상의 경우 디스크 크기를 최소 128GiB에서 최대 16TiB로 구성할 수 있습니다.

4.2.2.1. OpenShift Cluster Manager를 사용하여 머신 풀 디스크 볼륨 구성

클러스터 생성을 위한 사전 요구 사항

- 클러스터 설치 중에 기본 머신 풀의 노드 디스크 크기 조정을 선택하는 옵션이 있습니다.

클러스터 생성 절차

1. ROSA 클러스터 마법사에서 클러스터 설정으로 이동합니다.

2. 머신 풀 단계로 이동합니다.
3. 원하는 루트 디스크 크기를 선택합니다.
4. 다음을 선택하여 클러스터를 계속 생성합니다.

머신 풀 생성을 위한 사전 요구 사항

- 클러스터를 설치한 후 새 시스템 풀의 노드 디스크 크기 조정을 선택하는 옵션이 있습니다.

머신 풀 생성 절차

1. OpenShift Cluster Manager 로 이동하여 클러스터를 선택합니다.
2. 머신 풀 탭으로 이동합니다.
3. 머신 풀 추가를 클릭합니다.
4. 원하는 루트 디스크 크기를 선택합니다.
5. 머신 풀 추가 를 선택하여 머신 풀을 생성합니다.

4.2.2.2. ROSA CLI를 사용하여 머신 풀 디스크 볼륨 구성

클러스터 생성을 위한 사전 요구 사항

- 클러스터 설치 중에 기본 머신 풀의 루트 디스크 크기 조정을 선택하는 옵션이 있습니다.

클러스터 생성 절차

- 원하는 루트 디스크 크기에 맞게 OpenShift 클러스터를 생성할 때 다음 명령을 실행합니다.

```
$ rosa create cluster --worker-disk-size=<disk_size>
```

값은 GB, GiB, TB 또는 TiB일 수 있습니다. < **disk_size** >를 숫자 값 및 단위(예:**--worker-disk-size=200GiB**)로 바꿉니다. 숫자와 단위를 분리할 수 없습니다. 공백은 허용되지 않습니다.

머신 풀 생성을 위한 사전 요구 사항

- 클러스터를 설치한 후 새 시스템 풀의 루트 디스크 크기 조정을 선택하는 옵션이 있습니다.

머신 풀 생성 절차

1. 다음 명령을 실행하여 클러스터를 확장합니다.

```
$ rosa create machinepool --cluster=<cluster_id> \ 1
                          --disk-size=<disk_size> 2
```

1 기존 OpenShift 클러스터의 ID 또는 이름을 지정합니다.

2 작업자 노드 디스크 크기를 지정합니다. 값은 GB, GiB, TB 또는 TiB일 수 있습니다. &lt ;disk_size >를 숫자 값 및 단위(예:**--disk-size=200GiB**)로 바꿉니다. 숫자와 단위를 분리할 수 없습니다. 공백은 허용되지 않습니다.

2. AWS 콘솔에 로그인하여 새 머신 풀 디스크 볼륨 크기를 확인하고 EC2 가상 머신 루트 볼륨 크기를 찾습니다.

추가 리소스

- **rosa create machinepool** 하위 명령에 사용할 수 있는 인수의 자세한 목록은ROSA CLI를 사용하여 오브젝트 관리를 참조하십시오.

4.2.3. 머신 풀 삭제

워크로드 요구 사항이 변경되고 현재 머신 풀이 더 이상 요구 사항을 충족하지 않는 경우 머신 풀을 삭제할 수 있습니다.

Red Hat OpenShift Cluster Manager 또는 ROSA CLI(**rosa**)를 사용하여 머신 풀을 삭제할 수 있습니다.

4.2.3.1. OpenShift Cluster Manager를 사용하여 머신 풀 삭제

Red Hat OpenShift Cluster Manager를 사용하여 AWS 클러스터에서 Red Hat OpenShift Service의 머신 풀을 삭제할 수 있습니다.

사전 요구 사항

- ROSA 클러스터를 생성하셨습니다.
- 클러스터가 ready 상태입니다.
- 테인트 없이 기존 머신 풀이 있고 단일 AZ 클러스터용 인스턴스가 두 개 이상 있거나 다중 AZ 클러스터의 경우 세 개의 인스턴스가 있습니다.

절차

1. OpenShift Cluster Manager 에서 Cluster List 페이지로 이동하여 삭제할 시스템 풀이 포함된 클러스터를 선택합니다.
2. 선택한 클러스터에서 머신 풀 탭을 선택합니다.
3. 머신 풀 탭에서 삭제하려는 머신 풀의 옵션 메뉴  를 클릭합니다.
4. 삭제를 클릭합니다.
선택한 머신 풀이 삭제됩니다.

4.2.3.2. ROSA CLI를 사용하여 머신 풀 삭제

ROSA CLI를 사용하여 AWS 클러스터에서 Red Hat OpenShift Service의 머신 풀을 삭제할 수 있습니다.

참고

ROSA CLI **rosa** 버전 1.2.25 및 이전 버전의 사용자는 클러스터와 함께 생성된 머신 풀(ID='Default')을 삭제할 수 없습니다. ROSA CLI 버전 1.2.26 이상 사용자의 경우 클러스터와 함께 생성되는 머신 풀(ID='worker')은 테인트가 없는 클러스터 내에 하나의 머신 풀이 있고 단일 AZ 클러스터의 경우 두 개 이상의 복제본 또는 Multi-AZ 클러스터에 대한 세 개의 복제본을 삭제할 수 있습니다.

사전 요구 사항

- ROSA 클러스터를 생성하셨습니다.
- 클러스터가 ready 상태입니다.
- 테인트 없이 기존 머신 풀이 있고 Single-AZ 클러스터용 인스턴스가 두 개 이상 있거나 Multi-AZ 클러스터의 경우 세 개의 인스턴스가 있습니다.

절차

1. ROSA CLI에서 다음 명령을 실행합니다.

```
$ rosa delete machinepool -c=<cluster_name> <machine_pool_ID>
```

출력 예

```
? Are you sure you want to delete machine pool <machine_pool_ID> on cluster
<cluster_name>? (y/N)
```

2. y 를 입력하여 시스템 풀을 삭제합니다.

선택한 머신 풀이 삭제됩니다.

4.2.4. 수동으로 컴퓨팅 노드 확장

머신 풀에 자동 스케일링을 활성화하지 않은 경우 배포 요구에 맞게 풀의 컴퓨팅 (작업자라고도 함) 노드 수를 수동으로 스케일링할 수 있습니다.

각 머신 풀을 별도로 스케일링해야 합니다.

사전 요구 사항

- 워크스테이션에 최신 **Red Hat OpenShift Service on AWS(ROSA) CLI** 를 설치하고 구성하셨습니다.
- **ROSA CLI(rosa)**를 사용하여 **Red Hat** 계정에 로그인했습니다.
- **AWS(ROSA)** 클러스터에서 **Red Hat OpenShift Service**를 생성하셨습니다.
- 기존 머신 풀이 있습니다.

절차

1. 클러스터의 머신 풀을 나열합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

출력 예

```
ID       AUTOSCALING   REPLICAS   INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    DISK SIZE   SG IDs
default  No            2          m5.xlarge                us-east-1a          300GiB     sg-
0e375ff0ec4a6cfa2
mp1      No            2          m5.xlarge                us-east-1a          300GiB     sg-
0e375ff0ec4a6cfa2
```

2. 머신 풀에서 컴퓨팅 노드 복제본 수를 늘리거나 줄입니다.

```
$ rosa edit machinepool --cluster=<cluster_name> \
                  --replicas=<replica_count> \ 1
                  <machine_pool_id> 2
```

1. 단일 가용성 영역을 사용하여 **AWS(ROSA)(classic architecture)**에 **Red Hat OpenShift Service**를 배포한 경우 복제본 수는 영역의 시스템 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 **count**는 모든 영역에 걸쳐 시스템 풀의 총 컴퓨팅 노드 수를 정의하고 **3**개 중 여러 개여야 합니다.

2

& lt;machine_pool_id >를 이전 명령의 출력에 나열된 시스템 풀의 ID로 바꿉니다.

