

vRealize Automation Cloud Assembly 사용 및 관 리

2022년 10월

vRealize Automation 8.0

다음 VMware 웹 사이트에서 최신 기술 문서를 확인할 수 있습니다.

<https://docs.vmware.com/kr/>

VMware, Inc.
3401 Hillview Ave.
Palo Alto, CA 94304
www.vmware.com

VMware 코리아
서울시 강남구
영동대로 517
아셈타워 13층
(우) 06164
전화: +82 2 3016 6500
팩스: +82 2 3016 6501
www.vmware.com/kr

Copyright © 2022 VMware, Inc. All rights reserved. [저작권 및 상표 정보](#)

목차

1	vRealize Automation Cloud Assembly 개요	6
	vRealize Automation Cloud Assembly 작동 방식	6
2	조직에 vRealize Automation Cloud Assembly 설정	9
	vRealize Automation Cloud Assembly 사용자 역할이란?	9
	클라우드 계정 추가	12
	클라우드 계정을 사용한 작업에 필요한 자격 증명	12
	vRealize Automation에서 Microsoft Azure 클라우드 계정 생성	31
	vRealize Automation에서 Amazon Web Services 클라우드 계정 생성	31
	Google Cloud Platform 클라우드 계정 생성	32
	vCenter 클라우드 계정 생성	33
	NSX-V 클라우드 계정 생성	35
	NSX-T 클라우드 계정 생성	36
	VMware Cloud on AWS 클라우드 계정 생성	37
	다른 애플리케이션과 통합	39
	GitLab 및 GitHub 통합을 사용하는 방법	39
	외부 IPAM 통합 지점 구성	43
	최신 IPAM 통합 패키지로 업그레이드하는 방법	44
	vRealize Automation Cloud Assembly에서 My VMware 통합 구성	45
	Cloud Assembly에서 vRealize Orchestrator 통합 구성	46
	vRealize Automation Cloud Assembly에서 Kubernetes를 사용하는 방법	49
	vRealize Automation Cloud Assembly의 구성 관리	56
	vRealize Automation Cloud Assembly에서 Active Directory 통합을 생성하는 방법	61
	온보딩 계획이란?	62
	선택한 시스템을 단일 배포로 온보딩	63
	개별 배포로서 온보딩 규칙으로 필터링된 시스템	65
	고급 구성	71
	인터넷 프록시 서버를 구성하는 방법	71
	cloud-init 또는 cloudbase-init를 사용하여 Windows 템플릿을 설정하는 방법	75
	IPAM SDK를 사용하여 제공자별 외부 IPAM 통합 패키지를 생성하는 방법	76
3	vRealize Automation Cloud Assembly 사용 사례	77
	WordPress 사용 사례	77
	인프라 생성	78
	프로젝트 생성	85
	Blueprint 생성 및 확장	86

VMware Cloud on AWS 사용 사례	102
기본 VMware Cloud on AWS 워크플로 구성	103
VMware Cloud on AWS에서 격리된 네트워크 구성	116
제공자별 외부 IPAM 통합 사용 사례	121
다운로드 패키지를 배포하기 전에 Infoblox 애플리케이션에서 필요한 확장 가능 특성 추가	122
외부 IPAM 제공자 패키지 다운로드 및 배포	124
IPAM 통합 지점에 대한 실행 환경 생성	125
외부 IPAM 통합 지점 추가	126
IPAM 제공자 값을 사용하도록 네트워크 및 네트워크 프로파일 구성	129
IPAM 제공자 범위 할당을 사용하는 Blueprint 정의 및 배포	131
IPAM 통합에 대해 Infoblox 특정 속성 사용	135

4 리소스 인프라 구축 137

클라우드 영역을 추가하는 방법	137
클라우드 영역에 대해 알아보기	137
플레이버 매핑을 추가하는 방법	139
버전 매핑에 대해 알아보기	139
이미지 매핑을 추가하는 방법	140
이미지 매핑에 대해 알아보기	140
네트워크 프로파일을 추가하는 방법	143
네트워크 프로파일에 대해 알아보기	144
네트워크 및 네트워크 프로파일에서 IP 주소 사용	150
네트워크 및 네트워크 프로파일 사용	150
로드 밸런서 설정 사용	153
스토리지 프로파일을 추가하는 방법	155
스토리지 프로파일에 대해 알아보기	155
태그를 사용하는 방법	156
태그 지정 전략 생성	158
vRealize Automation Cloud Assembly에서 기능 태그 사용	159
vRealize Automation Cloud Assembly에서 제약 조건 태그 사용	160
표준 태그	162
vRealize Automation Cloud Assembly의 태그 처리 방식	163
간단한 태그 지정 구조를 설정하는 방법	163
리소스로 작업하는 방법	165
계산 리소스	165
네트워크 리소스	165
보안 리소스	166
스토리지 리소스	168
시스템 리소스	168
볼륨 리소스	168

리소스에 대해 알아보기 169

5 프로젝트 추가 및 관리 174

개발 팀용 프로젝트를 추가하는 방법 174

프로젝트에 대해 알아보기 176

프로젝트 태그 및 사용자 지정 속성 사용 176

배포 시간에 프로젝트가 작동하는 방식 177

6 배포 설계 179

Blueprint를 생성하기 전에 180

Blueprint를 생성하는 방법 180

간단한 Blueprint를 처음부터 생성하는 방법 182

Blueprint에 구성 요소를 선택하고 추가하는 방법 182

Blueprint 리소스를 연결하는 방법 183

유효한 Blueprint 코드를 생성하는 방법 184

간단한 Blueprint를 향상시키는 방법 186

사용자 입력으로 Blueprint를 사용자 지정하는 방법 187

구성 요소 배포 순서를 설정하는 방법 192

표현식을 사용하여 Blueprint 코드를 보다 다양하게 만드는 방법 193

Blueprint에서 시스템을 자동으로 초기화하는 방법 202

Blueprint에서 원격 액세스를 사용하도록 설정하는 방법 210

여러 버전의 Blueprint를 저장하는 방법 213

배포된 리소스의 이름을 사용자 지정하는 방법 215

리소스 속성 소개 217

일부 Blueprint 코드 예시 217

Blueprint의 vSphere 구성 요소 예 217

검토 가능한 Blueprint 221

네트워크, 보안 및 로드 밸런서 Blueprint 예 228

사용자 이름 및 암호 액세스를 사용하는 Puppet 지원 Blueprint 232

마켓플레이스를 사용하는 방법 241

확장성을 사용하여 애플리케이션 수명 주기를 연장 및 자동화하는 방법 242

확장성 작업 구독 242

확장성 워크플로 구독 261

확장성 구독에 대해 알아보기 268

7 배포 관리 277

활성 배포를 모니터링하는 방법 278

vRealize Automation Cloud Assembly 배포 실패 시 수행할 수 있는 작업 279

완료된 배포의 수명주기를 관리하는 방법 282

배포에서 실행할 수 있는 작업 285

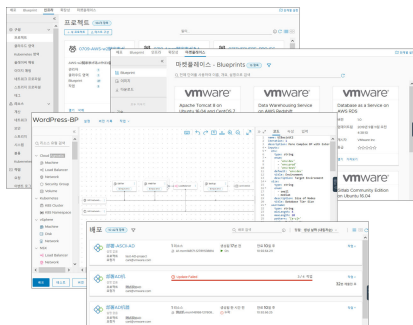
vRealize Automation Cloud Assembly 개요

1

vRealize Automation Cloud Assembly는 해당 리소스에 생성할 시스템, 애플리케이션 및 서비스를 배포할 수 있도록 공용 및 사설 클라우드 공급업체에 연결하는 데 사용됩니다. 여러분의 팀은 개발부터 테스트, 운영에 이르기까지 반복 워크플로를 지원하는 환경에서 코드로 구현되는 Blueprint(Blueprint-as-code)를 개발합니다. 프로비저닝 시 다양한 클라우드 벤더에 걸쳐 배포할 수 있습니다. 이 서비스는 관리되는 VMware SaaS 및 NaaS 기반 프레임워크입니다.

vRealize Automation Cloud Assembly 개요에는 다음과 같은 기본 기능이 포함되어 있습니다.

- [인프라] 탭에서는 클라우드 벤더 리소스와 사용자를 추가하고 구성할 수 있습니다. 또한 이 탭에는 배포된 Blueprint에 대한 정보도 표시됩니다.
- [마켓플레이스] 탭에서는 OVA 또는 OVF를 지원하는 Blueprint 라이브러리 및 액세스 기능을 구축하는 데 도움이 되는 VMware Solution Exchange Blueprint 및 이미지를 제공합니다.
- [Blueprint] 탭은 개발 홈 역할을 하며 캔버스 및 YAML 편집기를 사용하여 시스템과 애플리케이션을 개발하고 배포할 수 있습니다.
- [배포] 탭에는 프로비저닝된 리소스의 현재 상태가 표시됩니다. 세부 정보 및 기록 정보를 활용하여 배포를 관리할 수 있습니다.



본 장은 다음 항목을 포함합니다.

- vRealize Automation Cloud Assembly 작동 방식

vRealize Automation Cloud Assembly 작동 방식

vRealize Automation Cloud Assembly는 Blueprint 개발 및 배포 서비스입니다. 사용자와 팀 멤버는 이 서비스를 사용하여 클라우드 벤더 리소스에 시스템, 애플리케이션 및 서비스를 배포합니다.

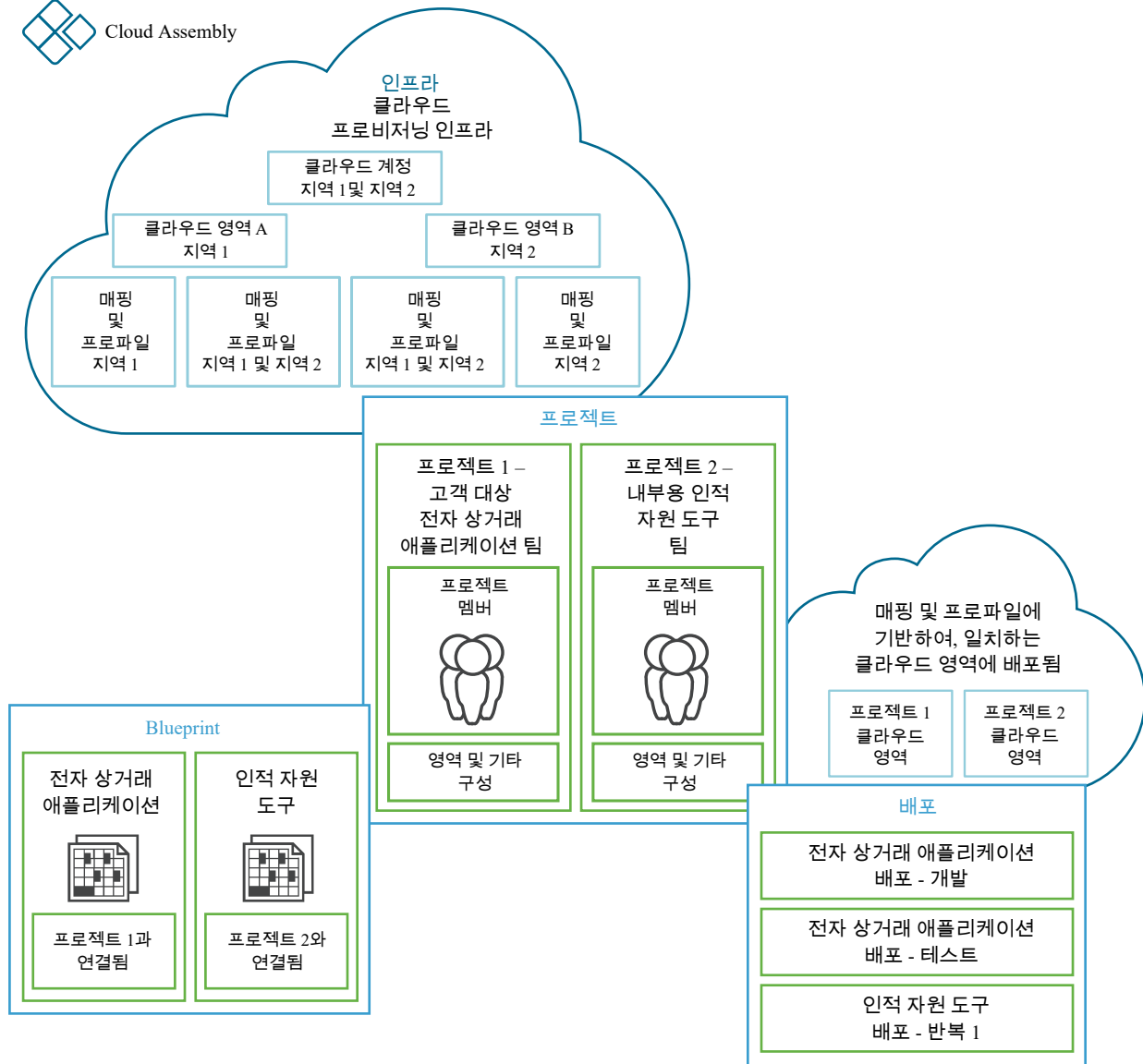
일반적으로 클라우드 관리자라고 하는 Cloud Assembly 관리자는 프로비저닝 인프라를 설정하고 사용자와 리소스를 그룹화하는 프로젝트를 생성합니다.

- 클라우드 벤더 계정을 추가합니다. [vRealize Automation Cloud Assembly에 클라우드 계정 추가 항목을 참조하십시오.](#)
- 개발자가 배포할 때 사용할 수 있는 지역 또는 데이터스토어, 즉 클라우드 영역을 결정합니다. [vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기 항목을 참조하십시오.](#)
- 클라우드 영역을 정의하는 정책을 생성합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축 항목을 참조하십시오.](#)
- 개발자를 클라우드 영역에 그룹화하는 프로젝트를 생성합니다. [vRealize Automation Cloud Assembly 프로젝트 태그 및 사용자 지정 속성 사용 항목을 참조하십시오.](#)

Blueprint 개발자는 프로젝트 하나 이상의 멤버입니다. 개발자는 Blueprint를 생성하여 프로젝트 중 하나에 연결된 클라우드 영역에 배포합니다.

- 캔버스를 사용하여 프로젝트를 위한 Blueprint를 개발합니다. 프로젝트 관리자는 마켓플레이스를 사용하여 Blueprint 및 지원 이미지를 VMware Solution Exchange에서 다운로드할 수 있습니다. [장 6 vRealize Automation Cloud Assembly 배포 설계 및 vRealize Automation Cloud Assembly 마켓플레이스를 사용하는 방법 항목을 참조하십시오.](#)
- 정책과 제약 조건을 기반으로 Blueprint를 프로젝트 클라우드 영역에 배포합니다.
- 사용되지 않는 애플리케이션을 삭제하는 등 배포를 관리합니다. [장 7 vRealize Automation Cloud Assembly 배포 관리 항목을 참조하십시오.](#)

vRealize Automation Cloud Assembly를 시작합니다. 인프라를 정의한 후 Blueprint를 생성하고 배포하는 방법의 예시를 보려면 [WordPress 사용 사례 항목을 참조하십시오.](#)



조직에 vRealize Automation Cloud Assembly 설정

2

Cloud Assembly 관리자는 사용자 역할을 이해하고 클라우드 계정 벤더 및 통합 애플리케이션과의 연결을 설정해야 합니다.

클라우드 계정 및 통합을 구성하는 경우 Cloud Assembly와 해당 대상 시스템 간의 통신을 구성하게 됩니다.

본 장은 다음 항목을 포함합니다.

- vRealize Automation Cloud Assembly 사용자 역할이란?
- vRealize Automation Cloud Assembly에 클라우드 계정 추가
- vRealize Automation를 다른 애플리케이션과 통합
- vRealize Automation Cloud Assembly의 온보딩 계획이란?
- vRealize Automation Cloud Assembly 환경에 대한 고급 구성

vRealize Automation Cloud Assembly 사용자 역할이란?

사용자 역할은 사용자가 vRealize Automation Cloud Assembly에서 보고 수행할 수 있는 작업을 결정합니다. 일부 역할은 조직 수준에서 정의되고 일부는 vRealize Automation Cloud Assembly에만 적용됩니다.

사용자 역할

사용자 역할은 조직 소유자가 vRealize Automation 콘솔에서 조직에 대해 정의합니다. 역할에는 조직 역할과 서비스 역할이라는 두 가지 유형이 있습니다.

조직 역할은 전역적이며 조직의 모든 서비스에 적용됩니다. 조직 수준 역할은 조직 소유자 또는 조직 멤버 역할입니다.

조직 역할에 대한 자세한 내용은 [vRealize Automation 관리](#)를 참조하십시오.

서비스별 사용 권한인 vRealize Automation Cloud Assembly 서비스 역할은 콘솔의 조직 수준에서도 할당됩니다.

표 2-1. 서비스 역할

역할	설명
Cloud Assembly 관리자	전체 사용자 인터페이스 및 API 리소스에 대해 읽기/쓰기 액세스 권한을 갖고 있어야 합니다. 클라우드 계정 추가, 새 프로젝트 생성 및 프로젝트 관리자 할당 등을 볼 수 있고 수행할 수 있는 유일한 사용자 역할입니다.
Cloud Assembly 사용자	Cloud Assembly 관리자 역할이 없는 모든 사용자입니다. vRealize Automation Cloud Assembly 프로젝트에서는 관리자가 프로젝트에 사용자를 프로젝트 멤버로 추가합니다. 관리자는 프로젝트 관리자를 추가할 수도 있습니다. 두 가지 역할에 대한 사용 권한은 아래에 정의되어 있습니다.

프로젝트 역할 및 사용 권한

프로젝트 역할, 프로젝트 관리자 및 프로젝트 멤버는 vRealize Automation Cloud Assembly에서 정의되며 프로젝트마다 다를 수 있습니다.

다음 표에서 사용 권한이 정의된 경우, 클라우드 관리자는 UI의 모든 영역에 대해 모든 권한을 갖습니다.

프로젝트 관리자는 클라우드 관리자가 생성한 인프라를 활용하여, 프로젝트의 멤버가 개발 작업에 필요한 리소스를 사용할 수 있도록 보장합니다.

표 2-2. 프로젝트 관리자 사용 권한

탭	노드 또는 영역	보기	생성	수정/삭제
인프라	구성 - 프로젝트	예(본인 프로젝트만)	아니요	예(본인 프로젝트만)
	구성 - 클라우드 영역	아니요	아니요	아니요
	구성 - 버전 매핑	예	아니요	아니요
	구성 - 이미지 매핑	예	아니요	아니요
	구성 - 네트워크 프로파일	예	아니요	아니요
	구성 - 스토리지 프로파일	예	아니요	아니요
	구성 - 태그	예	아니요	아니요
	리소스 - 계산	예	아니요	아니요
	리소스 - 네트워크	예	아니요	아니요
	리소스 - 스토리지	예	아니요	아니요
	리소스 - 시스템	예(본인 프로젝트만)	예	예(본인 프로젝트만)
	리소스 - 볼륨			
	작업 - 요청	예(본인 프로젝트만)	해당 없음	예(본인 프로젝트만)

표 2-2. 프로젝트 관리자 사용 권한 (계속)

탭	노드 또는 영역	보기	생성	수정/삭제
	작업 - 이벤트	예(본인 프로젝트만)	해당 없음	예(본인 프로젝트만)
	연결 - 클라우드 계정	아니요	아니요	아니요
	연결 - 통합		아니요	아니요
	연결 - 클라우드 프록시		아니요	아니요
	비용 - VMC 평가	예	아니요	아니요
	비용 - 전용 클라우드	예	아니요	아니요
	온보딩		아니요	아니요
Blueprint	Blueprint	예(본인 프로젝트에 대해서만)	예(본인 프로젝트에 대해서만)	예(본인 프로젝트에 대해서만)
배포	배포	예(본인 프로젝트에 대해서만)	해당 없음	예(본인 프로젝트에 대해서만)

프로젝트 멤버는 일반적으로 Blueprint를 생성하고 배포하는 개발자입니다.

표 2-3. 프로젝트 멤버 사용 권한

탭	노드 또는 영역	보기	생성	수정/삭제
인프라	구성 - 프로젝트	예(자신이 멤버로 속해 있는 프로젝트만)	아니요	아니요
	구성 - 클라우드 영역	아니요	아니요	아니요
	구성 - 버전 매핑	예	아니요	아니요
	구성 - 이미지 매핑	예	아니요	아니요
	구성 - 네트워크 프로파일	예	아니요	아니요
	구성 - 스토리지 프로파일	예	아니요	아니요
	구성 - 태그	예	아니요	아니요
	리소스 - 계산	예	아니요	아니요
	리소스 - 네트워크	예	아니요	아니요
	리소스 - 스토리지	예	아니요	아니요
	리소스 - 시스템	예(본인이 배포한 항목만)	예	예(본인이 배포한 항목만)
	리소스 - 볼륨			
	작업 - 요청	예(본인이 배포한 항목만)	해당 없음	예(본인이 배포한 항목만)

표 2-3. 프로젝트 멤버 사용 권한 (계속)

탭	노드 또는 영역	보기	생성	수정/삭제
	작업 - 이벤트	예(본인이 배포한 항목만)	해당 없음	예(본인이 배포한 항목만)
	연결 - 클라우드 계정	아니요	아니요	아니요
	연결 - 통합			
	연결 - 클라우드 프록시			
	비용 - VMC 평가	예	아니요	아니요
	비용 - 전용 클라우드	예	아니요	아니요
	온보딩			
Blueprint	Blueprint	예(본인 프로젝트에 대해서만)	예(본인 프로젝트에 대해서만)	예(본인 프로젝트에 대해서만)
배포	배포	예(프로젝트 배포를 모든 프로젝트 멤버와 공유하는 경우를 제외하고 본인 프로젝트에 대해서만)	해당 없음	예(프로젝트 배포를 모든 프로젝트 멤버와 공유하고 본인에게 2일차 작업을 실행할 권한이 있는 경우를 제외하고 본인 프로젝트에 대해서만)

vRealize Automation Cloud Assembly에 클라우드 계정 추가

클라우드 계정은 지역 또는 데이터 센터에서 데이터를 수집하고 해당 지역에 Blueprint를 배포하기 위해 vRealize Automation Cloud Assembly에서 사용하는 구성된 사용 권한입니다.

수집된 데이터에는 나중에 클라우드 영역과 연결하는 지역이 포함됩니다.

나중에 클라우드 영역, 매핑 및 프로파일을 구성할 때 이러한 항목들이 연결된 클라우드 계정을 선택합니다.

클라우드 관리자는 팀 멤버가 작업하는 프로젝트에 대해 클라우드 계정을 생성합니다. 네트워크 및 보안, 계산, 스토리지 및 태그 콘텐츠 같은 리소스 정보는 클라우드 계정에서 데이터 수집됩니다.

참고 클라우드 계정에 이미 지역에 배포되어 있는 연결된 시스템이 있는 경우 온보딩 계획을 사용하여 그러한 시스템을 vRealize Automation Cloud Assembly 관리로 가져올 수 있습니다. [vRealize Automation Cloud Assembly의 온보딩 계획이란?](#)의 내용을 참조하십시오.

배포에 사용되는 클라우드 계정을 제거하면 해당 배포에 속하는 리소스가 관리되지 않습니다.

vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명

vRealize Automation에서 클라우드 계정을 구성하고 작업하려면 다음과 같은 자격 증명이 있는지 확인합니다.

필수 클라우드 계정 자격 증명

수행할 작업	필요 사항
vRealize Automation Cloud Assembly에 가입 및 로그인	<p>VMware ID가 필요합니다.</p> <ul style="list-style-type: none"> ■ 회사 이메일 주소를 사용하여 My VMware 계정을 설정합니다.
vRealize Automation 서비스에 연결	<p>HTTPS 포트 443이 송신 트래픽을 위해 열려 있고 방화벽을 통해 다음에 액세스할 수 있어야 합니다.</p> <ul style="list-style-type: none"> ■ *.vmwareidentity.com ■ gaz.csp-vidm-prod.com ■ *.vmware.com <p>포트 및 프로토콜에 대한 자세한 내용은 VMware 포트 및 프로토콜을 참조하십시오.</p> <p>필수 포트 및 프로토콜에 대한 관련 정보는 다음을 참조하십시오.</p> <ul style="list-style-type: none"> ■ "설치" 도움말의 포트 및 프로토콜 ■ "참조 아키텍처" 도움말의 포트 요구 사항

수행할 작업	필요 사항
AWS(Ama zon Web Services) 클라우드 계정 추가	<p>읽기 및 쓰기 권한을 가진 고급 사용자 계정을 제공합니다. 사용자 계정은 AWS IAM(Identity and Access Management) 시스템에서 전원 액세스 정책(PowerUserAccess)의 멤버여야 합니다.</p> <ul style="list-style-type: none"> ■ 20자리 액세스 키 ID 및 해당하는 비밀 액세스 키 <p>외부 HTTP 인터넷 프록시를 사용 중인 경우에는 IPv4에 대해 해당 프록시를 구성해야 합니다.</p> <p>vRealize Automation ABX(Action-Based Extensibility) 및 외부 IPAM 통합에는 추가 권한이 필요할 수 있습니다.</p> <p>자동 크기 조정 기능을 허용하려면 다음 AWS 권한이 제안됩니다.</p> <ul style="list-style-type: none"> ■ 자동 크기 조정 작업: <ul style="list-style-type: none"> ■ autoscaling:DescribeAutoScalingInstances ■ autoscaling:AttachInstances ■ autoscaling>DeleteLaunchConfiguration ■ autoscaling:DescribeAutoScalingGroups ■ autoscaling>CreateAutoScalingGroup ■ autoscaling:UpdateAutoScalingGroup ■ autoscaling>DeleteAutoScalingGroup ■ autoscaling:DescribeLoadBalancers ■ 자동 크기 조정 리소스: <ul style="list-style-type: none"> ■ * <p>모든 크기 조정 리소스 권한을 제공합니다.</p> <p>AWS STS(Security Token Service) 기능이 AWS ID 및 액세스를 위해 제한된 권한의 임시 자격 증명을 지원하도록 하려면 다음 권한이 필요합니다.</p> <ul style="list-style-type: none"> ■ AWS STS 리소스: <ul style="list-style-type: none"> ■ * <p>모든 STS 리소스 권한을 제공합니다.</p> <p>EC2 기능을 허용하려면 다음과 같은 AWS 권한이 필요합니다.</p> <ul style="list-style-type: none"> ■ EC2 작업: <ul style="list-style-type: none"> ■ ec2:AttachVolume ■ ec2:AuthorizeSecurityGroupIngress ■ ec2>DeleteSubnet ■ ec2>DeleteSnapshot ■ ec2:DescribeInstances ■ ec2>DeleteTags ■ ec2:DescribeRegions ■ ec2:DescribeVolumesModifications ■ ec2>CreateVpc ■ ec2:DescribeSnapshots ■ ec2:DescribeInternetGateways ■ ec2>DeleteVolume ■ ec2:DescribeNetworkInterfaces ■ ec2:StartInstances ■ ec2:DescribeAvailabilityZones ■ ec2>CreateInternetGateway ■ ec2>CreateSecurityGroup ■ ec2:DescribeVolumes

수행할 작업	필요 사항
	<ul style="list-style-type: none"> ■ ec2:CreateSnapshot ■ ec2:ModifyInstanceAttribute ■ ec2:DescribeRouteTables ■ ec2:DescribeInstanceStatus ■ ec2:DetachVolume ■ ec2:RebootInstances ■ ec2:AuthorizeSecurityGroupEgress ■ ec2:ModifyVolume ■ ec2:TerminateInstances ■ ec2:DescribeSpotFleetRequestHistory ■ ec2:DescribeTags ■ ec2:CreateTags ■ ec2:RunInstances ■ ec2:DescribeNatGateways ■ ec2:StopInstances ■ ec2:DescribeSecurityGroups ■ ec2:CreateVolume ■ ec2:DescribeSpotFleetRequests ■ ec2:DescribeImages ■ ec2:DescribeVpcs ■ ec2>DeleteSecurityGroup ■ ec2>DeleteVpc ■ ec2:CreateSubnet ■ ec2:DescribeSubnets ■ ec2:RequestSpotFleet
	<p>참고 vRealize Automation ABX(Action-Based Extensibility) 또는 외부 IPAM 통합에는 SpotFleet 요청 권한이 필요하지 않습니다.</p>
■ EC2 리소스:	<ul style="list-style-type: none"> ■ * <p>모든 EC2 리소스 권한을 제공합니다.</p> <p>탄력적 로드 밸런싱 기능을 허용하려면 다음과 같은 AWS 권한이 필요합니다.</p>
■ 로드 밸런서 작업:	<ul style="list-style-type: none"> ■ elasticloadbalancing:DeleteLoadBalancer ■ elasticloadbalancing:DescribeLoadBalancers ■ elasticloadbalancing:RemoveTags ■ elasticloadbalancing:CreateLoadBalancer ■ elasticloadbalancing:DescribeTags ■ elasticloadbalancing:ConfigureHealthCheck ■ elasticloadbalancing:AddTags ■ elasticloadbalancing:CreateTargetGroup ■ elasticloadbalancing>DeleteLoadBalancerListeners ■ elasticloadbalancing:DeregisterInstancesFromLoadBalancer ■ elasticloadbalancing:RegisterInstancesWithLoadBalancer

수행할 작업

필요 사항

- elasticloadbalancing:CreateLoadBalancerListeners
- 로드 밸런서 리소스:
 - *

모든 로드 밸런서 리소스 권한을 제공합니다.

다음과 같은 AWS IAM(Identity and Access Management) 권한을 사용하도록 설정할 수 있지만 필수는 아닙니다.

- iam:SimulateCustomPolicy
- iam:GetUser
- iam:ListUserPolicies
- iam:GetUserPolicy
- iam:ListAttachedUserPolicies
- iam:GetPolicyVersion
- iam:ListGroupsForUser
- iam:ListGroupPolicies
- iam:GetGroupPolicy
- iam:ListAttachedGroupPolicies
- iam:ListPolicyVersions

수행할 작업	필요 사항
Microsoft Azure 클라우드 계정 추가	<p>Microsoft Azure 인스턴스를 구성하고, 사용 가능한 구독 ID를 제공하는 올바른 Microsoft Azure 구독을 얻습니다.</p> <p>Microsoft Azure 제품 설명서의 방법: 포털을 사용하여 리소스에 액세스할 수 있는 Azure AD 애플리케이션 및 서비스 사용자 생성에 설명된 대로 Active Directory 애플리케이션을 생성합니다.</p> <p>외부 HTTP 인터넷 프록시를 사용 중인 경우에는 IPv4에 대해 해당 프록시를 구성해야 합니다.</p> <p>다음 정보를 기록해 둡니다.</p> <ul style="list-style-type: none"> ■ 구독 ID <p>Microsoft Azure 구독에 액세스할 때 사용합니다.</p> ■ 테넌트 ID <p>Microsoft Azure 계정에 생성하는 Active Directory 애플리케이션에 대한 권한 부여 끝점입니다.</p> ■ 클라이언트 애플리케이션 ID <p>Microsoft Azure 개별 계정에서 Microsoft Active Directory에 대한 액세스를 제공합니다.</p> ■ 클라이언트 애플리케이션 비밀 키 <p>클라이언트 애플리케이션 ID와 쌍으로 연결할 고유 비밀 키입니다.</p> <p>Microsoft Azure 클라우드 계정을 생성하고 검증하기 위해 다음과 같은 사용 권한이 필요합니다.</p> <ul style="list-style-type: none"> ■ Microsoft Compute <ul style="list-style-type: none"> ■ Microsoft.Compute/virtualMachines/extensions/write ■ Microsoft.Compute/virtualMachines/extensions/read ■ Microsoft.Compute/virtualMachines/extensions/delete ■ Microsoft.Compute/virtualMachines/deallocate/action ■ Microsoft.Compute/virtualMachines/delete ■ Microsoft.Compute/virtualMachines/powerOff/action ■ Microsoft.Compute/virtualMachines/read ■ Microsoft.Compute/virtualMachines/restart/action ■ Microsoft.Compute/virtualMachines/start/action ■ Microsoft.Compute/virtualMachines/write ■ Microsoft.Compute/availabilitySets/write ■ Microsoft.Compute/availabilitySets/read ■ Microsoft.Compute/availabilitySets/delete ■ Microsoft.Compute/disks/delete ■ Microsoft.Compute/disks/read ■ Microsoft.Compute/disks/write ■ Microsoft Network <ul style="list-style-type: none"> ■ Microsoft.Network/loadBalancers/backendAddressPools/join/action ■ Microsoft.Network/loadBalancers/delete ■ Microsoft.Network/loadBalancers/read ■ Microsoft.Network/loadBalancers/write ■ Microsoft.Network/networkInterfaces/join/action ■ Microsoft.Network/networkInterfaces/read ■ Microsoft.Network/networkInterfaces/write ■ Microsoft.Network/networkInterfaces/delete ■ Microsoft.Network/networkSecurityGroups/join/action ■ Microsoft.Network/networkSecurityGroups/read

수행할 작업	필요 사항
	<ul style="list-style-type: none"> ■ Microsoft.Network/networkSecurityGroups/write ■ Microsoft.Network/networkSecurityGroups/delete ■ Microsoft.Network/publicIPAddresses/delete ■ Microsoft.Network/publicIPAddresses/join/action ■ Microsoft.Network/publicIPAddresses/read ■ Microsoft.Network/publicIPAddresses/write ■ Microsoft.Network/virtualNetworks/read ■ Microsoft.Network/virtualNetworks/subnets/delete ■ Microsoft.Network/virtualNetworks/subnets/join/action ■ Microsoft.Network/virtualNetworks/subnets/read ■ Microsoft.Network/virtualNetworks/subnets/write ■ Microsoft.Network/virtualNetworks/write ■ Microsoft Resources <ul style="list-style-type: none"> ■ Microsoft.Resources/subscriptions/resourcegroups/delete ■ Microsoft.Resources/subscriptions/resourcegroups/read ■ Microsoft.Resources/subscriptions/resourcegroups/write ■ Microsoft Storage <ul style="list-style-type: none"> ■ Microsoft.Storage/storageAccounts/delete ■ Microsoft.Storage/storageAccounts/listKeys/action ■ Microsoft.Storage/storageAccounts/read ■ Microsoft.Storage/storageAccounts/write ■ Microsoft Web <ul style="list-style-type: none"> ■ Microsoft.Web/sites/read ■ Microsoft.Web/sites/write ■ Microsoft.Web/sites/delete ■ Microsoft.Web/sites/config/read ■ Microsoft.Web/sites/config/write ■ Microsoft.Web/sites/config/list/action ■ Microsoft.Web/sites/publishxml/action ■ Microsoft.Web/serverfarms/write ■ Microsoft.Web/serverfarms/delete ■ Microsoft.Web/sites/hostruntime/functions/keys/read ■ Microsoft.Web/sites/hostruntime/host/read ■ Microsoft.web/sites/functions/masterkey/read <p>ABX(Action-Based Extensibility)가 있는 Microsoft Azure를 사용하는 경우 최소 사용 권한 외에 다음과 같은 사용 권한이 필요합니다.</p> <ul style="list-style-type: none"> ■ Microsoft.Web/sites/read ■ Microsoft.Web/sites/write ■ Microsoft.Web/sites/delete ■ Microsoft.Web/sites/config/read ■ Microsoft.Web/sites/config/write ■ Microsoft.Web/sites/config/list/action ■ Microsoft.Web/sites/publishxml/action ■ Microsoft.Web/serverfarms/write

수행할 작업	필요 사항
	<ul style="list-style-type: none"> ■ Microsoft.Web/serverfarms/delete ■ Microsoft.Web/sites/hostruntime/functions/keys/read ■ Microsoft.Web/sites/hostruntime/host/read ■ Microsoft.Web/sites/functions/masterkey/read <p>확장 기능과 함께 ABX(Action-Based Extensibility)가 있는 Microsoft Azure를 사용하는 경우 다음 사용 권한도 필요합니다.</p> <ul style="list-style-type: none"> ■ Microsoft.Compute/virtualMachines/extensions/write ■ Microsoft.Compute/virtualMachines/extensions/read ■ Microsoft.Compute/virtualMachines/extensions/delete