검증

1. 클러스터에서 사용 가능한 머신 풀을 나열합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

출력 예

```
ID       AUTOSCALING   REPLICAS   INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    DISK SIZE   SG IDs
default   No          2          m5.xlarge                  us-east-1a          300GiB      sg-
0e375ff0ec4a6cfa2
mp1       No          3          m5.xlarge                  us-east-1a          300GiB      sg-
0e375ff0ec4a6cfa2
```

2. 이전 명령의 출력에서 계산 노드 복제본 수가 시스템 풀에 대해 예상대로 표시되는지 확인합니다. 예제 출력에서 iPXE 1 머신 풀의 컴퓨팅 노드 복제본 수가 3으로 확장됩니다.

4.2.5. 노드 라벨

레이블은 Node 오브젝트에 적용되는 키-값 쌍입니다. 라벨을 사용하여 오브젝트 세트를 구성하고 Pod 예약을 제어할 수 있습니다.

클러스터 생성 중 또는 이후에 라벨을 추가할 수 있습니다. 레이블은 언제든지 수정하거나 업데이트할 수 있습니다.

추가 리소스

- 라벨에 대한 자세한 내용은 Kubernetes 라벨 및 선택자 개요를 참조하십시오.

4.2.5.1. 머신 풀에 노드 라벨 추가

언제든지 **compute**(작업자라고도 함) 노드의 라벨을 추가하거나 편집하여 사용자와 관련된 방식으로 노드를 관리합니다. 예를 들어 특정 노드에 워크로드 유형을 할당할 수 있습니다.

레이블은 키-값 쌍으로 할당됩니다. 각 키는 할당된 오브젝트에 고유해야 합니다.

사전 요구 사항

- 워크스테이션에 최신 **Red Hat OpenShift Service on AWS(ROSA) CLI** 를 설치하고 구성하셨습니다.
- **ROSA CLI(rosa)**를 사용하여 **Red Hat** 계정에 로그인했습니다.
- **AWS(ROSA)** 클러스터에서 **Red Hat OpenShift Service**를 생성하셨습니다.
- 기존 머신 풀이 있습니다.

절차

1. 클러스터의 머신 풀을 나열합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

출력 예

```
ID          AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SPOT INSTANCES
Default     No           2         m5.xlarge                  us-east-1a          N/A
db-nodes-mp  No          2         m5.xlarge                  us-east-1a          No
```

2. 머신 풀의 노드 레이블을 추가하거나 업데이트합니다.

- 자동 스케일링을 사용하지 않는 머신 풀의 노드 레이블을 추가하거나 업데이트하려면 다음 명령을 실행합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> \
                --replicas=<replica_count> \ 1
                --labels=<key>=<value>,<key>=<value> \ 2
                <machine_pool_id>
```

1

자동 스케일링을 사용하지 않는 머신 풀의 경우 노드 라벨을 추가할 때 복제본 수를 제공해야 합니다. **--replicas** 인수를 지정하지 않으면 명령이 완료되기 전에 복제본 수를 입력하라는 메시지가 표시됩니다. 단일 가용성 영역을 사용하여 **AWS(ROSA)**에 **Red Hat OpenShift Service**를 배포한 경우 복제본 수는 영역의 시스템 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 **count**는 모든 영역에 걸쳐 시스템 풀의 총 컴퓨팅 노드 수를 정의하고 **3**개 중 여러 개여야 합니다.

2

< key>=<value>,<key>=<value >를 쉼표로 구분된 키-값 쌍 목록(예: **--labels=key1=value1,key2=value 2**)로 바꿉니다. 이 목록은 노드 레이블에 대한 변경 사항을 지속적으로 덮어씁니다.

다음 예제에서는 **db-nodes-mp** 머신 풀에 라벨을 추가합니다.

```
$ rosa edit machinepool --cluster=mycluster --replicas=2 --
labels=app=db,tier=backend db-nodes-mp
```

출력 예

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

- 자동 스케일링을 사용하는 머신 풀의 노드 레이블을 추가하거나 업데이트하려면 다음 명령을 실행합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> \
                --min-replicas=<minimum_replica_count> \ 1
                --max-replicas=<maximum_replica_count> \ 2
```

```
--labels=<key>=<value>,<key>=<value> \ 3
<machine_pool_id>
```

1 2

자동 스케일링을 사용하는 머신 풀의 경우 최소 및 최대 컴퓨팅 노드 복제본 제한을 제공해야 합니다. 인수를 지정하지 않으면 명령이 완료되기 전에 값을 입력하라는 메시지가 표시됩니다. 클러스터 자동 스케일러는 지정한 제한을 초과하여 머신 풀 노드 수를 줄이거나 늘리지 않습니다. 단일 가용성 영역을 사용하여 **ROSA**를 배포한 경우 **--min-replicas** 및 **--max-replicas** 인수는 영역의 시스템 풀에 자동 스케일링 제한을 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 인수는 모든 영역에서 총 자동 스케일링 제한을 정의하고 개수는 **3**개여야 합니다.

3

< key>=<value>,<key>=<value >를 쉼표로 구분된 키-값 쌍 목록(예: **--labels=key1=value1,key2=value 2**)로 바꿉니다. 이 목록은 노드 레이블에 대한 변경 사항을 지속적으로 덮어씁니다.

다음 예제에서는 **db-nodes-mp** 머신 풀에 라벨을 추가합니다.

```
$ rosa edit machinepool --cluster=mycluster --min-replicas=2 --max-replicas=3 --
labels=app=db,tier=backend db-nodes-mp
```

출력 예

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

검증

1. 새 라벨을 사용하여 머신 풀의 세부 정보를 설명합니다.

```
$ rosa describe machinepool --cluster=<cluster_name> --machinepool=<machine-
pool-name>
```

출력 예

```
ID:                 db-nodes-mp
```

```
Cluster ID:              <ID_of_cluster>
Autoscaling:             No
Replicas:                2
Instance type:           m5.xlarge
Labels:                  app=db, tier=backend
Taints:
Availability zones:      us-east-1a
Subnets:
Spot instances:          No
Disk size:               300 GiB
Security Group IDs:
```

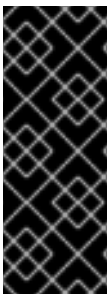
2. 출력의 시스템 풀에 레이블이 포함되어 있는지 확인합니다.

4.2.6. 머신 풀에 태그 추가

머신 풀에 컴퓨팅 노드(작업자 노드라고도 함)에 대한 태그를 추가하여 시스템 풀을 프로비저닝할 때 생성되는 AWS 리소스에 대한 사용자 지정 사용자 태그를 도입할 수 있습니다.

4.2.6.1. ROSA CLI를 사용하여 머신 풀에 태그 추가

ROSA CLI(명령줄 인터페이스)를 사용하여 AWS 클러스터에서 Red Hat OpenShift Service의 머신 풀에 태그를 추가할 수 있습니다.



중요

태그 키가 aws,red-hat-managed,red-hat-clustertype 또는 Name 이 아닌지 확인해야 합니다. 또한 kubernetes.io/cluster/ 로 시작하는 태그 키를 설정하지 않아야 합니다. 태그의 키는 128자를 초과할 수 없으며 태그의 값은 256자를 초과할 수 없습니다. Red Hat은 향후 예약된 태그를 추가할 수 있는 권한을 갖습니다.

사전 요구 사항

- 워크스테이션에 최신 AWS(aws), ROSA(rosa), OpenShift(oc) CLI를 설치하고 구성하셨습니다.
- rosa CLI를 사용하여 Red Hat 계정에 로그인했습니다.

- AWS(ROSA) 클러스터에서 Red Hat OpenShift Service를 생성하셨습니다.

절차

- 다음 명령을 실행하여 사용자 지정 태그로 머신 풀을 생성합니다.

```
$ rosa create machinepools --cluster=<name> --replicas=<replica_count> \
  --name <mp_name> --tags='<key> <value>,<key> <value>' 1
```

1 < key> <value>,<key> <value >를 각 태그의 키와 값으로 바꿉니다.

출력 예

```
$ rosa create machinepools --cluster=mycluster --replicas 2 --tags='tagkey1
tagvalue1,tagkey2 tagvaluev2'

I: Checking available instance types for machine pool 'mp-1'
I: Machine pool 'mp-1' created successfully on cluster 'mycluster'
I: To view the machine pool details, run 'rosa describe machinepool --cluster
mycluster --machinepool mp-1'
I: To view all machine pools, run 'rosa list machinepools --cluster mycluster'
```

검증

- describe 명령을 사용하여 태그가 있는 머신 풀의 세부 정보를 확인하고 출력에 시스템 풀에 태그가 포함되어 있는지 확인합니다.

```
$ rosa describe machinepool --cluster=<cluster_name> --machinepool=
<machinepool_name>
```

출력 예

```
ID:                         mp-1
Cluster ID:                 2baiirqa2141oreotoivp4sipq84vp5g
Autoscaling:                No
```

```
Replicas:                       2
Instance type:                  m5.xlarge
Labels:
Taints:
Availability zones:             us-east-1a
Subnets:
Spot instances:                 No
Disk size:                      300 GiB
Additional Security Group IDs:
Tags:                           red-hat-clustertype=rosa, red-hat-managed=true,
tagkey1=tagvalue1, tagkey2=tagvaluev2
```

4.2.7. 머신 풀에 테인트 추가

머신 풀에서 **compute**(작업자라고도 함) 노드의 테인트를 추가하여 예약된 **Pod**를 제어할 수 있습니다. 머신 풀에 테인트를 적용하면 **Pod** 사양에 테인트에 대한 허용 오차가 포함되지 않는 한 스케줄러는 풀의 노드에 **Pod**를 배치할 수 없습니다. **Red Hat OpenShift Cluster Manager** 또는 **AWS(ROSA) CLI(rosa) CLI**에서 **Red Hat OpenShift Service**를 사용하여 머신 풀에 테인트를 추가할 수 있습니다.

참고

클러스터에 테인트가 포함되지 않은 하나 이상의 머신 풀이 있어야 합니다.

4.2.7.1. OpenShift Cluster Manager를 사용하여 머신 풀에 테인트 추가

Red Hat OpenShift Cluster Manager를 사용하여 **AWS** 클러스터에서 **Red Hat OpenShift Service**의 머신 풀에 테인트를 추가할 수 있습니다.

사전 요구 사항

- **AWS** 클러스터에서 **Red Hat OpenShift Service**를 생성하셨습니다.
- 기존 머신 풀에는 테인트가 포함되지 않고 두 개 이상의 인스턴스가 포함됩니다.

절차

1. OpenShift Cluster Manager 로 이동하여 클러스터를 선택합니다.

2. 머신 풀 탭에서 테인트를 추가할 머신 풀의 옵션 메뉴  를 클릭합니다.

3. 테인트 편집을 선택합니다.

4. 테인트에 대한 키 및 값 항목을 추가합니다.

5. 드롭다운 메뉴에서 테인트의 영향을 선택합니다. 사용 가능한 옵션에는 **NoSchedule,PreferNoSchedule** 및 **NoExecute** 가 포함됩니다.

6. 선택 사항: 머신 풀에 테인트 를 추가하려면 테인트 추가를 선택합니다.

7. 저장을 클릭하여 머신 풀에 테인트를 적용합니다.

검증

1. 머신 풀 탭에서 머신 풀 옆에 있는 >을 선택하여 뷰를 확장합니다.

2. 확장된 보기의 **Taints** 아래에 테인트가 나열되어 있는지 확인합니다.

4.2.7.2. ROSA CLI를 사용하여 머신 풀에 테인트 추가

ROSA CLI를 사용하여 **AWS** 클러스터에서 **Red Hat OpenShift Service**의 머신 풀에 테인트를 추가할 수 있습니다.

참고

ROSA CLI rosa 버전 **1.2.25** 및 이전 버전의 사용자의 경우 클러스터와 함께 생성된 머신 풀**(ID=Default)** 내에서 테인트 수를 변경할 수 없습니다. **ROSA CLI** 버전 **1.2.26** 이상 사용자의 경우 클러스터와 함께 생성된 머신 풀**(ID=worker)** 내에서 테인트 수를 변경할 수 있습니다. 테인트 없이 하나 이상의 머신 풀이 있어야 하며 **Single-AZ** 클러스터당 두 개 이상의 복제본 또는 **Multi-AZ** 클러스터의 복제본 **3**개가 있어야 합니다.

사전 요구 사항

- 워크스테이션에 최신 AWS(aws), ROSA(rosa), OpenShift(oc) CLI를 설치하고 구성하셨습니다.
- rosa CLI를 사용하여 Red Hat 계정에 로그인했습니다.
- AWS(ROSA) 클러스터에서 Red Hat OpenShift Service를 생성하셨습니다.
- 기존 머신 풀에는 테인트가 포함되지 않고 두 개 이상의 인스턴스가 포함됩니다.

절차

1. 다음 명령을 실행하여 클러스터의 머신 풀을 나열합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

2. 머신 풀의 테인트를 추가하거나 업데이트합니다.

- 자동 스케일링을 사용하지 않는 머신 풀의 테인트를 추가하거나 업데이트하려면 다음 명령을 실행합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> \
                --replicas=<replica_count> \ 1
                --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 2
                <machine_pool_id>
```

1. 자동 스케일링을 사용하지 않는 머신 풀의 경우 테인트를 추가할 때 복제본 수를 제공해야 합니다. --replicas 인수를 지정하지 않으면 명령이 완료되기 전에 복제본 수를 입력하라는 메시지가 표시됩니다. 단일 가용성 영역을 사용하여 AWS(ROSA)에 Red Hat OpenShift Service를 배포한 경우 복제본 수는 영역의 시스템 풀에 프로비저닝할 컴퓨팅 노드 수를 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 count는 모든 영역에 걸쳐 시스템 풀의 총 컴퓨팅 노드 수를 정의하고 3개 중 여러 개여야 합니다.

2. < key>=<value>:<effect>,<key>=<value>:<effect >:<effect>를 각 테인트의 키, 값 및 효과(예: --

taints=key1=value1:NoSchedule,key2=value2:NoSchedule,key2=value2:NoExecute)로 바꿉니다. 사용 가능한 효과에는 NoSchedule,PreferNoSchedule 및 NoExecute 가 포함됩니다. 이 목록은 노드 테인트에 대한 수정 사항을 지속적으로 덮어씁니다.

다음 예제에서는 db-nodes-mp 머신 풀에 테인트를 추가합니다.

```
$ rosa edit machinepool --cluster=mycluster --replicas 2 --
taints=key1=value1:NoSchedule,key2=value2:NoExecute db-nodes-mp
```

출력 예

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

- 자동 스케일링을 사용하는 머신 풀의 테인트를 추가하거나 업데이트하려면 다음 명령을 실행합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> \
                --min-replicas=<minimum_replica_count> \ 1
                --max-replicas=<maximum_replica_count> \ 2
                --taints=<key>=<value>:<effect>,<key>=<value>:<effect> \ 3
                <machine_pool_id>
```

1 2

자동 스케일링을 사용하는 머신 풀의 경우 최소 및 최대 컴퓨팅 노드 복제본 제한을 제공해야 합니다. 인수를 지정하지 않으면 명령이 완료되기 전에 값을 입력하라는 메시지가 표시됩니다. 클러스터 자동 스케일러는 지정한 제한을 초과하여 머신 풀 노드 수를 줄이거나 늘리지 않습니다. 단일 가용성 영역을 사용하여 ROSA를 배포한 경우 --min-replicas 및 --max-replicas 인수는 영역의 시스템 풀에 자동 스케일링 제한을 정의합니다. 여러 가용성 영역을 사용하여 클러스터를 배포하는 경우 인수는 모든 영역에서 총 자동 스케일링 제한을 정의하고 개수는 3개여야 합니다.

3

< key>=<value>:<effect>,<key>=<value>:<effect >:<effect>를 각 테인트의 키, 값 및 효과(예: --taints=key1=value1:NoSchedule,key2=value2:NoSchedule,key2=value2:NoExecute)로 바꿉니다. 사용 가능한 효과에는 NoSchedule,PreferNoSchedule 및 NoExecute 가 포함됩니다. 이 목록은 노드 테인트에 대한 모든 수정 사항을 지속적으로 덮어씁니다.