수행할 작업	필요 사항
GCP(Google Cloud Platform) 클라우드 계정 추가	<p>Google Cloud Platform 클라우드 계정이 Google Cloud Platform 계산 엔진과 상호 작용합니다.</p> <p>Google Cloud Platform 클라우드 계정을 생성하고 유효성을 검사하려면 프로젝트 관리자 및 소유자 자격 증명이 필요합니다.</p> <p>외부 HTTP 인터넷 프록시를 사용 중인 경우에는 IPv4에 대해 해당 프록시를 구성해야 합니다.</p> <p>계산 엔진 서비스를 사용하도록 설정해야 합니다. vRealize Automation에서 클라우드 계정을 생성하는 경우 계산 엔진을 초기화할 때 생성한 서비스 계정을 사용합니다.</p> <p>사용자가 수행할 수 있는 작업에 따라 다음과 같은 계산 엔진 권한도 필요합니다.</p> <ul style="list-style-type: none"> ■ roles/compute.admin <p>모든 계산 엔진 리소스에 대한 모든 권한을 제공합니다.</p> ■ roles/iam.serviceAccountUser <p>서비스 계정으로 실행되도록 구성된 가상 시스템 인스턴스를 관리하는 사용자에게 액세스를 제공합니다. 다음 리소스 및 서비스에 대한 액세스 권한을 부여합니다.</p> <ul style="list-style-type: none"> ■ compute.* ■ resourceManager.projects.get ■ resourceManager.projects.list ■ serviceUsage.quotas.get ■ serviceUsage.services.get ■ serviceUsage.services.list ■ roles/compute.imageUser <p>이미지에 대한 다른 사용 권한 없이 이미지를 나열하고 읽을 수 있는 권한을 제공합니다. 프로젝트 수준에서 compute.imageUser 역할을 부여하면 사용자가 프로젝트의 모든 이미지를 나열할 수 있습니다. 또한 사용자는 프로젝트의 이미지를 기반으로 인스턴스 및 영구 디스크와 같은 리소스를 생성할 수 있습니다.</p> <ul style="list-style-type: none"> ■ compute.images.get ■ compute.images.getFromFamily ■ compute.images.list ■ compute.images.useReadOnly ■ resourceManager.projects.get ■ resourceManager.projects.list ■ serviceUsage.quotas.get ■ serviceUsage.services.get ■ serviceUsage.services.list ■ roles/compute.instanceAdmin <p>가상 시스템 인스턴스를 생성, 수정 및 삭제할 수 있는 권한을 제공합니다. 여기에는 디스크를 생성, 수정 및 삭제하고 쉼드된 VMBETA 설정을 구성할 수 있는 권한이 포함됩니다.</p> <p>가상 시스템 인스턴스(네트워크 또는 보안 설정이나 서비스 계정으로 실행되는 인스턴스는 제외)를 관리하는 사용자의 경우, 개별 인스턴스나 인스턴스를 포함하는 조직, 폴더 또는 프로젝트에 이 역할을 부여합니다.</p> <p>서비스 계정으로 실행되도록 구성된 가상 시스템 인스턴스를 관리하는 사용자에게는 roles/iam.serviceAccountUser 역할도 필요합니다.</p> <ul style="list-style-type: none"> ■ compute.acceleratorTypes ■ compute.addresses.get ■ compute.addresses.list ■ compute.addresses.use ■ compute.autoscalers

수행할 작업

필요 사항

- compute.diskTypes
- compute.disks.create
- compute.disks.createSnapshot
- compute.disks.delete
- compute.disks.get
- compute.disks.list
- compute.disks.resize
- compute.disks.setLabels
- compute.disks.update
- compute.disks.use
- compute.disks.useReadOnly
- compute.globalAddresses.get
- compute.globalAddresses.list
- compute.globalAddresses.use
- compute.globalOperations.get
- compute.globalOperations.list
- compute.images.get
- compute.images.getFromFamily
- compute.images.list
- compute.images.useReadOnly
- compute.instanceGroupManagers
- compute.instanceGroups
- compute.instanceTemplates
- compute.instances
- compute.licenses.get
- compute.licenses.list
- compute.machineTypes
- compute.networkEndpointGroups
- compute.networks.get
- compute.networks.list
- compute.networks.use
- compute.networks.useExternalIp
- compute.projects.get
- compute.regionOperations.get
- compute.regionOperations.list
- compute.regions
- compute.reservations.get
- compute.reservations.list
- compute.subnetworks.get
- compute.subnetworks.list
- compute.subnetworks.use
- compute.subnetworks.useExternalIp
- compute.targetPools.get
- compute.targetPools.list

수행할 작업	필요 사항
	<ul style="list-style-type: none"> ■ compute.zoneOperations.get ■ compute.zoneOperations.list ■ compute.zones ■ resourcemanager.projects.get ■ resourcemanager.projects.list ■ serviceusage.quotas.get ■ serviceusage.services.get ■ serviceusage.services.list ■ roles/compute.instanceAdmin.v1 <p>계산 엔진 인스턴스, 인스턴스 그룹, 디스크, 스냅샷 및 이미지에 대한 모든 권한을 제공합니다. 모든 계산 엔진 네트워킹 리소스에 대한 읽기 액세스도 제공합니다.</p> <hr/> <p>참고 인스턴스 수준에서 사용자에게 이 역할을 부여하면 해당 사용자가 새 인스턴스를 생성할 수 없습니다.</p> <hr/> <ul style="list-style-type: none"> ■ compute.acceleratorTypes ■ compute.addresses.get ■ compute.addresses.list ■ compute.addresses.use ■ compute.autoscalers ■ compute.backendBuckets.get ■ compute.backendBuckets.list ■ compute.backendServices.get ■ compute.backendServices.list ■ compute.diskTypes ■ compute.disks ■ compute.firewalls.get ■ compute.firewalls.list ■ compute.forwardingRules.get ■ compute.forwardingRules.list ■ compute.globalAddresses.get ■ compute.globalAddresses.list ■ compute.globalAddresses.use ■ compute.globalForwardingRules.get ■ compute.globalForwardingRules.list ■ compute.globalOperations.get ■ compute.globalOperations.list ■ compute.healthChecks.get ■ compute.healthChecks.list ■ compute.httpHealthChecks.get ■ compute.httpHealthChecks.list ■ compute.httpsHealthChecks.get ■ compute.httpsHealthChecks.list ■ compute.images ■ compute.instanceGroupManagers ■ compute.instanceGroups

수행할 작업

필요 사항

- compute.instanceTemplates
- compute.instances
- compute.interconnectAttachments.get
- compute.interconnectAttachments.list
- compute.interconnectLocations
- compute.interconnects.get
- compute.interconnects.list
- compute.licenseCodes
- compute.licenses
- compute.machineTypes
- compute.networkEndpointGroups
- compute.networks.get
- compute.networks.list
- compute.networks.use
- compute.networks.useExternallp
- compute.projects.get
- compute.projects.setCommonInstanceMetadata
- compute.regionBackendServices.get
- compute.regionBackendServices.list
- compute.regionOperations.get
- compute.regionOperations.list
- compute.regions
- compute.reservations.get
- compute.reservations.list
- compute.resourcePolicies
- compute.routers.get
- compute.routers.list
- compute.routes.get
- compute.routes.list
- compute.snapshots
- compute.sslCertificates.get
- compute.sslCertificates.list
- compute.sslPolicies.get
- compute.sslPolicies.list
- compute.sslPolicies.listAvailableFeatures
- compute.subnetworks.get
- compute.subnetworks.list
- compute.subnetworks.use
- compute.subnetworks.useExternallp
- compute.targetHttpProxies.get
- compute.targetHttpProxies.list
- compute.targetHttpsProxies.get
- compute.targetHttpsProxies.list
- compute.targetInstances.get

수행할 작업	필요 사항
	<ul style="list-style-type: none"> ■ compute.targetInstances.list ■ compute.targetPools.get ■ compute.targetPools.list ■ compute.targetSslProxies.get ■ compute.targetSslProxies.list ■ compute.targetTcpProxies.get ■ compute.targetTcpProxies.list ■ compute.targetVpnGateways.get ■ compute.targetVpnGateways.list ■ compute.urlMaps.get ■ compute.urlMaps.list ■ compute.vpnTunnels.get ■ compute.vpnTunnels.list ■ compute.zoneOperations.get ■ compute.zoneOperations.list ■ compute.zones ■ resourceManager.projects.get ■ resourceManager.projects.list ■ serviceusage.quotas.get ■ serviceusage.services.get ■ serviceusage.services.list
NSX-T 클라우드 계정 추가	<p>계정에 다음과 같은 읽기 및 쓰기 권한을 제공합니다.</p> <ul style="list-style-type: none"> ■ NSX-T 엔터프라이즈 관리자 역할 및 액세스 자격 증명 ■ NSX-T IP 주소 또는 FQDN <p>관리자에게는 이 페이지의 다음 "vCenter 기반 클라우드 계정에 대한 vSphere 에이전트 요구 사항" 섹션에 설명된 대로 vCenter Server에 대한 액세스 권한 "도" 필요합니다.</p>
NSX-V 클라우드 계정 추가	<p>계정에 다음과 같은 읽기 및 쓰기 권한을 제공합니다.</p> <ul style="list-style-type: none"> ■ NSX-V 엔터프라이즈 관리자 역할 및 액세스 자격 증명 ■ NSX-V IP 주소 또는 FQDN <p>관리자에게는 이 페이지의 다음 "vCenter 기반 클라우드 계정에 대한 vSphere 에이전트 요구 사항" 섹션에 설명된 대로 vCenter Server에 대한 액세스 권한 "도" 필요합니다.</p>

수행할 작업	필요 사항
vCenter 클라우드 계정 추가	<p>계정에 다음과 같은 읽기 및 쓰기 권한을 제공합니다.</p> <ul style="list-style-type: none"> ■ vCenter IP 주소 또는 FQDN <p>관리자에게는 이 페이지의 다음 "vCenter 기반 클라우드 계정에 대한 vSphere 에이전트 요구 사항" 섹션에 설명된 대로 vCenter Server에 대한 액세스 권한 "도" 필요합니다.</p>
VMC(VMware Cloud on AWS) 클라우드 계정 추가	<p>계정에 다음과 같은 읽기 및 쓰기 권한을 제공합니다.</p> <ul style="list-style-type: none"> ■ cloudadmin@vmc.local 계정 또는 CloudAdmin 그룹의 모든 사용자 계정 ■ NSX 엔터프라이즈 관리자 역할 및 액세스 자격 증명 ■ 조직의 VMware Cloud on AWS SDDC 환경에 대한 NSX 클라우드 관리자 액세스 권한 ■ 조직의 VMware Cloud on AWS SDDC 환경에 대한 관리자 액세스 권한 ■ 조직의 VMware Cloud on AWS 서비스에서 VMware Cloud on AWS 환경의 VMware Cloud on AWS API 토큰 ■ vCenter IP 주소 또는 FQDN <p>관리자에게는 이 페이지의 "vSphere 기반 클라우드 계정에 대한 vCenter 에이전트 요구 사항" 섹션에 나열된 모든 사용 권한이 있는 대상 VMware Cloud on AWS SDDC에 사용되는 vCenter에 대한 액세스 권한 "도" 필요합니다.</p> <p>VMware Cloud on AWS 클라우드 계정을 생성하고 사용하는 데 필요한 사용 권한에 대한 자세한 내용은 VMware Cloud on AWS 제품 설명서에서 "VMware Cloud on AWS 데이터 센터 관리"를 참조하십시오.</p>

vCenter 기반 클라우드 계정에 대한 vSphere 에이전트 요구 사항

다음 표에는 VMware Cloud on AWS 및 vCenter 클라우드 계정을 관리하는 데 필요한 사용 권한이 나열되어 있습니다. 끝점을 호스팅하는 클러스터만이 아닌 vCenter Server의 모든 클러스터에 해당 사용 권한이 설정되어야 합니다.

NSX-V, NSX-T, vCenter 및 VMware Cloud on AWS를 포함하여 모든 vCenter Server 기반 클라우드 계정에 대해 관리자는 vSphere 끝점 자격 증명 또는 호스트 vCenter Server에 대한 관리 액세스 권한을 제공하는 자격 증명을 가지고 있어야 하며, 이러한 자격 증명을 사용하여 에이전트 서비스가 vCenter에서 실행됩니다.

vSphere 에이전트 요구 사항에 대한 자세한 내용은 [VMware vSphere 제품 설명서](#)를 참조하십시오.

표 2-4. vSphere 에이전트가 vCenter Server 인스턴스를 관리하는 데 필요한 사용 권한

특성 값	사용 권한
데이터스토어	<ul style="list-style-type: none"> ■ 공간 할당 ■ 데이터스토어 찾아보기 ■ 하위 수준 파일 작업
데이터스토어 클러스터	데이터스토어 클러스터 구성
폴더	<ul style="list-style-type: none"> ■ 폴더 생성 ■ 폴더 삭제
글로벌	<ul style="list-style-type: none"> ■ 사용자 지정 특성 관리 ■ 사용자 지정 특성 설정
네트워크	네트워크 할당

표 2-4. vSphere 에이전트가 vCenter Server 인스턴스를 관리하는 데 필요한 사용 권한 (계속)

특성 값	사용 권한
사용 권한	사용 권한 수정
리소스	<ul style="list-style-type: none"> ■ VM을 리소스 풀에 할당 ■ 전원이 꺼진 가상 시스템 마이그레이션 ■ 전원이 켜진 가상 시스템 마이그레이션
컨텐츠 라이브러리	<p>컨텐츠 라이브러리에 대한 사용 권한을 할당하려면 관리자가 사용 권한을 사용자에게 글로벌 사용 권한으로 부여해야 합니다. 관련 정보는 VMware vSphere 설명서의 "vSphere 가상 시스템 관리"에서 컨텐츠 라이브러리에 대한 사용 권한의 계층적 상속을 참조하십시오.</p> <ul style="list-style-type: none"> ■ 라이브러리 항목 추가 ■ 로컬 라이브러리 생성 ■ 구독 라이브러리 생성 ■ 라이브러리 항목 삭제 ■ 로컬 라이브러리 삭제 ■ 구독 라이브러리 삭제 ■ 파일 다운로드 ■ 라이브러리 항목 제거 ■ 구독 라이브러리 제거 ■ 구독 정보 검색 ■ 스토리지 읽기 ■ 라이브러리 항목 동기화 ■ 구독 라이브러리 동기화 ■ 유형 검사 ■ 구성 설정 업데이트 ■ 파일 업데이트 ■ 라이브러리 업데이트 ■ 라이브러리 항목 업데이트 ■ 로컬 라이브러리 업데이트 ■ 구독 라이브러리 업데이트 ■ 구성 설정 보기
태그	<ul style="list-style-type: none"> ■ vSphere 태그 할당 또는 할당 취소 ■ vSphere 태그 생성 ■ vSphere 태그 범주 생성 ■ vSphere 태그 삭제 ■ vSphere 태그 범주 삭제 ■ vSphere 태그 편집 ■ vSphere 태그 범주 편집 ■ 범주에 대한 UsedBy 필드 수정 ■ 태그에 대한 UsedBy 필드 수정

표 2-4. vSphere 에이전트가 vCenter Server 인스턴스를 관리하는 데 필요한 사용 권한 (계속)

특성 값	사용 권한
vApp	<ul style="list-style-type: none"> ■ 가져오기 ■ vApp 애플리케이션 구성 <p>OVF 템플릿에 대해 그리고 콘텐츠 라이브러리에서 VM을 프로비저닝하려면 vApp.Import 애플리케이션 구성이 필요합니다.</p> <p>클라우드 구성 스크립팅에 대해 cloud-init를 사용할 때 vApp.vApp 애플리케이션 구성이 필요합니다. 이 설정을 사용하면 제품 정보 및 속성 같은 vApp의 내부 구조를 수정할 수 있습니다.</p>
가상 시스템 인벤토리	<ul style="list-style-type: none"> ■ 기존 항목에서 생성 ■ 새로 생성 ■ 이동 ■ 제거
가상 시스템 상호 작용	<ul style="list-style-type: none"> ■ CD 미디어 구성 ■ 콘솔 상호 작용 ■ 디바이스 연결 ■ 전원 끄기 ■ 전원 켜기 ■ 재설정 ■ 일시 중단 ■ 도구 설치
가상 시스템 - 구성	<ul style="list-style-type: none"> ■ 기존 디스크 추가 ■ 새 디스크 추가 ■ 디스크 제거 ■ 고급 ■ CPU 수 변경 ■ 리소스 변경 ■ 가상 디스크 확장 ■ 디스크 변경 내용 추적 ■ 메모리 ■ 디바이스 설정 수정 ■ 이름 바꾸기 ■ 주석 설정 ■ 설정 ■ 스왑 파일 배치

표 2-4. vSphere 에이전트가 vCenter Server 인스턴스를 관리하는 데 필요한 사용 권한 (계속)

특성 값	사용 권한
가상 시스템 - 프로비저닝	<ul style="list-style-type: none"> ■ 사용자 지정 ■ 템플릿 복제 ■ 가상 시스템 복제 ■ 템플릿 배포 ■ 사용자 지정 규격 읽기
가상 시스템 상태	<ul style="list-style-type: none"> ■ 스냅샷 생성 ■ 스냅샷 제거 ■ 스냅샷으로 되돌리기

vRealize Automation Cloud Assembly에서 사용할 Microsoft Azure 구성

vRealize Automation Cloud Assembly에서 Microsoft Azure 클라우드 계정을 생성하려면 몇 가지 정보를 수집하고 구성을 수행해야 합니다.

절차

- 1 Microsoft Azure 구독 및 테넌트 ID를 찾아서 기록해둡니다.
 - 구독 ID - Azure Portal의 왼쪽 도구 모음에 있는 [구독] 아이콘을 클릭하여 구독 ID를 확인합니다.
 - 테넌트 ID - Azure Portal에서 [도움말] 아이콘을 클릭하고 [진단 표시]를 선택합니다. 테넌트를 검색하여 찾으려면 ID를 기록해둡니다.
- 2 새 스토리지 계정 및 리소스 그룹을 생성하여 시작할 수 있습니다. 또는, 나중에 Blueprint에서 생성할 수 있습니다.
 - 스토리지 계정 - 다음 절차를 사용하여 계정을 구성합니다.
 - 1 Azure Portal의 사이드바에서 스토리지 계정 아이콘을 찾습니다. 올바른 구독이 선택되었는지 확인하고 **추가**를 클릭합니다. Azure 검색 필드에서 스토리지 계정을 검색할 수도 있습니다.
 - 2 스토리지 계정에 대한 필수 정보를 입력합니다. 구독 ID가 필요합니다.
 - 3 기존 리소스 그룹을 사용할지 아니면 새로 생성할지 선택합니다. 리소스 그룹 이름을 기록해둡니다. 나중에 필요할 수 있습니다.

참고 스토리지 계정의 위치를 저장합니다. 나중에 필요할 수 있습니다.

- 3 가상 네트워크를 생성합니다. 또는 적합한 기존 네트워크가 있는 경우 해당 네트워크를 선택할 수 있습니다.

네트워크를 생성하는 경우 [기존 리소스 그룹 사용]을 선택하고 이전 단계에서 생성한 그룹을 지정해야 합니다. 또한 이전에 지정한 것과 동일한 위치를 선택합니다. 개체가 사용할 적용 가능한 모든 구성 요소 간에 위치가 일치하지 않으면 Microsoft Azure에서 가상 시스템이나 기타 개체가 배포되지 않습니다.

- a 왼쪽 패널에서 [가상 네트워크] 아이콘을 찾아서 클릭하거나 가상 네트워크를 검색합니다. 올바른 구독을 선택하고 **추가**를 클릭합니다.
 - b 새 가상 네트워크의 고유 이름을 입력하고 나중을 위해 기록해둡니다.
 - c **주소 공간** 필드에 가상 네트워크에 적합한 IP 주소를 입력합니다.
 - d 올바른 구독이 선택되었는지 확인하고 **추가**를 클릭합니다.
 - e 나머지 기본 구성 정보를 입력합니다.
 - f 필요에 따라 다른 옵션을 수정할 수 있지만 대부분의 구성에서 기본값을 그대로 두어도 됩니다.
 - g **생성**을 클릭합니다.
- 4 vRA가 인증할 수 있도록 Azure Active Directory 애플리케이션을 설정합니다.
- a Azure 왼쪽 메뉴에서 **Active Directory** 아이콘을 찾아서 클릭합니다.
 - b **애플리케이션 등록**을 클릭하고 **추가**를 선택합니다.
 - c Azure 이름 유효성 검사를 준수하는 애플리케이션 이름을 입력합니다.
 - d 웹앱/API를 애플리케이션 유형으로 둡니다.
 - e [로그온 URL]은 사용에 적합한 것이면 무엇이든 가능합니다.
 - f **생성**을 클릭합니다.
- 5 Cloud Assembly에서 애플리케이션을 인증하기 위한 비밀 키를 생성합니다.
- a Azure에서 애플리케이션 이름을 클릭합니다.
나중에 사용할 수 있도록 애플리케이션 ID를 기록해둡니다.
 - b 다음 창에서 **모든 설정**을 클릭하고 설정 목록에서 [키]를 선택합니다.
 - c 새 키에 대한 설명을 입력하고 기간을 선택합니다.
 - d **저장**을 클릭하고 키 값을 안전한 위치에 복사합니다. 키 값은 나중에 검색할 수 없습니다.
 - e 왼쪽 메뉴에서 애플리케이션에 대해 **API 사용 권한**을 선택하고 **사용 권한 추가**를 클릭하여 새 사용 권한을 생성합니다.
 - f [API 선택] 페이지에서 Azure 서비스 관리를 선택합니다.
 - g **위임된 사용 권한**을 클릭합니다.
 - h [사용 권한 선택]에서 user_impersonation을 선택하고 **사용 권한 추가**를 클릭합니다.

- 6 Active Directory 애플리케이션이 Azure 구독에 연결할 수 있도록 권한을 부여합니다. 그래야 가상 시스템을 배포하고 관리할 수 있습니다.
 - a 왼쪽 메뉴에서 구독 아이콘을 클릭하고 새 구독을 선택합니다.
이름 텍스트를 클릭해야 패널이 미끄러질 수도 있습니다.
 - b [액세스 제어(IAM)] 옵션을 선택하여 구독에 대한 사용 권한을 살펴봅니다.
 - c [역할 할당 추가] 머리글 아래에서 **추가**를 클릭합니다.
 - d [역할] 드롭다운에서 [참가자]를 선택합니다.
 - e [액세스 할당] 드롭다운의 기본 선택 항목을 그대로 둡니다.
 - f [선택] 상자에 애플리케이션의 이름을 입력합니다.
 - g **저장**을 클릭합니다.
 - h 다른 역할을 추가하여 새 애플리케이션에 소유자, 참가자 및 독자 역할이 포함되도록 합니다.
 - i **저장**을 클릭합니다.

다음에 수행할 작업

Microsoft Azure 명령줄 인터페이스 도구를 설치해야 합니다. 이러한 도구는 Windows 및 Mac 운영 체제 모두에서 무료로 사용할 수 있습니다. 이러한 도구의 다운로드 및 설치에 대한 자세한 내용은 Microsoft 설 명서를 참조하십시오.

명령줄 인터페이스가 설치되어 있으면 새 구독에 인증해야 합니다.

- 1 터미널 창을 열고 Microsoft Azure 로그인을 입력합니다. 인증할 수 있는 URL과 짧은 코드가 제공됩 니다.
- 2 브라우저에서 디바이스의 애플리케이션으로부터 받은 코드를 입력합니다.
- 3 인증 코드를 입력하고 **계속**을 클릭합니다.
- 4 Azure 계정 및 로그인을 선택합니다.
여러 구독이 있는 경우 `azure account set <subscription-name>` 명령을 사용하여 올바른 구독이 선택되도록 합니다.
- 5 계속 진행하기 전에 `azure provider register microsoft.compute` 명령을 사용하여 Microsoft.Compute 제공자를 새 Azure 구독에 등록해야 합니다.

명령이 시간 초과되고 처음 실행할 때 오류가 생성되면 다시 실행합니다.

구성을 완료하면 `azure vm image list` 명령을 사용하여 사용 가능한 가상 시스템 이미지 이름을 검색할 수 있습니다. 원하는 이미지를 선택하여 제공된 URN을 기록해두면 나중에 Blueprint에서 사용할 수 있습 니다.

vRealize Automation에서 Microsoft Azure 클라우드 계정 생성

클라우드 관리자는 팀에서 vRealize Automation Blueprint를 배포할 계정 지역에 대한 Microsoft Azure 클라우드 계정을 생성할 수 있습니다.

Microsoft Azure 클라우드 계정이 vRealize Automation에서 작동하는 방식에 대한 사용 사례 예시를 보려면 [WordPress 사용 사례](#)를 참조하십시오.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목](#)을 참조하십시오.
- 필요한 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- vRealize Automation에서 사용할 Microsoft Azure 계정을 구성합니다. [vRealize Automation Cloud Assembly에서 사용할 Microsoft Azure 구성의 내용](#)을 참조하십시오.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성합니다. [vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.
- 2 Microsoft Azure 계정 유형을 선택하고 자격 증명과 기타 값을 입력합니다.
- 3 **검증**을 클릭합니다.
계정과 연결된 계정 지역이 수집됩니다.
- 4 이 리소스를 프로비저닝할 지역을 선택합니다.
- 5 효율성을 위해 **선택한 지역에 대한 클라우드 영역 생성**을 클릭합니다.
- 6 태그 지정 전략을 지원하기 위해 태그를 추가해야 하는 경우에는 기능 태그를 입력합니다. 태그를 사용하여 [vRealize Automation Cloud Assembly](#) 리소스 및 배포를 관리하는 방법 및 태그 지정 전략 생성 항목을 참조하십시오.
- 7 **저장**을 클릭합니다.

결과

계정이 vRealize Automation에 추가되고 선택한 지역을 지정된 클라우드 영역에서 사용할 수 있습니다.

다음에 수행할 작업

이 클라우드 계정에 대한 인프라 리소스를 생성합니다.

vRealize Automation에서 Amazon Web Services 클라우드 계정 생성

클라우드 관리자는 팀에서 vRealize Automation Blueprint를 배포할 계정 지역에 대한 AWS(Amazon Web Services) 클라우드 계정을 생성할 수 있습니다.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 필요한 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 필요한 AWS 관리자 자격 증명이 있는지 확인합니다.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성합니다. [vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.
- 2 AWS 계정 유형을 선택하고 자격 증명과 기타 값을 입력합니다.
- 3 **검증**을 클릭합니다.
계정과 연결된 계정 지역이 수집됩니다.
- 4 이 리소스를 프로비저닝할 지역을 선택합니다.
- 5 효율성을 위해 **선택한 지역에 대한 클라우드 영역 생성**을 클릭합니다.
- 6 태그 지정 전략을 지원하기 위해 태그를 추가해야 하는 경우에는 기능 태그를 입력합니다. 태그를 사용하여 [vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#) 및 [태그 지정 전략 생성](#) 항목을 참조하십시오.
- 7 **추가**를 클릭합니다.

결과

계정이 vRealize Automation에 추가되고 선택한 지역을 지정된 클라우드 영역에서 사용할 수 있습니다.

다음에 수행할 작업

이 클라우드 계정에 대한 인프라 리소스를 구성합니다.

vRealize Automation에서 Google Cloud Platform 클라우드 계정 생성

클라우드 관리자는 팀에서 vRealize Automation Blueprint를 배포할 계정 지역에 대한 GCP(Google Cloud Platform) 클라우드 계정을 생성할 수 있습니다.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 필요한 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.

- Google Cloud Platform JSON 보안 키에 액세스할 수 있는지 확인합니다.
- Google Cloud Platform 인스턴스에 대한 필수 보안 정보가 있는지 확인합니다. 인스턴스에서 또는 Google 설명서에서 이러한 정보의 대부분을 얻을 수 있습니다.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성합니다. [vRealize Automation](#)에 대한 인터넷 프록시 서버를 구성하는 방법의 내용을 참조하십시오.

절차

1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.

2 Google Cloud Platform 계정 유형을 선택하고 적절한 자격 증명 및 관련 정보를 입력합니다. 소스 GCP 계정 계산 엔진을 초기화할 때 생성한 서비스 계정을 사용합니다.

위의 **사전 요구 사항** 섹션에 명시된 바와 같이, 자격 증명 요구 사항은 [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 자격 증명에서 사용할 수 있습니다. [vRealize Automation](#)에서 클라우드 계정을 성공적으로 생성하려면 소스 GCP 계정에 계산 엔진 서비스가 사용되도록 설정되어 있어야 합니다.

[vRealize Automation](#)에서 프로젝트 ID는 Google Cloud Platform 끝점의 일부입니다. 클라우드 계정을 생성할 때 이를 지정합니다. 프로젝트별 전용 이미지의 데이터를 수집하는 동안 [vRealize Automation](#) GCP 어댑터는 Google Cloud Platform API를 쿼리합니다.

3 **검증**을 클릭합니다.

계정과 연결된 계정 지역이 수집됩니다.

4 이 리소스를 프로비저닝할 지역을 선택합니다.

5 효율성을 위해 **선택한 지역에 대한 클라우드 영역 생성**을 클릭합니다.

6 태그 지정 전략을 지원하는 태그가 필요한 경우에는 기능 태그를 입력합니다. 태그를 사용하여 [vRealize Automation Cloud Assembly](#) 리소스 및 배포를 관리하는 방법 및 태그 지정 전략 생성 항목을 참조하십시오.

7 **추가**를 클릭합니다.

결과

계정이 [vRealize Automation](#)에 추가되고 선택한 지역을 지정된 클라우드 영역에서 사용할 수 있습니다.

다음에 수행할 작업

이 클라우드 계정에 대한 인프라 리소스를 생성합니다.

vRealize Automation Cloud Assembly에서 vCenter 클라우드 계정 생성

[vRealize Automation Cloud Assembly](#) Blueprint를 배포하려는 계정 영역에 대한 vCenter 클라우드 계정을 추가합니다.

네트워크 및 보안을 위해 NSX-T 또는 NSX-V 클라우드 계정을 vCenter 클라우드 계정에 연결할 수 있습니다.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 클라우드 계정 지원을 위해 포트와 프로토콜을 올바르게 구성했는지 확인합니다. [vRealize Automation 제품 설명서](#)의 "vRealize Easy Installer를 사용하여 vRealize Automation 설치"에서 "vRealize Automation의 포트 및 프로토콜" 항목 및 "vRealize Automation 참조 아키텍처 가이드"에서 "포트 요구 사항" 항목을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.
- 2 vCenter 계정 유형을 선택하고 vCenter Server 호스트 IP 주소를 입력합니다.
- 3 vCenter Server 관리자 자격 증명을 입력하고 **검증**을 클릭합니다.
계정과 연결된 데이터 센터가 수집됩니다.
- 4 이 클라우드 계정에 대한 프로비저닝을 허용하려면 지정된 vCenter Server에서 사용 가능한 데이터 센터를 하나 이상 선택합니다.
- 5 효율성을 위해, 선택한 데이터 센터에 프로비저닝을 위한 클라우드 영역을 생성합니다.
조직의 클라우드 전략에 따라 별도의 단계로 클라우드 영역을 생성할 수도 있습니다.
클라우드 영역에 대한 자세한 내용은 [vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기](#) 항목을 참조하십시오.
- 6 기존 NSX 클라우드 계정을 선택합니다.
NSX 계정을 지금 선택하거나 나중에 클라우드 계정을 편집할 때 선택할 수 있습니다.
NSX-V 클라우드 계정에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 NSX-V 클라우드 계정 생성](#) 항목을 참조하십시오.
NSX-T 클라우드 계정에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 NSX-T 클라우드 계정 생성](#) 항목을 참조하십시오.
- 7 태그 지정 전략을 지원하기 위해 태그를 추가하려면 기능 태그를 입력합니다.
태그를 지금 추가하거나 나중에 클라우드 계정을 편집할 때 추가할 수 있습니다. 태그 지정에 대한 자세한 내용은 [태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#) 항목을 참조하십시오.
- 8 **저장**을 클릭합니다.

결과

클라우드 계정이 추가되고 선택한 데이터 센터를 지정된 클라우드 영역에서 사용할 수 있습니다. 시스템 및 볼륨과 같은 수집된 데이터가 [인프라] 탭의 리소스 섹션에 나열됩니다.

다음에 수행할 작업

이 클라우드 계정에 대한 나머지 인프라 리소스를 구성합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly에서 NSX-V 클라우드 계정 생성

네트워크 및 보안을 위해 NSX-V 클라우드 계정을 생성하고 vCenter 클라우드 계정과 연결할 수 있습니다.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 이 NSX 클라우드 계정에서 사용할 vCenter 클라우드 계정이 있는지 확인합니다. [vRealize Automation Cloud Assembly에서 vCenter 클라우드 계정 생성](#) 항목을 참조하십시오.
- 클라우드 계정 지원을 위해 포트와 프로토콜을 올바르게 구성했는지 확인합니다. [vRealize Automation 제품 설명서](#)의 "vRealize Easy Installer를 사용하여 vRealize Automation 설치"에서 "vRealize Automation의 포트 및 프로토콜" 항목 및 "vRealize Automation 참조 아키텍처 가이드"에서 "포트 요구 사항" 항목을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.
- 2 NSX-V 계정 유형을 선택하고 NSX-V 호스트 IP 주소를 입력합니다.
- 3 NSX 관리자 자격 증명을 입력하고 **검증**을 클릭합니다.
계정과 연결된 자산이 수집됩니다.
NSX 호스트 IP 주소를 사용할 수 없으면 검증이 실패합니다.
- 4 가능한 경우, 이 NSX-V 계정과 연결하는 vCenter 클라우드 계정을 나타내는 vCenter 끝점을 선택합니다.
- 5 태그 지정 전략을 지원하기 위해 태그를 추가하려면 기능 태그를 입력합니다.
나중에 기능 태그를 추가하거나 제거할 수 있습니다. [태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#)의 내용을 참조하십시오.
- 6 **저장**을 클릭합니다.

다음에 수행할 작업

vCenter 클라우드 계정을 생성하거나 편집하여 NSX 클라우드 계정과 연결할 수 있습니다. [vRealize Automation Cloud Assembly에서 vCenter 클라우드 계정 생성](#)의 내용을 참조하십시오.

이 클라우드 계정에서 사용하는 데이터 센터에 사용할 하나 이상의 클라우드 영역을 생성하고 구성합니다. [vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기](#) 항목을 참조하십시오.

이 클라우드 계정에 대한 인프라 리소스를 구성합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly에서 NSX-T 클라우드 계정 생성

네트워크 및 보안을 위해 NSX-T 클라우드 계정을 생성하고 vCenter 클라우드 계정과 연결할 수 있습니다.

배포에서 Fault Tolerance 및 고가용성을 용이하게 하기 위해 각 NSX-T 데이터 센터 끝점은 NSX Manager 3개의 클러스터를 나타냅니다.

- vRealize Automation은 NSX Manager 중 하나를 가리킬 수 있습니다. 이 옵션을 사용하면 하나의 NSX Manager가 vRealize Automation에서 API 호출을 받습니다.
- vRealize Automation은 클러스터의 가상 IP를 가리킬 수 있습니다. 이 옵션을 사용하면 하나의 NSX Manager가 VIP의 제어를 담당합니다. 이 Manager는 vRealize Automation에서 API 호출을 수신합니다. 장애가 발생할 경우 클러스터의 다른 노드가 VIP의 제어를 담당하고 vRealize Automation에서 API 호출을 수신합니다.

VIP 구성에 대한 자세한 내용은 [VMware NSX-T Data Center 설명서](#)의 "NSX-T Data Center 설치 가이드"에서 "클러스터의 VIP(가상 IP) 주소 구성"을 참조하십시오.

- vRealize Automation은 로드 밸런서 VIP를 가리켜서 3개의 NSX Manager에 대한 호출을 로드 밸런싱할 수 있습니다. 이 옵션을 사용하면 3개의 모든 NSX Manager가 vRealize Automation에서 API 호출을 수신합니다.

타사 로드 밸런서 또는 NSX-T 로드 밸런서에서 VIP를 구성할 수 있습니다.

대규모 환경에서는 이 옵션을 사용하여 3개의 NSX Manager 간에 vRealize Automation API 호출을 분할하는 것이 좋습니다.

사전 요구 사항

- 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명](#) 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 이 NSX 클라우드 계정에서 사용할 vCenter 클라우드 계정이 있는지 확인합니다. [vRealize Automation Cloud Assembly에서 vCenter 클라우드 계정 생성](#) 항목을 참조하십시오.

- 클라우드 계정 지원을 위해 포트와 프로토콜을 올바르게 구성했는지 확인합니다. [vRealize Automation 제품 설명서](#)의 "vRealize Easy Installer를 사용하여 vRealize Automation 설치"에서 "vRealize Automation의 포트 및 프로토콜" 항목 및 "vRealize Automation 참조 아키텍처 가이드"에서 "포트 요구 사항" 항목을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭합니다.
- 2 NSX-T 계정 유형을 선택하고 NSX-T 끝점 관리자 인스턴스 또는 VIP에 대한 호스트 IP 주소를 입력합니다(위 참조).
- 3 NSX 관리자 자격 증명을 입력하고 **검증**을 클릭합니다.
계정과 연결된 자산이 수집됩니다.
NSX 호스트 IP 주소를 사용할 수 없으면 검증이 실패합니다.
- 4 사용 가능한 경우 이 NSX-T 클라우드 계정에 연결할 vCenter 클라우드 계정을 나타내는 vCenter 끝점을 선택합니다.
- 5 태그 지정 전략을 지원하기 위해 태그를 추가하려면 기능 태그를 입력합니다.
나중에 기능 태그를 추가하거나 제거할 수 있습니다. [태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#)의 내용을 참조하십시오.
- 6 **저장**을 클릭합니다.

다음에 수행할 작업

이 NSX 클라우드 계정에 연결할 vCenter 클라우드 계정을 생성하거나 편집할 수 있습니다. [vRealize Automation Cloud Assembly](#)에서 vCenter 클라우드 계정 생성의 내용을 참조하십시오.

이 클라우드 계정에서 사용하는 데이터 센터에 사용할 하나 이상의 클라우드 영역을 생성하고 구성합니다. [vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기](#) 항목을 참조하십시오.

이 클라우드 계정에 대한 인프라 리소스를 구성합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#)의 내용을 참조하십시오.

vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성

클라우드 관리자는 팀에서 vRealize Automation Blueprint를 배포할 계정 지역에 대한 VMware Cloud on AWS 클라우드 계정을 생성할 수 있습니다.

VMware Cloud on AWS는 vRealize Automation에서 고유한 구성 절차가 필요합니다. 클라우드 계정에 대한 API 토큰 값 설정 및 클라우드 프록시에 대한 게이트웨이 방화벽 규칙 설정을 비롯하여, VMware Cloud on AWS에 대해 vRealize Automation를 올바르게 구성하려면 [VMware Cloud on AWS 사용 사례 워크플로](#)를 참조하십시오.