다음 예제에서는 **db-nodes-mp** 머신 풀에 테인트를 추가합니다.

```
$ rosa edit machinepool --cluster=mycluster --min-replicas=2 --max-replicas=3 --
taints=key1=value1:NoSchedule,key2=value2:NoExecute db-nodes-mp
```

출력 예

```
I: Updated machine pool 'db-nodes-mp' on cluster 'mycluster'
```

검증

1. 새 테인트를 사용하여 머신 풀의 세부 정보를 설명합니다.

```
$ rosa describe machinepool --cluster=<cluster_name> --machinepool=
<machinepool_name>
```

출력 예

```
ID:                  db-nodes-mp
Cluster ID:          <ID_of_cluster>
Autoscaling:         No
Replicas:            2
Instance type:       m5.xlarge
Labels:
Taints:              key1=value1:NoSchedule, key2=value2:NoExecute
Availability zones:  us-east-1a
Subnets:
Spot instances:      No
Disk size:           300 GiB
Security Group IDs:
```

2. 출력에서 머신 풀에 테인트가 포함되어 있는지 확인합니다.

4.2.8. 추가 리소스

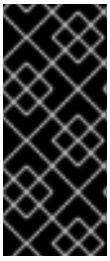
- 머신 풀 정보
- 자동 스케일링 정보
- 자동 스케일링 활성화
- 자동 스케일링 비활성화
- ROSA(클래식 아키텍처) 서비스 정의

4.3. 로컬 영역에서 머신 풀 구성

이 문서에서는 ROSA(Red Hat OpenShift Service on AWS)를 사용하여 머신 풀에서 로컬 영역을 구성하는 방법을 설명합니다.

4.3.1. 로컬 영역에서 머신 풀 구성

다음 단계를 사용하여 로컬 영역에서 시스템 풀을 구성합니다.



중요

AWS 로컬 영역은 AWS 4.12의 Red Hat OpenShift Service에서 지원됩니다. 로컬 영역을 활성화하는 방법에 대한 자세한 내용은 Red Hat Knowledgebase 문서를 참조하십시오.

사전 요구 사항

- Red Hat OpenShift Service on AWS (ROSA)는 일반적으로 선택한 상위 리전에서 사용할 수 있습니다. 특정 AWS 리전에서 사용할 수 있는 로컬 영역을 확인하려면 AWS 일반적으로 사용 가능한 위치 목록을 참조하십시오.
- ROSA 클러스터는 처음에 기존 Amazon VPC(BYO-VPC)로 구축되었습니다.

- ROSA 클러스터의 최대 전송 단위(MTU)는 1200으로 설정됩니다.

중요

일반적으로 로컬 영역의 Amazon EC2 인스턴스와 리전의 Amazon EC2 인스턴스 간 최대 전송 단위(MTU)는 1300입니다. AWS 문서 의 로컬 영역 작동 방식을 참조하십시오. 오버헤드를 설명하려면 클러스터 네트워크 MTU가 항상 EC2 MTU보다 작아야 합니다. 특정 오버헤드는 네트워크 플러그인에 의해 결정됩니다. 예를 들면 다음과 같습니다.

  - OVN-Kubernetes: 100바이트
  - OpenShift SDN: 50바이트

네트워크 플러그인은 MTU를 줄일 수 있는 추가 기능을 제공할 수 있습니다. 자세한 내용은 설명서를 확인하십시오.

- AWS 계정에는 로컬 영역이 활성화되어 있습니다.
- AWS 계정에는 클러스터와 동일한 VPC에 대한 로컬 영역 서브넷 이 있습니다.
- AWS 계정에는 NAT 게이트웨이로의 경로가 있는 라우팅 테이블과 연결된 서브넷이 있습니다.
- AWS 계정에는 연결된 서브넷에 'kubernetes.io/cluster/<infra_id>: shared' 태그가 있습니다.

절차

1. 다음 ROSA CLI(rosa) 명령을 실행하여 클러스터에 머신 풀을 만듭니다.

```
$ rosa create machinepool -c <cluster-name> -i
```

2. ROSA CLI에서 머신 풀의 서브넷 및 인스턴스 유형을 추가합니다. 몇 분 후에 클러스터에서

노드를 프로비저닝합니다.

```
I: Enabling interactive mode 1
? Machine pool name: xx-lz-xx 2
? Create multi-AZ machine pool: No 3
? Select subnet for a single AZ machine pool (optional): Yes 4
? Subnet ID: subnet-<a> (region-info) 5
? Enable autoscaling (optional): No 6
? Replicas: 2 7
I: Fetching instance types 8
? disk-size (optional): 9
```

1

대화형 모드를 활성화합니다.

2

시스템 풀의 이름을 지정합니다. 영숫자 및 최대 30자로 제한됩니다.

3

이 옵션을 no로 설정합니다.

4

이 옵션을 yes로 설정합니다.

5

목록에서 서브넷 ID를 선택합니다.

6

자동 스케일링을 활성화하려면 yes를 선택합니다.

7

머신 풀의 머신 수를 선택합니다. 이 번호는 1에서 180까지 어느 곳에나 있을 수 있습니다.

8

목록에서 인스턴스 유형을 선택합니다. 선택한 로컬 영역에서 지원되는 인스턴스 유형만 표시됩니다.

9

선택 사항: 작업자 노드 디스크 크기를 지정합니다. 값은 GB, GiB, TB 또는 TiB일 수 있습니다. 숫자 값과 단위를 설정합니다(예: '200GiB'). 숫자와 단위를 분리할 수 없습니다. 공백은 허용되지 않습니다.

3. 로컬 영역에서 머신 풀을 프로비저닝할 서브넷 ID를 제공합니다.

일반적으로 사용 가능한 AWS 로컬 영역 위치 목록은 AWS 로컬 영역의 AWS 로컬 영역 목록을 참조하십시오.

4.4. 클러스터의 노드 자동 스케일링 정보

자동 스케일러 옵션은 클러스터의 머신 수를 자동으로 확장하도록 구성할 수 있습니다.

리소스가 부족하여 현재 노드에서 pod를 예약할 수 없거나 배포 요구를 충족시키기 위해 다른 노드가 필요한 경우 클러스터 자동 스케일러는 클러스터 크기를 늘립니다. 클러스터 자동 스케일러는 사용자가 지정한 제한을 초과하여 클러스터 리소스를 늘리지 않습니다.

또한 클러스터 자동 스케일러는 리소스 사용이 적고 중요한 pod가 모두 다른 노드에 적합한 경우와 같이 상당한 기간 동안 일부 노드가 지속적으로 필요하지 않은 경우 클러스터 크기를 줄입니다.

자동 스케일링을 활성화하는 경우 최소 및 최대 작업자 노드 수를 설정해야 합니다.

참고

클러스터 소유자 및 조직 관리자만 클러스터를 확장하거나 삭제할 수 있습니다.

4.4.1. 클러스터에서 노드 자동 스케일링 활성화

작업자 노드에서 자동 스케일링을 활성화하여 기존 클러스터에 대한 머신 풀 정의를 편집하여 사용 가능한 노드 수를 늘리거나 줄일 수 있습니다.

Red Hat OpenShift Cluster Manager를 사용하여 기존 클러스터에서 노드 자동 스케일링 활성화

OpenShift Cluster Manager 콘솔에서 머신 풀 정의에서 작업자 노드에 대한 자동 스케일링을 활성화합니다.

절차

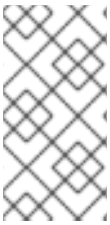
1. OpenShift Cluster Manager 에서 클러스터 목록 페이지로 이동하여 자동 스케일링을 활성화할 클러스터를 선택합니다.

2. 선택한 클러스터에서 머신 풀 탭을 선택합니다.

3. 자동 스케일링을 활성화하려는 머신 풀 끝에 있는 옵션 메뉴  를 클릭하고 편집을 선택합니다.

4. Edit machine pool 대화 상자에서 자동 스케일링 활성화 확인란을 선택합니다.

5. 저장을 선택하여 이러한 변경 사항을 저장하고 머신 풀에 대한 자동 스케일링을 활성화합니다.