사전 요구 사항

- vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 비롯해 필요한 VMware Cloud on AWS 관리자 자격 증명에 있는지 그리고 포트 443에서 HTTPS 액세스를 사용하도록 설정했는지 확인합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.](#)
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성합니다. [vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하고 **클라우드 계정 추가**를 클릭한 후 VMware Cloud on AWS 계정 유형을 선택합니다.
- 2 사용 가능한 SDDC에 액세스하려면 조직의 **VMC API 토큰**을 추가합니다.
새 토큰을 생성하거나 연결된 **API 토큰** 페이지에서 조직에 대한 기존 토큰을 사용할 수 있습니다. 자세한 내용은 [샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성](#)의 내용을 참조하십시오.
- 3 배포에 사용할 SDDC를 선택합니다.
NSX-V SDDC는 지원되지 않으며 목록에 표시되지 않습니다.
vCenter 및 NSX-T Manager IP 주소/FQDN 값은 SDDC를 기반으로 자동으로 채워집니다.
- 4 기본값인 cloudadmin@vmc.local이 아닌 경우 지정된 SDDC에 대한 vCenter 사용자 이름 및 암호를 입력합니다.
- 5 **검증**을 클릭하여 지정된 vCenter에 대한 액세스 권한을 확인하고 vCenter가 실행 중인지 확인합니다.
계정과 연결된 데이터 센터가 수집됩니다.
- 6 효율성을 위해 선택한 SDDC에 프로비저닝을 위한 클라우드 영역을 생성합니다.
조직의 클라우드 전략에 따라 별도의 단계로 클라우드 영역을 생성할 수도 있습니다.
- 7 태그 지정 전략을 지원하기 위해 태그를 추가해야 하는 경우에는 기능 태그를 입력합니다.
나중에 기능 태그를 추가하거나 제거할 수 있습니다. [태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#)의 내용을 참조하십시오.
- 8 **저장**을 클릭합니다.

결과

클라우드 계정이 추가되고 선택한 SDDC를 지정된 클라우드 영역에서 사용할 수 있습니다.

다음에 수행할 작업

VMware Cloud on AWS에 대해 vRealize Automation를 올바르게 구성하려면 [VMware Cloud on AWS 사용 사례](#)의 내용을 참조하십시오.

vRealize Automation 외부의 VMware Cloud on AWS에 대한 관련 정보는 [VMware Cloud on AWS 설명서](#)를 참조하십시오.

vRealize Automation를 다른 애플리케이션과 통합

통합을 통해 vRealize Automation에 외부 시스템을 추가할 수 있습니다.

통합에는 vRealize Orchestrator, 구성 관리 및 기타 외부 시스템(예: GitHub, Ansible, Puppet) 및 Infoblox와 같은 외부 IPAM 제공자가 포함됩니다.

참고 외부 인터넷에 액세스할 수 없으며 통합에 필요한 경우 인터넷 서버 프록시를 구성할 수 있습니다. [vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly에서 GitLab 및 GitHub 통합을 사용하는 방법

vRealize Automation Cloud Assembly는 소스 제어 하에 Blueprint 및 작업 스크립트를 관리할 수 있도록 GitLab 및 GitHub 저장소와의 통합을 지원합니다. 이 기능은 배포와 관련된 프로세스의 감사 및 책임을 용이하게 합니다.

vRealize Automation Cloud Assembly로 Git 통합을 설정하려면 지정된 모든 사용자에게 대한 액세스 권한으로 구성된 적절한 로컬 Git 저장소가 있어야 합니다. 또한 Blueprint를 Git에서 검색할 수 있도록 특정 구조로 저장해야 합니다. GitLab 또는 GitHub와의 통합을 생성하려면 Cloud Assembly에서 **인프라 > 연결 > 통합**을 선택한 다음 적절한 항목을 선택합니다. 대상 저장소에 대한 URL과 토큰이 필요합니다.

Git 통합이 기존 저장소에 구성되면 선택된 프로젝트와 연결된 모든 Blueprint를 자격 있는 사용자가 사용할 수 있게 됩니다. 이러한 Blueprint는 기존 배포와 함께 사용하거나 새 배포의 기초로 사용할 수 있습니다. 프로젝트를 추가할 때 Git에서 프로젝트가 저장되는 위치와 방법에 관한 몇 가지 속성을 선택해야 합니다.

vRealize Automation Cloud Assembly에서 직접 Git 저장소에 작업을 저장할 수 있습니다. 작업 스크립트 버전을 Git에 직접 지정하거나 vRealize Automation Cloud Assembly에서 버전을 생성할 수도 있습니다. vRealize Automation Cloud Assembly에서 작업 버전을 생성하면 자동으로 Git에 버전으로 저장됩니다. Blueprint는 조금 더 복잡합니다. vRealize Automation Cloud Assembly에서 Git 통합에 직접 추가할 수 없기 때문입니다. Git 인스턴스에 직접 저장해야 합니다. 그런 다음, vRealize Automation Cloud Assembly에서 Blueprint 관리 페이지로 작업할 때 Git에서 검색할 수 있습니다.

시작하기 전에

Blueprint를 GitLab 또는 GitHub에서 검색할 수 있도록 특정 구조로 생성하고 저장해야 합니다.

- GitLab에 올바르게 통합되도록 Blueprint를 구성하고 저장합니다. 유효한 Blueprint만 GitLab으로 가져올 수 있습니다.
 - Blueprint에 대해 지정된 폴더를 하나 이상 생성합니다.
 - 모든 Blueprint는 `blueprint.yaml` 파일 내에 저장되어야 합니다.
 - Blueprint 맨 위에 `name:` 및 `version:` 속성이 포함되어 있는지 확인합니다.
- 해당하는 저장소의 API 키를 추출합니다. Git 계정에서 오른쪽 상단 모서리에 있는 로그인을 선택하고 [설정] 메뉴로 이동합니다. **액세스 토큰**을 선택하고 토큰에 이름을 지정한 후 만료 날짜를 설정합니다. 그런 다음, API를 선택하고 토큰을 생성합니다. 결과 값을 복사하여 저장합니다.

Git 통합에 사용되는 모든 Blueprint에 대해 다음 지침을 준수해야 합니다.

- 각 Blueprint는 별도의 폴더에 상주해야 합니다.
- 모든 Blueprint의 이름을 `blueprint.yaml`로 지정해야 합니다.
- 모든 Blueprint YAML 파일은 `name` 및 `version` 필드를 사용해야 합니다.
- 올바른 Blueprint만 가져옵니다.
- Git에서 가져온 초안 Blueprint를 업데이트할 때 해당 콘텐츠가 최상위 버전의 콘텐츠와 다르다면 이후 동기화 시 초안이 업데이트되지 않고 새 버전이 생성됩니다. Blueprint를 업데이트하고 Git의 추가 동기화도 허용하려면 마지막 변경 후 새 버전을 생성해야 합니다.
- **vRealize Automation Cloud Assembly에서 GitLab Blueprint 통합 구성**
 이 절차에서는 저장소에서 Blueprint로 작업하고 지정된 프로젝트와 연결되어 있는 저장된 Blueprint를 자동으로 다운로드할 수 있도록 vRealize Automation Cloud Assembly에서 GitLab 통합을 구성하는 방법을 보여줍니다. GitLab에서 Blueprint를 사용하려면 적절한 GitLab 인스턴스에 대한 연결을 생성한 다음, 원하는 Blueprint를 해당 인스턴스에 저장해야 합니다.
- **vRealize Automation Cloud Assembly에서 GitHub 통합 구성**
 vRealize Automation Cloud Assembly에서 GitHub 클라우드 기반 저장소 호스팅 서비스를 통합할 수 있습니다.

vRealize Automation Cloud Assembly에서 GitLab Blueprint 통합 구성

이 절차에서는 저장소에서 Blueprint로 작업하고 지정된 프로젝트와 연결되어 있는 저장된 Blueprint를 자동으로 다운로드할 수 있도록 vRealize Automation Cloud Assembly에서 GitLab 통합을 구성하는 방법을 보여줍니다. GitLab에서 Blueprint를 사용하려면 적절한 GitLab 인스턴스에 대한 연결을 생성한 다음, 원하는 Blueprint를 해당 인스턴스에 저장해야 합니다.

GitLab 통합이 기존 저장소에 구성되면 선택된 프로젝트와 연결된 모든 Blueprint를 자격 있는 사용자가 사용할 수 있게 됩니다. 이러한 Blueprint는 기존 배포와 함께 사용하거나 새 배포의 기초로 사용할 수 있습니다. 프로젝트를 추가할 때 GitLab에서 프로젝트가 저장되는 위치와 방법에 관한 몇 가지 속성을 선택해야 합니다.

참고 새로운 또는 업데이트된 Blueprint를 vRealize Automation Cloud Assembly에서 Git 저장소로 푸시할 수 없습니다. 또한 새 Blueprint를 vRealize Automation Cloud Assembly에서 저장소로 푸시할 수 없습니다. Blueprint를 저장소에 추가하려면 개발자가 Git 인터페이스를 사용해야 합니다.

Git에서 가져온 초안 Blueprint를 업데이트할 때 해당 콘텐츠가 최상위 버전의 콘텐츠와 다르면 이후 동기화 시 초안이 업데이트되지 않고 새 버전이 생성됩니다. Blueprint를 업데이트하고 Git의 추가 동기화도 허용하려면 마지막 변경 후 새 버전을 생성해야 합니다.

GitLab에서 사용할 Blueprint를 설정하고 필요한 정보를 수집한 후에는 GitLab 인스턴스와 통합을 설정해야 합니다. 그런 다음, 지정된 Blueprint를 GitLab으로 가져올 수 있습니다. 이 절차에 대한 비디오 데모는 <https://www.youtube.com/watch?v=h0vqo63Sdgg>에서 볼 수 있습니다.

사전 요구 사항

- 해당하는 저장소의 API 키를 추출합니다. GitLab 계정에서 오른쪽 상단 모서리에 있는 로그인을 선택하고 [설정] 메뉴로 이동합니다. [액세스 토큰]을 선택하고 토큰에 이름을 지정한 후 만료 날짜를 설정합니다. 그런 다음, API를 선택하고 토큰을 생성합니다. 결과 값을 복사하여 저장합니다.

vRealize Automation Cloud Assembly로 Git 통합을 설정하려면 지정된 모든 사용자에게 대한 액세스 권한으로 구성된 적절한 로컬 Git 저장소가 있어야 합니다. 또한 Blueprint를 GitLab에서 검색할 수 있도록 특정 구조로 생성하고 저장해야 합니다.

- GitLab에 올바르게 통합되도록 Blueprint를 구성하고 저장합니다. 유효한 Blueprint만 GitLab으로 가져올 수 있습니다. vRealize Automation Cloud Assembly에서 GitLab 및 GitHub 통합을 사용하는 방법의 내용을 참조하십시오.

절차

- 1 vRealize Automation Cloud Assembly에서 GitLab 환경과 통합을 설정합니다.
 - a **인프라 > 통합 > 새로 추가**를 선택하고 GitLab을 선택합니다.
 - b GitLab 인스턴스에 대한 URL을 입력합니다. SaaS(Software as a Service) GitLab은 대부분의 경우 `gitlab.com`입니다.
 - c 지정된 GitLab 인스턴스에 대한 **토큰**(API 키라고도 함)을 입력합니다. GitLab 인스턴스에서 토큰을 추출하는 방법에 대한 자세한 내용은 위의 사전 요구 사항을 참조하십시오.
 - d 적절한 이름과 설명을 추가합니다.
 - e **검증**을 클릭하여 연결을 확인합니다.

- f 필요한 경우 기능 태그를 추가합니다. 자세한 내용은 [vRealize Automation Cloud Assembly에서 기능 태그 사용](#)의 내용을 참조하십시오.
 - g **추가**를 클릭합니다.
- 2 적절한 저장소에서 Blueprint를 수락하도록 GitLab 연결을 구성합니다.
- a **인프라통합**을 선택하고 적절한 GitLab 통합을 선택합니다.
 - b **프로젝트**를 선택합니다.
 - c **새 프로젝트**를 선택하고 프로젝트의 이름을 생성합니다.
 - d GitLab 내 **저장소** 경로를 입력합니다. 일반적으로 이는 저장소 이름에 추가된 기본 계정의 사용자 이름입니다.
 - e 사용하려는 적절한 GitLab **분기**를 입력합니다.
 - f 해당하는 경우 **폴더** 이름을 입력합니다. 비워 두면 모든 폴더를 사용할 수 있습니다.
 - g 적절한 **유형**을 입력합니다. 해당하는 경우 폴더 이름을 입력합니다. 비워두면 모든 폴더를 사용할 수 있습니다.
 - h **다음**을 클릭하여 저장소 추가를 완료합니다.
- 다음**을 클릭하면 자동화된 동기화 작업이 시작되어 Blueprint를 플랫폼으로 가져옵니다.
- 동기화 작업이 완료되면 Blueprint를 가져왔다는 메시지가 표시됩니다.

결과

이제 GitLab에서 Blueprint를 검색할 수 있습니다.

vRealize Automation Cloud Assembly에서 GitHub 통합 구성

vRealize Automation Cloud Assembly에서 GitHub 클라우드 기반 저장소 호스팅 서비스를 통합할 수 있습니다.

vRealize Automation Cloud Assembly에서 GitHub 통합을 구성하려면 올바른 GitHub 토큰이 필요합니다. 토큰 생성 및 찾기에 대한 자세한 내용은 [GitHub 설명서](#)를 참조하십시오.

사전 요구 사항

- GitHub에 액세스할 수 있어야 합니다.
- GitHub에 올바르게 통합되도록 Blueprint를 구성하고 저장합니다. 유효한 Blueprint만 GitHub로 가져올 수 있습니다. [vRealize Automation Cloud Assembly에서 GitLab 및 GitHub 통합을 사용하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.
- 2 GitHub를 선택합니다.
- 3 [GitHub 구성] 페이지에 필요한 정보를 입력합니다.

4 **검증**을 클릭하여 통합을 확인합니다.

5 태그 지정 전략을 지원하기 위해 태그를 추가해야 하는 경우에는 기능 태그를 입력합니다. 태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법 및 태그 지정 전략 생성 항목을 참조하십시오.

6 **추가**를 클릭합니다.

결과

GitHub는 vRealize Automation Cloud Assembly Blueprint에서 사용할 수 있습니다.

다음에 수행할 작업

이제 GitHub에서 Blueprint를 검색할 수 있습니다.

vRealize Automation에서 외부 IPAM 통합 지점 구성

제공자별 외부 IPAM 통합 지점을 생성하여 Blueprint 배포에 사용되는 IP 주소를 관리할 수 있습니다. 외부 IPAM 통합 지점을 사용하는 경우 IP 주소는 vRealize Automation가 아니라 지정된 IPAM 제공자에서 가져오고 이 제공자가 관리합니다.

vRealize Automation의 VM 및 Blueprint 배포에 대한 IP 주소 및 DNS 설정을 관리하기 위해 제공자별 IPAM 통합 지점을 생성할 수 있습니다.

사전 요구 사항을 구성하는 방법과 샘플 워크플로 컨텍스트 내에서 제공자별 외부 IPAM 통합 지점을 생성하는 방법에 대한 자세한 내용은 [vRealize Automation에서 외부 IPAM 통합 지점 추가](#) 항목을 참조하십시오.

외부 IPAM 파트너 및 벤더가 IPAM 솔루션을 vRealize Automation와 통합할 수 있도록 필요한 자산을 생성하는 방법에 대한 자세한 내용은 [IPAM SDK를 사용하여 vRealize Automation에 대한 제공자별 외부 IPAM 통합 패키지를 생성하는 방법](#) 항목을 참조하십시오.

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명](#) 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 IPAM 제공자(예: Infoblox 또는 Bluecat)의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- IPAM 제공자(예: Infoblox 또는 BlueCat)에 대한 배포된 통합 패키지에 액세스할 수 있는지 확인합니다. 배포된 패키지는 처음에 IPAM 제공자 또는 vRealize Automation 마켓플레이스에서 .zip 다운로드 후 vRealize Automation에 배포됩니다.
- IPAM 제공자에 대해 구성된 실행 환경에 액세스할 수 있는지 확인합니다.

- ABX(Action-Based Extensibility) 온-프레미스 내장형 실행 환경을 사용 중인 경우 vRealize Automation 네트워크에 송신 트래픽을 gcr.io 및 storage.googleapis.com과 같은 외부 사이트로 전달할 수 있는 HTTP 프록시 서버가 있는지 확인합니다. 자세한 내용은 [vRealize Automation 8.x의 프록시](#) 뒤에서 [Docker 이미지 가져오기\(75180\)](#)를 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.

- 2 **IPAM**을 클릭합니다.

- 3 **제공자** 드롭다운 목록에서 구성된 IPAM 제공자 패키지를 선택합니다.

목록이 비어 있으면 **제공자 패키지 가져오기**를 클릭하고 기존 제공자 패키지 .zip 파일로 이동한 후 선택합니다. .zip 파일이 없으면 제공자의 웹 사이트나 vRealize Automation **마켓플레이스** 탭에서 가져올 수 있습니다.

- 4 제공자의 호스트 이름과 같은 다른 필수 필드(있는 경우)와 함께 외부 IPAM 제공자의 계정에 대한 관리자 사용자 이름 및 암호 자격 증명을 입력합니다.

- 5 **실행 환경** 드롭다운 목록에서 기존 실행 환경(예: 온-프레미스 작업 기반 확장성 통합 지점)을 선택합니다.

실행 환경은 vRealize Automation과 IPAM 제공자 간의 통신을 지원합니다.

IPAM 프레임워크는 ABX(Action-Based Extensibility) 온-프레미스 내장형 실행 환경만 지원합니다.

참고 Amazon Web Services 또는 Microsoft Azure 클라우드 계정을 통합 실행 환경으로 사용하는 경우에는 IPAM 제공자 장치를 인터넷에서 액세스할 수 있고, NAT 또는 방화벽이 뒤에 있지 않으며, 공개적으로 확인할 수 있는 DNS 이름이 있는지 확인하십시오. IPAM 제공자에 액세스할 수 없는 경우 Amazon Web Services Lambda 또는 Microsoft Azure 함수가 연결할 수 없으며 통합이 실패합니다.

- 6 **검증**을 클릭합니다.

- 7 외부 IPAM 제공자의 자체 서명된 인증서를 신뢰하라는 메시지가 표시되면 **수락**을 클릭합니다.

자체 서명된 인증서를 수락하면 검증 작업을 계속하여 완료할 수 있습니다.

- 8 이 IPAM 통합 지점의 이름을 입력하고 **추가**를 클릭하여 새 IPAM 통합 지점을 저장합니다.

데이터 수집 작업은 모방됩니다. 네트워크 및 IP 주소는 외부 IPAM 제공자에서 데이터 수집됩니다.

vRealize Automation에서 최신 IPAM 통합 패키지로 업그레이드하는 방법

기존 외부 IPAM 통합 지점을 업그레이드하여 최신 버전의 벤더 특정 IPAM 통합 패키지를 받을 수 있습니다.

외부 IPAM 공급자나 VMware는 특정 벤더의 소스 IPAM 통합 패키지를 업그레이드할 수 있습니다. 예를 들어 Infoblox용 외부 IPAM 통합 패키지가 여러 번 업그레이드되었습니다. 명명된 IPAM 통합 지점을 사용하는 기존 vRealize Automation 인프라 설정을 유지하려면 새 IPAM 통합 지점을 생성하는 대신 IPAM 통합 지점을 편집하여 업데이트된 IPAM 통합 패키지를 소싱할 수 있습니다.

사전 요구 사항

이 절차에서는 이미 외부 IPAM 통합 지점을 생성했고 최신 버전의 벤더 특정 IPAM 통합 패키지를 사용하여 해당 통합 지점을 업그레이드하려는 경우를 가정합니다.

외부 IPAM 통합 지점 생성 방법에 대한 자세한 내용은 [vRealize Automation](#)에서 **외부 IPAM 통합 지점 추가** 항목을 참조하십시오.

- 클라우드 관리자 자격 증명이 있는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 **자격 증명** 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 IPAM 제공자의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- IPAM 제공자를 위해 배포된 통합 패키지에 액세스할 수 있는지 확인합니다. 배포된 패키지는 처음에 IPAM 제공자 웹 사이트 또는 [vRealize Automation](#) 마켓플레이스에서 .zip 다운로드로 가져온 후 [vRealize Automation](#)에 배포됩니다.

제공자 패키지 .zip 파일을 다운로드 및 배포하고 IPAM 통합 페이지에서 **제공자** 값으로 사용할 수 있도록 설정하는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly](#)에서 **사용할 외부 IPAM 제공자 패키지 다운로드 및 배포** 항목을 참조하십시오.

- IPAM 제공자에 대해 구성된 실행 환경에 액세스할 수 있는지 확인합니다. 실행 환경은 일반적으로 ABX(작업 기반 확장성) 온-프레미스 내장형 통합 지점입니다.

실행 환경 특성에 대한 자세한 내용은 [vRealize Automation](#)에서 **IPAM 통합 지점에 대한 실행 환경 생성** 항목을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합 IPAM**을 선택하고 기존 IPAM 통합 지점을 엽니다.
- 2 **제공자 관리**를 클릭합니다.
- 3 업데이트된 IPAM 통합 패키지로 이동하고 가져옵니다.
- 4 **검증**을 클릭한 후 **저장**을 클릭합니다.

vRealize Automation Cloud Assembly에서 My VMware 통합 구성

My VMware를 vRealize Automation Cloud Assembly와 통합하여 Blueprint를 위한 VMware Marketplace 액세스와 같은 VMware 관련 작업 및 기능을 지원할 수 있습니다.

조직별로 하나의 My VMware 통합만 생성할 수 있습니다.

사전 요구 사항

My VMware에 대한 적절한 사용 권한이 있는 사용자 계정이 있어야 합니다.

- 사용자를 My VMware 계정에 초대하는 것에 대한 자세한 내용은 [KB 2070555](#)를 참조하십시오.

- My VMware 계정에서 사용자 사용 권한을 할당하는 것에 대한 자세한 내용은 [KB 2006977](#)을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.
- 2 [My VMware]를 선택합니다.
- 3 My VMware 구성 페이지에 필요한 정보를 입력합니다.
- 4 태그 지정 전략을 지원하는 태그가 필요한 경우에는 기능 태그를 입력합니다. 태그를 사용하여 [vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#) 및 [태그 지정 전략 생성 항목](#)을 참조하십시오.
- 5 **추가**를 클릭합니다.

결과

My VMware를 Blueprint에서 사용할 수 있습니다.

다음에 수행할 작업

My VMware 구성 요소를 원하는 Blueprint에 추가합니다.

Cloud Assembly에서 vRealize Orchestrator 통합 구성

하나 이상의 vRealize Orchestrator 통합을 구성하여 확장성의 일부로 워크플로를 사용할 수 있습니다.

vRealize Automation에는 확장성 구독에 사용할 수 있는 미리 구성된 vRealize Orchestrator 인스턴스가 포함되어 있습니다. 또한 vRealize Automation 클라우드 서비스 콘솔에서 내장된 vRealize Orchestrator의 클라이언트에 액세스할 수도 있습니다.

vRealize Automation Cloud Assembly에 vRealize Orchestrator를 통합하면 외부 vRealize Orchestrator 인스턴스를 추가하고 포함된 워크플로 라이브러리를 확장성 구독에 사용할 수 있습니다. 자세한 내용은 [확장성 워크플로 구독 항목](#)을 참조하십시오.

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. 자세한 내용은 [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- vRealize Orchestrator 7.x 인스턴스를 버전 8.0으로 마이그레이션합니다. "VMware vRealize Orchestrator 설치, 구성 및 마이그레이션"에서 "독립형 vRealize Orchestrator를 외부 vRealize Orchestrator 8.0으로 마이그레이션"을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택합니다.
- 2 **통합 추가**를 클릭합니다.
- 3 vRealize Orchestrator를 선택합니다.

- 4 vRealize Automation Cloud Assembly에 vRealize Orchestrator의 URL을 입력합니다.
- 5 통합을 검증하려면 **검증**을 클릭합니다.
- 6 vRealize Orchestrator 통합의 이름을 입력합니다.
- 7 (선택 사항) vRealize Orchestrator 통합에 대한 설명을 입력합니다.
- 8 (선택 사항) 기능 태그를 추가합니다. 기능 태그에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 기능 태그 사용](#) 항목을 참조하십시오.

참고 기능 태그를 사용하여 여러 vRealize Orchestrator 통합을 관리할 수 있습니다. [프로젝트 제약 조건으로 여러 vRealize Orchestrator 통합 관리](#)의 내용을 참조하십시오.

9 추가를 클릭합니다.

vRealize Orchestrator 통합이 저장됩니다.

다음에 수행할 작업

통합이 구성되었고 워크플로가 추가되었는지 확인하려면 **확장성 > 라이브러리 > 워크플로**를 선택합니다.

프로젝트 제약 조건으로 여러 vRealize Orchestrator 통합 관리

프로젝트 제약 조건을 사용하여 워크플로 구독에서 사용되는 vRealize Orchestrator 통합을 관리할 수 있습니다.

vRealize Automation Cloud Assembly는 워크플로 구독에서 사용할 수 있는 여러 vRealize Orchestrator 서버의 통합을 지원합니다. 연성 또는 경성 프로젝트 제약 조건으로 프로젝트에서 프로비저닝된 Blueprint에 사용되는 vRealize Orchestrator 통합을 관리할 수 있습니다. 프로젝트 제약 조건에 대한 자세한 내용은 [vRealize Automation Cloud Assembly 프로젝트 태그 및 사용자 지정 속성 사용](#) 항목을 참조하십시오.

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. 자세한 내용은 [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- vRealize Automation Cloud Assembly에서 둘 이상의 vRealize Orchestrator 통합을 구성합니다. 자세한 내용은 [Cloud Assembly에서 vRealize Orchestrator 통합 구성](#) 항목을 참조하십시오.
- vRealize Orchestrator 통합에 기능 태그를 추가합니다. 기능 태그에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 기능 태그 사용](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 프로젝트**로 이동하여 프로젝트를 선택합니다.
- 2 **프로비저닝** 탭을 선택합니다.
- 3 **확장성 제약 조건** 텍스트 상자에 vRealize Orchestrator 통합의 기능 태그를 입력하고 연성 또는 경성 프로젝트 제약 조건으로 설정합니다.

4 저장을 클릭합니다.

결과

Blueprint를 배포하는 경우, vRealize Automation Cloud Assembly는 프로젝트 제약 조건을 사용하여 워크플로 구독에서 사용되는 vRealize Orchestrator 통합을 관리합니다.

다음에 수행할 작업

또는 기능 태그를 사용하여 클라우드 계정 수준에서 여러 vRealize Orchestrator 통합을 관리할 수 있습니다. 자세한 내용은 [클라우드 계정 기능 태그를 사용하여 여러 vRealize Orchestrator 통합 관리](#) 항목을 참조하십시오.

클라우드 계정 기능 태그를 사용하여 여러 vRealize Orchestrator 통합 관리

기능 태그를 사용하여 워크플로 구독에서 사용되는 vRealize Orchestrator 통합을 관리할 수 있습니다.

vRealize Automation Cloud Assembly는 워크플로 구독에서 사용할 수 있는 여러 vRealize Orchestrator 서버의 통합을 지원합니다. 클라우드 계정에 기능 태그를 추가하여 워크플로 구독에 사용되는 vRealize Orchestrator 통합을 관리할 수 있습니다.

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. 자세한 내용은 [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- vRealize Automation Cloud Assembly에서 둘 이상의 vRealize Orchestrator 통합을 구성합니다. 자세한 내용은 [Cloud Assembly에서 vRealize Orchestrator 통합 구성](#) 항목을 참조하십시오.
- vRealize Orchestrator 통합에 기능 태그를 추가합니다. 기능 태그에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 기능 태그 사용](#) 항목을 참조하십시오.

절차

1 **인프라 > 연결 > 클라우드 계정**으로 이동합니다.

2 클라우드 계정을 선택합니다.

3 사용하려는 vRealize Orchestrator 통합의 기능 태그를 입력합니다.

기능 태그가 연성 제약 조건으로 자동 변환됩니다. 통합을 관리하는 데 경성 제약 조건을 사용하려면 프로젝트 제약 조건을 사용해야 합니다. 자세한 내용은 [프로젝트 제약 조건으로 여러 vRealize Orchestrator 통합 관리](#) 항목을 참조하십시오.

4 **저장**을 클릭합니다.

결과

Blueprint를 배포하는 경우 vRealize Automation Cloud Assembly는 연결된 클라우드 계정의 태그 지정을 사용하여 워크플로 구독에서 사용되는 vRealize Orchestrator 통합을 관리합니다.

vRealize Automation Cloud Assembly에서 Kubernetes를 사용하는 방법

PKS(Pivotal Container Service) 또는 Red Hat OpenShift를 vRealize Automation Cloud Assembly와 통합하여 Kubernetes 리소스를 관리하고 배포할 수 있습니다. vRealize Automation Cloud Assembly에서 외부 Kubernetes 리소스를 통합할 수도 있습니다.

PKS 또는 OpenShift 통합을 생성하면 적용 가능한 Kubernetes 클러스터를 vRealize Automation Cloud Assembly에서 사용할 수 있고, Kubernetes 구성 요소를 vRealize Automation Cloud Assembly에 추가 및 생성하여 클러스터 및 컨테이너 애플리케이션의 관리를 지원할 수 있습니다. 이러한 애플리케이션은 Service Broker 카탈로그에서 사용할 수 있는 셀프 서비스 배포의 기반을 형성합니다.

■ vRealize Automation Cloud Assembly에서 PKS 통합 구성

온-프레미스 및 클라우드에서 PKS 리소스 연결을 구성하여 vRealize Automation Cloud Assembly의 PKS 및 Kubernetes 통합 및 관리 기능을 지원할 수 있습니다.

■ vRealize Automation Cloud Assembly에서 Kubernetes 클러스터 및 네임스페이스 사용

vRealize Automation Cloud Assembly에서 Kubernetes 배포의 기반이 되는 Kubernetes 클러스터 및 네임스페이스의 구성을 추가하고, 보고, 관리할 수 있습니다.

■ vRealize Automation Cloud Assembly에서 Kubernetes 영역 구성

Kubernetes 영역을 사용하면 클라우드 관리자가 vRealize Automation Cloud Assembly 배포에 사용되는 Kubernetes 클러스터 및 네임스페이스의 정책 기반 배치를 정의할 수 있습니다. 관리자는 이 페이지를 사용하여 Kubernetes 네임스페이스의 프로비저닝에 사용할 수 있는 클러스터를 지정하고 클러스터에 허용되는 속성을 지정할 수 있습니다.

■ vRealize Automation Cloud Assembly에서 Blueprint에 Kubernetes 구성 요소 추가

Kubernetes 구성 요소를 vRealize Automation Cloud Assembly Blueprint에 추가할 때 클러스터를 추가하거나 사용자가 다양한 구성으로 네임스페이스를 생성하도록 선택할 수 있습니다. 일반적으로 이러한 선택은 액세스 제어 요구 사항, Kubernetes 구성 요소 구성 방법 및 배포 요구 사항에 따라 결정됩니다.

■ Kubernetes와 vRealize Automation Cloud Assembly 확장성 사용

vRealize Automation Cloud Assembly는 Kubernetes 클러스터 배포와 관련된 일반적인 작업에 해당하는 표준 이벤트 항목 집합을 제공합니다. 사용자는 이러한 항목을 원하는 대로 구독할 수 있으며 구독한 항목과 관련된 이벤트가 발생하면 알림을 받습니다. 이벤트 알림을 기반으로 실행되도록 vRO 워크플로를 구성할 수도 있습니다.

vRealize Automation Cloud Assembly에서 PKS 통합 구성

온-프레미스 및 클라우드에서 PKS 리소스 연결을 구성하여 vRealize Automation Cloud Assembly의 PKS 및 Kubernetes 통합 및 관리 기능을 지원할 수 있습니다.

PKS 통합을 통해 온-프레미스 및 클라우드의 PKS 인스턴스와 PKS 및 외부 클러스터에 프로비저닝된 Kubernetes 클러스터를 관리할 수 있습니다. 정책 기반 리소스 배치를 지원하려면 Kubernetes 프로파일을 생성하여 프로젝트와 연결해야 합니다.

사전 요구 사항

- UAA 인증을 사용하여 적절히 구성된 PKS(Pivotal Container Service) 서버를 구성해야 합니다.
- 클라우드 관리자 자격 증명이 있는지 확인합니다. 자세한 내용은 [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.
- 2 VMware Enterprise PKS를 선택합니다.
- 3 생성하는 PKS 클라우드 계정에 대한 IP 주소 또는 FQDN과 PKS 주소를 입력합니다.
 - IP 주소는 PKS 사용자 인증 서버의 FQDN이나 IP 주소입니다.
 - PKS 주소는 기본 PKS 서버의 FQDN이나 IP 주소입니다.
- 4 PKS 서버가 로컬인지 아니면 공용 클라우드 또는 사설 클라우드에 있는지 선택합니다.
- 5 PKS 서버 및 기타 관련 정보에 대해 적절한 **사용자 이름** 및 **암호**를 입력합니다.
- 6 태그 지정 전략을 지원하는 태그를 사용하는 경우에는 기능 태그를 입력합니다. 태그를 사용하여 [vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법](#) 및 [태그 지정 전략 생성](#) 항목을 참조하십시오.
- 7 **추가**를 클릭합니다.

결과

Kubernetes 영역을 생성하여 프로젝트에 할당하거나 외부 Kubernetes 클러스터를 검색하여 프로젝트에 할당할 수 있습니다. 또한 대규모 그룹과 조직 간의 클러스터 관리에 유용한 Kubernetes 네임스페이스를 추가하거나 생성할 수 있습니다.

다음에 수행할 작업

적절한 Kubernetes 영역을 생성하거나 선택한 다음 클러스터 또는 네임스페이스를 하나 이상 선택하여 프로젝트에 할당합니다. 그런 다음 Blueprint를 생성하고 게시하면 Kubernetes를 사용하는 셀프 서비스 배포를 사용자가 생성할 수 있도록 설정됩니다.

vRealize Automation Cloud Assembly에서 Kubernetes 클러스터 및 네임스페이스 사용

vRealize Automation Cloud Assembly에서 Kubernetes 배포의 기반이 되는 Kubernetes 클러스터 및 네임스페이스의 구성을 추가하고, 보고, 관리할 수 있습니다.

인프라 > 리소스 > Kubernetes 페이지에서 액세스 권한이 있는 Kubernetes 클러스터 및 네임스페이스를 보고, 추가하고, 관리할 수 있습니다. 일반적으로 이 페이지에서는 배포된 클러스터 및 네임스페이스를 쉽게 관리할 수 있습니다.

- **클러스터:** 클러스터는 하나 이상의 물리적 시스템에서 분산된 Kubernetes 노드 그룹입니다. 이 페이지에는 vRealize Automation Cloud Assembly 인스턴스에서 사용하도록 구성된 프로비저닝된 클러스터와 배포 해제된 클러스터가 표시됩니다. 클러스터를 클릭하면 현재 상태에 대한 정보를 볼 수 있습니다. 클러스터를 배포하는 경우 클라우드 관리자만 액세스할 수 있는 **Kubconfig** 파일에 대한 링크가 포함됩니다. 이 파일은 네임스페이스 목록을 포함하여 클러스터에 대한 전체 관리자 권한을 부여합니다.
- **네임스페이스:** 네임스페이스는 관리자가 클러스터 리소스를 분리할 수 있는 방법을 제공하는 가상 클러스터입니다. 대규모 사용자 및 조직 그룹 간의 리소스 관리를 용이하게 합니다. 역할 기반 액세스 제어의 한 가지 형태로, 클라우드 관리자는 사용자가 배포를 요청하면 프로젝트에 네임스페이스를 추가할 수 있게 한 다음, 해당 네임스페이스를 나중에 Kubernetes 클러스터 페이지에서 관리할 수 있습니다. 네임스페이스를 배포할 때 **Kubconfig** 파일에 대한 링크가 포함됩니다. 이 링크를 통해 유효한 사용자(예: 개발자)가 해당 네임스페이스의 일부 측면을 보고 관리할 수 있습니다.

새 클러스터 또는 기존 클러스터를 구성하는 경우 마스터 IP 주소와 마스터 호스트 이름 중 어디에 연결할지를 선택해야 합니다.

vRealize Automation Cloud Assembly에서 Kubernetes 클러스터 사용

이 페이지의 옵션을 사용하여 새 클러스터, 기존 클러스터 또는 외부 클러스터를 vRealize Automation Cloud Assembly에 추가할 수 있습니다.

1 인프라 > 리소스 > Kubernetes를 선택하고 [클러스터] 탭이 활성화 상태인지 확인합니다.

vRealize Automation Cloud Assembly 인스턴스에 대해 현재 구성된 클러스터가 있으면 이 페이지에 표시됩니다.

2 신규 또는 기존 클러스터를 추가하거나 클러스터를 배포하는 경우 다음 표에 따라 적절한 옵션을 선택합니다.

옵션	설명	세부 정보
배포	vRealize Automation Cloud Assembly에 새 클러스터 추가	이 클러스터가 배포될 PKS 클라우드 계정과 원하는 계획 및 노드 수를 지정해야 합니다.
기존 추가	프로젝트에서 작동하도록 기존 클러스터를 구성합니다.	PKS 클라우드 계정, 사용할 클러스터 및 대상 개발자를 위한 적절한 프로젝트를 지정해야 합니다. 공유 범위도 지정해야 합니다. 전역적으로 공유하려면 Kubernetes 영역 및 네임스페이스를 적절하게 구성해야 합니다.
외부 추가	PKS와 연결되지 않을 수 있는 vanilla Kubernetes 클러스터를 vRealize Automation Cloud Assembly에 추가합니다.	클러스터가 연결된 프로젝트를 지정하고 원하는 클러스터의 IP 주소를 입력한 후 이 클러스터에 연결하는 데 필요한 클라우드 프록시 및 인증서 정보를 선택해야 합니다.

3 추가를 클릭하여 vRealize Automation Cloud Assembly 내에서 클러스터를 사용할 수 있도록 만듭니다.

vRealize Automation Cloud Assembly에서 Kubernetes 네임스페이스 사용

클라우드 관리자가 네임스페이스를 사용하면 Kubernetes 클러스터 리소스를 그룹화하고 관리하는 데 유용합니다. 사용자에게 네임스페이스는 배포를 위한 Kubernetes 클러스터의 영역입니다. 관리자와 사용자는 **인프라 > 리소스 > Kubernetes** 페이지에 있는 [네임스페이스] 탭을 사용하여 네임스페이스에 액세스할 수 있습니다.

vRealize Automation Cloud Assembly의 리소스에 Kubernetes 네임스페이스를 추가하는 방법이 여러 가지 있습니다. 다음 절차는 일반적인 방법 중 하나를 요약한 것입니다.

- 1 **인프라 > 리소스 > Kubernetes**를 선택하고 [네임스페이스] 탭을 클릭합니다.
- 2 새 네임스페이스를 추가하려면 **새 네임스페이스**를 클릭합니다. 기존 네임스페이스를 추가하려면 **네임스페이스 추가**를 클릭합니다.
- 3 네임스페이스에 대한 **이름**과 **설명**을 입력합니다.
이 시점에서 Kubernetes 리소스에 사용할 네임스페이스가 추가되었지만 특정 항목에 연결되어 있지는 않습니다.
- 4 이 네임스페이스에 연결할 **클러스터**를 지정합니다.
- 5 **생성**을 클릭하여 vRealize Automation Cloud Assembly에 네임스페이스를 추가합니다.

vRealize Automation Cloud Assembly에서 Kubernetes 영역 구성

Kubernetes 영역을 사용하면 클라우드 관리자가 vRealize Automation Cloud Assembly 배포에 사용되는 Kubernetes 클러스터 및 네임스페이스의 정책 기반 배치를 정의할 수 있습니다. 관리자는 이 페이지를 사용하여 Kubernetes 네임스페이스의 프로비저닝에 사용할 수 있는 클러스터를 지정하고 클러스터에 허용되는 속성을 지정할 수 있습니다.

클라우드 관리자는 Kubernetes 영역을 Cloud Assembly에 대해 구성된 PKS 클라우드 계정 또는 프로젝트와 연결되지 않은 외부 Kubernetes 클러스터와 연결할 수 있습니다.

Kubernetes 영역을 생성할 때 제공자별 리소스 여러 개를 영역에 할당할 수 있으며, 이러한 리소스는 작업자, 마스터, 사용 가능한 CPU, 메모리의 수와 기타 구성 설정 측면에서 새로 프로비저닝된 클러스터에 설정할 수 있는 속성을 나타냅니다. PKS 제공자의 경우 PKS 계획에 해당합니다. 관리자는 여러 클러스터를 Kubernetes 영역에 할당하여 새로 프로비저닝된 Kubernetes 네임스페이스를 배치하는 데 사용할 수도 있습니다. 관리자는 등록되지 않았거나 CMX에 의해 관리되지 않고 미리 선택된 클러스터 제공자를 통해 프로비저닝된 클러스터만 할당할 수 있습니다. 관리자는 여러 Kubernetes 영역을 단일 프로젝트에 할당하여 해당 프로젝트 내에서 발생하는 배치 작업에 모두 사용할 수 있습니다.

클라우드 관리자는 여러 수준에 우선 순위를 지정할 수 있습니다.

- 프로젝트 내 Kubernetes 영역 우선 순위.
- Kubernetes 영역 내 리소스 우선 순위.
- Kubernetes 영역 내 클러스터 우선 순위.

클라우드 관리자는 여러 수준에 태그를 할당할 수도 있습니다.

- Kubernetes 영역별 기능 태그.

- 리소스 할당별 태그.
- 클러스터 할당별 태그.

Service Broker에는 Service Broker 관리자가 카탈로그에서 프로비저닝된 Kubernetes 네임스페이스 및 클러스터에 대한 배치 정책을 생성할 수 있도록 기존 Kubernetes 영역에 액세스할 수 있게 하는 Kubernetes 영역 페이지 버전이 포함되어 있습니다.

사전 요구 사항

적절한 PKS 배포와의 통합을 구성합니다. [vRealize Automation Cloud Assembly](#)에서 **PKS 통합 구성** 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > Kubernetes 영역**을 선택한 다음, **새 Kubernetes 영역**을 클릭합니다.
- 2 이 영역을 적용할 PKS 통합 **계정**의 이름을 입력합니다.
- 3 Kubernetes 영역의 **이름**과 **설명**을 추가합니다.
- 4 필요한 경우 기능 태그를 추가합니다. 자세한 내용은 [vRealize Automation Cloud Assembly](#)에서 **기능 태그 사용**의 내용을 참조하십시오.
- 5 **저장**을 클릭합니다.
- 6 [주문형] 탭을 클릭하고 클러스터 프로비저닝에 사용할 영역에 대해 **PKS 계획**을 적절하게 추가합니다.

하나 이상의 계획을 선택하여 우선 순위를 할당할 수 있습니다. 숫자가 낮을수록 우선 순위가 높습니다. 우선 순위 할당은 태그 기반 선택에 부수적입니다.
- 7 [클러스터] 탭을 클릭한 다음, **추가** 버튼을 클릭하여 Kubernetes 클러스터를 영역에 추가합니다. 외부 클러스터로 작업하는 경우 선택하면 [vRealize Automation Cloud Assembly](#)에 자동으로 등록됩니다.

[vRealize Automation Cloud Assembly](#)의 [Kubernetes 클러스터] 페이지에서 클러스터에 Kubernetes 네임스페이스를 추가할 수 있습니다.

결과

Kubernetes 영역은 [vRealize Automation Cloud Assembly](#) 배포에서 사용하도록 구성됩니다.

다음에 수행할 작업

프로젝트에 Kubernetes 영역을 할당합니다.

- 1 **인프라 > 구성 > 프로젝트**를 선택한 다음 Kubernetes 영역에 연결할 프로젝트를 선택합니다.
- 2 [프로젝트] 페이지에서 [Kubernetes 프로비저닝] 탭을 클릭합니다.
- 3 **Kubernetes 영역 추가**를 클릭하고 방금 생성한 영역을 추가합니다. 해당되는 경우 여러 영역을 사용할 수 있으며 영역의 우선 순위도 설정할 수 있습니다.
- 4 **저장**을 클릭합니다.

영역을 프로젝트에 할당한 후에는 Blueprint 페이지를 사용하여 Kubernetes 영역 및 프로젝트 구성을 기반으로 배포를 프로비저닝할 수 있습니다.

vRealize Automation Cloud Assembly에서 Blueprint에 Kubernetes 구성 요소 추가

Kubernetes 구성 요소를 vRealize Automation Cloud Assembly Blueprint에 추가할 때 클러스터를 추가하거나 사용자가 다양한 구성으로 네임스페이스를 생성하도록 선택할 수 있습니다. 일반적으로 이러한 선택은 액세스 제어 요구 사항, Kubernetes 구성 요소 구성 방법 및 배포 요구 사항에 따라 결정됩니다.

vRealize Automation Cloud Assembly에서 Blueprint에 Kubernetes 구성 요소를 추가하려면 Blueprint를 클릭하고 **새로 만들기**를 선택한 다음, 왼쪽 메뉴에서 **Kubernetes** 옵션을 찾아서 확장합니다. 그런 다음 원하는 항목(클러스터 또는 KBS 네임스페이스 중 하나)을 캔버스로 끌어서 선택합니다.

프로젝트와 연결된 Kubernetes 클러스터를 Blueprint에 추가하는 것은 Kubernetes 리소스를 유효한 사용자가 사용할 수 있게 만드는 가장 간단한 방법입니다. 다른 Cloud Assembly 리소스와 마찬가지로 클러스터에서 태그를 사용하여 배포 위치를 제어할 수 있습니다. 클러스터 배포의 할당 단계에서 태그를 사용하여 영역 및 PKS 계획을 선택할 수 있습니다.

이러한 방식으로 클러스터를 추가하면 유효한 모든 사용자가 자동으로 사용할 수 있게 됩니다.

Blueprint 예시

첫 번째 Blueprint 예시는 태그 지정으로 제어되는 간단한 Kubernetes 배포에 대한 Blueprint를 보여줍니다. Kubernetes 영역은 [새 Kubernetes 영역] 페이지에서 구성된 두 가지 배포 계획으로 생성되었습니다. 이 경우, placement:tag라는 태그가 영역의 기능으로 추가되었으며 Blueprint의 유사한 제약 조건을 매칭하는 데 사용되었습니다. 태그로 구성된 영역이 둘 이상인 경우, 우선 순위 번호가 가장 낮은 영역이 선택됩니다.

```
formatVersion: 1
inputs: {}
resources:
  Cluster_provisioned_from_tag:
    type: Cloud.K8S.Cluster
    properties:
      hostname: 109.129.209.125
      constraints:
        -tag: 'placement tag'
        port: 7003
        workers: 1
        connectBy: hostname
```

두 번째 Blueprint 예시는 배포를 요청할 때 사용자가 원하는 클러스터 호스트 이름을 입력할 수 있도록 `$(input.hostname)`이라는 변수를 사용하여 Blueprint를 설정하는 방법을 보여줍니다. 태그는 클러스터 배포의 리소스 할당 단계에서 영역과 PKS 계획을 선택하는 데 사용할 수도 있습니다.

```
formatVersion: 1
inputs:
  hostname:
    type: string
    title: Cluster hostname
resources:
```

```
Cloud_K8S_Cluster_1:
  type: Cloud.K8S.Cluster
  properties:
    hostname: ${input.hostname}
    port: 8443
    connectBy: hostname
    workers: 1
```

네임스페이스를 사용하여 클러스터 사용량을 관리하려는 경우, 배포를 요청할 때 사용자가 입력하는 네임스페이스 이름을 대체하도록 Blueprint에 `name: ${input.name}`이라는 변수를 설정할 수 있습니다. 이러한 종류의 배포에는 다음 예시와 같은 Blueprint를 생성합니다.

```
1 formatVersion: 1
2 inputs:
3 name:
4   type: string
5   title: "Namespace name"
6 resources:
7   Cloud_K8S_Namespace_1:
8     type: Cloud.K8S.Namespace
9     properties:
10      name: ${input.name}
```

사용자는 **인프라 > 리소스 > Kubernetes 클러스터** 페이지에서 액세스할 수 있는 kubeconfig 파일을 통해 배포된 클러스터를 관리할 수 있습니다. 원하는 클러스터의 페이지에서 카드를 찾은 후 **Kubeconfig**를 클릭합니다.

Kubernetes와 vRealize Automation Cloud Assembly 확장성 사용

vRealize Automation Cloud Assembly는 Kubernetes 클러스터 배포와 관련된 일반적인 작업에 해당하는 표준 이벤트 항목 집합을 제공합니다. 사용자는 이러한 항목을 원하는 대로 구독할 수 있으며 구독한 항목과 관련된 이벤트가 발생하면 알림을 받습니다. 이벤트 알림을 기반으로 실행되도록 vRO 워크플로를 구성할 수도 있습니다.

다음 항목은 vRealize Automation Cloud Assembly의 **확장성 > 라이브러리 > 이벤트 항목** 페이지에서 구독할 수 있습니다. 이러한 항목을 보려면 이벤트 항목 검색 텍스트 상자에서 **Kubernetes**를 검색합니다.

- Kubernetes 클러스터 할당
- Kubernetes 클러스터 사후 프로비저닝
- Kubernetes 클러스터 사후 제거
- Kubernetes 클러스터 프로비저닝
- Kubernetes 클러스터 제거

항목 중 하나를 클릭하면 수집 및 전송되는 모든 정보를 표시하는 해당 항목에 대한 스키마를 볼 수 있습니다. 이 스키마 정보를 사용하여 다양한 알림 및 관리 및 보고 작업을 설정할 수 있습니다.

확장성 > 라이브러리 > 작업 페이지에서 CMX 관련 작업에 대한 작업 스크립트를 설정할 수 있습니다. 작업 스크립트는 다양한 용도로 사용할 수 있습니다(예: Kubernetes 클러스터 프로비저닝의 DNS 레코드 생성). DNS 레코드를 생성하는 경우에는 Kubernetes 클러스터 사후 프로비저닝 항목의 `masternodeips` 필드를 작업 스크립트에서 REST 명령과 함께 사용하여 DNS 레코드를 생성할 수 있습니다.

[구독] 페이지에서는 이벤트 항목과 작업 스크립트 간의 관계를 정의합니다. vRealize Automation Cloud Assembly의 [구독] 페이지에서 이러한 구성 요소를 살펴보고 관리할 수 있습니다.

vRealize Automation Cloud Assembly의 구성 관리

vRealize Automation Cloud Assembly는 배포의 구성 및 편차를 관리할 수 있도록 Puppet Enterprise 및 Ansible 오픈 소스와의 통합을 지원합니다.

Puppet 통합

Puppet 기반 구성 관리를 통합하려면 vSphere 워크로드가 있는 공용 또는 사설 클라우드에 유효한 Puppet Enterprise 인스턴스가 설치되어 있어야 합니다. 이 외부 시스템과 vRealize Automation Cloud Assembly 인스턴스 간에 연결을 설정해야 합니다. 그런 다음 Puppet 구성 관리를 해당 Blueprint에 추가하여 vRealize Automation Cloud Assembly에서 사용 가능하게 만들 수 있습니다.

vRealize Automation Cloud Assembly Blueprint 서비스 Puppet 제공자는 배포된 계산 리소스에 Puppet 에이전트를 설치, 구성 및 실행합니다. Puppet 제공자는 다음 사전 요구 사항을 충족하는 SSH 및 WinRM 연결을 지원합니다.

- SSH 연결:
 - 사용자 이름은 슈퍼 사용자 또는 NOPASSWD로 명령을 실행할 수 있는 `sudo` 권한을 가진 사용자여야 합니다.
 - 해당 사용자에게 대해 `requiretty`를 사용하지 않도록 설정해야 합니다.
 - 배포 계산 리소스에서 `cURL`을 사용할 수 있어야 합니다.
- WinRM 연결:
 - 배포 계산 리소스에서 PowerShell 2.0을 사용할 수 있어야 합니다.
 - vRealize Orchestrator 설명서에 명시된 대로 Windows 템플릿을 구성해야 합니다.

DevOps 관리자는 Puppet Master에 대한 연결을 관리하고 특정 배포에 Puppet 역할 또는 구성 규칙을 적용하는 작업을 수행합니다. 배포 후에는 구성 관리를 지원하도록 구성된 가상 시스템이 지정된 Puppet Master에 등록됩니다.

가상 시스템이 배포되면 사용자가 외부 시스템으로 Puppet Master를 추가 또는 삭제하거나 Puppet Master에 할당된 프로젝트를 업데이트할 수 있습니다. 아울러, 해당 사용자는 시스템이 더 이상 사용되지 않는 경우 Puppet Master에서 배포된 가상 시스템의 등록을 해제할 수 있습니다.

Ansible 오픈 소스 통합

Ansible 통합을 설정할 때 Ansible 설치 지침에 따라 Ansible 오픈 소스를 설치합니다. 설치에 대한 자세한 내용은 Ansible 설명서를 참조하십시오.

Ansible은 기본적으로 호스트 키 검사를 사용합니다. known_hosts 파일에 다른 키를 사용하여 호스트를 다시 설치하면 오류 메시지가 나타납니다. known_hosts 파일에 호스트가 나열되지 않으면 시작 시 키를 입력해야 합니다. /etc/ansible/ansible.cfg 또는 ~/.ansible.cfg 파일에서 다음 설정을 사용하여 호스트 키 확인을 사용하지 않도록 설정할 수 있습니다.

```
[defaults]
host_key_checking = False
localhost_warning = False

[paramiko_connection]
record_host_keys = False

[ssh_connection]
#ssh_args = -C -o ControlMaster=auto -o ControlPersist=60s
ssh_args = -o UserKnownHostsFile=/dev/null
```

호스트 키 검사 오류를 방지하려면 host_key_checking 및 record_host_keys를 **False**로 설정하고 ssh_args에 추가 옵션 UserKnownHostsFile=/dev/null을 설정합니다. 또한 인벤토리가 처음에 비어 있는 경우 Ansible은 호스트 목록이 비어 있음을 경고합니다. 이로 인해 플레이 북 구문 검사가 실패합니다.

Ansible Vault를 사용하면 암호 또는 키와 같은 중요한 정보를 일반 텍스트가 아닌 암호화된 파일에 저장할 수 있습니다. Vault는 암호로 암호화됩니다. vRealize Automation Cloud Assembly에서, Ansible은 Vault를 사용하여 호스트 시스템에 대한 ssh 암호와 같은 데이터를 암호화합니다. 이는 Vault 암호 경로가 설정된 것으로 가정합니다.

ansible.cfg 파일을 수정하여 다음 형식으로 암호 파일의 위치를 지정할 수 있습니다.

```
vault_password_file = /path to/file.txt
```

Ansible이 암호를 자동으로 검색하도록 ANSIBLE_VAULT_PASSWORD_FILE 환경 변수를 설정할 수도 있습니다. 예를 들어 ANSIBLE_VAULT_PASSWORD_FILE=~/.vault_pass.txt와 같이 설정할 수 있습니다.

vRealize Automation Cloud Assembly는 Ansible 인벤토리 파일을 관리하므로 vRealize Automation Cloud Assembly 사용자가 인벤토리 파일에 대한 **rwX** 액세스 권한이 있는지 확인해야 합니다.

```
cat ~/var/tmp/vmware/provider/user_defined_script/${ls -t ~/var/tmp/vmware/provider/
user_defined_script/ | head -1)/log.txt
```

vRealize Automation Cloud Assembly 오픈 소스 통합으로 루트가 아닌 사용자를 사용하려는 경우 vRealize Automation Cloud Assembly 오픈 소스 제공자가 사용하는 명령을 실행하려면 사용자에게 일련의 사용 권한이 필요합니다. 다음 명령은 사용자의 **sudoers** 파일에서 설정해야 합니다.

```
Defaults:myuser !requiretty
```

askpass 애플리케이션이 지정되지 않은 관리 그룹에 사용자가 속하지 않은 경우 사용자의 **sudoers** 파일에서 다음 명령을 설정합니다.

```
myuser ALL=(ALL) NOPASSWD: ALL
```

Ansible 통합을 설정하는 동안 오류가 발생하거나 다른 문제가 발생하면, Ansible 제어 시스템의 'cat~/var/tmp/vmware/provider/user_defined_script/\$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ | head -1)/'에 있는 log.txt 파일을 참조하십시오.

vRealize Automation Cloud Assembly에서 Puppet Enterprise 통합 구성

vRealize Automation Cloud Assembly는 Puppet Enterprise 구성 관리와의 통합을 지원합니다.

Cloud Assembly에 외부 시스템으로 Puppet Enterprise를 추가하면 기본적으로 모든 프로젝트에서 Puppet Enterprise를 사용할 수 있는데 이를 특정 프로젝트로 제한할 수 있습니다.

Puppet Enterprise를 통합을 추가하려면 마스터의 Puppet 마스터 이름과 마스터의 호스트 이름 또는 IP 주소가 있어야 합니다.

오류 또는 정보 용도로 Puppet 로그를 확인해야 하는 경우, 다음 위치에서 해당 로그를 찾을 수 있습니다.

설명	로그 위치
관련 이벤트 생성 및 설치에 대한 로그	배포된 시스템 `~/var/tmp/vmware/provider/user_defined_script/\$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ head -1)/`에 로그가 있습니다. 전체 로그는 log.txt 파일을 참조하십시오. 세부 Puppet 에이전트 로그에 대한 자세한 내용은 https://puppet.com/docs/puppet/4.8/services_agent_unix.html#logging 을 참조하십시오.
Puppet 삭제 및 실행 관련 작업에 대한 로그	PE, `~/var/tmp/vmware/provider/user_defined_script/\$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ head -1)/`에 로그가 있습니다. 전체 로그는 log.txt 파일을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.
- 2 Puppet을 선택합니다.
- 3 Puppet 구성 페이지에 필요한 정보를 입력합니다.
- 4 **검증**을 클릭하여 통합을 확인합니다.
- 5 **추가**를 클릭합니다.

결과

Puppet을 Blueprint에서 사용할 수 있습니다.

다음에 수행할 작업

Puppet 구성 요소를 원하는 Blueprint에 추가합니다.

- 1 Blueprint 메뉴의 [컨텐츠 관리] 머리글 아래에 있는 Puppet을 선택하고 Puppet 구성 요소를 캔버스에 끌어다 놓습니다.
- 2 오른쪽 창에서 Puppet 속성을 입력합니다.

속성	설명
마스터	이 Blueprint에 사용할 Puppet 기본 시스템의 이름을 입력합니다.
환경	Puppet 기본 시스템의 환경을 선택합니다.
역할	이 Blueprint에 사용할 Puppet 역할을 선택합니다.
에이전트 실행 간격	Puppet 에이전트가 이 Blueprint와 관련하여 배포된 가상 시스템에 적용할 구성 세부 정보를 Puppet 기본 시스템에서 풀링하는 주기입니다.

3 Puppet 구성 속성에 대한 YAML 코드를 보려면 오른쪽 창에서 [코드] 탭을 클릭합니다.

vRealize Automation Cloud Assembly에서 Ansible 오픈 소스 통합 구성

vRealize Automation Cloud Assembly는 Ansible 오픈 소스 구성 관리와의 통합을 지원합니다. 통합을 구성한 후에는 Ansible 구성 요소를 새 배포 또는 기존 배포에 추가할 수 있습니다.

Ansible 오픈 소스를 vRealize Automation Cloud Assembly와 통합하는 경우 구성 관리 자동화를 위해 새 시스템이 프로비저닝될 때 지정된 순서로 하나 이상의 Ansible 플레이북을 실행하도록 Ansible 오픈 소스를 구성할 수 있습니다. 배포의 Blueprint에서 원하는 플레이북을 지정합니다.

Ansible 통합을 설정할 때 리소스 관리용 정보를 정의하는 인벤토리 파일 경로를 비롯해 Ansible Open 소스 호스트 시스템을 지정해야 합니다. 또한 Ansible 오픈 소스 인스턴스 액세스를 위해 이름과 암호를 제공해야 합니다. 나중에 Ansible 구성 요소를 배포에 추가할 때 키 기반 인증을 사용하도록 연결을 업데이트할 수 있습니다.

기본적으로 Ansible에서는 ssh를 사용하여 물리적 시스템에 연결합니다. Windows 시스템을 osType Windows 속성과 함께 Blueprint에 지정된 대로 사용하는 경우 connection_type 변수가 자동으로 winm으로 설정됩니다.

Ansible 통합은 IP 주소를 사용하지 않는 물리적 시스템을 지원합니다. AWS, Azure 및 GCP와 같은 공용 클라우드에서 프로비저닝된 시스템의 경우, 생성된 리소스의 주소 속성은 시스템이 공용 네트워크에 연결되어 있는 경우에만 시스템의 공용 IP 주소로 채워집니다. 공용 네트워크에 연결되어 있지 않은 시스템의 경우 Ansible 통합은 시스템에 연결되어 있는 네트워크에서 IP 주소를 찾습니다. 여러 개의 네트워크가 연결되어 있다면 최소 deviceIndex(즉 시스템에 연결된 NIC(네트워크 인터페이스 카드)의 인덱스)가 있는 네트워크를 찾습니다. deviceIndex 속성이 Blueprint에 지정되어 있지 않은 경우 통합은 연결된 첫 번째 네트워크를 사용합니다.

vRealize Automation Cloud Assembly에서 통합을 위한 Ansible 오픈 소스 구성에 대한 자세한 내용은 [vRealize Automation Cloud Assembly의 구성 관리](#) 항목을 참조하십시오.

사전 요구 사항

- Ansible 제어 시스템은 Ansible 버전 2.6.0 이상을 사용해야 합니다.
- 사용자에게 Ansible 인벤토리 파일이 있는 디렉토리에 대한 읽기/쓰기 액세스 권한이 있어야 합니다. 또한 사용자에게 인벤토리 파일(이미 있는 경우)에 대한 읽기/쓰기 액세스 권한이 있어야 합니다.

- `sudo` 옵션과 함께 루트가 아닌 사용자를 사용하는 경우 `sudoers` 파일에 다음이 설정되어 있는지 확인합니다.

```
Defaults:user_name !requiretty
```

및

```
username ALL=(ALL) NOPASSD: ALL
```

- `/etc/ansible/ansible.cfg` 또는 `~/.ansible.cfg`에 `host_key_checking = False`를 설정하여 호스트 키 검사가 사용되지 않도록 설정합니다.
- 다음 줄을 `/etc/ansible/ansible.cfg` 또는 `~/.ansible.cfg` 파일에 추가하여 Vault 암호가 설정되도록 해야 합니다.

```
vault password_file = /path/to/password_file
```

Vault 암호 파일에는 일반 텍스트 형식의 암호가 포함되어 있으며, Blueprint 또는 배포에서 다음 예에 표시된 것과 같이 ACM과 노드 간에 사용할 사용자 이름 및 암호 조합을 제공하는 경우에만 사용됩니다.

```
echo 'myStr0ng9@88w0rd' > ~/.ansible_vault_password.txt
echo 'ANSIBLE_VAULT_PASSWORD_FILE=~/.ansible_vault_password.txt' > ~/.profile      #
Instead of this way, you can also set it setting
'vault_password_file=~/.ansible_vault_password.txt' in either /etc/ansible/ansible.cfg or
~/.ansible.cfg
```

- 플레이 북을 실행하는 동안 호스트 키 실패를 방지하려면 `/etc/ansible/ansible config`에 다음 설정을 포함하는 것이 좋습니다.

```
[paramiko_connection]
record_host_keys = False

[ssh_connection]
#ssh_args = -C -o ControlMaster=auto -o ControlPersist=60s
ssh_args = -o UserKnownHostsFile=/dev/null          # If you already have any
options set for ssh_args, just add the additional option shown here at the end.
```

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.
- 2 **[Ansible]**을 클릭합니다.
[Ansible 구성] 페이지가 나타납니다.
- 3 Ansible 오픈 소스 인스턴스에 대한 호스트 이름, 인벤토리 파일 경로 및 기타 필수 정보를 입력합니다.
- 4 **검증**을 클릭하여 통합을 확인합니다.
- 5 **추가**를 클릭합니다.

결과

Ansible을 Blueprint에서 사용할 수 있습니다.

다음에 수행할 작업

Ansible 구성 요소를 원하는 Blueprint에 추가합니다.

- 1 Blueprint 캔버스 페이지에서, Blueprint 옵션 메뉴의 [구성 관리] 머리글 아래에서 [Ansible]을 선택하고 Ansible 구성 요소를 캔버스로 끕니다.
- 2 오른쪽 패널을 사용하여 적절한 Ansible 속성을 구성합니다(예: 실행할 플레이북 지정).

Ansible에서 사용자는 변수를 단일 호스트에 할당한 다음 나중에 이를 플레이북에서 사용할 수 있습니다.

Ansible 오픈 소스 통합을 사용하면 이러한 호스트 변수를 Blueprint에 지정할 수 있습니다.

hostVariables 속성은 Ansible 제어 시스템이 예상하는 올바른 YAML 형식이어야 하며, 이 콘텐츠는 다음 위치에 배치됩니다.

```
parent_directory_of_inventory_file/host_vars/host_ip_address/vra_user_host_vars.yml
```

Ansible 인벤토리 파일의 기본 위치는 Cloud Assembly의 [통합] 페이지에 추가된 Ansible 계정에서 정의됩니다. Ansible 통합은 Blueprint에서 hostVariable YAML 구문을 검증하지 않지만 형식 또는 구문이 잘못된 경우 플레이북을 실행할 때 Ansible 제어 시스템에서 예외가 발생합니다.

다음 Blueprint YAML 조각은 hostVariables 속성의 사용 예를 보여 줍니다.

```
Cloud_Ansible_1:
  type: Cloud.Ansible
  properties:
    host: '${resource.AnsibleLinuxVM.*}'
    osType: linux
    account: ansible-CAVA
    username: ${input.username}
    password: ${input.password}
    maxConnectionRetries: 20
    groups:
      - linux_vms
    playbooks:
      provision:
        - /root/ansible-playbooks/install_web_server.yml
    hostVariables: |
      message: Hello ${env.requestedBy}
      project: ${env.projectName}
```

vRealize Automation Cloud Assembly에서 Active Directory 통합을 생성하는 방법

vRealize Automation Cloud Assembly는 가상 시스템을 프로비저닝하기 전에 Active Directory 서버 내에 지정된 OU(조직 구성 단위)에서 바로 사용할 수 있는 컴퓨터 계정을 생성할 수 있도록 Active Directory 서버와의 통합을 지원합니다.

Active Directory 통합은 Active Directory 서버에 대한 LDAP 연결만 지원합니다.

사전 요구 사항

- vCenter 온-프레미스와 Active Directory 통합을 구성하는 경우 Active Directory 통합을 위해 확장성 클라우드 프록시를 구성해야 합니다. **확장성 > 작업 > 통합**을 선택하고 **확장성 작업 온-프레미스**를 선택합니다.
- 클라우드에서 Active Directory와의 통합을 구성하는 경우에는 Microsoft Azure 또는 Amazon Web Services 계정이 있어야 합니다.
- Active Directory 서버는 LDAP 서버 연결을 사용해야 합니다.
- Active Directory 통합에 사용할 수 있는 적절한 클라우드 영역과 이미지 및 버전 매핑으로 구성된 프로젝트가 있어야 합니다.
- Active Directory 통합을 프로젝트와 연결하기 전에 Active Directory에서 원하는 OU를 사전 생성해야 합니다.

절차

- 1 **인프라 > 연결 > 통합**을 선택한 다음, **새 통합**을 선택합니다.
- 2 **Active Directory**를 클릭합니다.
- 3 **요약** 탭에서 적절한 LDAP 호스트 및 환경 이름을 입력합니다.
- 4 LDAP 서버에 대한 이름과 암호를 입력합니다.
- 5 Active Directory에서 원하는 사용자 및 그룹에 대해 적절한 기본 DN을 입력합니다.

참고 Active Directory 통합당 DN은 하나만 지정할 수 있습니다.

- 6 **유효성 검증**을 클릭하여 통합이 작동하는지 확인합니다.
- 7 이 통합의 이름과 설명을 입력합니다.
- 8 **저장**을 클릭합니다.
- 9 **프로젝트** 탭을 클릭하여 Active Directory 통합에 프로젝트를 추가합니다.

프로젝트 추가 대화 상자에서 프로젝트 이름과 상대 DN을 선택해야 합니다. 이 DN은 [요약] 탭에 지정된 기본 DN 내에 있는 DN입니다.

- 10 **저장**을 클릭합니다.

결과

이제 Active Directory와 통합된 프로젝트를 Blueprint에 연결할 수 있습니다. 이 Blueprint을 사용하여 시스템을 프로비저닝하면 지정된 Active Directory와 조직 구성 단위에서 시스템이 미리 준비됩니다.

vRealize Automation Cloud Assembly의 온보딩 계획이란?

워크로드 온보딩 계획을 사용하여 대상 지역이나 데이터 센터의 클라우드 계정 유형에서 데이터가 수집되지만 아직 vRealize Automation Cloud Assembly 프로젝트를 통해 관리되지 않는 시스템을 식별합니다.

vRealize Automation Cloud Assembly 외부에 배포된 시스템이 포함된 클라우드 계정을 추가하는 경우 해당 시스템을 온보딩할 때까지 Cloud Assembly에서 관리되지 않습니다. 온보딩 계획을 사용하여 관리되지 않는 시스템을 vRealize Automation Cloud Assembly 관리 대상으로 가져옵니다. 계획을 생성하고 시스템으로 채운 다음 계획을 실행하여 시스템을 가져옵니다. 온보딩 계획을 사용하면 Blueprint를 생성하고 하나 이상의 배포를 생성할 수도 있습니다.

하나 이상의 관리되지 않는 시스템을 단일 계획으로 온보딩할 수 있습니다. 수동으로 또는 필터링 규칙을 사용하여 시스템을 선택할 수 있습니다. 필터링 규칙은 시스템 이름, 상태, IP 주소 및 태그와 같은 조건을 기반으로 온보딩을 위한 시스템을 선택합니다.

- 시간당 단일 온보딩 계획 내에서 최대 3,500개의 관리되지 않는 시스템을 온보딩할 수 있습니다.
- 시간당 여러 온보딩 계획 내에서 최대 17,000개의 관리되지 않는 시스템을 동시에 온보딩할 수 있습니다.

워크로드 온보딩에 사용할 수 있는 시스템은 특정 클라우드 계정 유형 및 지역과 관련된 **리소스 > 시스템** 페이지에서 [원본] 열에 Discovered 레이블로 표시됩니다. 데이터 수집된 시스템만 나열됩니다. 시스템을 온보딩한 후에는 [원본] 열에 Deployed로 표시됩니다.

워크로드 온보딩 계획을 실행하는 사용자는 시스템 소유자로 자동 할당됩니다.

온보딩 예시

온보딩 기술의 예시는 예: 선택한 시스템을 vRealize Automation Cloud Assembly의 단일 배포로 온보딩 및 예: vRealize Automation Cloud Assembly에서 개별 배포로서 온보딩 규칙으로 필터링된 시스템 항목을 참조하십시오.

온보딩 이벤트 구독

계획을 실행하면 Deployment Onboarded 이벤트가 생성됩니다. 확장성 탭 옵션을 사용하면 이러한 배포 이벤트를 구독하고, 이벤트에 대해 작업을 수행할 수 있습니다.

예: 선택한 시스템을 vRealize Automation Cloud Assembly의 단일 배포로 온보딩

이 예에서는 관리되지 않는 두 개의 시스템을 단일 vRealize Automation Cloud Assembly 배포로 온보딩하고 계획의 모든 시스템에 대한 단일 Blueprint를 생성합니다.

클라우드 계정을 생성할 경우, 이 클라우드 계정에 연결된 모든 시스템의 데이터가 수집되어 **인프라 > 리소스 > 시스템** 페이지에 표시됩니다. 클라우드 계정에 vRealize Automation Cloud Assembly 외부에 배포된 시스템이 있는 경우에는 온보딩 계획을 사용하여 vRealize Automation Cloud Assembly가 해당 시스템을 관리하도록 할 수 있습니다.

사전 요구 사항

- 필요한 사용자 역할이 있는지 확인합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.
- vRealize Automation Cloud Assembly의 온보딩 계획이란? 항목을 검토합니다.
- vRealize Automation Cloud Assembly 프로젝트를 생성하고 준비합니다.

이 절차에는 기본 Wordpress 사용 사례의 일부 단계가 포함됩니다. [WordPress 사용 사례](#) 항목을 참조하십시오.

- 프로젝트를 생성하고, 사용자를 추가하고, 프로젝트에 사용자 역할을 할당합니다. [WordPress 사용 사례: 프로젝트 생성](#) 항목을 참조하십시오.
- 프로젝트에 대한 Amazon Web Services 클라우드 계정을 생성합니다. [WordPress 사용 사례: 클라우드 계정 추가](#) 항목을 참조하십시오.

이 절차의 Amazon Web Services 클라우드 계정은 클라우드 계정이 vRealize Automation Cloud Assembly에 추가되기 전에 배포되고 vRealize Automation Cloud Assembly 이외의 애플리케이션이 배포한 시스템을 포함합니다.

- 온보딩할 시스템이 **시스템** 페이지에 포함되어 있는지 확인합니다. [시스템 리소스](#) 항목을 참조하십시오.

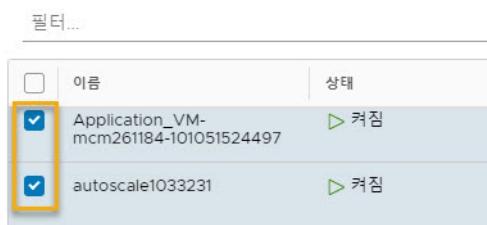
절차

- 1 **인프라 > 온보딩**으로 이동합니다.
- 2 **새 온보딩 계획**을 클릭하고 샘플 값을 입력합니다.

설정	샘플 값
계획 이름	VC-sqa-deployments
설명	OurCo-AWS 클라우드 계정의 AWS 시스템 온보딩 계획 샘플
클라우드 계정	OurCo-AWS
기본 프로젝트	WordPress

- 3 **생성**을 클릭합니다.
- 4 계획의 **배포** 탭에서 **시스템 선택**을 클릭하고 시스템을 하나 이상 선택한 후 **확인**을 클릭합니다.

시스템 선택



- 5 모든 시스템이 포함된 **단일 배포 생성**을 선택하고 **생성**을 클릭합니다.
- 6 새 배포 이름 옆의 확인란을 클릭하고 **Blueprint...**를 클릭합니다.
- 7 **Cloud Assembly 형식으로 Blueprint 생성**을 클릭합니다.

8 Blueprint 이름을 입력하고 **저장**을 클릭합니다.

Blueprint 구성

배포 Deployment-c8a6e0f9-790a-411b-b0e9-4d2e79ce118e

☐ 없음(런타임 스냅샷 사용)

☒ Cloud Assembly 형식으로 Blueprint 생성

Blueprint 이름: BP_Sample_1

Blueprint 미리 보기

```

1 ---
2 resources:
3   VMware-vRO-Appliance-SAAS-1127:
4     type: "Cloud.vSphere.Machine"
5     properties:
6       imageRef: "no_image_available"
7       cpuCount: 2
8       totalMemoryMB: 6144
9   VMware-Cloud-Services-Data-Collector-7.2.0.25668-11138205_OVF10:
10    type: "Cloud.vSphere.Machine"
11    properties:
12      imageRef: "no_image_available"
13      cpuCount: 4
14      totalMemoryMB: 12288
  
```

취소 저장

참고 온보딩 계획이 vSphere 시스템을 사용하는 경우에는 온보딩 프로세스가 완료된 후 Blueprint를 편집해야 합니다. 온보딩 프로세스는 소스 vSphere 시스템과 해당 시스템 템플릿을 연결할 수 없으며 결과로 생성된 Blueprint에서는 Blueprint 코드에 imageRef: "no image available" 항목이 포함됩니다. imageRef: 필드에 올바른 템플릿 이름을 지정해야 Blueprint를 배포할 수 있습니다. 온보딩 프로세스가 완료된 후 Blueprint를 쉽게 찾고 업데이트하려면 배포의 **Blueprint 구성** 페이지에서 **Blueprint 이름** 옵션을 사용합니다. 자동 생성된 Blueprint 이름을 기록하거나 원하는 Blueprint 이름을 입력하고 기록합니다. 온보딩이 완료되면 Blueprint를 찾아서 열고 imageRef: 필드의 "no image available" 항목을 올바른 템플릿 이름으로 바꿉니다.