참고

또한 대화형 모드를 사용하여 클러스터를 생성할 때 기본 머신 풀에서 자동 스케일링을 구성할 수 있습니다.

ROSA CLI를 사용하여 기존 클러스터에서 노드 자동 스케일링 활성화

로드를 기반으로 작업자 노드 수를 동적으로 확장 또는 축소하도록 자동 스케일링을 구성합니다.

자동 스케일링은 **AWS** 계정에 올바른 **AWS** 리소스 할당량을 보유하는 데 따라 달라집니다. AWS 콘솔에서 리소스 할당량 및 요청 할당량 증가를 확인합니다.

절차

1. 클러스터에서 머신 풀 **ID**를 식별하려면 다음 명령을 입력합니다.

```
$ rosa list machinepools --cluster=<cluster_name>
```

출력 예

```
ID      AUTOSCALING  REPLICAS  INSTANCE TYPE  LABELS    TAINTS
AVAILABILITY ZONES    SUBNETS    SPOT INSTANCES  DISK SIZE  SG IDs
worker  No           2         m5.xlarge                  us-east-2a                  No           300
GiB
mp1     No           2         m5.xlarge                  us-east-2a                  No           300
GiB
```

2. 구성할 머신 풀의 ID를 가져옵니다.

3. 머신 풀에서 자동 스케일링을 활성화하려면 다음 명령을 입력합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> <machinepool_ID> --enable-
autoscaling --min-replicas=<number> --max-replicas=<number>
```

예제

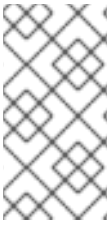
2에서 5개의 작업자 노드 사이를 스케일링하도록 복제본 수가 설정된 mycluster 클러스터에서 ID iPXE 1 이 있는 머신 풀에서 자동 스케일링을 활성화합니다.

```
$ rosa edit machinepool --cluster=mycluster mp1 --enable-autoscaling --min-
replicas=2 --max-replicas=5
```

4.4.2. 클러스터에서 자동 스케일링 노드 비활성화

작업자 노드에서 자동 스케일링을 비활성화하여 기존 클러스터에 대한 머신 풀 정의를 편집하여 사용 가능한 노드 수를 늘리거나 줄일 수 있습니다.

Red Hat OpenShift Cluster Manager 또는 AWS CLI에서 Red Hat OpenShift Service를 사용하여 클러스터에서 자동 스케일링을 비활성화할 수 있습니다.



참고

또한 대화형 모드를 사용하여 클러스터를 생성할 때 기본 머신 풀에서 자동 스케일링을 구성할 수 있습니다.

Red Hat OpenShift Cluster Manager를 사용하여 기존 클러스터에서 자동 스케일링 노드 비활성화

OpenShift Cluster Manager의 시스템 풀 정의에서 작업자 노드의 자동 스케일링을 비활성화합니다.

절차

1. OpenShift Cluster Manager 에서 **Cluster List** 페이지로 이동하여 비활성화해야 하는 자동 스케일링이 있는 클러스터를 선택합니다.
2. 선택한 클러스터에서 머신 풀 탭을 선택합니다.
3. 자동 스케일링을 사용하여 머신 풀 끝에 있는 옵션 메뉴

를 클릭하고 편집을 선택합니다.

4. **Edit machine pool** (시스템 풀 편집) 대화 상자에서 자동 스케일링 활성화 확인란을 선택 취소합니다.
5. 저장을 선택하여 이러한 변경 사항을 저장하고 시스템 풀에서 자동 스케일링을 비활성화합니다.

ROSA CLI를 사용하여 기존 클러스터에서 자동 스케일링 노드 비활성화

ROSA(Red Hat OpenShift Service on AWS) CLI, rosa 를 사용하여 시스템 풀 정의에서 작업자 노드의 자동 스케일링을 비활성화합니다.

절차

- 다음 명령을 실행합니다.

```
$ rosa edit machinepool --cluster=<cluster_name> <machinepool_ID> --enable-
autoscaling=false --replicas=<number>
```

예제

mycluster 라는 클러스터에서 기본 머신 풀에서 자동 스케일링을 비활성화합니다.