- 9 배포 이름 확인란을 클릭하고 **실행**을 클릭한 다음, **계획 실행** 페이지에서 **실행**을 다시 클릭합니다.
선택한 Amazon Web Services 시스템이 해당하는 Blueprint와 함께 단일 배포로 온보딩됩니다.
- 10 **Blueprint** 탭을 클릭한 다음, Blueprint 이름을 클릭하여 Blueprint를 열고 검토합니다.
- 11 **배포** 탭을 클릭한 다음, 배포 이름을 클릭하여 배포를 열고 검토합니다.

예: vRealize Automation Cloud Assembly에서 개별 배포로서 온보딩 규칙으로 필터링된 시스템

이 예시에서는 필터링 규칙을 사용하여 상태가 [On]이고 이름이 BG로 시작하는 시스템을 온보딩합니다. 또한 계획의 각 시스템에 대해 별도의 vRealize Automation Cloud Assembly Blueprint 및 배포를 생성합니다.

클라우드 계정을 생성할 경우, 이 클라우드 계정에 연결된 모든 시스템의 데이터가 수집되어 **인프라 > 리소스 > 시스템** 페이지에 표시됩니다. 클라우드 계정에 vRealize Automation Cloud Assembly 외부에 배포된 시스템이 있는 경우에는 온보딩 계획을 사용하여 vRealize Automation Cloud Assembly가 해당 시스템을 관리하도록 할 수 있습니다.

사전 요구 사항

- 필요한 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- [vRealize Automation Cloud Assembly의 온보딩 계획이란?](#) 항목을 검토합니다.
- vRealize Automation Cloud Assembly 프로젝트를 생성하여 준비하고 하나 이상의 클라우드 계정으로 채웁니다.

이를 위해서는 단계별 설정 절차의 몇 가지 기본적인 단계를 수행해야 합니다.

- 프로젝트를 생성하고, 사용자를 추가하고, 프로젝트에 사용자 역할을 할당합니다. [WordPress 사용 사례: 프로젝트 생성](#) 항목을 참조하십시오.
- 프로젝트의 지정된 지역에 하나 이상의 클라우드 계정을 생성합니다. [WordPress 사용 사례: 클라우드 계정 추가](#) 항목을 참조하십시오.
- 온보딩할 시스템이 **시스템** 페이지에 포함되어 있는지 확인합니다. [시스템 리소스](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 온보딩**으로 이동합니다.
- 2 **새 온보딩 계획**을 클릭하고 값을 입력합니다.

설정	샘플 값
계획 이름	ob_rules_1
설명	Machine onboarding with rules1
클라우드 계정	rs-aws
기본 프로젝트	rs-project

새 온보딩 계획



계획 이름 ob_rules_1

설명

Machine onboarding with rules1

사전 요구 사항

클라우드 계정을 추가하고 온보딩할 시스템이 있는 계산 리소스에 대한 클라우드 영역을 생성합니다.

사용자가 한 명 이상인 프로젝트를 생성하고 클라우드 영역에 대한 프로젝트 액세스 권한을 부여합니다.

클라우드 계정 Q rs-aws



기본 프로젝트 Q rs-project



취소

생성

3 생성을 클릭합니다.

ob_rules_1

요약 규칙 시스템 배포

계획 이름 ob_rules_1

설명 Machine onboarding with rules1

계획 상태

마지막 실행 없음

소스 정보

클라우드 계정 346test_vc_account

배포 태그 키

적용

대상 구성

기본 프로젝트 Q 123

저장 실행 취소

4 규칙 탭을 클릭한 후 규칙 추가를 클릭합니다.

하나 이상의 규칙을 생성하여 구체적인 시스템 특징을 기반으로 온보딩 대상 시스템 그룹을 선택할 수 있습니다.

ob_rules_1

요약 **규칙** 시스템 배포

규칙을 사용하여 이 계획에 시스템을 추가합니다. ①

규칙 추가 편집 삭제

☐ 이름

5 규칙 이름을 입력합니다(예: ob_rules_1).

추가 규칙

생성 이 계획의 시스템을 채우는 데 사용되는 필터 기반 규칙입니다.

규칙 이름 * ob_rules_1

6 필터를 추가하여 규칙을 구축합니다.

이 예시에서는 **필터** 드롭다운 메뉴에서 **상태** 및 **이름** 필터를 사용하여 이름에 BG*가 포함되어 있고 상태가 On인 모든 시스템을 지정합니다.

규칙 이름 *

ob_rules_1

필터...

속성	상태	주소	생성 시간	태그
임의:				
이름:	켜짐	10.184.68.223	2020년 1월 09일	
상태:				
주소:				
태그:				

4 시스템

규칙 이름 *

ob_rules_1

이름: BG* × 상태: 켜짐

7 저장을 클릭합니다.

규칙을 더 만들 수도 있지만 이 예시에서는 하나의 규칙을 사용합니다.

ob_rules_1

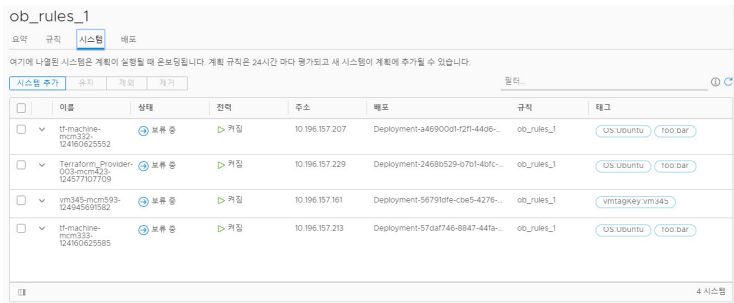
요약 **규칙** 시스템 배포

규칙을 사용하여 이 계획에 시스템을 추가합니다. ①

규칙 추가 편집 삭제

<input type="checkbox"/> 이름	상태	필터
<input type="checkbox"/> ob_rules_1	확인	Name:BG*

- 8 시스템 탭을 클릭합니다. 이 예시에서는 시스템 4개가 선택되어 있는데, 그 중 3개는 이름이 BG로 시작하고 나머지 1개는 이름에 BG가 포함되어 있습니다.

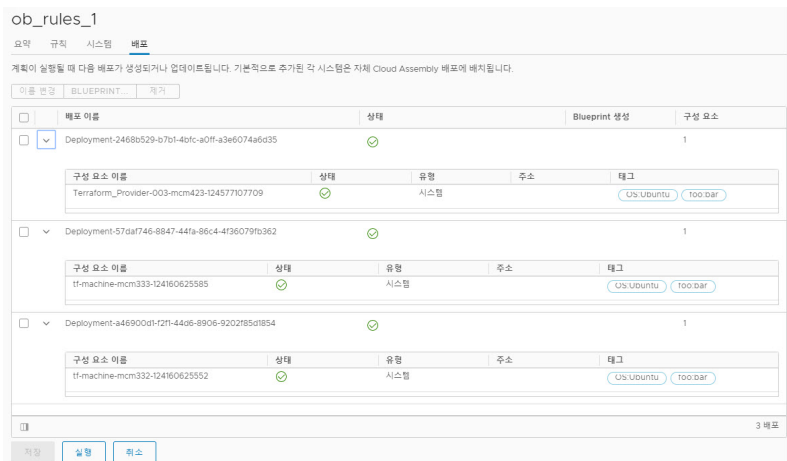


- 9 이름이 BG로 시작하지 않는 시스템의 확인란을 선택한 후 **제외**를 클릭하여 해당 시스템을 제거합니다.



- 10 배포 탭을 클릭합니다.

이름이 BG로 시작하고 전원이 On 상태에 있는 시스템 3개에 대한 배포 준비가 완료되었습니다. 기본적으로 각 시스템에 대해 별도의 Blueprint와 배포가 생성됩니다.



- 11 배포 이름 3개 옆의 확인란을 클릭한 후 **Blueprint, Cloud Assembly** 형식으로 **Blueprint** 생성을 차례로 클릭하고 **저장**을 클릭합니다.

Blueprint 구성

3 선택된 배포

☐ 없음(런타임 스냅샷 사용)

☒ Cloud Assembly 형식으로 Blueprint 생성

Blueprint 미리 보기

여러 배포가 선택됨

취소

저장

참고 온보딩 계획이 vSphere 시스템을 사용하는 경우에는 온보딩 프로세스가 완료된 후 Blueprint를 편집해야 합니다. 온보딩 프로세스는 소스 vSphere 시스템과 해당 시스템 템플릿을 연결할 수 없으며 결과로 생성된 Blueprint에서는 Blueprint 코드에 imageRef: "no image available" 항목이 포함됩니다. imageRef: 필드에 올바른 템플릿 이름을 지정해야 Blueprint를 배포할 수 있습니다. 온보딩 프로세스가 완료된 후 Blueprint를 쉽게 찾고 업데이트하려면 배포의 **Blueprint 구성** 페이지에서 **Blueprint 이름** 옵션을 사용합니다. 자동 생성된 Blueprint 이름을 기록하거나 원하는 Blueprint 이름을 입력하고 기록합니다. 온보딩이 완료되면 Blueprint를 찾아서 열고 imageRef: 필드의 "no image available" 항목을 올바른 템플릿 이름으로 바꿉니다.

- 12** **배포** 페이지에서 **배포** 이름 3개 옆의 확인란을 클릭한 후 **실행**을 클릭합니다.

The screenshot shows the Azure portal interface for a resource group named 'ob_rules_1'. The 'Blueprints' tab is active, showing a table of blueprints. The first blueprint, 'Deployment-24688520-67b1-46fc-a0ff-a3e6074a6035', is selected. Below the table, the details of the selected blueprint are shown, including its 'Configuration details' and 'Tags'. The 'Configuration details' section shows the blueprint name 'Deployment-57dad746-6847-441a-96c4-4136079fb362' and its 'Configuration details' section, which includes the 'Configuration details' and 'Tags'.

13 확인 메시지가 표시되면 **실행**을 클릭하여 시스템을 온보딩합니다.

계획 실행

×

계획 이름	ob_rules_1
설명	Machine onboarding with rules1
클라우드 계정	346test_vc_account
기본 프로젝트	123
배포	3
마지막 실행	없음

취소

실행

계획이 실행되고 시스템이 vRealize Automation Cloud Assembly 관리 대상으로 온보딩됩니다. 각 시스템에 대해 별도의 Blueprint와 배포가 생성됩니다.

vRealize Automation Cloud Assembly 환경에 대한 고급 구성

프로젝트 구성, 통합 및 배포를 추가로 지원하도록 vRealize Automation Cloud Assembly 환경을 구성할 수 있습니다.

사용자 및 로그 사용, 고객 환경 향상 프로그램 참여 또는 탈퇴와 같은 관리 방법에 대한 관련 정보와 추가 정보는 [vRealize Automation 관리](#) 도움말을 참조하십시오.

vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법

인터넷에 직접 액세스할 수 없는 격리된 네트워크에 vRealize Automation 8.0.1 이상이 설치된 경우 프록시 기능을 통해 인터넷을 허용하도록 인터넷 프록시 서버를 구성할 수 있습니다. 인터넷 프록시 서버는 HTTP 및 HTTPS를 지원합니다.

vRealize Automation을 사용하여 AWS(Amazon Web Services), Microsoft Azure, GCP(Google Cloud Platform)와 같은 공용 클라우드 제공자 및 IPAM, Ansible, Puppet과 같은 외부 통합 지점을 구성하고 사용하려면 내부 vRealize Automation 인터넷 프록시 서버에 액세스하도록 인터넷 프록시 서버를 구성해야 합니다.

vRealize Automation에는 인터넷 프록시 서버와 통신하는 내부 프록시 서버가 포함되어 있습니다. 이 서버는 `vraccli proxy set ...` 명령으로 구성된 경우 프록시 서버와 통신합니다. 조직에 대해 인터넷 프록시 서버를 구성하지 않은 경우 vRealize Automation 내부 프록시 서버가 인터넷에 직접 연결하려고 시도합니다.

제공된 `vraccli` 명령줄 유틸리티를 사용하여 인터넷 프록시 서버를 사용하도록 vRealize Automation을 설정할 수 있습니다. `vraccli API`를 사용하는 방법에 대한 정보를 보려면 `vraccli` 명령줄에서 `--help` 인수를 사용하면 됩니다(예: `vraccli proxy --help`).

인터넷 프록시 서버에 액세스하려면 vRealize Automation에 내장된 ABX(작업 기반 확장성) 온-프레미스 내장형 제어를 사용해야 합니다.

참고 인터넷 프록시를 통한 Workspace ONE Access(이전 명칭 VMware Identity Manager)에 대한 액세스는 지원되지 않습니다. `vraccli set vidm` 명령을 사용하여 인터넷 프록시 서버를 통해 Workspace ONE Access에 액세스할 수 없습니다.

인터넷 프록시 서버는 기본 IP 형식으로 IPv4가 필요합니다. TLS(HTTPS) 인증서 트래픽에는 인터넷 프로토콜 제한, 인증 또는 중간자(man-in-the-middle) 작업이 필요하지 않습니다.

사전 요구 사항

- 송신 트래픽을 외부 사이트로 전달할 수 있는 vRealize Automation 네트워크에서 인터넷 프록시 서버로 사용할 수 있는 기존의 HTTP 또는 HTTPS 서버가 있는지 확인합니다. IPv4에 대해 연결을 구성해야 합니다.
- 대상 인터넷 프록시 서버가 기본 IP 형식으로 IPv6이 아닌 IPv4를 지원하도록 구성되어 있는지 확인합니다.
- 인터넷 프록시 서버가 TLS를 사용 중이고 해당 클라이언트에 대해 HTTPS 연결이 필요한 경우에는 프록시 구성을 설정하기 전에 다음 명령 중 하나를 사용하여 서버 인증서를 가져와야 합니다.

- `vraccli certificate proxy --set path_to_proxy_certificate.pem`

- `vraccli certificate proxy --set stdin`

대화형 입력에 대해 `stdin` 매개 변수를 사용합니다.

절차

- 1 Kubernetes에서 사용되는 포트 또는 컨테이너에 대한 프록시 구성을 생성합니다. 이 예에서는 HTTP 체계를 사용하여 프록시 서버에 액세스합니다.

```
vraccli proxy set --host http://proxy.vmware.com:3128
```

- 2 프록시 구성을 표시합니다.

```
vraccli proxy show
```

결과는 다음과 유사합니다.

```
{
  "enabled": true,
  "host": "10.244.4.51",
  "java-proxy-exclude": "*.local|*.localdomain|localhost|10.244.*|192.168.*|172.16.*|kubernetes|sc2-rdops-vm06-dhcp-198-120.eng.vmware.com|10.192.204.9|*.eng.vmware.com|sc2-rdops-vm06-dhcp-204-9.eng.vmware.com|10.192.213.146|sc2-rdops-vm06-dhcp-213-146.eng.vmware.com|10.192.213.151|sc2-rdops-vm06-dhcp-213-151.eng.vmware.com",
  "java-user": null,
  "password": null,
  "port": 3128,
  "proxy-exclude":
  ".local,.localdomain,localhost,10.244.,192.168.,172.16.,kubernetes,sc2-rdops-vm06-
```

```

dhcp-198-120.eng.vmware.com,10.192.204.9,.eng.vmware.com,sc2-rdops-vm06-
dhcp-204-9.eng.vmware.com,10.192.213.146,sc2-rdops-vm06-
dhcp-213-146.eng.vmware.com,10.192.213.151,sc2-rdops-vm06-dhcp-213-151.eng.vmware.com",
  "scheme": "http",
  "upstream_proxy_host": null,
  "upstream_proxy_password_encoded": "",
  "upstream_proxy_port": null,
  "upstream_proxy_user_encoded": "",
  "user": null,
  "internal.proxy.config": "dns_v4_first on \nhttp_port 0.0.0.0:3128\nlogformat squid
%ts.%03tu %6tr %>a %Ss/%03>Hs %<st %rm %ru %[un %Sh/%<a %mt\naccess_log stdio:/tmp/logger
squid\ncoredump_dir /\ncache deny all \nappend_domain .prelude.svc.cluster.local\nacl
mylan src 10.0.0.0/8\nacl mylan src 127.0.0.0/8\nacl mylan src 192.168.3.0/24\nacl proxy-
exclude dstdomain .local\nacl proxy-exclude dstdomain .localdomain\nacl proxy-exclude
dstdomain localhost\nacl proxy-exclude dstdomain 10.244.\n\nacl proxy-exclude dstdomain
192.168.\n\nacl proxy-exclude dstdomain 172.16.\n\nacl proxy-exclude dstdomain kubernetes\n\nacl
proxy-exclude dstdomain 10.192.204.9\n\nacl proxy-exclude dstdomain .eng.vmware.com\n\nacl
proxy-exclude dstdomain 10.192.213.146\n\nacl proxy-exclude dstdomain
10.192.213.151\n\nalways_direct allow proxy-exclude\n\nhttp_access allow mylan\n\nhttp_access
deny all\n\n# End autogen configuration\n",
  "internal.proxy.config.type": "default"
}

```

참고 조직에 대해 인터넷 프록시 서버를 구성한 경우 위의 예에서 'default' 대신

"internal.proxy.config.type": "non-default"가 표시됩니다. 암호는 보안을 위해 표시되지 않습니다.

참고 -proxy-exclude 매개 변수를 사용하는 경우 기본값을 편집해야 합니다. 예를 들어 인터넷 프록시 서버를 사용하여 액세스할 수 없는 도메인으로 acme.com을 추가하려는 경우 다음 단계를 사용합니다.

- a vracli proxy default-no-proxy를 입력하여 기본 proxy-exclude 설정을 가져옵니다. 이것은 자동으로 생성되는 도메인 및 네트워크의 목록입니다.
- b 값을 편집하여 .acme.com을 추가합니다.
- c vracli proxy set --proxy-exclude ...를 입력하여 구성 설정을 업데이트합니다.
- d /opt/scripts/deploy.sh 명령을 실행하여 환경을 다시 배포합니다.

- 3 (선택 사항) 인터넷 프록시 서버에서 DNS 도메인, FQDN 및 IP 주소에 액세스하지 못하도록 합니다.

항상 parameter `--proxy-exclude`를 사용하여 proxy-exclude 변수의 기본값을 수정합니다.

`exclude.vmware.com` 도메인을 추가하려면 우선 `vrali proxy show` 명령을 사용하고, proxy-exclude 변수를 복사한 다음, 아래와 같이 `vracli proxy set ...` 명령을 사용하여 도메인 값을 추가합니다.

```
vracli proxy set --host http://proxy.vmware.com:3128 --proxy-exclude
"exclude.vmware.com,docker-
registry.prelude.svc.cluster.local,localhost,.local,.cluster.local,10.244.,192.,172.16.,sc-
rdops-vm11-dhcp-75-38.eng.vmware.com,10.161.75.38,.eng.vmware.com"
```

참고 값을 바꾸는 대신 요소를 proxy-exclude에 추가합니다. proxy-exclude 기본값을 삭제하면 vRealize Automation이 제대로 작동하지 않습니다. 이러한 상황이 발생하면 프록시 구성을 삭제하고 다시 시작합니다.

- 4 `vracli proxy set ...` 명령으로 인터넷 프록시 서버를 설정한 후에는 `vracli proxy apply` 명령을 사용하여 인터넷 프록시 서버 구성을 업데이트하고 최신 프록시 설정을 활성화할 수 있습니다.
- 5 아직 수행하지 않은 경우 다음 명령을 실행하여 스크립트 변경 내용을 활성화합니다.

```
/opt/scripts/deploy.sh
```

- 6 (선택 사항) 필요하다면 포트 22에서 외부 액세스를 지원하도록 프록시 서버를 구성합니다.

Puppet 및 Ansible과 같은 통합을 지원하려면 프록시 서버에서 포트 22가 관련 호스트에 액세스할 수 있도록 허용해야 합니다.

예제: 샘플 Squid 구성

1단계를 기준으로, Squid 프록시를 설정하는 경우 `/etc/squid/squid.conf`에서 다음 샘플에 맞추어 구성을 조정할 수 있습니다.

```
acl localnet src 192.168.11.0/24

acl SSL_ports port 443

acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl CONNECT method CONNECT

http_access allow !Safe_ports
http_access allow CONNECT !SSL_ports
http_access allow localnet
```

```

http_port 0.0.0.0:3128

maximum_object_size 5 GB
cache_dir ufs /var/spool/squid 20000 16 256
coredump_dir /var/spool/squid
refresh_pattern ^ftp: 1440 20% 10080
refresh_pattern ^gopher: 1440 0% 1440
refresh_pattern -i (/cgi-bin/|\?) 0 0% 0
refresh_pattern (Release|Packages(.gz)*)$ 0 20% 2880
refresh_pattern . 0 20% 4320

client_persistent_connections on
server_persistent_connections on

```

vRealize Automation에서 cloud-init 또는 cloudbase-init를 사용하여 Windows 템플릿을 설정하는 방법

Windows 배포에서 사용자 지정 이미지를 생성하고 사용하기 위한 cloud-init 및 cloudbase-init 스크립팅을 지원하도록 vRealize Automation 환경을 구성할 수 있습니다.

클라우드 구성 스크립트는 이미지 매핑 및 Blueprint 코드에서 지원됩니다. 클라우드 구성 스크립팅은 cloud-init 및 cloudbase-init 규칙 및 형식을 준수합니다. cloud-init 및 cloudbase-init에 대한 관련 정보는 <https://cloudbase.it/cloudbase-init>를 참조하십시오.

참고 Windows용 cloud-init 또는 cloudbase-init 구성에 대한 자세한 내용은 다음 VMware 블로그를 참조하십시오.

- [Windows Cloud-Init solution](#) 블로그 문서
- [Windows guest initialization with Cloudbase-Init in vCenter](#) 블로그 문서

Linux용 cloud-init 구성에 대한 관련 정보는 블로그 게시물 [vSphere용 vRealize Automation 클라우드 지원 Ubuntu 템플릿 구축](#)을 참조하십시오.

vRealize Automation에서 cloud-init 및 클라우드 구성 스크립트를 사용하는 방법에 대한 자세한 내용은 다음 항목을 참조하십시오.

- [vRealize Automation Cloud Assembly의 이미지 매핑에 대해 알아보기](#)
- [vRealize Automation Cloud Assembly Blueprint에서 시스템을 자동으로 초기화하는 방법](#)

또한 다음 cloud-init 및 cloudbase-init 벤더 페이지도 참조하십시오.

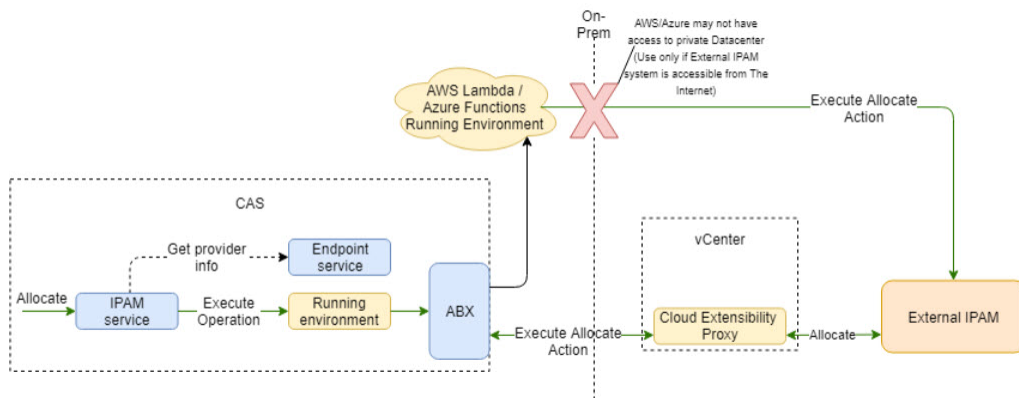
- <https://cloud-init.io>
- <https://cloudinit.readthedocs.io/en/latest/index.html>
- <https://cloudinit.readthedocs.io/en/latest/topics/examples.html#yaml-examples>
- <https://cloudbase.it/cloudbase-init>
- <https://cloudbase-init.readthedocs.io/en/latest/services.html>

- <https://cloudbase-init.readthedocs.io/en/latest/userdata.html#userdata>

IPAM SDK를 사용하여 vRealize Automation에 대한 제공자별 외부 IPAM 통합 패키지를 생성하는 방법

외부 IPAM 벤더 및 파트너는 IPAM SDK를 다운로드하고 사용하여 vRealize Automation에서 제공자별 IPAM 솔루션을 지원할 수 있도록 하는 IPAM 통합 패키지를 생성할 수 있습니다.

제공된 IPAM SDK를 사용하여 vRealize Automation에 대한 사용자 지정 IPAM 통합 패키지를 구축하고 배포하는 프로세스는 [VMware Cloud Assembly에 대한 제공자별 IPAM 통합 패키지 생성 및 배포에 설명](#)되어 있습니다. 문서의 설명대로 [VMware Solution Exchange](#) 마켓플레이스에서 "VMware vRealize Automation 타사 IPAM SDK"를 다운로드할 수 있습니다.



IPAM SDK를 사용하여 벤더별 IPAM 통합 패키지를 생성하기 전에 vRealize Automation에 대한 패키지가 이미 있는지 확인합니다. IPAM 제공자 웹 사이트, [VMware Solution Exchange](#) 마켓플레이스 및 vRealize Automation [마켓플레이스](#) 탭에서 제공자별 IPAM 통합 패키지가 있는지 확인할 수 있습니다.

제공자별 외부 IPAM 통합 사용 사례 예제는 벤더에 따라 다르며 유용한 참조 정보도 포함되어 있습니다.

vRealize Automation Cloud Assembly 사용 사례

3

이러한 사용 사례는 vRealize Automation Cloud Assembly에서 리소스 인프라를 생성한 다음, 애플리케이션을 설계하고 해당 인프라에 배포하는 예를 보여줍니다.

사용 사례는 예시 값만 제공합니다. 실제 환경 구조 및 이름 지정 규칙은 다를 수 있습니다.

본 장은 다음 항목을 포함합니다.

- [WordPress 사용 사례](#)
- [VMware Cloud on AWS 사용 사례](#)
- [제공자별 외부 IPAM 통합 사용 사례](#)

WordPress 사용 사례

이 중단 간 vRealize Automation Cloud Assembly 사용 사례는 인프라를 생성하고 이 인프라에 WordPress 사이트를 배포하는 예시를 보여 줍니다.

WordPress 사이트를 완료하는 프로세스를 이해하려면 단계별 설정을 확인하십시오.

여기에 나와 있는 값은 사용 사례 예시에만 해당합니다. 사용자 환경에서 이러한 값을 그대로 사용해서는 안 됩니다.

클라우드 인프라 및 배포 요구 사항에 맞게 고유한 값으로 대체하거나 예시의 값에서 추론할 수 있는 부분에 대해 생각해 보십시오.

절차

1 [WordPress 사용 사례: 인프라 생성](#)

클라우드 관리자는 엔지니어링 팀이 나중에 개발 및 테스트하여 WordPress 사이트를 운영할 수 있는 리소스를 먼저 구성해야 합니다.

2 [WordPress 사용 사례: 프로젝트 생성](#)

프로젝트는 프로비저닝을 수행할 수 있는 사용자를 지정하고 프로비저닝 가능한 양을 구성합니다.

3 [WordPress 사용 사례: Blueprint 생성 및 확장](#)

개발자는 모든 클라우드 벤더에 배포 가능한 일반적인 vRealize Automation Cloud Assembly Blueprint 형식으로 WordPress 사이트를 정의할 수 있습니다.

WordPress 사용 사례: 인프라 생성

클라우드 관리자는 엔지니어링 팀이 나중에 개발 및 테스트하여 WordPress 사이트를 운영할 수 있는 리소스를 먼저 구성해야 합니다.

인프라에는 클라우드 대상 및 WordPress 사이트에 필요한 시스템, 네트워크 및 스토리지에 대한 정의가 포함됩니다.

절차

1 WordPress 사용 사례: 클라우드 계정 추가

이 단계에서는 클라우드 관리자가 클라우드 계정 2개를 추가합니다. 프로젝트는 AWS에서 개발 및 테스트 작업을 수행한 후 Azure에서 운영 환경으로 전환합니다.

2 WordPress 사용 사례: 클라우드 영역 추가

이 단계에서는 클라우드 관리자가 개발, 테스트 및 운영 환경에 대해 하나씩 총 3개의 클라우드 영역을 추가합니다.

3 WordPress 사용 사례: 플레이버 매핑 추가

이 단계에서는 클라우드 관리자가 배포 환경마다 다를 수 있는 용량 요구 사항을 고려하기 위해 플레이버 매핑을 추가합니다.

4 WordPress 사용 사례: 이미지 매핑 추가

이 단계에서, 클라우드 관리자는 Ubuntu를 위한 이미지 매핑, WordPress 서버를 위한 호스트 및 해당 MySQL 데이터베이스 서버를 추가합니다.

5 WordPress 사용 사례: 네트워크 프로파일 추가

이 단계에서는 클라우드 관리자가 각 클라우드 영역에 네트워크 프로파일을 추가합니다.

6 WordPress 사용 사례: 스토리지 프로파일 추가

이 단계에서는 클라우드 관리자가 각 클라우드 영역에 스토리지 프로파일을 추가합니다.

WordPress 사용 사례: 클라우드 계정 추가

이 단계에서는 클라우드 관리자가 클라우드 계정 2개를 추가합니다. 프로젝트는 AWS에서 개발 및 테스트 작업을 수행한 후 Azure에서 운영 환경으로 전환합니다.

절차

1 **인프라 > 연결 > 클라우드 계정**으로 이동합니다.

2 **클라우드 계정 추가**를 클릭하고 Amazon Web Services를 선택한 후 값을 입력합니다.

설정	샘플 값
액세스 키 ID	R5SDR3PXVV2ZW8B7YNSM
비밀 액세스 키	SZXAINXU4UHNAQ1E156S
이름	OurCo-AWS

설정	샘플 값
설명	WordPress
기능	cloud:aws

모든 값은 사용 사례만 사용되는 샘플입니다. 계정 세부 사항은 다를 수 있습니다.

- 3 자격 증명을 확인하려면 **검증**을 클릭합니다.
- 4 **추가**를 클릭합니다.
- 5 새로 추가된 계정 **구성**을 편집하여 **us-east-1** 및 **us-west-2** 지역으로의 프로비저닝을 허용합니다.
- 6 **클라우드 계정 추가**를 클릭하고 **Microsoft Azure**를 선택한 후 값을 입력합니다.

설정	샘플 값
구독 ID	ef2avpf-dfdv-zxlugi1i-g4h0-i8ep2jwp4c9arbf
테넌트 ID	dso9wv3-4zgc-5nrcy5h3m-4skf-nnovp40wfxsro22r
클라이언트 애플리케이션 ID	bg224oq-3ptp-mbhi6aa05-q511-uflyjr2sttyik6bs
클라이언트 애플리케이션 비밀 키	7uqxi57-0wtn-kymgf9wcj-t2l7-e52e4nu5fig4pmdd
이름	OurCo-Azure
설명	WordPress
기능	cloud:az

- 7 자격 증명을 확인하려면 **검증**을 클릭합니다.
- 8 **추가**를 클릭합니다.
- 9 새로 추가된 계정 **구성**을 편집하여 **East US** 지역으로의 프로비저닝을 허용합니다.

다음에 수행할 작업

프로젝트가 WordPress 사이트를 배포할 클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

WordPress 사용 사례: 클라우드 영역 추가

이 단계에서는 클라우드 관리자가 개발, 테스트 및 운영 환경에 대해 하나씩 총 3개의 클라우드 영역을 추가합니다.

클라우드 영역은 프로젝트가 WordPress 사이트를 지원하기 위해 시스템, 네트워크 및 스토리지를 배포하는 리소스입니다.

사전 요구 사항

클라우드 계정을 추가합니다. [WordPress 사용 사례: 클라우드 계정 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 클라우드 영역**으로 이동합니다.
- 2 **새 클라우드 영역**을 클릭하고 개발 환경에 대한 값을 입력합니다.

클라우드 영역 설정	샘플 값
계정/지역	OurCo-AWS/us-east-1
이름	OurCo-AWS-US-East
설명	WordPress
배치 정책	기본값
기능 태그	env:dev

모든 값은 사용 사례만 사용되는 샘플입니다. 실제 영역 세부 정보는 다를 수 있습니다.

- 3 **계산**을 클릭하고, 필요한 영역이 표시되는지 확인합니다.
- 4 **생성**을 클릭합니다.
- 5 테스트 및 운영 환경에 대한 값을 사용하여 이 프로세스를 두 번 반복합니다.

클라우드 영역 설정	샘플 값
계정/지역	OurCo-AWS/us-west-2
이름	OurCo-AWS-US-West
설명	WordPress
배치 정책	기본값
기능 태그	env:test

클라우드 영역 설정	샘플 값
계정/지역	OurCo-Azure/East US
이름	OurCo-Azure-East-US
설명	WordPress
배치 정책	기본값
기능 태그	env:prod

다음에 수행할 작업

크기가 서로 다른 시스템 배포를 고려하기 위해 플레이버 매핑을 추가합니다. [WordPress 사용 사례: 플레이버 매핑 추가](#) 항목을 참조하십시오.

WordPress 사용 사례: 플레이버 매핑 추가

이 단계에서는 클라우드 관리자가 배포 환경마다 다를 수 있는 용량 요구 사항을 고려하기 위해 플레이버 매핑을 추가합니다.

플레이버 매핑은 비공식적인 용어로 티셔츠 크기 조정이라고 합니다.

사전 요구 사항

클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 플레이버 매핑**으로 이동합니다. 각 클라우드 영역에 대해 소규모, 중간 규모 및 대규모 플레이버를 허용해야 합니다.
- 2 **새 플레이버 매핑**을 클릭하고 개발 클라우드 영역에 대한 값을 입력합니다.

설정	샘플 값
플레이버 이름	small
계정/지역 값	OurCo-AWS/us-east-1 t2.micro
계정/지역 값	OurCo-AWS/us-west-2 t2.micro
계정/지역 값	OurCo-Azure/East US Standard_A0

모든 값은 사용 사례만 사용되는 샘플입니다. 실제 플레이버는 다를 수 있습니다.

- 3 **생성**을 클릭합니다.
- 4 중형 및 대형 플레이버에 대한 값을 사용하여 이 프로세스를 두 번 반복합니다.

설정	샘플 값
플레이버 이름	medium
계정/지역 값	OurCo-AWS/us-east-1 t2.medium
계정/지역 값	OurCo-AWS/us-west-2 t2.medium
계정/지역 값	OurCo-Azure/East US Standard_A3

설정	샘플 값
플레이버 이름	large
계정/지역 값	OurCo-AWS/us-east-1 t2.large

설정	샘플 값
계정/지역	OurCo-AWS/us-west-2
값	t2.large
계정/지역	OurCo-Azure/East US
값	Standard_A7

다음에 수행할 작업

이미지 매핑을 추가하여 운영 체제를 계획합니다. [WordPress 사용 사례: 이미지 매핑 추가](#) 항목을 참조하십시오.

WordPress 사용 사례: 이미지 매핑 추가

이 단계에서, 클라우드 관리자는 Ubuntu를 위한 이미지 매핑, WordPress 서버를 위한 호스트 및 해당 MySQL 데이터베이스 서버를 추가합니다.

각 클라우드 영역에는 Ubuntu 이미지 매핑이 필요합니다.

사전 요구 사항

클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 이미지 매핑**으로 이동합니다.
- 2 **새 이미지 매핑**을 클릭하고 Ubuntu 서버에 대한 값을 입력합니다.

설정	샘플 값
이미지 이름	ubuntu-16
계정/지역	OurCo-AWS/us-east-1
값	ubuntu-16.04-server-cloudimg-amd64
계정/지역	OurCo-AWS/us-west-2
값	ubuntu-16.04-server-cloudimg-amd64
계정/지역	OurCo-Azure/East US
값	azul-zulu-ubuntu-1604-923eng

모든 값은 사용 사례만 사용되는 샘플입니다. 이미지는 달라집니다.

- 3 **생성**을 클릭합니다.

다음에 수행할 작업

네트워크를 추가합니다. [WordPress 사용 사례: 네트워크 프로파일 추가](#) 항목을 참조하십시오.

WordPress 사용 사례: 네트워크 프로파일 추가

이 단계에서는 클라우드 관리자가 각 클라우드 영역에 네트워크 프로파일을 추가합니다.

관리자는 WordPress 시스템이 사용할 네트워크 및 로드 밸런서의 다른 측면에서 사용할 두 번째 네트워크를 각 프로파일에 추가합니다. 두 번째 네트워크는 결과적으로 사용자가 연결할 때 사용됩니다.

사전 요구 사항

클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 네트워크 프로파일**로 이동합니다.
- 2 **새 네트워크 프로파일**을 클릭하고 개발 클라우드 영역에 사용할 프로파일을 생성합니다.