```
$ rosa edit machinepool --cluster=mycluster default --enable-autoscaling=false --
replicas=3
```

4.4.3. 추가 리소스

- 문제 해결: 자동 확장은 노드 축소되지 않음
- machinepools 정보
- 컴퓨팅 노드 관리
- ROSA CLI로 오브젝트 관리

4.5. 컨테이너 메모리 및 위험 요구 사항을 충족하도록 클러스터 메모리 구성

클러스터 관리자는 다음과 같은 방법으로 애플리케이션 메모리를 관리하여 클러스터를 효율적으로 작동할 수 있습니다.

- 컨테이너화된 애플리케이션 구성 요소의 메모리 및 위험 요구 사항을 확인하고 해당 요구 사항에 맞게 컨테이너 메모리 매개변수를 구성합니다.
- 구성된 컨테이너 메모리 매개변수를 최적으로 준수하도록 컨테이너화된 애플리케이션 런타임(예: OpenJDK)을 구성합니다.
- 컨테이너에서 실행과 연결된 메모리 관련 오류 조건을 진단 및 해결합니다.

4.5.1. 애플리케이션 메모리 관리 이해

계속하기 전에 AWS의 Red Hat OpenShift Service가 컴퓨팅 리소스를 관리하는 방법에 대한 개요를 완전히 확인하는 것이 좋습니다.

각 종류의 리소스(메모리, CPU, 스토리지)에 대해 AWS의 Red Hat OpenShift Service를 사용하면 선택적 요청 및 제한 값을 Pod의 각 컨테이너에 배치할 수 있습니다.

메모리 요청 및 메모리 제한에 대해 다음 사항에 유의하십시오.

- 메모리 요청
  - 메모리 요청 값을 지정하면 AWS 스케줄러에서 Red Hat OpenShift Service에 영향을 미칩니다. 스케줄러는 노드에 컨테이너를 예약할 때 메모리 요청을 고려한 다음 컨테이너 사용을 위해 선택한 노드에서 요청된 메모리를 차단합니다.
  - 노드의 메모리가 소진되면 AWS의 Red Hat OpenShift Service는 메모리 사용량이 메모리 요청을 가장 많이 초과하는 컨테이너를 제거하는 우선 순위를 지정합니다. 메모리 소모가 심각한 경우 노드 OOM 종료자는 유사한 메트릭을 기반으로 컨테이너에서 프로세스를 선택하고 종료할 수 있습니다.
  - 클러스터 관리자는 메모리 요청 값에 할당량을 할당하거나 기본값을 할당할 수 있습니다.
  - 클러스터 관리자는 클러스터 과다 할당을 관리하기 위해 개발자가 지정하는 메모리 요청 값을 덮어쓸 수 있습니다.
- 메모리 제한
  - 메모리 제한 값을 지정하면 컨테이너의 모든 프로세스에 할당될 수 있는 메모리에 대한 하드 제한을 제공합니다.
  - 컨테이너의 모든 프로세스에서 할당한 메모리가 메모리 제한을 초과하면 노드의 OOM(Out of Memory) 종료자에서 즉시 컨테이너의 프로세스를 선택하여 종료합니다.
  - 메모리 요청 및 제한을 둘 다 지정하면 메모리 제한 값이 메모리 요청보다 크거나 같아야 합니다.
  - 클러스터 관리자는 메모리 제한 값에 할당량을 할당하거나 기본값을 할당할 수 있습니

다.

- 최소 메모리 제한은 **12MB**입니다. 메모리를 할당할 수 없음 **Pod** 이벤트로 인해 컨테이너가 시작되지 않으면 메모리 제한이 너무 낮은 것입니다. 메모리 제한을 늘리거나 제거합니다. 제한을 제거하면 **Pod**에서 바인딩되지 않은 노드 리소스를 사용할 수 있습니다.

4.5.1.1. 애플리케이션 메모리 전략 관리

AWS의 **Red Hat OpenShift Service**에서 애플리케이션 메모리 크기를 조정하는 단계는 다음과 같습니다.

1. 예상되는 컨테이너 메모리 사용량 확인

   필요한 경우 경험적으로 예상되는 평균 및 최대 컨테이너 메모리 사용량을 결정합니다(예: 별도의 부하 테스트를 통해). 컨테이너에서 잠재적으로 병렬로 실행될 수 있는 모든 프로세스를 고려해야 합니다(예: 기본 애플리케이션에서 보조 스크립트를 생성하는지의 여부).

2. 위험 유형 확인

   제거와 관련된 위험 유형을 확인합니다. 위험 성향이 낮으면 컨테이너는 예상되는 최대 사용량과 백분율로 된 안전 범위에 따라 메모리를 요청해야 합니다. 위험 성향이 높으면 예상되는 사용량에 따라 메모리를 요청하는 것이 더 적합할 수 있습니다.

3. 컨테이너 메모리 요청 설정

   위 내용에 따라 컨테이너 메모리 요청을 설정합니다. 요청이 애플리케이션 메모리 사용량을 더 정확하게 나타낼수록 좋습니다. 요청이 너무 높으면 클러스터 및 할당량 사용이 비효율적입니다. 요청이 너무 낮으면 애플리케이션 제거 가능성이 커집니다.

4. 필요한 경우 컨테이너 메모리 제한 설정

   필요한 경우 컨테이너 메모리 제한을 설정합니다. 제한을 설정하면 컨테이너에 있는 모든 프로세스의 메모리 사용량 합계가 제한을 초과하는 경우 컨테이너 프로세스가 즉시 종료되는 효과가 있어 이로 인한 장단점이 발생합니다. 다른 한편으로는 예상치 못한 과도한 메모리 사용을 조기에 확인할 수 있습니다("빠른 실패"). 그러나 이로 인해 프로세스가 갑자기 종료됩니다.

AWS 클러스터의 일부 **Red Hat OpenShift Service**는 제한 값을 설정해야 할 수 있습니다. 일부는 제한을 기반으로 요청을 덮어쓸 수 있습니다. 일부 애플리케이션 이미지는 요청 값보다 탐

지하기 쉽게 설정되는 제한 값을 사용합니다.

메모리 제한을 설정하는 경우 예상되는 최대 컨테이너 메모리 사용량과 백분율로 된 안전 범위 이상으로 설정해야 합니다.

5. 애플리케이션이 튜닝되었는지 확인

적절한 경우 구성된 요청 및 제한 값과 관련하여 애플리케이션이 튜닝되었는지 확인합니다. 이 단계는 특히 **JVM**과 같이 메모리를 풀링하는 애플리케이션과 관련이 있습니다. 이 페이지의 나머지 부분에서는 이 작업에 대해 설명합니다.

4.5.2. AWS에서 Red Hat OpenShift Service에 대한 OpenJDK 설정 이해

기본 **OpenJDK** 설정은 컨테이너화된 환경에서 제대로 작동하지 않습니다. 따라서 컨테이너에서 **OpenJDK**를 실행할 때마다 몇 가지 추가 **Java** 메모리 설정을 항상 제공해야 합니다.

JVM 메모리 레이아웃은 복잡하고 버전에 따라 다르며 자세한 설명은 이 문서의 범위를 벗어납니다. 그러나 최소한 다음 세 가지 메모리 관련 작업은 컨테이너에서 **OpenJDK**를 실행하기 위한 시작점으로서 중요합니다.

1. **JVM** 최대 힙 크기를 덮어씁니다.
2. 적절한 경우 **JVM**에서 사용하지 않는 메모리를 운영 체제에 제공하도록 유도합니다.
3. 컨테이너 내의 모든 **JVM** 프로세스가 적절하게 구성되었는지 확인합니다.

컨테이너에서 실행하기 위해 **JVM** 워크로드를 최적으로 튜닝하는 것은 이 문서의 범위를 벗어나며 다양한 **JVM** 옵션을 추가로 설정하는 작업이 포함될 수 있습니다.

4.5.2.1. JVM 최대 힙 크기를 덮어쓰는 방법 이해

대다수의 **Java** 워크로드에서 **JVM** 힙은 메모리를 가장 많이 사용하는 단일 소비 항목입니다. 현재 **OpenJDK**는 기본적으로 **OpenJDK**가 컨테이너에서 실행되는지의 여부와 관계없이 컴퓨팅 노드 메모리의 최대 **1/4(1/-XX:MaxRAMFraction)**을 힙에 사용할 수 있도록 허용합니다. 따라서 특히 컨테이너 메모리 제한도 설정되어 있는 경우 이 동작을 덮어쓰는 것이 중요합니다.

위 작업은 두 가지 이상의 방법으로 수행할 수 있습니다.

- 컨테이너 메모리 제한이 설정되어 있고 JVM에서 실험 옵션을 지원하는 경우 -XX:+UnlockExperimentalVMOptions -XX:+UseCGroupMemoryLimitForHeap을 설정합니다.

참고

UseCGroupMemoryLimitForHeap 옵션이 JDK 11에서 제거되었습니다. 대신 -XX:+UseContainerSupport를 사용합니다.

이 명령은 -XX:MaxRAM을 컨테이너 메모리 제한으로 설정하고 최대 힙 크기(-XX:MaxHeapSize / -Xmx)를 1/-XX:MaxRAMFraction(기본값: 1/4)으로 설정합니다.

- -XX:MaxRAM, -XX:MaxHeapSize 또는 -Xmx 중 하나를 직접 덮어씁니다.

이 옵션을 수행하려면 값을 하드 코딩해야 하지만 안전한 여백을 계산할 수 있다는 장점이 있습니다.

4.5.2.2. JVM에서 사용하지 않는 메모리를 운영 체제에 제공하도록 유도하는 방법 이해

기본적으로 OpenJDK는 사용하지 않는 메모리를 운영 체제에 적극적으로 반환하지 않습니다. 이는 대다수의 컨테이너화된 Java 워크로드에 적합할 수 있습니다. 그러나 추가 프로세스가 네이티브인지 추가 JVM인지 또는 이 둘의 조합인지와 관계없이 컨테이너 내에서 추가 활성 프로세스가 JVM과 공존하는 워크로드는 주목할 만한 예외입니다.

Java 기반 에이전트는 다음 JVM 인수를 사용하여 JVM에서 사용하지 않는 메모리를 운영 체제에 제공하도록 유도할 수 있습니다.

```
-XX:+UseParallelGC
-XX:MinHeapFreeRatio=5 -XX:MaxHeapFreeRatio=10 -XX:GCTimeRatio=4
-XX:AdaptiveSizePolicyWeight=90.
```

이러한 인수는 할당된 메모리가 사용 중인 메모리의 110%(-XX:MaxHeapFreeRatio)를 초과할 때마다 힙 메모리를 운영 체제에 반환하기 위한 것으로, 가비지 수집기에서 최대 20%(-XX:GCTimeRatio)의 CPU 시간을 사용합니다. 애플리케이션 힙 할당은 항상 초기 힙 할당(-XX:InitialHeapSize / -Xms로 덮어씀)보다 적지 않습니다. 자세한 내용은 OpenShift에서 Java 풋프린트 튜닝(1부), OpenShift에서 Java 풋프린트 튜닝(2부), OpenJDK 및 컨테이너에서 확인할 수 있습니다.

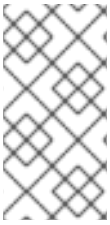
4.5.2.3. 컨테이너 내의 모든 JVM 프로세스를 적절하게 구성하는 방법 이해

동일한 컨테이너에서 여러 개의 JVM이 실행되는 경우 모든 JVM이 올바르게 구성되어 있는지 확인해야 합니다. 워크로드가 많은 경우 각 JVM에 백분율로 된 메모리 예산을 부여하여 추가 안전 범위를 충분히 유지해야 합니다.

많은 Java 툴은 다양한 환경 변수(JAVA_OPTS,GRADLE_OPTS 등)를 사용하여 JVM을 구성하며 올바른 설정이 올바른 JVM으로 전달되도록 하는 것이 어려울 수 있습니다.

OpenJDK는 항상 JAVA_TOOL_OPTIONS 환경 변수를 준수하고 JAVA_TOOL_OPTIONS에 지정된 값은 JVM 명령줄에 지정된 다른 옵션에서 덮어씁니다. 기본적으로 이러한 옵션이 Java 기반 에이전트 이미지에서 실행되는 모든 JVM 워크로드에 기본적으로 사용되도록 AWS Jenkins Maven 에이전트 이미지의 Red Hat OpenShift Service를 설정합니다.