네트워크 프로파일 설정	샘플 값
계정/지역	OurCo-AWS/us-east-1
이름	devnets
설명	WordPress
기능 태그	env:dev

- 3 **네트워크**를 클릭하고 **네트워크 추가**를 클릭합니다.

- 4 wpnet 및 appnet-public을 선택하고 **추가**를 클릭합니다.

모든 값은 사용 사례만 사용되는 샘플입니다. 네트워크 이름은 다를 수 있습니다.

- 5 **생성**을 클릭합니다.

이 Wordpress 예제에서는 네트워크 정책 또는 네트워크 보안 설정을 지정할 필요가 없습니다.

- 6 프로세스를 두 번 반복하여 Wordpress 예제 테스트 클라우드 영역과 운영 클라우드 영역에 사용할 네트워크 프로파일을 생성합니다. 각 프로세스에서 wpnet 네트워크와 appnet-public 네트워크를 추가합니다.

네트워크 프로파일 설정	샘플 값
계정/지역	OurCo-AWS/us-west-2
이름	testnets
설명	WordPress
기능 태그	env:test

네트워크 프로파일 설정	값
계정/지역	OurCo-Azure/East US
이름	prodnets

네트워크 프로파일 설정	값
설명	WordPress
기능 태그	env:prod

다음에 수행할 작업

스토리지를 추가합니다. [WordPress 사용 사례: 스토리지 프로파일 추가](#) 항목을 참조하십시오.

WordPress 사용 사례: 스토리지 프로파일 추가

이 단계에서는 클라우드 관리자가 각 클라우드 영역에 스토리지 프로파일을 추가합니다.

관리자는 운영 영역에 고속 스토리지를 배치하고 개발 및 테스트 영역에 일반 스토리지를 배치합니다.

사전 요구 사항

클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 스토리지 프로파일**로 이동합니다.
- 2 **새 스토리지 프로파일**을 클릭하고 개발 클라우드 영역에 사용할 프로파일을 생성합니다.

계정/지역을 선택하면 추가적인 필드가 나타납니다.

스토리지 프로파일 설정	샘플 값
계정/지역	OurCo-AWS/us-east-1
이름	OurCo-AWS-US-East-Disk
설명	WordPress
디바이스 유형	EBS
볼륨 유형	범용 SSD
기능 태그	usage:general

모든 값은 사용 사례만 사용되는 샘플입니다.

- 3 **생성**을 클릭합니다.
- 4 이 프로세스를 반복하여 테스트 클라우드 영역에 사용할 프로파일을 생성합니다.

스토리지 프로파일 설정	샘플 값
계정/지역	OurCo-AWS/us-west-2
이름	OurCo-AWS-US-West-Disk
설명	WordPress

스토리지 프로파일 설정	샘플 값
디바이스 유형	EBS
볼륨 유형	범용 SSD
기능 태그	usage:general

- 5 이 프로세스를 반복하여 운영 클라우드 영역에 사용할 프로파일을 생성합니다. 운영 클라우드 영역은 Azure 영역이므로 설정이 다릅니다.

스토리지 프로파일 설정	샘플 값
계정/지역	OurCo-Azure/East US
이름	OurCo-Azure-East-US-Disk
설명	WordPress
스토리지 유형	관리 디스크
디스크 유형	프리미엄 LRS
OS 디스크 캐시	읽기 전용
데이터 디스크 캐시	읽기 전용
기능 태그	usage:fast

다음에 수행할 작업

사용자를 식별하고 프로비저닝 설정을 정의하기 위해 프로젝트를 생성합니다. [WordPress 사용 사례: 프로젝트 생성](#) 항목을 참조하십시오.

WordPress 사용 사례: 프로젝트 생성

프로젝트는 프로비저닝을 수행할 수 있는 사용자를 지정하고 프로비저닝 가능한 양을 구성합니다.

프로젝트는 사용자 및 프로비저닝 설정을 정의합니다.

- 사용자 및 사용자 역할의 사용 권한 수준
- 클라우드 영역에 프로비저닝할 때의 배포 우선 순위
- 클라우드 영역당 최대 배포 인스턴스 수

사전 요구 사항

클라우드 영역을 추가합니다. [WordPress 사용 사례: 클라우드 영역 추가](#) 항목을 참조하십시오.

절차

- 1 **인프라 > 구성 > 프로젝트**로 이동합니다.
- 2 **새 프로젝트**를 클릭하고 WordPress를 이름으로 입력합니다.

3 사용자를 클릭하고 **사용자 추가**를 클릭합니다.**4** 사용자의 이메일 주소와 역할을 추가합니다.

사용자를 성공적으로 추가하려면 VMware Cloud Services 관리자가 해당 사용자를 위해 vRealize Automation Cloud Assembly에 대한 액세스 권한을 부여해야 합니다.

여기에 나와 있는 주소는 단지 사용 사례를 위한 샘플입니다.

- chris.ladd@ourco.com, 멤버
- kerry.mott@ourco.com, 멤버
- pat.tubb@ourco.com, 관리자

5 프로비저닝을 클릭하고 **클라우드 영역 추가**를 클릭합니다.**6** 사용자가 배포할 수 있는 클라우드 영역을 추가합니다.

프로젝트 클라우드 영역 설정	샘플 값
클라우드 영역	OurCo-AWS-US-East
프로비저닝 우선 순위	1
인스턴스 제한	5
클라우드 영역	OurCo-AWS-US-West
프로비저닝 우선 순위	1
인스턴스 제한	5
클라우드 영역	OurCo-Azure-East-US
프로비저닝 우선 순위	0
인스턴스 제한	1

7 생성을 클릭합니다.**8 인프라 > 구성 > 클라우드 영역**으로 이동하여 **WordPress 사용 사례: 클라우드 영역 추가**에서 생성한 영역을 엽니다.**9 프로젝트**를 클릭하고, WordPress가 영역에 프로비저닝할 수 있는 프로젝트인지 확인합니다.**10 WordPress 사용 사례: 클라우드 영역 추가**에서 생성한 다른 영역을 확인합니다.

다음에 수행할 작업

기본 Blueprint를 생성합니다.

WordPress 사용 사례: Blueprint 생성 및 확장

개발자는 모든 클라우드 벤더에 배포 가능한 일반적인 vRealize Automation Cloud Assembly Blueprint 형식으로 WordPress 사이트를 정의할 수 있습니다.

사용 사례 Blueprint는 WordPress 애플리케이션 서버, MySQL 데이터베이스 서버 및 AWS, Azure 또는 vSphere 기반 클라우드에 배포 가능한 지원 구성 요소로 구성됩니다. 처음에는 Blueprint에 구성 요소 몇 개만 포함되지만 기존 구성 요소를 수정하고 더 많은 구성 요소를 추가하면서 그 크기가 증가합니다.

WordPress 사용 사례: 인프라 생성의 예제에는 클라우드 관리자가 설정한 인프라가 포함되어 있습니다.

- 클라우드 계정 2개(AWS 및 Azure)
- 클라우드 영역 환경 3개
 - 개발 - OurCo-AWS-US-East
 - 테스트 - OurCo-AWS-US-West
 - 운영 - OurCo-Azure-East-US
- 각 영역에 대해 소규모, 중간 규모 및 대규모 계산 리소스가 포함된 플레이버 매핑
- 각 영역에 대해 구성된 Ubuntu 16 이미지 매핑
- 각 영역에 대해 내부 및 외부 서브넷이 포함된 네트워크 프로파일(devnets, testnets, prodnets)
- 아카이브 디스크를 지원하는 스토리지, 개발 및 테스트를 위한 일반 스토리지 및 운영을 위한 고속 스토리지
- WordPress 프로젝트에는 세 가지 클라우드 영역 환경 모두 및 사용 사례를 사용해 볼 수 있는 사용자 포함

사전 요구 사항

인프라 값에 익숙해야 합니다. 예를 들어 이 사용 사례 예제에서는 개발 및 테스트 환경에서는 AWS를 사용하고 운영 환경에서는 Azure를 사용합니다. 고유한 Blueprint를 생성할 때는 예시에 나와 있는 값을 일반적으로 클라우드 관리자가 설정한 실제 값으로 대체하십시오.

절차

1 WordPress 사용 사례: 기본 Blueprint 생성

개발자인 경우 애플리케이션 서버 1개만 포함되어 있는 Blueprint 같이 최소한의 WordPress 구성 요소만 포함되어 있는 vRealize Automation Cloud Assembly Blueprint로 작업을 시작합니다.

2 WordPress 사용 사례: 기본 Blueprint 테스트

개발 중에는 일반적으로 필수 구성 요소를 사용하여 vRealize Automation Cloud Assembly Blueprint를 구축한 다음 Blueprint의 증가에 따라 배포 및 테스트를 수행합니다.

3 WordPress 사용 사례: Blueprint 확장

기본 vRealize Automation Cloud Assembly Blueprint를 생성하고 테스트한 후에는 개발, 테스트를 거쳐 최종적으로 운영 환경에 배포할 수 있는 여러 계층의 애플리케이션으로 확장합니다.

WordPress 사용 사례: 기본 Blueprint 생성

개발자인 경우 애플리케이션 서버 1개만 포함되어 있는 Blueprint 같이 최소한의 WordPress 구성 요소만 포함되어 있는 vRealize Automation Cloud Assembly Blueprint로 작업을 시작합니다.

vRealize Automation Cloud Assembly는 인프라를 코드로 구현하는 도구입니다. 먼저 설계 캔버스로 구성 요소를 끌어 와서 작업을 시작합니다. 그런 다음 캔버스의 오른쪽에 있는 코드 편집기를 사용하여 세부 정보를 완성합니다.

코드 편집기를 사용하면 코드를 직접 입력하고, 잘라내고, 붙여 넣을 수 있습니다. 코드 편집이 불편하면 캔버스에서 리소스를 선택하고 코드 편집기 **속성** 탭을 클릭한 다음 그 곳에 값을 입력하면 됩니다. 입력한 값이 마치 직접 입력한 것처럼 코드에 나타납니다.

사전 요구 사항

인프라에 익숙해야 합니다. 여기에 나와 있는 예시에서는 **WordPress 사용 사례: 인프라 생성**의 인프라 값을 사용하지만 이러한 값은 실제 값으로 대체해야 합니다.

절차

- 1 **Blueprint**로 이동하여 **새로 만들기**를 클릭합니다.
- 2 Blueprint 이름을 **Wordpress-BP**로 지정합니다.
- 3 **WordPress** 프로젝트를 선택하고 **생성**을 클릭합니다.
- 4 Blueprint 설계 페이지의 왼쪽에서 클라우드 애그노스틱 시스템 2개를 캔버스로 끌어 옵니다.
이 두 시스템은 WordPress 애플리케이션 서버(WebTier)와 MySQL 데이터베이스 서버(DBTier)로 사용됩니다.
- 5 오른쪽에서 시스템 **YAML** 코드를 편집하여 이름, 이미지, 플레이버 및 제약 조건 태그를 추가합니다.

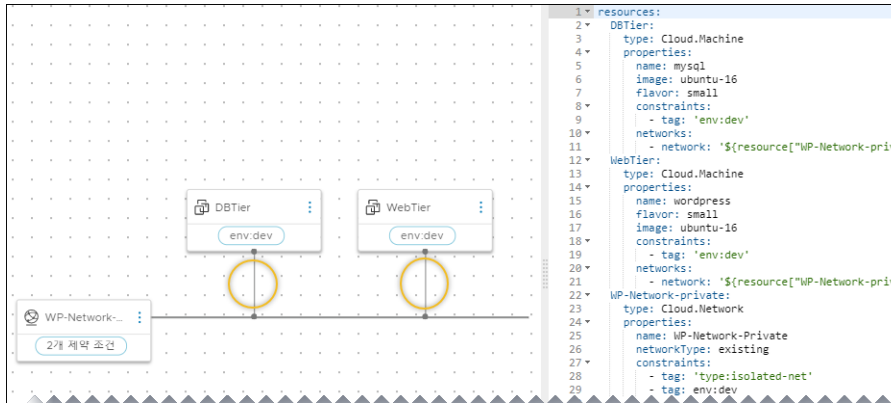
```
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: 'ubuntu-16'
      flavor: 'small'
      constraints: - tag: env:dev
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      image: 'ubuntu-16'
      flavor: 'small'
      constraints: - tag: env:dev
```

- 6 클라우드 애그노스틱 네트워크를 캔버스로 끌어 와서 해당 코드를 편집합니다.

```
WP-Network-Private:
  type: Cloud.Network
  properties:
    name: WP-Network-Private
    networkType: existing
    constraints: - tag: 'type:isolated-net' - tag: 'env:dev'
```

- 7 시스템을 네트워크에 연결합니다.

선이 네트워크 블록과 만나는 지점을 클릭하여 누른 상태에서 시스템 블록까지 끌고 가서 마우스 버튼을 놓습니다.



편집기를 확인하면 두 시스템에 네트워크 코드가 추가되는 것을 볼 수 있습니다.

```
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: 'ubuntu-16'
      flavor: 'small'
      constraints:
        - tag: env:dev
      networks: - network: '${resource["WP-Network-Private"].id}'
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      image: 'ubuntu-16'
      flavor: 'small'
      constraints:
        - tag: env:dev
      networks: - network: '${resource["WP-Network-Private"].id}'
```

8 사용자 입력 프롬프트를 추가합니다.

다음과 같은 경우에는 여러 옵션을 사용하도록 사용 사례 인프라가 설정되었습니다. 예를 들면 다음과 같습니다.

- 개발, 테스트 및 운영을 위한 클라우드 영역 환경
- 소규모, 중간 규모 및 대규모 시스템을 위한 플레이버 매핑
- 일반적인 사용 환경 및 고속 사용 환경을 위한 스토리지 디스크 속도

특정 옵션을 Blueprint에 직접 설정할 수도 있지만, Blueprint 배포 시 사용자가 옵션을 선택할 수 있게 하는 방법이 더 좋습니다. 사용자에게 입력을 요청하면 하드 코딩된 Blueprint 여러 개를 사용하는 대신 다양한 방식으로 배포할 수 있는 하나의 Blueprint를 생성할 수 있습니다.

- a 배포 시 사용자가 시스템 크기 및 대상 환경을 선택할 수 있도록 코드에 inputs 섹션을 생성합니다. 선택할 수 있는 값을 정의합니다.

```
inputs:
  env:
    type: string
    enum:
      - 'env:dev'
      - 'env:prod'
      - 'env:test'
    default: 'env:dev'
    title: Environment
    description: Target Environment
  size:
    type: string
    enum:
      - small
      - medium
      - large
    description: Size of Nodes
    title: Tier Machine Size
```

- b 코드의 resources 섹션에 사용자의 선택을 요청하는 `${input.input-name}` 코드를 추가합니다.

```
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: 'ubuntu-16'
      flavor: '${input.size}'
      constraints:
        - tag: '${input.env}'
    networks:
      - network: '${resource["WP-Network-Private"].id}'
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      image: 'ubuntu-16'
      flavor: '${input.size}'
      constraints:
        - tag: '${input.env}'
    networks:
      - network: '${resource["WP-Network-Private"].id}'
  WP-Network-Private:
    type: Cloud.Network
    properties:
```

```
name: WP-Network-Private
networkType: existing
constraints:
  - tag: 'type:isolated-net'
  - tag: '${input.env}'
```

- 9 마지막으로 다음의 예시를 사용하여 WebTier 및 DBTier 코드를 향상시킵니다. WP-Network-Private 코드는 추가적으로 변경할 필요가 없습니다.

향상된 기능에는 데이터베이스 서버로의 로그인 액세스, 데이터베이스 디스크 및 배포 시점의 cloudConfig 초기화 스크립트가 포함됩니다.

구성 요소	예
추가적인 DBTier 입력	<pre> username: type: string minLength: 4 maxLength: 20 pattern: '[a-z]+' title: Database Username description: Database Username userpassword: type: string pattern: '[a-z0-9A-Z@#]+\$' encrypted: true title: Database Password description: Database Password databaseDiskSize: type: number default: 4 maximum: 10 title: MySQL Data Disk Size description: Database Disk Size </pre>
DBTier 리소스	<pre> DBTier: type: Cloud.Machine properties: name: mysql image: ubuntu-16 flavor: '\${input.size}' constraints: - tag: '\${input.env}' networks: - network: '\${resource["WP-Network-Private"].id}' assignPublicIpAddress: true remoteAccess: authentication: usernamePassword username: '\${input.username}' password: '\${input.userpassword}' cloudConfig: #cloud-config repo_update: true repo_upgrade: all packages: - mysql-server runcmd: - sed -e '/bind-address/ s/^#*/#/' -i /etc/mysql/mysql.conf.d/ mysqlld.cnf - service mysql restart - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';" - mysql -e "FLUSH PRIVILEGES;" attachedDisks: [] </pre>
WebTier 리소스	<pre> WebTier: type: Cloud.Machine properties: name: wordpress flavor: '\${input.size}' </pre>

구성 요소	예
	<pre> image: ubuntu-16 constraints: - tag: '\${input.env}' networks: - network: '\${resource["WP-Network-Private"].id}' assignPublicIpAddress: true cloudConfig: #cloud-config repo_update: true repo_upgrade: all packages: - apache2 - php - php-mysql - libapache2-mod-php - php-mcrypt - mysql-client runcmd: - mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://wordpress.org/latest.tar.gz && tar -xzf /var/www/html/ latest.tar.gz -C /var/www/html/mywordpresssite --strip-components 1 - i=0; while [\$i -le 5]; do mysql --connect-timeout=3 -h \$ {DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break sleep 15; i=\$((i+1)); done - mysql -u root -pmysqlpassword -h \${DBTier.networks[0].address} -e "create database wordpress_blog;" - mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/ html/mywordpresssite/wp-config.php - sed -i -e s/"define('DB_NAME', 'database_name_here');"/"define('DB_NAME', 'wordpress_blog');"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e s/"define('DB_USER', 'username_here');"/"define('DB_USER', 'root');"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e s/"define('DB_PASSWORD', 'password_here');"/"define('DB_PASSWORD', 'mysqlpassword');"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e s/"define('DB_HOST', 'localhost');"/"define('DB_HOST', '\$ {DBTier.networks[0].address}');"/ /var/www/html/mywordpresssite/wp- config.php - service apache2 reload </pre>

예제: 완성된 기본 Blueprint 코드 예시

```

inputs:
  env:
    type: string
    enum:
      - 'env:dev'
      - 'env:prod'
      - 'env:test'
    default: 'env:dev'
    title: Environment
    description: Target Environment
  size:
    type: string
    enum:

```

```

    - small
    - medium
    - large
  description: Size of Nodes
  title: Tier Machine Size
username:
  type: string
  minLength: 4
  maxLength: 20
  pattern: '[a-z]+'
  title: Database Username
  description: Database Username
userpassword:
  type: string
  pattern: '[a-z0-9A-Z@#&$]+'
  encrypted: true
  title: Database Password
  description: Database Password
databaseDiskSize:
  type: number
  default: 4
  maximum: 10
  title: MySQL Data Disk Size
  description: Database Disk Size
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: ubuntu-16
      flavor: '${input.size}'
      constraints:
        - tag: '${input.env}'
    networks:
      - network: '${resource["WP-Network-Private"].id}'
        assignPublicIpAddress: true
    remoteAccess:
      authentication: usernamePassword
      username: '${input.username}'
      password: '${input.userpassword}'
    cloudConfig: |
      #cloud-config
      repo_update: true
      repo_upgrade: all

    packages:
      - mysql-server

    runcmd:
      - sed -e '/bind-address/ s/^#*#/' -i /etc/mysql/mysql.conf.d/mysqld.cnf
      - service mysql restart
      - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';"
      - mysql -e "FLUSH PRIVILEGES;"
    attachedDisks: []
  WebTier:

```

```

type: Cloud.Machine
properties:
  name: wordpress
  flavor: '${input.size}'
  image: ubuntu-16
  constraints:
    - tag: '${input.env}'
  networks:
    - network: '${resource["WP-Network-Private"].id}'
      assignPublicIpAddress: true
  cloudConfig: |
    #cloud-config
    repo_update: true
    repo_upgrade: all

  packages:
    - apache2
    - php
    - php-mysql
    - libapache2-mod-php
    - php-mcrypt
    - mysql-client

  runcmd:
    - mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
    - i=0; while [ $i -le 5 ]; do mysql --connect-timeout=3 -h $
{DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
    - mysql -u root -pmysqlpassword -h ${DBTier.networks[0].address} -e "create database
wordpress_blog;"
    - mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
    - sed -i -e s/"define( 'DB_NAME', 'database_name_here' );"/"define( 'DB_NAME',
'wordpress_blog' );"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define( 'DB_USER', 'username_here' );"/"define( 'DB_USER', 'root' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_PASSWORD',
'password_here' );"/"define( 'DB_PASSWORD', 'mysqlpassword' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_HOST',
'localhost' );"/"define( 'DB_HOST', '${DBTier.networks[0].address}' );"/ /var/www/html/
mywordpresssite/wp-config.php
    - service apache2 reload
WP-Network-Private:
  type: Cloud.Network
  properties:
    name: WP-Network-Private
    networkType: existing
    constraints:
      - tag: 'type:isolated-net'
      - tag: '${input.env}'

```

다음에 수행할 작업

문서를 검사하고 배포하여 Blueprint를 테스트합니다.

WordPress 사용 사례: 기본 Blueprint 테스트

개발 중에는 일반적으로 필수 구성 요소를 사용하여 vRealize Automation Cloud Assembly Blueprint를 구축한 다음 Blueprint의 증가에 따라 배포 및 테스트를 수행합니다.

배포가 원하는 방식으로 작동하는지 확인하기 위해 Blueprint를 여러 번 테스트하고 배포할 수 있습니다. 점차적으로 더 많은 구성 요소를 추가하면서 다시 테스트하고 다시 배포합니다.

사전 요구 사항

기본 Blueprint를 생성합니다. [WordPress 사용 사례: 기본 Blueprint 생성](#) 항목을 참조하십시오.

절차

- 1 Blueprint를 클릭하고 WordPress-BP Blueprint를 엽니다.

설계 캔버스 및 코드 편집기에 기본 Blueprint가 나타납니다.

- 2 Blueprint 구문, 배치 및 기본 유효성을 검사하려면 왼쪽 아래에서 **테스트**를 클릭합니다.

- 3 입력 값을 입력하고 **테스트**를 클릭합니다.

Environment	env:dev	▼ ⓘ
Database Tier Size *	small	▼ ⓘ
Database Username *	ouradmin	
Database Password *	••••••	
MySQL Data Disk Size	4	⬆️⬆️ ⓘ

이 테스트는 시뮬레이션일 뿐이며 실제로 가상 시스템이나 다른 리소스를 배포하지는 않습니다. 시뮬레이션에서는 잠재적 문제(예: Blueprint의 경성 제약 조건과 일치하는 리소스 기능이 정의되어 있지 않음)를 노출합니다.

테스트에는 **프로비저닝 다이어그램**에 대한 링크가 포함되어 있으며 여기에서 시뮬레이션된 배포 흐름을 검사하고 발생한 오류를 확인할 수 있습니다.

요청 세부 정보

NETWORK ALLOCATION

요청: WP-Network

오류: Could not find any profile to match network 'WP-Network' of type 'PUBLIC' with constraints '[env:dev]'.

요청 유형: Allocation

네트워크 유형: PUBLIC

인터넷 연결:

제약 조건: env:dev:hard

프로젝트: wordpress project

네트워크 제약 조건: 없음

스토리지 제약 조건: 없음

확장성 제약 조건: 없음

시뮬레이션에 성공해도 Blueprint를 오류 없이 배포할 수 있다고 보장되는 것은 아닙니다.

- Blueprint가 시뮬레이션을 통과한 후 왼쪽 아래에서 **배포**를 클릭합니다.
- 새 배포 생성**을 선택합니다.
- 배포 이름을 **WordPress for OurCo**로 지정하고 **다음**을 클릭합니다.
- 입력 값을 입력하고 **배포**를 클릭합니다.
- Blueprint가 성공적으로 배포되었는지 확인하려면 **배포** 아래를 확인합니다.

배포가 실패한 경우, 해당 이름을 클릭한 후 **기록** 탭을 클릭하여, 문제 해결을 도와 줄 메시지를 봅니다.

타임 스탬프	상태	리소스 유형	리소스 이름	세부 정보
2020년 1월 21일 오전 9:41:32	REQUEST_FINISHED			
2020년 1월 21일 오전 9:41:31	COMPLETION_FINISHED			
2020년 1월 21일 오전 9:41:14	COMPLETION_IN_PROGRESS			
2020년 1월 21일 오전 9:40:51	CREATE_FINISHED	Cloud.Machine	Cloud_vSphere_Machine_1[...	
2020년 1월 21일 오전 9:33:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[...	Request is in stage STARTED and substage RESOURCE_COUNTED
2020년 1월 21일 오전 9:31:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[...	

일부 기록 항목의 맨 오른쪽에 **프로비저닝 다이어그램** 링크가 있을 수 도 있습니다. 다이어그램은 시뮬레이션된 것과 유사하며, 프로비저닝 프로세스에서 vRealize Automation Cloud Assembly 결정 시점의 순서도를 검사할 수 있습니다.

인프라 > 작업 > 요청 아래에서 더 많은 순서도를 확인할 수 있습니다.

- 9 애플리케이션이 작동하는지 확인하려면 브라우저에서 **WordPress** 시작 페이지를 엽니다.
 - a **WordPress** 서버가 완전히 생성되고 초기화될 때까지 기다립니다.
환경에 따라 초기화에 30분 이상이 소요될 수 있습니다.
 - b 사이트 FQDN 또는 IP 주소를 찾으려면 **배포 > 토폴로지**로 이동합니다.
 - c 캔버스에서 **WebTier**를 클릭하면 오른쪽 패널에 IP 주소가 표시됩니다.
 - d **WordPress** 시작 페이지에 대한 전체 URL의 일부로 IP 주소를 입력합니다.
이 사용 사례에서 전체 URL은 다음과 같습니다.
`http://{IP-address}/mywordpresssite`
또는
`http://{IP-address}/mywordpresssite/wp-admin/install.php`
- 10 브라우저에서 **WordPress**를 검사한 후 애플리케이션 작업이 더 필요하면 **Blueprint**를 변경한 후 **기존 배포 업데이트** 옵션을 사용하여 다시 배포합니다.
- 11 **Blueprint** 버전 관리를 고려합니다. 버전을 관리하면 변경 내용 때문에 배포가 실패할 경우에 작동하는 버전으로 되돌릴 수 있습니다.
 - a **Blueprint** 설계 페이지에서 **버전**을 클릭합니다.
 - b 버전 생성 페이지에서 **WP-1.0**을 입력합니다.
버전 이름에는 공백을 입력하지 마십시오.
 - c **생성**을 클릭합니다.
검토하거나 특정 버전으로 되돌리려면 설계 페이지에서 **버전 기록** 탭을 클릭합니다.
- 12 이제 기본 배포가 가능하므로 애플리케이션 및 데이터베이스 서버의 CPU와 메모리를 늘려서 첫 번째 배포 시 기능 향상을 테스트합니다.
둘 모두에서 노드를 보통 크기로 업데이트합니다. 동일한 **Blueprint**를 사용하여, 배포 시 **중간**을 선택한 후 애플리케이션을 다시 배포한 후 다시 확인합니다.

다음에 수행할 작업

더 많은 구성 요소를 추가하여 **Blueprint**를 운영 가능한 애플리케이션으로 확장합니다.

WordPress 사용 사례: Blueprint 확장

기본 vRealize Automation Cloud Assembly **Blueprint**를 생성하고 테스트한 후에는 개발, 테스트를 거쳐 최종적으로 운영 환경에 배포할 수 있는 여러 계층의 애플리케이션으로 확장합니다.

Blueprint를 확장하려면 다음과 같은 향상된 기능을 추가합니다.

- 용량 증가를 위해 애플리케이션 서버를 클러스터링하는 옵션
- 애플리케이션 서버 앞의 공용 네트워크 및 로드 밸런서

■ 아카이브 스토리지가 있는 백업 서버

사전 요구 사항

기본 Blueprint를 생성하여 테스트합니다. [WordPress 사용 사례: 기본 Blueprint 생성 및 WordPress 사용 사례: 기본 Blueprint 테스트](#) 항목을 참조하십시오.

절차

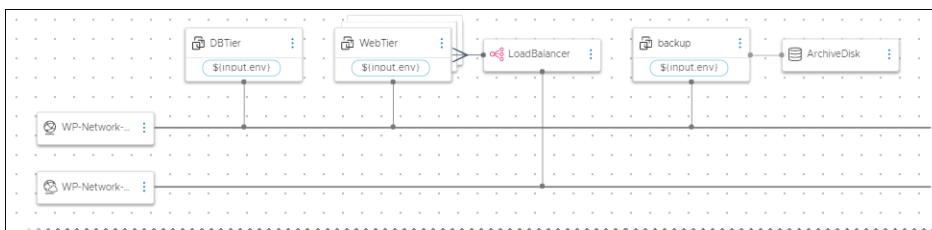
1 Blueprint를 클릭하고 WordPress-BP Blueprint를 엽니다.

설계 캔버스 및 코드 편집기에 기본 Blueprint가 나타납니다.

2 코드 예시 및 그림을 참조하여 내용을 추가하거나 변경합니다.

GUI를 사용하여 로드 밸런서와 같은 새로운 리소스를 캔버스에 끈 다음 코드 편집기에서 구성을 완료합니다.

- WordPress 애플리케이션 서버를 클러스터로 만들려면 count 입력 프롬프트를 추가합니다.
- 클라우드 애그노스틱 로드 밸런서를 추가합니다.
- 로드 밸런서를 WordPress 애플리케이션 서버 클러스터에 연결합니다.
- 클라우드 애그노스틱 백업 시스템을 추가합니다.
- 백업 시스템을 전용/내부 네트워크에 연결합니다.
- 클라우드 애그노스틱 공용/외부 네트워크를 추가합니다.
- 로드 밸런서를 공용 네트워크에 연결합니다.
- 아카이브 디스크로 사용하기 위해 클라우드 애그노스틱 스토리지 볼륨을 추가합니다.
- 아카이브 디스크를 백업 시스템에 연결합니다.
- 스토리지 디스크 속도에 대한 archiveusage 입력 프롬프트를 추가합니다.
- 스토리지 디스크 크기에 대한 archiveDiskSize 입력 프롬프트를 추가합니다.



3 기본 Blueprint에 대해 수행했던 것과 동일한 방식으로 배포, 테스트 및 변경합니다.

기존 배포를 업데이트하거나 새 인스턴스를 배포하여 배포를 비교할 수 있습니다.

목표는 운영 배포에 사용할 수 있는 신뢰할 수 있고 반복 가능한 Blueprint에 도달하는 것입니다.

예제: 완성된 확장 Blueprint 코드 예

```

inputs:
  env:
    type: string
    enum:
      - 'env:dev'
      - 'env:prod'
      - 'env:test'
    default: 'env:dev'
    title: Environment
    description: Target Environment
  size:
    type: string
    enum:
      - small
      - medium
      - large
    description: Size of Nodes
    title: Tier Machine Size
  username:
    type: string
    minLength: 4
    maxLength: 20
    pattern: '[a-z]+'
    title: Database Username
    description: Database Username
  userpassword:
    type: string
    pattern: '[a-z0-9A-Z@#&]+$'
    encrypted: true
    title: Database Password
    description: Database Password
  databaseDiskSize:
    type: number
    default: 4
    maximum: 10
    title: MySQL Data Disk Size
    description: Database Disk Size

count: type: integer default: 2 maximum: 5 minimum: 2 title: WordPress Cluster Size
description: WordPress Cluster Size (Number of Nodes) archiveDiskSize: type: number default:
4 maximum: 10 title: WordPress Archive Disk Size description: Archive Storage Disk Speed
archiveusage: type: string enum: - 'usage:general' - 'usage:fast' description: Archive
Storage Disk Speed title: Archive Disk Speed
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: ubuntu-16
      flavor: '${input.size}'
      constraints:
        - tag: '${input.env}'

```



```

networks:
  - network: '${resource["WP-Network-Private"].id}'
    assignPublicIpAddress: true
remoteAccess:
  authentication: usernamePassword
  username: '${input.username}'
  password: '${input.userpassword}'
cloudConfig: |
  #cloud-config
  repo_update: true
  repo_upgrade: all

  packages:
    - mysql-server

  runcmd:
    - sed -e '/bind-address/ s/^#*\/#/' -i /etc/mysql/mysql.conf.d/mysqld.cnf
    - service mysql restart
    - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';"
    - mysql -e "FLUSH PRIVILEGES;"

  attachedDisks: []
WebTier:
  type: Cloud.Machine
  properties:
    name: wordpress
    flavor: '${input.size}'
    image: 'ubuntu-16'
    count: '${input.count}'
  constraints:
    - tag: '${input.env}'
  networks:
    - network: '${resource["WP-Network-Private"].id}'
      assignPublicIpAddress: true
  storage: disks: - capacityGb: '${input.archiveDiskSize}' name: ArchiveDisk
  cloudConfig: |
    #cloud-config
    repo_update: true
    repo_upgrade: all

    packages:
      - apache2
      - php
      - php-mysql
      - libapache2-mod-php
      - php-mcrypt
      - mysql-client

    runcmd:
      - mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
      - i=0; while [ $i -le 10 ]; do mysql --connect-timeout=3 -h $
${DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
      - mysql -u root -pmysqlpassword -h ${DBTier.networks[0].address} -e "create database

```

```
wordpress_blog;"
- mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
- sed -i -e s/"define( 'DB_NAME', 'database_name_here' );"/"define( 'DB_NAME',
'wordpress_blog' );"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define( 'DB_USER', 'username_here' );"/"define( 'DB_USER', 'root' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_PASSWORD',
'password_here' );"/"define( 'DB_PASSWORD', 'mysqlpassword' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_HOST',
'localhost' );"/"define( 'DB_HOST', '${DBTier.networks[0].address}' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_HOST',
'localhost' );"/"define( 'DB_HOST', '${DBTier.networks[0].address}' );"/ /var/www/html/
mywordpresssite/wp-config.php
- service apache2 reload

LoadBalancer: type: Cloud.LoadBalancer properties: name: myapp-lb network: '${resource["WP-
Network-Public"].id}' instances: - '${WebTier.id}' routes: - protocol: HTTP port: '80'
instanceProtocol: HTTP instancePort: '80' healthCheckConfiguration: protocol: HTTP port: '80'
urlPath: /mywordpresssite/wp-admin/install.php intervalSeconds: 6 timeoutSeconds: 5
unhealthyThreshold: 2 healthyThreshold: 2 internetFacing: true
WP-Network-Private:
  type: Cloud.Network
  properties:
    name: WP-Network-Private
    networkType: existing
    constraints:
      - tag: 'type:isolated-net'
      - tag: '${input.env}'

WP-Network-Public: type: Cloud.Network properties: name: WP-Network-Public networkType:
public constraints: - tag: 'type:public-net' - tag: '${input.env}' backup: type:
Cloud.Machine properties: name: backup flavor: '${input.size}' image: 'ubuntu-16' networks: -
network: '${resource["WP-Network-Private"].id}' constraints: - tag: '${input.env}'
attachedDisks: - source: '${ArchiveDisk.id}' ArchiveDisk: type: Cloud.Volume properties:
name: ArchiveDisk capacityGb: 5 constraints: - tag: '${input.archiveusage}' - tag: '$
{input.env}'
```

다음에 수행할 작업

고유한 인프라를 정의하고 고유한 Blueprint를 생성합니다.

[장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#) 및 [장 6 vRealize Automation Cloud Assembly 배포 설계 항목](#)을 참조하십시오.

VMware Cloud on AWS 사용 사례

이 vRealize Automation Cloud Assembly 사용 사례는 VMware Cloud on AWS 환경에 배포할 리소스 인프라 및 Blueprint 설정을 정의하는 프로세스를 보여줍니다.

이 절차를 수행하려면 [VMware Cloud on AWS 시작 설명서](#)의 "소프트웨어 정의 데이터 센터 배포 및 관리"에 설명된 대로 관리자가 귀사의 VMware Cloud on AWS SDDC 데이터 센터를 이미 구성한 상태여야 합니다.

VMware Cloud on AWS에 대한 환경을 구성하는 프로세스를 이해하려면 순차적 설정을 확인하십시오. 여기에 나와 있는 값은 사용 사례 예시에만 해당합니다. 클라우드 인프라 및 배포 요구 사항에 맞게 고유한 값으로 대체하거나 예시의 값에서 추론할 수 있는 부분에 대해 생각해 보십시오.

유사한 워크플로에 대한 자세한 비디오는 [Cloud Assembly용 VMware Cloud on AWS를 구성하는 방법](#)의 "VMware 클라우드 관리 기술 마케팅"에서 확인할 수 있습니다.

절차

- 1 [vRealize Automation Cloud Assembly에서 기본 VMware Cloud on AWS 워크플로 구성](#)
이 vRealize Automation Cloud Assembly 사용 사례는 VMware Cloud on AWS 환경에 배포하기 위해 리소스 인프라와 해당 Blueprint를 정의하는 프로세스를 보여 줍니다.
- 2 [vRealize Automation Cloud Assembly의 VMware Cloud on AWS 워크플로에서 격리된 네트워크 구성](#)
이 절차에서는 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대해 격리된 네트워크를 추가합니다.

vRealize Automation Cloud Assembly에서 기본 VMware Cloud on AWS 워크플로 구성

이 vRealize Automation Cloud Assembly 사용 사례는 VMware Cloud on AWS 환경에 배포하기 위해 리소스 인프라와 해당 Blueprint를 정의하는 프로세스를 보여 줍니다.

이 절차에서 기존 VMware Cloud on AWS 환경의 리소스에 대한 Blueprint 배포를 지원하는 인프라를 구성합니다.