```
JAVA_TOOL_OPTIONS="-XX:+UnlockExperimentalVMOptions
-XX:+UseCGroupMemoryLimitForHeap -Dsun.zip.disableMemoryMapping=true"
```



참고

UseCGroupMemoryLimitForHeap 옵션이 JDK 11에서 제거되었습니다. 대신 -XX:+UseContainerSupport를 사용합니다.

이러한 설정을 통해 추가 옵션이 필요하지 않다고 보장할 수는 없지만 유용한 시작점이 될 수 있습니다.

4.5.3. Pod 내에서 메모리 요청 및 제한 찾기

Pod 내에서 메모리 요청 및 제한을 동적으로 검색하려는 애플리케이션에서는 Downward API를 사용해야 합니다.

절차

- MEMORY_REQUEST 및 MEMORY_LIMIT 스탠자를 추가하도록 Pod를 구성합니다.

  a. 다음과 유사한 YAML 파일을 생성합니다.

```
apiVersion: v1
kind: Pod
```

```
metadata:
  name: test
spec:
  securityContext:
    runAsNonRoot: true
    seccompProfile:
      type: RuntimeDefault
  containers:
  - name: test
    image: fedora:latest
    command:
    - sleep
    - "3600"
    env:
    - name: MEMORY_REQUEST (1)
      valueFrom:
        resourceFieldRef:
          containerName: test
          resource: requests.memory
    - name: MEMORY_LIMIT (2)
      valueFrom:
        resourceFieldRef:
          containerName: test
          resource: limits.memory
    resources:
      requests:
        memory: 384Mi
      limits:
        memory: 512Mi
    securityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop: [ALL]
```

(1) 이 스탠자를 추가하여 애플리케이션 메모리 요청 값을 검색합니다.

(2) 이 스탠자를 추가하여 애플리케이션 메모리 제한 값을 검색합니다.

b. 다음 명령을 실행하여 **Pod**를 생성합니다.

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

검증

1. 원격 쉘을 사용하여 **Pod**에 액세스합니다.

```
$ oc rsh test
```

2. 요청된 값이 적용되었는지 확인합니다.

```
$ env | grep MEMORY | sort
```

출력 예

```
MEMORY_LIMIT=536870912
MEMORY_REQUEST=402653184
```

참고

메모리 제한 값은 **/sys/fs/cgroup/memory/memory.limit_in_bytes** 파일을 통해 컨테이너 내부에서도 확인할 수 있습니다.

4.5.4. OOM 종료 정책 이해

Red Hat OpenShift Service on AWS는 컨테이너에 있는 모든 프로세스의 총 메모리 사용량이 메모리 제한을 초과하거나 노드 메모리 소모가 심각한 경우 컨테이너의 프로세스를 종료할 수 있습니다.

프로세스가 **OOM(Out of Memory)** 종료되면 컨테이너가 즉시 종료될 수 있습니다. 컨테이너 **PID 1** 프로세스에서 **SIGKILL**을 수신하면 컨테이너가 즉시 종료됩니다. 그 외에는 컨테이너 동작이 기타 프로세스의 동작에 따라 달라집니다.

예를 들어 컨테이너 프로세스가 코드 **137**로 종료되면 **SIGKILL** 신호가 수신되었음을 나타냅니다.

컨테이너가 즉시 종료되지 않으면 다음과 같이 **OOM** 종료를 탐지할 수 있습니다.

1. 원격 쉘을 사용하여 **Pod**에 액세스합니다.

```
# oc rsh test
```

2. 다음 명령을 실행하여 **/sys/fs/cgroup/memory/memory.oom_control**에서 현재 **OOM** 종료 수를 확인합니다.

```
$ grep '^oom_kill ' /sys/fs/cgroup/memory/memory.oom_control
```

출력 예

```
oom_kill 0
```

3. 다음 명령을 실행하여 **OOM** 종료를 유도합니다.

```
$ sed -e '' </dev/zero
```

출력 예

```
Killed
```

4. 다음 명령을 실행하여 **sed** 명령의 종료 상태를 확인합니다.

```
$ echo $?
```

출력 예

```
137
```

137 코드는 컨테이너 프로세스가 코드 137로 종료되었음을 나타냅니다. 이 코드는 SIGKILL 신호가 수신되었음을 나타냅니다.

5. 다음 명령을 실행하여 /sys/fs/cgroup/memory/memory.oom_control에서 OOM 종료 카운터가 증가했는지 확인합니다.

```
$ grep '^oom_kill ' /sys/fs/cgroup/memory/memory.oom_control
```

출력 예

```
oom_kill 1
```

Pod에서 하나 이상의 프로세스가 OOM 종료된 경우 나중에 Pod가 종료되면(즉시 여부와 관계없이) 단계는 실패, 이유는 OOM 종료가 됩니다. restartPolicy 값에 따라 OOM 종료된 Pod가 다시 시작될 수 있습니다. 재시작되지 않는 경우 복제 컨트롤러와 같은 컨트롤러는 Pod의 실패 상태를 확인하고 새 Pod를 생성하여 이전 Pod를 교체합니다.

다음 명령을 사용하여 Pod 상태를 가져옵니다.

```
$ oc get pod test
```

출력 예

```
NAME      READY     STATUS      RESTARTS   AGE
test      0/1       OOMKilled   0          1m
```

- Pod가 재시작되지 않은 경우 다음 명령을 실행하여 Pod를 확인합니다.

```
$ oc get pod test -o yaml
```

출력 예

```
...
status:
  containerStatuses:
  - name: test
    ready: false
    restartCount: 0
    state:
      terminated:
        exitCode: 137
        reason: OOMKilled
  phase: Failed
```

- 재시작된 경우 다음 명령을 실행하여 **Pod**를 확인합니다.

```
$ oc get pod test -o yaml
```

출력 예

```
...
status:
  containerStatuses:
  - name: test
    ready: true
    restartCount: 1
    lastState:
      terminated:
        exitCode: 137
        reason: OOMKilled
    state:
      running:
  phase: Running
```

4.5.5. Pod 제거 이해

Red Hat OpenShift Service on AWS는 노드의 메모리가 소모되면 노드에서 **Pod**를 제거할 수 있습니다. 메모리 소모 범위에 따라 제거가 정상적으로 수행되지 않을 수 있습니다. 정상적인 제거에서는 프로세스가 아직 종료되지 않은 경우 각 컨테이너의 기본 프로세스(**PID 1**)에서 **SIGTERM** 신호를 수신한 다음 잠

시 후 **SIGKILL** 신호를 수신합니다. 비정상적인 제거에서는 각 컨테이너의 기본 프로세스에서 **SIGKILL** 신호를 즉시 수신합니다.

제거된 **Pod**의 단계는 실패, 이유는 제거됨입니다. **restartPolicy** 값과 관계없이 재시작되지 않습니다. 그러나 복제 컨트롤러와 같은 컨트롤러는 **Pod**의 실패 상태를 확인하고 새 **Pod**를 생성하여 이전 **Pod**를 교체합니다.