사전 요구 사항

- vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 클라우드 계정을 생성하고 구성하려면 우선 기존 VMware Cloud on AWS SDDC 환경에서 조직의 일부여야 합니다. VMware Cloud on AWS 서비스 구성에 대한 자세한 내용은 [VMware Cloud on AWS 설명서](#)를 참조하십시오.
- vCenter에 있는 기존 VMware Cloud on AWS 호스트 SDDC와 vRealize Automation Cloud Assembly에 있는 VMware Cloud on AWS 클라우드 계정 간에 필요한 연결을 용이하게 하려면 네트워크 연결을 제공하고 VPN 또는 유사한 네트워킹 수단을 사용하여 방화벽 규칙을 추가해야 합니다. [vRealize Automation에서 VMware Cloud on AWS 클라우드 계정에 연결할 VMware Cloud on AWS SDDC 준비](#)의 내용을 참조하십시오.

절차

- 1 [vRealize Automation에서 VMware Cloud on AWS 클라우드 계정에 연결할 VMware Cloud on AWS SDDC 준비](#)
vRealize Automation Cloud Assembly 온-프레미스 환경에서 VMware Cloud on AWS 클라우드 계정을 사용하는 경우 vCenter의 SDDC와 vRealize Automation의 VMware Cloud on AWS 클라우드 계정 간에 통신을 지원하도록 네트워크 연결을 생성해야 합니다.

- 2 샘플 워크플로 내의 **vRealize Automation**에서 **VMware Cloud on AWS** 클라우드 계정 생성
이 단계에서는 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정을 생성합니다.
- 3 **vRealize Automation Cloud Assembly**에서 **VMware Cloud on AWS** 배포에 대한 클라우드 영역 생성
이 단계에서는 클라우드 영역을 생성하고 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS로 작업할 때 CloudAdmin 사용자가 액세스할 수 있는 계산 리소스를 지정합니다.
- 4 **vRealize Automation Cloud Assembly**에서 **VMware Cloud on AWS** 배포를 위한 네트워크 및 스토리지 프로파일 구성
이 단계에서는 네트워크 프로파일 및 스토리지 프로파일을 구성하여 vRealize Automation Cloud Assembly의 VMware Cloud on AWS CloudAdmin 사용자가 사용할 수 있는 리소스를 지정합니다.
- 5 **vRealize Automation Cloud Assembly**에서 **VMware Cloud on AWS** 배포를 지원하는 프로젝트 생성
이 단계에서는 VMware Cloud on AWS 배포에 사용할 수 있는 리소스를 제어하는 데 사용하는 vRealize Automation Cloud Assembly 프로젝트를 정의합니다.
- 6 **vRealize Automation Cloud Assembly**에서 **VMware Cloud on AWS** 배포를 지원하도록 Blueprint 설계에 **vCenter** 시스템 리소스 정의
이 단계에서는 vCenter 시스템 리소스를 설계 캔버스에 끌어와서 VMware Cloud on AWS 배포에 대한 설정을 추가합니다.

vRealize Automation에서 VMware Cloud on AWS 클라우드 계정에 연결할 VMware Cloud on AWS SDDC 준비

vRealize Automation Cloud Assembly 온-프레미스 환경에서 VMware Cloud on AWS 클라우드 계정을 사용하는 경우 vCenter의 SDDC와 vRealize Automation의 VMware Cloud on AWS 클라우드 계정 간에 통신을 지원하도록 네트워크 연결을 생성해야 합니다.

vCenter의 기존 VMware Cloud on AWS 호스트 SDDC와 vRealize Automation의 VMware Cloud on AWS 클라우드 계정 사이에 필요한 연결을 용이하게 하려면 VPN 또는 유사한 네트워킹 수단을 사용하여 두 요소 간에 네트워크 연결을 제공해야 합니다.

절차

- 1 공용 인터넷 또는 AWS Direct Connect를 통해 VPN 연결을 구성합니다.
VMware Cloud on AWS 설명서에서 "VMware Cloud on AWS 네트워킹 및 보안" 을 참조하십시오.
- 2 관리 네트워크의 개인 IP 주소에서 vCenter Server FQDN을 확인할 수 있는지 확인합니다.
VMware Cloud on AWS 설명서에서 "VMware Cloud on AWS 네트워킹 및 보안" 을 참조하십시오.

3 필요한 방화벽 규칙을 구성합니다.

통신을 지원하도록 SDDC의 VMware Cloud on AWS 콘솔에서 관리 게이트웨이 방화벽 규칙을 구성해야 합니다. 규칙은 **관리 게이트웨이** 방화벽 규칙 섹션에 있어야 합니다. 방화벽 규칙은 SDDC 콘솔에서 **네트워킹 및 보안** 탭의 옵션을 사용하여 생성합니다.

- HTTPS(TCP 443) 서비스에 대한 ESXi로의 네트워크 트래픽을 vRealize Automation 장치/서버 또는 vRealize Automation 로드 밸런서 VIP의 검색된 IP 주소로 제한합니다.
- ICMP(All ICMP), SSO(TCP 7444) 및 HTTPS(TCP 443) 서비스에 대한 vCenter로의 네트워크 트래픽을 vRealize Automation 장치/서버 또는 vRealize Automation 로드 밸런서 VIP의 검색된 IP 주소로 제한합니다.
- HTTPS(TCP 443) 서비스에 대한 NSX-T Manager로의 네트워크 트래픽을 vRealize Automation 장치/서버 또는 vRealize Automation 로드 밸런서 VIP의 검색된 IP 주소로 제한합니다.

필요한 방화벽 규칙은 다음 표에 요약되어 있습니다.

표 3-1. 필요한 관리 게이트웨이 방화벽 규칙 요약

이름	소스	대상	서비스
vCenter	온-프레미스 데이터 센터의 CIDR 블록	vCenter	임의(모든 트래픽)
vCenter ping	임의	vCenter	ICMP(모든 ICMP)
NSX Manager	온-프레미스 데이터 센터의 CIDR 블록	NSX Manager	임의(모든 트래픽)
온-프레미스에서 ESXi로 ping	온-프레미스 데이터 센터의 CIDR 블록	ESXi 관리만	ICMP(모든 ICMP)
온-프레미스에서 ESXi로 원격 콘솔 및 프로비저닝	온-프레미스 데이터 센터의 CIDR 블록	ESXi 관리만	TCP 902
온-프레미스에서 SDDC VM으로	온-프레미스 데이터 센터의 CIDR 블록	SDDC 논리적 네트워크의 CIDR 블록	임의(모든 트래픽)
SDDC VM에서 온-프레미스로	SDDC 논리적 네트워크의 CIDR 블록	온-프레미스 데이터 센터의 CIDR 블록	임의(모든 트래픽)

관련 정보는 [VMware Cloud on AWS 설명서](#)에서 "VMware Cloud on AWS 네트워킹 및 보안" 및 "VMware Cloud on AWS 작업 가이드" 를 참조하십시오.

결과

필요한 게이트웨이 액세스 및 방화벽 규칙을 구성한 후에는 VMware Cloud on AWS 클라우드 계정 생성 프로세스를 계속할 수 있습니다.

샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성

이 단계에서는 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정을 생성합니다.

관련 정보는 [VMware Cloud on AWS 설명서](#)를 참조하십시오.

달리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- 이 절차는 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 비롯해 필요한 관리자 자격 증명이 있고 포트 443에서 HTTPS 액세스를 사용하도록 설정했다고 가정합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.](#)
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- vCenter의 기존 VMware Cloud on AWS 호스트 SDDC와 vRealize Automation의 VMware Cloud on AWS 클라우드 계정 사이에 필요한 연결을 용이하게 하려면, VPN 또는 유사한 네트워킹 수단을 사용하여 네트워크 연결 및 방화벽 규칙을 제공해야 합니다. [vRealize Automation에서 VMware Cloud on AWS 클라우드 계정에 연결할 VMware Cloud on AWS SDDC 준비의 내용](#)을 참조하십시오. 외부 HTTP 인터넷 프록시를 사용 중인 경우에는 해당 프록시를 IPv4에 대해 구성해야 합니다.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성합니다. [vRealize Automation에 대한 인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 클라우드 계정**을 선택합니다.
- 2 **클라우드 계정 추가**를 클릭하고 VMware Cloud on AWS를 선택한 후 값을 입력합니다.

샘플 값 및 지원 정보는 다음 표에 나와 있습니다.

설정	샘플 값 및 지침	설명
VMC API 토큰	<ol style="list-style-type: none"> 1 VMC API 토큰 줄 끝에 있는 "i" 도움말 아이콘을 클릭하고 도움말 텍스트 상자에서 API 토큰 페이지를 클릭하여 조직의 내 계정 페이지에서 API 토큰 탭을 엽니다. 2 토큰 생성을 클릭하여 새 API 토큰 생성 옵션을 표시합니다. 3 새 토큰 이름(예: myinitials_mytoken)을 입력합니다. 4 토큰 TTL을 만료되지 않음으로 설정합니다. 만료되도록 설정된 토큰을 생성하는 경우, 토큰이 만료되면 vRealize Automation에서 VMware Cloud on AWS 작업이 중지되며, 새 토큰으로 클라우드 계정을 업데이트할 때까지 계속 작동하지 않습니다. 5 범위 정의 섹션에서 모든 역할을 선택합니다. <div data-bbox="499 863 884 1052" data-label="Image"> </div> 6 생성을 클릭합니다. 7 [생성된 토큰] 페이지에서 복사를 클릭하고 계속을 클릭합니다. 8 새 클라우드 계정 페이지로 돌아가서 복사한 토큰을 VMC API 토큰 행에 붙여넣고 API 토큰 적용을 클릭합니다. <div data-bbox="499 1297 842 1507" data-label="Image"> </div> 	<p>새 토큰을 생성하거나 연결된 API 토큰 페이지에서 조직에 대한 기존 토큰을 사용할 수 있습니다.</p> <p>범위 정의 섹션에서 API 토큰에 필요한 최소 역할은 다음과 같습니다.</p> <ul style="list-style-type: none"> ■ 조직 역할 <ul style="list-style-type: none"> ■ 조직 멤버 ■ 조직 소유자 ■ 서비스 역할 - VMware Cloud on AWS <ul style="list-style-type: none"> ■ 관리자 ■ NSX Cloud 관리자 ■ NSX Cloud 감사자 <p>참고 생성된 토큰을 복사, 다운로드 또는 인쇄합니다. 이 페이지에서 나가면 생성된 토큰을 검색할 수 없습니다.</p> <p>생성 또는 제공된 토큰을 적용하여 조직의 VMware Cloud on AWS 구독에서 사용할 수 있는 SDDC 환경에 연결하고 SDDC 이름 목록을 채웁니다.</p> <p>vRealize Automation 및 VMware Cloud on AWS 서비스가 서로 다른 조직에 있으면 VMware Cloud on AWS 조직으로 전환한 다음 토큰을 생성해야 합니다.</p> <p>API 토큰에 대한 자세한 내용은 API 토큰 생성을 참조하십시오.</p>
SDDC 이름	<p>이 예에서는 Datacenter:Datacenter-abz를 선택합니다.</p> <p>유효한 SDDC 이름이 vCenter 및 NSX-T FQDN 항목에 자동으로 채워집니다. 클라우드 프록시가 SDDC에 이미 배포되어 있다면 클라우드 프록시 값도 자동으로 채워집니다.</p>	<p>VMware Cloud on AWS 구독의 사용 가능한 SDDC 목록에서 선택합니다. SDDC 목록은 VMware Cloud on AWS API 토큰을 기반으로 합니다.</p> <p>NSX-V SDDC는 vRealize Automation에서 지원되지 않기 때문에 사용 가능한 SDDC 목록에 표시되지 않습니다.</p>

설정	샘플 값 및 지침	설명
vCenter IP 주소/ FQDN	주소는 선택한 SDDC에 기반하여 자동으로 채워집니다.	지정된 SDDC에 있는 vCenter Server의 IP 주소 또는 FQDN을 입력합니다. IP 주소는 기본적으로 개인 IP 주소로 설정됩니다. SDDC에 액세스하는 데 사용되는 네트워크 연결 유형에 따라 기본 주소는 지정된 SDDC에 있는 NSX Manager 서버의 IP 주소와 다를 수 있습니다.
NSX Manager IP 주소/FQDN	주소는 선택한 SDDC에 기반하여 자동으로 채워집니다.	지정된 SDDC에 있는 NSX Manager의 IP 주소 또는 FQDN을 지정합니다. IP 주소는 기본적으로 개인 IP 주소로 설정됩니다. SDDC에 액세스하는 데 사용되는 네트워크 연결 유형에 따라 기본 주소는 지정된 SDDC에 있는 NSX Manager 서버의 IP 주소와 다를 수 있습니다. VMware Cloud on AWS 클라우드는 NSX-T를 지원합니다.
vCenter 사용자 이름 및 암호	cloudadmin@vmc.local이라는 사용자 이름이 자동으로 채워집니다.	지정된 SDDC의 vCenter 사용자 이름을 입력합니다(기본값과 다른 경우). 지정된 사용자에게 CloudAdmin 자격 증명이 필요합니다. 사용자에게 CloudGlobalAdmin 자격 증명이 필요하지 않습니다. 사용자 암호를 입력합니다.
검증	검증을 클릭합니다.	검증 기능은 지정된 vCenter에 대한 사용자의 액세스 권한을 확인하고 vCenter가 실행 중인지 확인합니다.
이름 및 설명	클라우드 계정 이름으로 OurCo-VMC 를 입력합니다. 클라우드 계정 설명으로 Sample deployment for VMC 를 입력합니다.	
이러한 데이터 센터로의 프로비저닝 허용	이 정보는 읽기 전용입니다.	지정된 VMware Cloud on AWS SDDC 환경에 있는 사용 가능한 데이터 센터가 나열됩니다.
클라우드 영역 생성	확인란을 선택 취소합니다. 이 예의 경우 나중에 워크플로에서 클라우드 영역을 생성합니다.	vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기 항목을 참조하십시오.
기능 태그	비워 둡니다. 이 워크플로는 기능 태그를 사용하지 않습니다.	조직의 태그 전략에 따라 태그를 사용하십시오. 태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법 및 태그 지정 전략 생성 항목을 참조하십시오.

3 추가를 클릭합니다.

결과

시스템 및 볼륨 같은 리소스 데이터가 VMware Cloud on AWS SDDC 데이터 센터에서 수집되어 vRealize Automation **인프라** 탭의 **리소스** 섹션에 나열됩니다.

다음에 수행할 작업

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대한 클라우드 영역 생성.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대한 클라우드 영역 생성

이 단계에서는 클라우드 영역을 생성하고 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS로 작업할 때 CloudAdmin 사용자가 액세스할 수 있는 계산 리소스를 지정합니다.

VMware Cloud on AWS의 두 가지 기본 관리자 자격 증명은 CloudGlobalAdmin과 CloudAdmin입니다. vRealize Automation Cloud Assembly는 CloudAdmin 사용자를 지원하도록 설계되었습니다. VMware Cloud on AWS CloudAdmin 사용자가 사용할 수 있는 리소스에 배포합니다. VMware Cloud on AWS CloudGlobalAdmin 자격 증명이 필요한 리소스에는 배포하지 마십시오.

클라우드 영역은 프로젝트 Blueprint가 시스템, 네트워크 및 스토리지를 배포하는 계산 리소스를 식별합니다. vRealize Automation Cloud Assembly 클라우드 영역에 대해 [알아보기](#) 항목을 참조하십시오.

달리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- 샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성 절차를 완료합니다.
- 이 절차에서는 사용자에게 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 포함하여 필요한 관리자 자격 증명이 있다고 가정합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 [자격 증명](#) 항목을 참조하십시오.
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. vRealize Automation Cloud Assembly [사용자 역할이란?](#) 항목을 참조하십시오.

절차

1 **인프라 > 구성 > 클라우드 영역**을 선택합니다.

2 **새 클라우드 영역**을 클릭하고 VMware Cloud on AWS 환경에 대한 값을 입력합니다.

설정	샘플 값
계정/지역	OurCo-VMC/Datacenter:Datacenter-abz 이전 단계인 샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성에서 정의한 클라우드 계정 및 연결된 지역입니다.
이름	VMC_cloud_zone-1
설명	VMware Cloud on AWS 리소스만

설정	샘플 값
배치 정책	기본값
기능 태그	비워 둡니다. 이 워크플로는 기능 태그를 사용하지 않습니다.

3 계산 탭을 클릭합니다.

- 4 아래 영역 1에 표시된 대로, CloudAdmin 사용자가 사용할 수 있는 계산 리소스를 찾아서 선택합니다. 이 예에서는 Cluster 1/ Compute-ResourcePool이라는 리소스를 사용합니다.

Cluster 1/ Compute-ResourcePool은 VMware Cloud on AWS에 대한 기본 계산 리소스입니다.

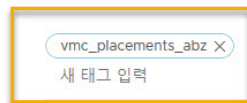


- 5 위의 영역 2에 표시된 대로 태그 이름 vmc_placements_abz를 추가합니다.

태그

개체 1개 선택됨

태그 추가

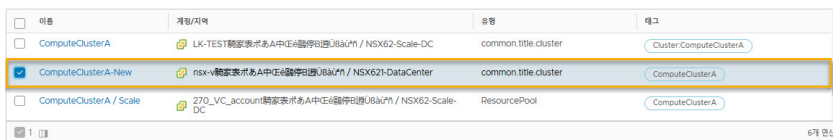


태그 제거

태그 없음 ①

- 6 필터 태그 섹션에 vmc_placements_abz를 입력하여 이 클라우드 영역에서 사용되는 계산 리소스를 필터링합니다.

7 저장을 클릭합니다.



이 예에서는 이름이 Cluster 1/ Compute-ResourcePool인 계산 리소스만 CloudAdmin 사용자에게 제공됩니다.

다음에 수행할 작업

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성

이 단계에서는 네트워크 프로파일 및 스토리지 프로파일을 구성하여 vRealize Automation Cloud Assembly의 VMware Cloud on AWS CloudAdmin 사용자가 사용할 수 있는 리소스를 지정합니다.

이미지 및 플레이버 값도 필요하지만 VMware Cloud on AWS 사용자 자격 증명과 관련된 고유 항목은 없습니다. 이 예에서는 Blueprint를 정의할 때 플레이버 값은 small을 사용하고 이미지 값은 ubuntu-16을 사용합니다.

매핑 및 프로파일에 대한 일반적인 내용은 [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#) 항목을 참조하십시오.

다들 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- 클라우드 영역을 생성합니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대한 클라우드 영역 생성 항목을 참조하십시오.
- 이 절차에서는 사용자에게 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 포함하여 필요한 관리자 자격 증명이 있다고 가정합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.

절차

1 VMware Cloud on AWS 배포에 대한 네트워크 프로파일을 정의합니다.

a 인프라 > 구성 > 네트워크 프로파일을 선택하고 새 네트워크 프로파일을 클릭합니다.

설정	샘플 값
계정/지역	OurCo-VMC/Datacenter:Datacenter-abz
이름	vmc-network1
설명	VMware Cloud on AWS CloudAdmin 자격 증명이 있는 Blueprint 관리자가 액세스할 수 있는 네트워크가 포함되어 있습니다.

참고 샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성에서 생성한 VMware Cloud on AWS 클라우드 계정 및 이와 일치하는 SDDC 데이터 센터를 선택합니다.

b 네트워크 탭을 클릭하고 네트워크 추가를 클릭합니다.

c CloudAdmin 자격 증명이 있는 VMware Cloud on AWS 사용자가 배포할 수 있는 네트워크(예: sddc-cgw-network-1)를 선택합니다.

네트워크 추가



<input type="checkbox"/>	이름	계정/지역	영역	네트워크 도메인
<input checked="" type="checkbox"/>	ESO_PKS_VC01_VM_PKS	1114VC아7중泰國@臺術		ESO_PKS_VC01_DVS01
<input type="checkbox"/>	ESO_PKS_VC01_Mgmt	1114VC아7중泰國@臺術		ESO_PKS_VC01_DVS01

2 네트워크 프로파일을 저장합니다.

3 VMware Cloud on AWS 배포에 대한 스토리지 프로파일을 정의합니다.

CloudAdmin 사용자가 액세스할 수 있는 데이터스토어/클러스터를 대상으로 하는 스토리지 프로파일을 구성합니다.

a 인프라 > 구성 > 스토리지 프로파일을 선택하고 새 스토리지 프로파일을 클릭합니다.

설정	샘플 값
계정/지역	OurCo-VMC/Datacenter:Datacenter-abz 샘플 워크플로 내의 vRealize Automation에서 VMware Cloud on AWS 클라우드 계정 생성에서 생성한 VMware Cloud on AWS 클라우드 계정 및 이와 일치하는 SDDC 데이터 센터를 선택합니다.
이름	vmc-storage1
설명	VMware Cloud on AWS CloudAdmin 자격 증명이 있는 Blueprint 관리자가 배포할 수 있는 데이터스토어 클러스터가 포함되어 있습니다.

b 데이터스토어/클러스터 드롭다운 메뉴에서 **WorkloadDatastore** 데이터스토어를 선택합니다.



vRealize Automation Cloud Assembly의 VMware Cloud on AWS인 경우 스토리지 정책에 **WorkloadDatastore** 데이터스토어를 사용하여 VMware Cloud on AWS 배포를 지원해야 합니다.

4 스토리지 프로파일을 저장합니다.

다음에 수행할 작업

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 지원하는 프로젝트 생성.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 지원하는 프로젝트 생성

이 단계에서는 VMware Cloud on AWS 배포에 사용할 수 있는 리소스를 제어하는 데 사용하는 vRealize Automation Cloud Assembly 프로젝트를 정의합니다.

프로젝트에 대한 자세한 내용은 배포 시간에 vRealize Automation Cloud Assembly 프로젝트가 작동하는 방식 항목을 참조하십시오.

달리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성 절차를 완료합니다.

- 이 절차에서는 사용자에게 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 포함하여 필요한 관리자 자격 증명이라고 가정합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.](#)
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.

절차

1 **인프라 > 구성 > 프로젝트**를 선택합니다.

2 **새 프로젝트**를 클릭하고 VMC_proj-1_abz를 프로젝트 이름으로 입력합니다.

3 **사용자**를 클릭하고 **사용자 추가**를 클릭합니다.

사용자에게 조직의 VMware Cloud on AWS 구독에 대한 CloudAdmin 자격 증명が必要です.

- chris.gray@ourco.com, 관리자
- kerry.white@ourco.com, Member

4 **프로비저닝**을 클릭한 다음 **클라우드 영역 추가**를 클릭합니다.

5 이전 단계에서 구성한 클라우드 영역을 추가합니다.

설정	샘플 값
클라우드 영역	VMC_cloud_zone-1 이 클라우드 영역은 이전 단계인 vRealize Automation Cloud Assembly 에서 VMware Cloud on AWS 배포에 대한 클라우드 영역 생성에서 생성했습니다.
프로비저닝 우선 순위	1
인스턴스 제한	3

6 이 예에서는 다른 옵션을 무시합니다.

다음에 수행할 작업

VMware Cloud on AWS 환경에 배포할 Blueprint를 생성합니다. [vRealize Automation Cloud Assembly](#)에서 [VMware Cloud on AWS](#) 배포를 지원하도록 [Blueprint 설계에 vCenter 시스템 리소스 정의](#) 항목을 참조하십시오.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 지원하도록 Blueprint 설계에 vCenter 시스템 리소스 정의

이 단계에서는 vCenter 시스템 리소스를 설계 캔버스에 끌어와서 VMware Cloud on AWS 배포에 대한 설정을 추가합니다.

사용 가능한 VMware Cloud on AWS 리소스에 배포할 수 있는 Blueprint 설계를 생성합니다.

달리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- 이 절차에서는 사용자에게 **Blueprint** 설계자 자격 증명이 있다고 가정합니다. **vRealize Automation Cloud Assembly 사용자 역할이란?** 항목을 참조하십시오.
- 이 절차에서는 vCenter(Datacenter:Datacenter-abz)의 대상 SDDC에 대해 **VMware Cloud on AWS CloudAdmin** 자격 증명이 있다고 가정합니다. **vRealize Automation**에서 클라우드 계정 작업에 **필요한 자격 증명** 항목을 참조하십시오.
- 앞의 섹션에 설명된 대로 리소스 인프라 및 프로젝트를 구성합니다.

절차

- 1 **설계** 탭을 클릭한 다음, **새로 만들기**를 클릭합니다.

설정	샘플 값
이름	vmc-bp_abz
설명	1
프로젝트	VMC_proj-1_abz 이전에 생성한 프로젝트이며 이전에 생성한 클라우드 영역을 지원합니다. 이제 프로젝트가 클라우드 영역과 연결되며, 이 영역은 차례로 이전에 생성한 VMware Cloud on AWS 클라우드 계정/지역과 연결됩니다.

- 2 **vSphere** 시스템 리소스를 캔버스로 씁니다.
- 3 시스템 리소스에서 다음(굵게 표시됨) **Blueprint** 리소스 코드를 편집합니다.

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: ubuntu-1604
      cpuCount: 1
      totalMemoryMB: 1024
      folderName: Workloads
```

image에는 배포 요구 사항에 적합한 값을 지정할 수 있습니다.

VMware Cloud on AWS 배포를 지원하려면 **Blueprint** 설계 코드에 `folderName: Workloads` 문을 추가해야 합니다. `folderName: Workloads` 설정은 VMware Cloud on AWS SDDC 환경에서 CloudAdmin 자격 증명을 지원하며 필수입니다.

참고: 위의 코드 샘플에 나와 있는 `folderName: Workloads` 설정이 필요하지만 위에 나와 있는 대로 **Blueprint** 설계 코드에 직접 추가하거나 연결된 클라우드 영역 또는 프로젝트에 추가할 수 있습니다. 설정이 이러한 세 위치 중 하나 이상에 지정된 경우 우선 순위는 다음과 같습니다.

- 프로젝트 설정은 **Blueprint** 설계 설정과 클라우드 영역 설정을 재정의합니다.

- Blueprint 설계 설정은 클라우드 영역 설정을 재정의합니다.

참고: 필요에 따라 `cpuCount` 및 `totalMemoryMB` 설정을 `flavor`(크기 조정) 항목으로 교체할 수 있습니다. 아래를 참조하세요.

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: ubuntu-1604
      flavor: small
      folderName: Workloads
```

클라우드 영역의 폴더 값이 **Workloads**로 설정된 경우 클라우드 영역 폴더 값을 재정의하려는 경우가 아니면 Blueprint 설계에 `folderName` 속성을 설정할 필요가 없습니다.

다음에 수행할 작업

네트워크 격리를 추가하여 기본 VMware Cloud on AWS 워크플로를 확장합니다. [vRealize Automation Cloud Assembly의 VMware Cloud on AWS 워크플로](#)에서 격리된 네트워크 구성 항목을 참조하십시오.

vRealize Automation Cloud Assembly의 VMware Cloud on AWS 워크플로에서 격리된 네트워크 구성

이 절차에서는 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대해 격리된 네트워크를 추가합니다.

VMware Cloud on AWS 클라우드 계정을 정의할 때 VMware Cloud on AWS 서비스에 구성된 NSX-T 설정을 사용할 수 있습니다. VMware Cloud on AWS 서비스에서 NSX-T 설정을 구성하는 방법에 대한 자세한 내용은 VMware Cloud on AWS [제품 설명서](#)를 참조하십시오.

vRealize Automation Cloud Assembly는 VMware Cloud on AWS에서 NSX-T를 지원합니다. VMware Cloud on AWS에는 NSX-V가 지원되지 않습니다.

vRealize Automation Cloud Assembly는 VMware Cloud on AWS 배포에 대한 네트워크 분리를 지원합니다. VMware Cloud on AWS에는 다른 네트워크 방법이 지원되지 않습니다.

기본 VMware Cloud on AWS 워크플로에 대한 확장은 vRealize Automation Cloud Assembly Blueprint에 사용할 격리된 네트워크를 생성하는 다음과 같은 방법을 설명합니다.

- 주문형 네트워크 기반 격리 구성.
- 주문형 보안 그룹 기반 격리 구성.

사전 요구 사항

이 절차는 기본 VMware Cloud on AWS 워크플로를 확장합니다. 여기에는 **VMware Cloud on AWS 사용 사례** 워크플로에 구성한 것과 동일한 클라우드 계정 및 지역, 클라우드 영역, 프로젝트 및 네트워크 프로파일이 사용됩니다.

절차

1 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대해 격리된 네트워크 정의

다음 절차 중 하나를 사용하여 VMware Cloud on AWS 배포에 대한 네트워크 격리를 구성할 수 있습니다.

2 vRealize Automation Cloud Assembly에서 VMware Cloud on AWS의 네트워크 격리를 지원하도록 Blueprint의 네트워크 구성 요소 정의

이 단계에서는 네트워크 시스템 구성 요소를 vRealize Automation Cloud Assembly Blueprint 캔버스에 끌어와서, 격리된 네트워크 배포에 대한 설정을 대상 VMware Cloud on AWS 환경에 추가합니다.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포에 대해 격리된 네트워크 정의

다음 절차 중 하나를 사용하여 VMware Cloud on AWS 배포에 대한 네트워크 격리를 구성할 수 있습니다.

- vRealize Automation Cloud Assembly에서 주문형 네트워크 기반 격리 구성
- vRealize Automation Cloud Assembly에서 주문형 보안 그룹 기반 격리 구성

vRealize Automation Cloud Assembly에서 주문형 네트워크 기반 격리 구성

vRealize Automation Cloud Assembly 네트워크 프로파일에 주문형 네트워크 설정을 지정하고 사용하여 VMware Cloud on AWS 배포 요구 사항에 맞게 네트워크 격리를 구성할 수 있습니다.

보안 그룹을 사용하거나 주문형 네트워크 설정을 사용하여 격리된 네트워크를 지정할 수 있습니다. 이 예에서는 네트워크 프로파일에 주문형 네트워크 설정을 지정하여 네트워크 격리를 구성합니다. 나중에, Blueprint에서 네트워크에 액세스하고 VMware Cloud on AWS 배포에 Blueprint를 사용합니다.

다리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- vRealize Automation Cloud Assembly에서 기본 VMware Cloud on AWS 워크플로 구성 워크플로를 완료합니다.
- vRealize Automation Cloud Assembly의 VMware Cloud on AWS 워크플로에서 격리된 네트워크 구성 항목을 검토합니다.

- 이 절차에서는 사용자에게 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 포함하여 필요한 관리자 자격 증명이라고 가정합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.

절차

- 1 기본 VMware Cloud on AWS 워크플로에 사용한 네트워크 프로파일(예: vmc-network1)을 엽니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성의 내용을 참조하십시오.
- 2 **네트워크** 탭에서 아무것도 선택할 필요가 없습니다.
- 3 **네트워크 정책** 탭을 클릭합니다.
- 4 **주문형 네트워크 생성** 옵션 선택하고 기본 cgw 네트워크 도메인을 선택합니다. 적절한 CIDR 및 서브넷 크기를 지정합니다.

요약 네트워크 **네트워크 정책** 보안 그룹

다음 설정은 아웃바운드 및 전용 네트워크를 생성할 때 사용됩니다. ⓘ

☐ 주문형 네트워크 또는 주문형 **주문형 네트워크**

☒ 주문형 네트워크 생성 ⓘ 이러한 설정을 사용하여 각 배포에 대한 서브넷이 생성됩니다. 이 방법은 IP 주소가 적은 소규모 네트워크에 더 적합할 수 있습니다.

네트워크 도메인 * 10.10.6 0/24 ⓘ

CIDR * /29(~6 IP 주소) ⓘ

서브넷 크기 * /29(~6 IP 주소) ⓘ

- 5 **저장**을 클릭합니다.

이 네트워크 프로파일을 사용하면 시스템이 기본 네트워크 도메인의 네트워크에 배포됩니다. 네트워크는 전용 또는 아웃바운드 네트워크 액세스를 사용하여 다른 네트워크와 격리됩니다.

다음에 수행할 작업

Blueprint에 네트워크 구성 요소를 구성합니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS의 네트워크 격리를 지원하도록 Blueprint의 네트워크 구성 요소 정의 항목을 참조하십시오.

vRealize Automation Cloud Assembly에서 주문형 보안 그룹 기반 격리 구성

vRealize Automation Cloud Assembly 네트워크 프로파일에 주문형 보안 그룹을 지정하고 사용하여 VMware Cloud on AWS 배포 요구 사항에 맞게 네트워크 격리를 구성할 수 있습니다.

보안 그룹을 사용하거나 주문형 네트워크 설정을 사용하여 격리된 네트워크를 지정할 수 있습니다. 이 예에서는 네트워크 프로파일에 주문형 보안 그룹을 지정하여 네트워크 격리를 구성합니다. 나중에, Blueprint에 네트워크를 지정하고 VMware Cloud on AWS 배포에 Blueprint를 사용합니다.

다리 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

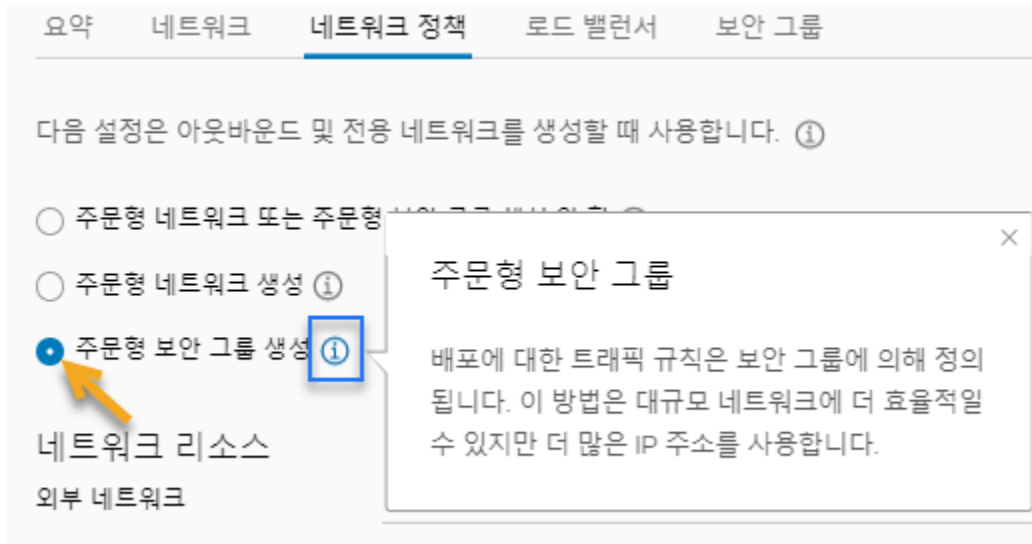
사전 요구 사항

- vRealize Automation Cloud Assembly에서 기본 VMware Cloud on AWS 워크플로 구성 워크플로를 완료합니다.
- vRealize Automation Cloud Assembly의 VMware Cloud on AWS 워크플로에서 격리된 네트워크 구성 항목을 검토합니다.
- 이 절차에서는 사용자에게 vCenter의 대상 SDDC에 대한 VMware Cloud on AWS CloudAdmin 자격 증명을 포함하여 필요한 관리자 자격 증명이 있다고 가정합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 이 절차에서는 사용자에게 클라우드 관리자 사용자 역할이 있다고 가정합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.

절차

- 1 기본 VMware Cloud on AWS 워크플로에 사용한 네트워크 프로파일(예: vmc-network1)을 엽니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성의 내용을 참조하십시오.
- 2 기본 VMware Cloud on AWS 워크플로에 사용한 기존 네트워크를 선택합니다(예: sddc-cgw-network-1). vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 위한 네트워크 및 스토리지 프로파일 구성의 내용을 참조하십시오.
- 3 **네트워크 정책** 탭을 클릭합니다.

4 주문형 보안 그룹 생성 옵션을 선택합니다.



5 저장을 클릭합니다.

이 네트워크 프로파일을 사용하면 시스템이 선택한 네트워크에 배포되고 새 보안 그룹 정책에 의해 격리됩니다. 새 보안 정책에는 전용 또는 아웃바운드 네트워크 액세스가 허용됩니다.

다음에 수행할 작업

Blueprint에 네트워크 구성 요소를 구성합니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS의 네트워크 격리를 지원하도록 Blueprint의 네트워크 구성 요소 정의 항목을 참조하십시오.

vRealize Automation Cloud Assembly에서 VMware Cloud on AWS의 네트워크 격리를 지원하도록 Blueprint의 네트워크 구성 요소 정의

이 단계에서는 네트워크 시스템 구성 요소를 vRealize Automation Cloud Assembly Blueprint 캔버스에 끌어와서, 격리된 네트워크 배포에 대한 설정을 대상 VMware Cloud on AWS 환경에 추가합니다.

이전에 생성한 Blueprint에 네트워크 격리를 추가합니다. Blueprint는 VMware Cloud on AWS 환경에 대한 배포를 지원하는 프로젝트 및 클라우드 영역과 이미 연결되어 있으며 격리를 위해 구성된 네트워크 및 네트워크 프로파일과도 연결되어 있습니다.

다들 명시되지 않는 한, 이 절차에서 입력하는 단계 값은 예시 워크플로에만 해당됩니다.

사전 요구 사항

- vRealize Automation Cloud Assembly에서 주문형 보안 그룹 기반 격리 구성 또는 vRealize Automation Cloud Assembly에서 주문형 네트워크 기반 격리 구성 절차를 완료합니다.
- 이 절차에서는 사용자에게 Blueprint 설계자 자격 증명이 있다고 가정합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.

- 이 절차에서는 vCenter의 대상 SDDC에 대해 VMware Cloud on AWS CloudAdmin 자격 증명이 있다고 가정합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.