```
$ oc get pod test
```

출력 예

```
NAME    READY   STATUS    RESTARTS   AGE
test    0/1     Evicted   0          1m
```

```
$ oc get pod test -o yaml
```

출력 예

```
...
status:
  message: 'Pod The node was low on resource: [MemoryPressure].'
  phase: Failed
  reason: Evicted
```

5장. PID 제한 구성

PID(프로세스 ID)는 시스템에서 현재 실행 중인 각 프로세스 또는 스레드에 Linux 커널에서 할당한 고유 식별자입니다. 시스템에서 동시에 실행할 수 있는 프로세스 수는 Linux 커널의 4,194,304로 제한됩니다. 이 숫자는 메모리, CPU 및 디스크 공간과 같은 다른 시스템 리소스에 대한 제한된 액세스의 영향을 받을 수도 있습니다.

AWS 4.11 이상의 Red Hat OpenShift Service에서 Pod는 기본적으로 최대 4,096 PID를 가질 수 있습니다. 워크로드에 그 이상으로 필요한 경우 KubeletConfig 오브젝트를 구성하여 허용되는 최대 PID 수를 늘릴 수 있습니다.

4.11 이전 버전을 실행하는 AWS 클러스터의 Red Hat OpenShift Service는 기본 PID 제한이 1024 를 사용합니다.

5.1. 프로세스 ID 제한 이해

AWS의 Red Hat OpenShift Service에서 클러스터에서 작업을 예약하기 전에 PID(프로세스 ID) 사용에 대해 지원되는 다음 두 가지 제한 사항을 고려하십시오.

- Pod당 최대 PID 수입니다.

  기본값은 AWS 4.11 이상에서 Red Hat OpenShift Service의 4,096입니다. 이 값은 노드에 설정된 podPidsLimit 매개변수에 의해 제어됩니다.

- 노드당 최대 PID 수입니다.

  기본값은 노드 리소스에 따라 다릅니다. AWS의 Red Hat OpenShift Service에서 이 값은 --system-reserved 매개변수에 의해 제어되며, 노드의 총 리소스에 따라 각 노드에 PID를 예약합니다.

Pod가 Pod당 허용되는 최대 PID 수를 초과하면 Pod가 올바르게 작동을 중지하고 노드에서 제거될 수 있습니다. 자세한 내용은 제거 신호 및 임계값에 대한 Kubernetes 문서를 참조하십시오.

노드가 노드당 허용되는 최대 PID 수를 초과하면 새 프로세스에 PID를 할당할 수 없으므로 노드가 불안정해질 수 있습니다. 추가 프로세스를 생성하지 않고 기존 프로세스를 완료할 수 없는 경우 전체 노드를 사용할 수 없게 되고 재부팅이 필요할 수 있습니다. 이 경우 실행 중인 프로세스 및 애플리케이션에 따라

데이터가 손실될 수 있습니다. 고객 관리자 및 Red Hat 사이트 안정성 엔지니어링은 이 임계값에 도달하면 알림을 받으며 작업자 노드에 PIDPressure 경고가 표시됩니다.

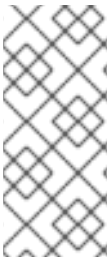
5.2. AWS POD에서 RED HAT OPENSHIFT SERVICE에 대해 더 높은 프로세스 ID 제한을 설정하는 위험

Pod의 podPidsLimit 매개변수는 해당 Pod에서 동시에 실행할 수 있는 최대 프로세스 및 스레드 수를 제어합니다.

podPidsLimit 의 값을 기본값인 4,096에서 최대 16,384로 늘릴 수 있습니다. podPidsLimit 을 변경하려면 영향을 받는 노드를 재부팅해야 하므로 이 값을 변경하면 애플리케이션의 다운타임이 발생할 수 있습니다.

노드당 다수의 Pod를 실행 중이고 노드에 podPidsLimit 값이 높은 경우 노드의 PID 최대값이 초과될 위험이 있습니다.

노드의 PID 최대값을 초과하지 않고 단일 노드에서 동시에 실행할 수 있는 최대 Pod 수를 찾으려면 podPidsLimit 값으로 3,650,000을 나눕니다. 예를 들어 podPidsLimit 값이 16,384이고 Pod가 프로세스 ID 수에 가깝게 사용할 것으로 예상되는 경우 단일 노드에서 222 Pod를 안전하게 실행할 수 있습니다.



참고

memory, CPU 및 사용 가능한 스토리지는 podPidsLimit 값이 적절하게 설정된 경우에도 동시에 실행할 수 있는 최대 Pod 수를 제한할 수 있습니다. 자세한 내용은 "환경 계획" 및 "제한 및 확장성"을 참조하십시오.

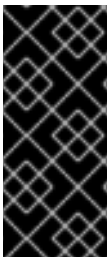
추가 리소스

- 인스턴스 유형
- 환경 계획
- 제한 및 확장성

5.3. AWS 클러스터의 기존 RED HAT OPENSHIFT SERVICE에 더 높은 프로세스 ID 제한 설정

--pod-pids-limit 매개변수를 변경하는 KubeletConfig 오브젝트를 생성하거나 편집하여 AWS의 기존

Red Hat OpenShift Service에 더 높은 podPidsLimit 을 설정할 수 있습니다.



중요

기존 클러스터에서 podPidsLimit 을 변경하면 클러스터의 비컨트롤 플레인 노드가 한 번에 하나씩 재부팅됩니다. 클러스터의 최대 사용 시간을 초과하여 이러한 변경을 수행하고 모든 노드가 재부팅될 때까지 클러스터 업그레이드 또는 중단을 방지합니다.

사전 요구 사항

- AWS 클러스터에 Red Hat OpenShift Service가 있습니다.
- ROSA CLI(rosa)를 설치했습니다.
- OpenShift CLI(oc)가 설치되어 있습니다.
- ROSA CLI를 사용하여 Red Hat 계정에 로그인했습니다.

절차

1. KubeletConfig 오브젝트를 생성하거나 편집하여 PID 제한을 변경합니다.

- 기본 PID 제한을 처음 변경하는 경우 다음 명령을 실행하여 KubeletConfig 오브젝트를 생성하고 --pod-pids-limit 값을 설정합니다.

```
$ rosa create kubeletconfig -c <cluster_name> --name <kubeletconfig_name> --
pod-pids-limit=<value>
```

참고

ROSA Classic 클러스터당 하나의 KubeletConfig 오브젝트만 지원되므로 --name 매개변수는 ROSA Classic 클러스터에서 선택 사항입니다.

예를 들어 다음 명령은 클러스터 my-cluster 의 Pod당 최대 16,384개의 PID를 설정합니다.

```
$ rosa create kubeletconfig -c my-cluster --name set-high-pids --pod-pids-limit=16384
```

- 이전에 **KubeletConfig** 오브젝트를 생성한 경우 다음 명령을 실행하여 기존 **KubeletConfig** 오브젝트를 편집하고 **--pod-pids-limit** 값을 설정합니다.

```
$ rosa edit kubeletconfig -c <cluster_name> --name <kubeletconfig_name> --pod-pids-limit=<value>
```

작업자 노드의 클러스터 전체 롤링 재부팅이 트리거됩니다.

2. 다음 명령을 실행하여 모든 작업자 노드가 재부팅되었는지 확인합니다.

```
$ oc get machineconfigpool
```

출력 예

```
NAME     CONFIG                 UPDATED  UPDATING  DEGRADED  MACHINECOUNT  READYMACHINECOUNT  UPDATEDMACHINECOUNT  DEGRADEDMACHINECOUNT  AGE
master   rendered-master-06c9c4…  True     False     False     3             3                   3                    0                     4h42m
worker   rendered-worker-f4b64…   True     False     False     4             4                   4                    0                     4h42m
```

검증

클러스터의 각 노드가 재부팅되면 새 설정이 있는지 확인할 수 있습니다.

- **KubeletConfig** 오브젝트에서 **Pod Pids** 제한을 확인합니다.

```
$ rosa describe kubeletconfig --cluster=<cluster_name>
```

다음 예와 같이 새 **PIDs** 제한이 출력에 표시됩니다.

출력 예

```
Pod Pids Limit:                 16384
```

5.4. 클러스터에서 사용자 정의 구성 제거

구성 세부 정보가 포함된 KubeletConfig 오브젝트를 제거하여 클러스터에서 사용자 지정 구성을 제거할 수 있습니다.

사전 요구 사항

- AWS 클러스터에 기존 Red Hat OpenShift Service가 있습니다.
- ROSA CLI(rosa)를 설치했습니다.
- ROSA CLI를 사용하여 Red Hat 계정에 로그인했습니다.

절차

- 관련 사용자 지정 KubeletConfig 오브젝트를 삭제하여 클러스터에서 사용자 지정 구성을 제거합니다.

```
$ rosa delete kubeletconfig --cluster <cluster_name> --name <kubeletconfig_name>
```

검증 단계

- 사용자 정의 KubeletConfig 오브젝트가 클러스터에 나열되어 있지 않은지 확인합니다.

```
$ rosa describe kubeletconfig --name <cluster_name>
```