절차

- 1 이전 워크플로에서 생성한 Blueprint를 엽니다. vRealize Automation Cloud Assembly에서 VMware Cloud on AWS 배포를 지원하도록 Blueprint 설계에 vCenter 시스템 리소스 정의의 내용을 참조하십시오.
- 2 Blueprint 설계 페이지의 왼쪽에 있는 구성 요소에서 네트워크 구성 요소를 캔버스 위로 끌어 놓습니다.
- 3 네트워크 구성 요소 YAML 코드를 편집하여 네트워크 유형을 private 또는 outbound(굵은 글꼴로 표시)로 지정합니다.

```
resources: Cloud_Network_1:
  type: Cloud.Network
  properties:
    name: "vmc_isolated"
    networkType: private
```

또는

```
resources: Cloud_Network_1:
  type: Cloud.Network
  properties:
    name: "vmc_isolated"
    networkType: outbound
```

다음에 수행할 작업

Blueprint를 배포하거나 단을 준비가 되었습니다.

제공자별 외부 IPAM 통합 사용 사례

외부 IPAM 제공자를 사용하여 Blueprint 배포에 대한 IP 주소 할당을 관리할 수 있습니다. 이 예제 절차에서는 vRealize Automation Cloud Assembly에서 외부 IPAM 통합을 구성하고 외부 IPAM 제공자로부터 IP 주소 할당을 받는 Blueprint를 배포하는 방법을 설명합니다. 이 예에서는 Infoblox를 사용하여 외부 IPAM 제공자로 IPAM 통합을 구성하는 방법을 설명합니다.

이 절차에서는 기존 IPAM 제공자 패키지(이 경우 Infoblox 패키지)와 기존 실행 환경을 사용하여 제공자별 IPAM 통합 지점을 만듭니다. 외부 IPAM 제공자의 IP 주소 할당을 지원하도록 기존 네트워크를 구성하고 네트워크 프로파일을 생성합니다. 마지막으로, 네트워크 및 네트워크 프로파일과 일치하는 Blueprint를 생성하고 외부 IPAM 제공자로부터 가져온 IP 값을 사용하여 네트워크 시스템을 배포합니다.

IPAM 제공자 패키지를 확보하고 구성하는 방법과 IPAM 제공자 통합을 지원하기 위해 클라우드 확장성 기록시에 액세스할 수 있는 실행 환경을 구성하는 방법에 대한 정보는 참조로 포함됩니다.

보이는 값은 예시 값입니다. 사용자 환경에서 이러한 값을 그대로 사용해서는 안 됩니다.

조직의 요구 사항에 맞게 고유한 값으로 대체하거나 예시의 값에서 추론할 수 있는 부분에 대해 생각해 보십시오.

절차

- 1 **vRealize Automation와 통합하기 위해 Infoblox 애플리케이션에서 필요한 확장 가능 특성 추가**
Infoblox 웹 사이트 또는 VMware Marketplace에서 vRealize Automation와 통합하기 위해 Infoblox 제공자 패키지(`infoblox.zip`)를 다운로드하고 배포하려면, 그 전에 Infoblox에서 필요한 확장성 특성을 추가해야 합니다.
- 2 **vRealize Automation Cloud Assembly에서 사용할 외부 IPAM 제공자 패키지 다운로드 및 배포**
vRealize Automation Cloud Assembly에서 외부 IPAM 통합 지점을 정의하려면, 먼저 구성된 IPAM 제공자 패키지가 필요합니다. 제공자 패키지는 IPAM 제공자의 웹 사이트나 vRealize Automation Cloud Assembly 마켓플레이스에서 다운로드할 수 있습니다.
- 3 **vRealize Automation에서 IPAM 통합 지점에 대한 실행 환경 생성**
vRealize Automation에서 외부 IPAM 통합 지점을 정의하려면 먼저 IPAM 제공자와 vRealize Automation 간에 중개자 역할을 하는 기존 실행 환경에 액세스하거나 새로 생성해야 합니다. 실행 환경은 일반적으로 Amazon Web Services 또는 Microsoft Azure 클라우드 계정이거나 클라우드 확장성 프록시에 연결된 온-프레미스 작업 기반 확장성 통합 지점입니다.
- 4 **vRealize Automation에서 외부 IPAM 통합 지점 추가**
vRealize Automation는 외부 IPAM 제공자와의 통합을 지원합니다. 제공자별 IPAM 통합 지점을 사용하여 Blueprint 배포를 위한 IP 주소 및 관련 네트워크 특성을 가져오고 관리할 수 있습니다.
- 5 **vRealize Automation Cloud Assembly에서 IPAM 제공자 값을 사용하도록 네트워크 및 네트워크 프로파일 구성**
내부 vRealize Automation Cloud Assembly가 아닌, 외부 IPAM 제공자가 관리하고 외부 IPAM 제공자로부터 가져오는 IP 주소 값을 사용하도록 네트워크를 정의할 수 있습니다.
- 6 **IPAM 제공자 범위 할당을 사용하는 vRealize Automation Cloud Assembly Blueprint 정의 및 배포**
외부 IPAM 제공자로부터 IP 주소 할당을 가져오고 관리하는 Blueprint를 정의할 수 있습니다.
- 7 **vRealize Automation에서 IPAM 통합에 대해 Infoblox 특정 속성 및 확장 가능 특성 사용**
Infoblox용 외부 IPAM 통합을 포함하는 vRealize Automation 프로젝트에 대해 Infoblox 특정 속성을 사용할 수 있습니다.

vRealize Automation와 통합하기 위해 Infoblox 애플리케이션에서 필요한 확장 가능 특성 추가

Infoblox 웹 사이트 또는 VMware Marketplace에서 vRealize Automation와 통합하기 위해 Infoblox 제공자 패키지(`infoblox.zip`)를 다운로드하고 배포하려면, 그 전에 Infoblox에서 필요한 확장성 특성을 추가해야 합니다.

이 절차는 vRealize Automation Cloud Assembly와 Infoblox 통합을 위해 외부 IPAM 통합 지점을 생성하는 경우에 적용됩니다.

infoblox.zip 다운로드를 사용하려면, 먼저 조직 계정 관리자 자격 증명을 사용하여 Infoblox 계정에 로그인하고 다음과 같은 Infoblox 확장 가능 특성을 미리 생성해야 합니다.

- VMware NIC index
- VMware resource ID
- Tenant ID
- CMP Type
- VM ID
- VM Name

사전 요구 사항

- Infoblox 계정이 있고 조직의 Infoblox 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- Infoblox WAPI 버전이 지원되는지 확인합니다. Infoblox와 IPAM 통합은 Infoblox WAPI 버전 v2.7에 따라 다릅니다. WAPI v2.7을 지원하는 모든 Infoblox 장치가 지원됩니다.
- vRealize Automation에서 IPAM 통합에 대해 Infoblox 특정 속성 및 확장 가능 특성 사용 항목을 검토합니다.

절차

- 1 관리자 자격 증명을 사용하여 Infoblox 계정에 로그인합니다.

이것은 **인프라 > 연결 > 통합 >** 메뉴 시퀀스를 사용하여 vRealize Automation Cloud Assembly에서 외부 IPAM 통합 지점을 생성할 때 지정하는 것과 동일한 관리자 이름과 암호 자격 증명입니다.

- 2 Infoblox 설명서에 설명된 절차를 사용하여 Infoblox 애플리케이션에서 다음 필수 확장 가능 특성을 생성합니다.

- VMware NIC index - 유형 정수
- VMware resource ID - 유형 문자열
- Tenant ID - 유형 문자열
- CMP Type - 유형 문자열
- VM ID - 유형 문자열
- VM Name - 유형 문자열

이 절차는 Infoblox 설명서 항목 [About Extensible Attributes](#)의 "Adding Extensible Attributes" 섹션에 설명되어 있습니다. [Managing Extensible Attributes](#) 항목도 참조하십시오.

다음에 수행할 작업

필요한 특성을 추가하면 vRealize Automation Cloud Assembly에서 사용할 외부 IPAM 제공자 패키지 다운로드 및 배포에 설명된 대로 Infoblox 패키지를 다운로드하고 배포하는 프로세스를 재개할 수 있습니다.

vRealize Automation Cloud Assembly에서 사용할 외부 IPAM 제공자 패키지 다운로드 및 배포

vRealize Automation Cloud Assembly에서 외부 IPAM 통합 지점을 정의하려면, 먼저 구성된 IPAM 제공자 패키지가 필요합니다. 제공자 패키지는 IPAM 제공자의 웹 사이트나 vRealize Automation Cloud Assembly 마켓플레이스에서 다운로드할 수 있습니다.

제공자별 통합 패키지는 IPAM 제공자의 웹 사이트, [VMware Solution Exchange](#) 마켓플레이스 또는 vRealize Automation Cloud Assembly [마켓플레이스](#) 탭에서 구할 수 있습니다.

참고 이 예에서는 VMware에서 제공하는 Infoblox 패키지인 Infoblox.zip을 사용합니다. 이 패키지는 VMware Solution Exchange 마켓플레이스에서 다음과 같은 버전으로 다운로드할 수 있습니다.

- [vRA Cloud Infoblox 플러그인 버전 1.1](#) - vRealize Automation 8.1 지원
- [vRA Cloud Infoblox 플러그인 버전 1.0](#) - vRealize Automation 8.0.1 지원
- [vRA Cloud Infoblox 플러그인 버전 0.1](#) - vRealize Automation 8.0 지원

Infoblox와 IPAM 통합은 Infoblox WAPI 버전 v2.7에 따라 다릅니다. WAPI v2.7을 지원하는 모든 Infoblox 장치가 지원됩니다.

다른 IPAM 제공자를 위한 IPAM 통합 패키지를(마켓플레이스에 아직 없는 경우) 만드는 방법에 대한 자세한 내용은 [IPAM SDK를 사용하여 vRealize Automation에 대한 제공자별 외부 IPAM 통합 패키지를 생성하는 방법](#) 항목을 참조하십시오.

IPAM 제공자 패키지에는 메타데이터 및 기타 구성과 함께 패키지로 구성된 스크립트가 포함되어 있습니다. 스크립트에는 vRealize Automation Cloud Assembly가 외부 IPAM 제공자와 함께 수행하는 작업에 사용되는 소스 코드가 포함되어 있습니다. 예제 작업에는 Allocate an IP address for a virtual machine, Fetch a list of IP ranges from the provider 및 Update the MAC address of a host record in the provider가 포함됩니다.

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. [vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명](#) 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 IPAM 제공자(예: [Infoblox](#) 또는 [Bluecat](#))의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- Infoblox를 외부 IPAM 제공자로 사용하는 경우, 계속하기 전에 Infoblox 계정에 필요한 확장 가능 특성을 추가했는지 확인합니다. [vRealize Automation과 통합하기 위해 Infoblox 애플리케이션에서 필요한 확장 가능 특성 추가](#) 항목을 참조하십시오.

절차

- 1 VMware Solution Exchange 마켓플레이스에서 **vRA Cloud Infoblox 플러그인 버전 0.1**(vRealize Automation 8.0) 또는 **vRA Cloud Infoblox 플러그인 버전 1.0**(vRealize Automation 8.0.1) 패키지 페이지로 이동합니다.
- 2 로그인하고 플러그인 패키지를 다운로드합니다.
- 3 Infoblox에 필요한 확장 가능 특성을 추가하지 않았다면 지금 추가합니다. **vRealize Automation**와 통합하기 위해 **Infoblox 애플리케이션**에서 필요한 확장 가능 특성 추가의 내용을 참조하십시오.

결과

이제 **vRealize Automation**에서 외부 IPAM 통합 지점 추가에 설명된 대로 **통합 > 통합 추가 > IPAM > 제공자 관리 > 패키지 가져오기** 메뉴 시퀀스를 사용하여 배포를 위한 패키지를 사용할 수 있습니다.

vRealize Automation에서 IPAM 통합 지점에 대한 실행 환경 생성

vRealize Automation에서 외부 IPAM 통합 지점을 정의하려면 먼저 IPAM 제공자와 vRealize Automation 간에 중개자 역할을 하는 기존 실행 환경에 액세스하거나 새로 생성해야 합니다. 실행 환경은 일반적으로 Amazon Web Services 또는 Microsoft Azure 클라우드 계정이거나 클라우드 확장성 프록시에 연결된 온-프레미스 작업 기반 확장성 통합 지점입니다.

외부 IPAM 통합에는 실행 환경이 필요합니다. IPAM 통합 지점을 정의할 때 사용 가능한 실행 환경을 지정하여 vRealize Automation Cloud Assembly와 IPAM 제공자 간에 연결을 생성합니다.

IPAM 통합은 Amazon Web Services Lambda, Microsoft Azure Functions와 같은 FaaS(Function-as-a-Service) 제공자 또는 ABX(Action-Based Extensibility) 온-프레미스 내장형 통합 지점이 지원하는 실행 환경에서 다운로드한 제공자별 스크립트 또는 플러그인 세트를 사용합니다. 실행 환경은 외부 IPAM 제공자(예: Infoblox)에 연결하는 데 사용됩니다.

참고 Infoblox IPAM 통합 지점에는 ABX(Action-Based Extensibility) 온-프레미스 내장형 통합 지점이 필요합니다.

런타임 환경 유형마다 장점과 단점이 있습니다.

- **ABX(Action-Based Extensibility) 통합 지점**
 - 무료이며, 추가 벤더 사용 비용 없음
 - 공개적으로 액세스할 수 없는 NAT/방화벽 뒤에 있는 온-프레미스 데이터 센터에 상주하는 IPAM 벤더 장치에 연결 가능(예: Infoblox)
 - 상용 클라우드 벤더보다 느리고 성능 안정성이 약간 낮음
- **Amazon Web Services**
 - 관련 벤더 FaaS 연결/사용 비용이 있음
 - 공개적으로 액세스할 수 없는 NAT/방화벽 뒤에 있는 온-프레미스 데이터 센터에 상주하는 IPAM 벤더 장치에 연결 불가능

- 빠르고 매우 안정적인 성능
- Microsoft Azure
 - 관련 벤더 FaaS 연결/사용 비용이 있음
 - 공개적으로 액세스할 수 없는 NAT/방화벽 뒤에 있는 온-프레미스 데이터 센터에 상주하는 IPAM 벤더 장치에 연결 불가능
 - 빠르고 매우 안정적인 성능

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 [자격 증명](#) 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 IPAM 제공자(예: [Infoblox](#) 또는 [Bluecat](#))의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- IPAM 제공자(예: [Infoblox](#) 또는 [BlueCat](#))를 위해 배포된 통합 패키지에 액세스할 수 있는지 확인합니다. 배포된 패키지는 처음에 IPAM 제공자 웹 사이트 또는 [vRealize Automation Cloud Assembly](#) 마켓플레이스에서 .zip 다운로드로 가져온 후 [vRealize Automation Cloud Assembly](#)에서 배포됩니다.

제공자 패키지 .zip 파일을 배포하고 IPAM 통합 페이지에서 **제공자** 값으로 사용할 수 있도록 설정하는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly](#)에서 [사용할 외부 IPAM 제공자 패키지 다운로드 및 배포](#) 항목을 참조하십시오.

절차

- 1 IPAM 통합 실행 환경으로 사용할 온-프레미스 FaaS 기반 확장성 작업을 생성하려면 **확장성 > 라이브러리 > 작업**을 선택합니다.
- 2 **새 작업**을 클릭하고 작업 이름과 설명을 입력한 후 프로젝트를 지정합니다.
- 3 **FaaS 제공자** 드롭다운 메뉴에서 **온-프레미스**를 선택합니다.
- 4 양식을 작성하여 확장성 작업을 정의합니다.



실행 환경에 대한 관련 정보는 이 [Infoblox IPAM 플러그인 1.1 통합 블로그 비디오](#)(약 24분 후 표시됨)를 참조하십시오.

vRealize Automation에서 외부 IPAM 통합 지점 추가

vRealize Automation는 외부 IPAM 제공자와의 통합을 지원합니다. 제공자별 IPAM 통합 지점을 사용하여 Blueprint 배포를 위한 IP 주소 및 관련 네트워크 특성을 가져오고 관리할 수 있습니다.

이 예시에서는 외부 IPAM 제공자를 사용하여 조직의 계정에 대한 액세스를 지원하기 위해 외부 IPAM 통합 지점을 생성합니다. 이 예시 워크플로에서 IPAM 제공자는 Infoblox이고 제공자별 통합 패키지가 이미 있습니다. 이러한 지점은 Infoblox 통합에만 해당되지만 다른 외부 IPAM 제공자에 대한 IPAM 통합을 생성하는 경우 참조로 사용할 수 있습니다.

제공자별 통합 패키지는 IPAM 제공자의 웹 사이트, [VMware Solution Exchange](#) 마켓플레이스 또는 vRealize Automation Cloud Assembly [마켓플레이스](#) 탭에서 구할 수 있습니다.

이 예에서는 VMware에서 제공하는 Infoblox 패키지인 Infoblox.zip을 사용합니다. 이 패키지는 VMware Solution Exchange 마켓플레이스에서 다음과 같은 버전으로 다운로드할 수 있습니다.

- [vRA Cloud Infoblox 플러그인 버전 1.1](#) - vRealize Automation 8.1 지원
- [vRA Cloud Infoblox 플러그인 버전 1.0](#) - vRealize Automation 8.0.1 지원
- [vRA Cloud Infoblox 플러그인 버전 0.1](#) - vRealize Automation 8.0 지원

사전 요구 사항

- 클라우드 관리자 자격 증명이 있는지 확인합니다. [vRealize Automation](#)에서 클라우드 계정 작업에 필요한 [자격 증명](#) 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. [vRealize Automation Cloud Assembly](#) [사용자 역할이란?](#) 항목을 참조하십시오.
- 외부 IPAM 제공자의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- IPAM 제공자를 위해 배포된 통합 패키지에 액세스할 수 있는지 확인합니다. 배포된 패키지는 처음에 IPAM 제공자 웹 사이트 또는 VMware Solution Exchange 마켓플레이스에서 .zip 다운로드로 가져온 후 vRealize Automation에 배포됩니다.

제공자 패키지 .zip 파일을 다운로드 및 배포하고 IPAM 통합 페이지에서 **제공자** 값으로 사용할 수 있도록 설정하는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly](#)에서 [사용할 외부 IPAM 제공자 패키지 다운로드 및 배포](#) 항목을 참조하십시오.

- IPAM 제공자에 대해 구성된 실행 환경에 액세스할 수 있는지 확인합니다. 실행 환경은 일반적으로 ABX(작업 기반 확장성) 온-프레미스 내장형 통합 지점입니다.

실행 환경 특성에 대한 자세한 내용은 [vRealize Automation](#)에서 [IPAM 통합 지점에 대한 실행 환경 생성](#) 항목을 참조하십시오.

- Infoblox 애플리케이션에서 필요한 확장 가능 특성을 사용하도록 설정합니다. [vRealize Automation](#)와 통합하기 위해 [Infoblox](#) 애플리케이션에서 필요한 확장 가능 특성 추가의 내용을 참조하십시오.
- 외부 인터넷에 액세스할 수 없는 경우 인터넷 서버 프록시를 구성할 수 있습니다. [vRealize Automation](#)에 대한 [인터넷 프록시 서버를 구성하는 방법](#)의 내용을 참조하십시오.

절차

- 1 **인프라 > 연결 > 통합**을 선택하고 **통합 추가**를 클릭합니다.

2 IPAM을 클릭합니다.

3 **제공자** 드롭다운 목록에서 구성된 IPAM 제공자 패키지(예: *Infoblox_hrg*)를 선택합니다.

목록이 비어 있으면 **제공자 패키지 가져오기**를 클릭하고 기존 제공자 패키지 .zip 파일로 이동한 후 선택합니다. 제공자 .zip 파일이 없으면 IPAM 제공자의 웹 사이트나 vRealize Automation Cloud Assembly **마켓플레이스** 탭에서 가져올 수 있습니다.

vCenter에서 제공자 패키지 .zip 파일을 배포하고 [통합] 페이지에서 **제공자** 값으로 사용할 수 있도록 설정하는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 사용할 외부 IPAM 제공자 패키지 다운로드 및 배포 항목](#)을 참조하십시오.

최신 버전의 벤더 IPAM 통합 패키지를 사용하도록 기존 IPAM 통합을 업그레이드하는 방법에 대한 자세한 내용은 [vRealize Automation에서 최신 IPAM 통합 패키지로 업그레이드하는 방법](#)에서 참조하십시오.

4 제공자의 호스트 이름과 같은 다른 필수 필드(있는 경우)와 함께 외부 IPAM 제공자의 계정에 대한 관리자 사용자 이름 및 암호 자격 증명을 입력합니다.

이 예시에서는 다음 단계를 사용하여 Infoblox IPAM 제공자의 호스트 이름을 가져옵니다.

- a 별도의 브라우저 탭에서 Infoblox 관리자 자격 증명을 사용하여 IPAM 제공자 계정에 로그인합니다.
- b 호스트 이름 URL을 복사합니다.
- c IPAM 통합 페이지의 **호스트 이름** 필드에 호스트 이름 URL을 붙여넣습니다.

5 **실행 환경** 드롭다운 목록에서 기존 온-프레미스 작업 기반 확장성 통합 지점(예: *Infoblox_abx_intg*)을 선택합니다.

실행 환경은 vRealize Automation와 외부 IPAM 제공자 간의 통신을 지원합니다.

참고 Amazon Web Services 또는 Microsoft Azure 클라우드 계정을 통합 실행 환경으로 사용하는 경우에는 IPAM 제공자 장치를 인터넷에서 액세스할 수 있고, NAT 또는 방화벽이 뒤에 있지 않으며, 공개적으로 확인할 수 있는 DNS 이름이 있는지 확인하십시오. IPAM 제공자에 액세스할 수 없는 경우 Amazon Web Services Lambda 또는 Microsoft Azure 함수가 연결할 수 없으며 통합이 실패합니다. 관련 정보는 [vRealize Automation에서 IPAM 통합 지점에 대한 실행 환경 생성 항목](#)을 참조하십시오.

IPAM 프레임워크는 ABX(Action-Based Extensibility) 온-프레미스 내장형 실행 환경만 지원합니다.

참고 Infoblox IPAM 통합 지점에는 ABX(Action-Based Extensibility) 온-프레미스 내장형 통합 지점이 필요합니다.

구성된 클라우드 계정 또는 통합 지점을 사용하면 연결된 클라우드 확장성 프로시를 통해 vRealize Automation와 IPAM 제공자(이 예에서는 Infoblox) 간의 통신이 가능합니다. 이미 생성된 제공자를 선택하거나 새로 생성할 수 있습니다.

실행 환경을 생성하는 방법에 대한 자세한 내용은 [vRealize Automation에서 IPAM 통합 지점에 대한 실행 환경 생성 항목](#)을 참조하십시오.

6 검증을 클릭합니다.

이 예에서는 실행 환경에 대해 온 프레미스 작업 기반 확장성 통합을 사용하기 때문에 유효성 검사 작업을 볼 수 있습니다.

a **확장성** 탭을 클릭합니다.

b **작업 > 작업 실행**을 클릭하고 필터에서 **모든 실행** 또는 **통합 실행**을 선택하여 끝점 검증 작업이 시작되어 실행 중임을 확인합니다.

7 IPAM 제공자의 자체 서명된 인증서를 신뢰하라는 메시지가 표시되면 **수락**을 클릭합니다.

자체 서명된 인증서를 수락하면 검증 작업을 계속하여 완료할 수 있습니다.

8 이 IPAM 통합 지점의 **이름**(예: *Infoblox_Integration*)과 **설명**(예: *팀 HRG에 ABX 통합을 사용하는 Infoblox IPAM*)을 입력합니다.

9 **추가**를 클릭하여 새로운 외부 IPAM 통합 지점을 저장합니다.

데이터 수집 작업은 모방됩니다. 네트워크 및 IP 범위는 IPAM 제공자가 수집한 데이터입니다. 다음과 같이 데이터 수집 작업을 볼 수 있습니다.

a **확장성** 탭을 클릭합니다.

b **작업 > 작업 실행**을 클릭하고 데이터 수집 작업이 시작되어 실행되고 있는지 확인합니다. 작업 실행 콘텐츠를 열어서 볼 수 있습니다.

결과

이제 네트워크 및 네트워크 프로파일에 제공자별 외부 IPAM 통합을 사용할 수 있습니다.

vRealize Automation Cloud Assembly에서 IPAM 제공자 값을 사용하도록 네트워크 및 네트워크 프로파일 구성

내부 vRealize Automation Cloud Assembly가 아닌, 외부 IPAM 제공자가 관리하고 외부 IPAM 제공자로부터 가져오는 IP 주소 값을 사용하도록 네트워크를 정의할 수 있습니다.

조직의 외부 IPAM 제공자 계정에 정의한 기존 IP 설정에 액세스하도록 네트워크를 정의할 수 있습니다. 이 단계는 이전 단계에서 생성한 Infoblox 제공자 통합을 확장합니다.

이 예시에서는 vCenter에서 데이터를 수집한 기존 네트워크로 네트워크 프로파일을 구성합니다. 그런 다음 외부 IPAM 제공자(이 경우 Infoblox)에서 IP 정보를 가져오도록 해당 네트워크를 구성합니다. 이 네트워크 프로파일과 일치할 수 있는 vRealize Automation Cloud Assembly에서 프로비저닝하는 가상 시스템은 외부 IPAM 제공자로부터 IP 및 기타 TCP/IP 관련 설정을 가져옵니다.

네트워크에 대한 자세한 내용은 [네트워크 리소스](#) 항목을 참조하십시오. 네트워크 프로파일에 대한 자세한 내용은 [vRealize Automation Cloud Assembly 네트워크 프로파일을 추가하는 방법](#) 및 [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

사전 요구 사항

다음 일련의 단계는 IPAM 제공자 통합 워크플로의 컨텍스트에서 표시됩니다. 제공자별 외부 IPAM 통합 사용 사례의 내용을 참조하십시오.

- 클라우드 관리자 자격 증명이 있는지 확인합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 자격 증명 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. vRealize Automation Cloud Assembly 사용자 역할이란? 항목을 참조하십시오.
- 외부 IPAM 제공자(예: Infoblox 또는 Bluecat)의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다. 이 예시 워크플로에서 IPAM 제공자는 Infoblox입니다.
- IPAM 제공자에 대한 IPAM 통합 지점이 있는지 확인합니다. vRealize Automation에서 외부 IPAM 통합 지점 추가의 내용을 참조하십시오.

절차

- 1 네트워크를 구성하려면 **인프라 > 리소스 > 네트워크**를 클릭합니다.
- 2 **네트워크** 탭에서 IPAM 제공자 통합 지점에 사용할 기존 네트워크를 선택합니다. 이 예시에서 네트워크 이름은 *net.23.117-only-IPAM*입니다.

나열된 네트워크는 조직에 있는 vCenter에서 vRealize Automation Cloud Assembly가 데이터를 수집했습니다.
- 3 외부 IPAM 제공자로부터 값을 얻으려면 **계정/지역, 이름 및 네트워크 도메인**을 제외하고, 다음을 비롯한 다른 모든 네트워크 설정이 비어 있는지 확인합니다.
 - 도메인(8단계의 참고 참조)
 - CIDR
 - 기본 게이트웨이
 - DNS 서버
 - DNS 검색 도메인
- 4 **IP 범위** 탭을 클릭하고 **IPAM IP 범위 추가**를 클릭합니다.
- 5 **네트워크** 메뉴에서 방금 구성한 네트워크(예: *net.23.117-only-IPAM*)를 선택합니다.
- 6 **제공자** 메뉴에서 이전에 워크플로에서 생성한 *Infoblox_Integration* IPAM 통합 지점을 선택합니다.
- 7 지금 보이는 **주소 공간** 드롭다운 메뉴에서 나열된 네트워크 보기 중 하나를 선택합니다.

Infoblox의 주소 공간을 네트워크 보기라고 합니다.

네트워크 보기는 IPAM 제공자 계정에서 가져옵니다. 이 예시에서는 방금 구성한 네트워크 서브넷(예: *net.23.117-only-IPAM*), 이전에 워크플로에서 생성한 *Infoblox_Integration* 통합 지점 및 이름이 *default*인 주소 공간을 사용합니다.

나열된 주소 공간 값은 외부 IPAM 제공자로부터 가져옵니다.

- 8 선택한 주소 공간에 사용할 수 있는 표시된 네트워크 목록에서 하나 이상의 네트워크(예: 10.23.117.0/24)를 선택합니다.

이 예시에서는 선택한 네트워크에 대한 **도메인** 및 **DNS 서버** 열 값에 Infoblox의 값이 포함되어 있습니다.

참고 3단계에서 vRealize Automation Cloud Assembly에 도메인이 지정된 네트워크를 선택한 다음, 도메인 값이 포함된 외부 IPAM 제공자 주소 공간에서 네트워크를 선택하면, 외부 IPAM 제공자 네트워크의 도메인 값이 vRealize Automation Cloud Assembly에 지정된 도메인보다 우선합니다. IPAM IP 범위 설정에 도메인 값이 없으면(위에서 설명한 대로 Cloud Assembly 또는 외부 IPAM 제공자에 지정됨) 프로비저닝이 실패합니다.

- 9 **추가**를 클릭하여 네트워크에 대한 IPAM IP 범위를 저장합니다.

IP 범위 테이블에 범위가 표시됩니다.

- 10 **IP 주소** 탭을 클릭합니다.

외부 IPAM 제공자의 새 주소 범위를 사용하여 시스템을 프로비저닝한 후에는 **IP 주소** 테이블에 새 레코드가 표시됩니다.

- 11 네트워크를 사용하도록 네트워크 프로파일을 구성하려면 **인프라 > 구성 > 네트워크 프로파일**을 클릭합니다.
- 12 네트워크 프로파일에 이름(예: *Infoblox-NP*)을 지정하고 다음 샘플 설정을 추가합니다.

- 요약 탭

- vSphere 클라우드 계정/지역을 지정합니다.
- 네트워크 프로파일에 대한 기능 태그(예: 이름이 *infoblox_abx*인)를 추가합니다.
기능 태그를 기록해 둡니다. Blueprint에서 프로비저닝 연결을 수행하기 위해 기능 태그를 Blueprint 제약 조건 태그로 사용해야 하기 때문입니다.

- 네트워크 탭

- 이전에 생성한 네트워크를 추가합니다(예: *net.23.117-only-IPAM*).

- 13 **저장**을 클릭하여 이 설정으로 네트워크 프로파일을 저장합니다.

결과

네트워크 및 네트워크 프로파일 설정이 이제 외부 IPAM 통합을 지원하도록 구성되었으며 Blueprint 내에서 사용할 수 있습니다.

IPAM 제공자 범위 할당을 사용하는 vRealize Automation Cloud Assembly Blueprint 정의 및 배포

외부 IPAM 제공자로부터 IP 주소 할당을 가져오고 관리하는 Blueprint를 정의할 수 있습니다.

외부 IPAM 통합 워크플로의 마지막 단계에서는, 이전에 정의한 네트워크 및 네트워크 프로파일을 조직의 Infoblox 계정에 연결하는 Blueprint를 정의하고 배포하여, vRealize Automation Cloud Assembly가 아닌 외부 IPAM 제공자로부터 배포된 VM에 대한 IP 주소 할당을 확보하고 관리합니다.

이 워크플로에서는 Infoblox를 외부 IPAM 제공자로 사용하며 일부 단계에서 예제 값은 Infoblox에 고유합니다. 단, 이러한 절차는 다른 외부 IPAM 통합에도 적용될 수 있습니다.

Blueprint를 배포하고 VM이 시작되면, 배포에서 각 VM에 사용된 IP 주소는 **리소스 > 네트워크** 페이지에 네트워크 항목으로 나타나며, IPAM 제공자 계정의 IPAM 제공자 네트워크 및 호스트 vCenter에 배포된 각 VM에 대한 vSphere Web Client 레코드에 새 호스트 레코드로 나타납니다.

사전 요구 사항

다음 일련의 단계는 외부 IPAM 제공자 통합 워크플로의 컨텍스트에 기반합니다. **제공자별 외부 IPAM 통합 사용 사례**의 내용을 참조하십시오.

- 클라우드 관리자 자격 증명이 있는지 확인합니다. vRealize Automation에서 클라우드 계정 작업에 필요한 **자격 증명** 항목을 참조하십시오.
- 클라우드 관리자 사용자 역할이 있는지 확인합니다. vRealize Automation Cloud Assembly **사용자 역할이란?** 항목을 참조하십시오.
- 외부 IPAM 제공자(예: Infoblox 또는 BlueCat)의 계정이 있고, IPAM 제공자를 통해 조직의 계정에 대한 올바른 액세스 자격 증명이 있는지 확인합니다.
- 호스트 vCenter에 배포된 VM에 대한 vSphere 웹 클라이언트 레코드에서 상태 레코드를 표시하는 데 필요한 호스트 계정 및 역할 요구 사항에 대한 관리자 액세스 권한이 있는지 확인합니다.
- 외부 IPAM 제공자에 대한 IPAM 통합 지점이 있는지 확인합니다. vRealize Automation에서 **외부 IPAM 통합 지점 추가**의 내용을 참조하십시오.
- 의도한 IPAM 통합 지점에 대해 외부 IPAM 통합을 지원하는 vRealize Automation Cloud Assembly 네트워크 및 네트워크 프로파일을 구성했는지 확인합니다. vRealize Automation Cloud Assembly에서 **IPAM 제공자 값을 사용하도록 네트워크 및 네트워크 프로파일 구성**의 내용을 참조하십시오.
- 프로젝트 및 클라우드 영역이 IPAM 통합 지점 및 네트워크 또는 네트워크 프로파일의 태그와 일치하도록 태그가 지정되어 있는지 확인합니다. 필요한 경우 사용자 지정 리소스 이름 지정을 지원하도록 프로젝트를 구성합니다.

Blueprint에서 프로젝트 및 클라우드 영역의 역할 및 기타 인프라 요소의 역할에 대한 자세한 내용은 **WordPress 사용 사례** 항목을 참조하십시오. 태그 지정에 대한 자세한 내용은 **태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법** 항목을 참조하십시오.

프로젝트의 설정을 사용한, 사용자에게 맞게 VM 이름 지정에 대한 자세한 내용은 vRealize Automation Cloud Assembly를 사용하여 배포된 리소스의 이름을 사용자 지정하는 방법 항목을 참조하십시오.

절차

- 1 **Blueprints > 새로 만들기**를 클릭하고 **새 Blueprint** 페이지에 다음 정보를 입력하고 **생성**을 클릭합니다.

- **이름** = ipam-bpa
- **설명** = Infoblox IPAM 통합을 사용하는 Blueprint
- **프로젝트** = 123VC

- 2 이 예에서는 클라우드 애그노스틱 시스템 구성 요소와 클라우드 애그노스틱 네트워크 구성 요소를 Blueprint 캔버스에 추가하고 두 구성 요소를 연결합니다.
- 3 Blueprint 코드를 편집하여 네트워크 프로파일에 추가한 기능 태그와 일치하는 네트워크 구성 요소에 제약 조건 태그를 추가합니다. 이 예에서는 해당 태그 값이 *infoblox_abx*입니다.
- 4 Blueprint 코드를 편집하여 네트워크 할당 유형이 *정적*인 것으로 지정합니다.

이 예에서 지정된 IP 주소 10.23.117.4는 연결된 네트워크 프로파일의 네트워크에 대해 선택한 외부 IPAM 주소 공간에서 현재 사용 가능한 것으로 알려져 있습니다. 정적 할당 설정은 필수이지만 주소 값은 아닙니다. 특정 주소에서 외부 IP 주소 선택을 시작하려는 경우에는 지정할 수 있지만 필수가 아닙니다. 주소 값을 지정하지 않으면 외부 IPAM 제공자는 외부 IPAM 네트워크에서 사용 가능한 다음 주소를 선택합니다.

- 5 다음 예제와 비교하여 Blueprint 코드를 확인합니다.

```
formatVersion: 1
inputs: {}
resources:
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      networkType: existing
      name: ipam
      constraints:
        - tag: infoblox_abx
  Cloud_Machine_1:
    type: Cloud.Machine
    properties:
      image: ubuntu
      flavor: small
      networks:
```

```
- network: '${resource.Cloud_Network_1.id}'
  assignment: static
  address: 10.23.117.4
  name: '${resource.Cloud_Network_1.name}'
```

참고 기본적으로 Infoblox 통합은 Infoblox의 기본 DNS 보기에 DNS 호스트 레코드를 생성합니다.

Infoblox 관리자가 *사용자 지정* DNS 보기를 생성한 경우, 시스템 구성 요소의

Infoblox.IPAM.Network.dnsView 속성을 사용하여 기본 통합 동작을 덮어쓰고 명명된 보기를 지정할 수 있습니다. 예를 들어, 다음 속성을 Cloud_Machine_1 구성 요소에 추가하여 Infoblox에서 명명된 DNS 보기를 지정할 수 있습니다.

```
Cloud_Machine_1:
  type: Cloud.Machine
  properties:
    image: ubuntu
    flavor: small
    Infoblox.IPAM.Network.dnsView:<DNS-보기-이름>
```

DNS 보기 구성 및 사용에 대한 자세한 내용은 Infoblox 제품 설명서의 [DNS 보기](#)를 참조하십시오.

- 6 Blueprint 페이지에서 **배포**를 클릭하고 배포 이름을 *Infoblox-1*로 지정한 후 **배포 유형** 페이지에서 **배포**를 클릭합니다.
- 7 Blueprint가 배포되는 동안 **확장성** 탭을 클릭하고 **작업 > 작업 실행**을 선택하여 *Infoblox_AllocateIP_n* 확장성 작업이 실행 중인지 확인합니다.
확장성 작업이 완료되고 시스템이 프로비저닝되면 *Infoblox_Update_n* 작업이 MAC 주소를 Infoblox로 전파합니다.
- 8 Infoblox 계정에 로그인하여 열면 연결된 10.23.117.0/24 네트워크의 IPAM 주소에 대한 새 호스트 레코드를 볼 수 있습니다. Infoblox의 DNS 탭을 열어서 새 DNS 호스트 레코드를 볼 수도 있습니다.
- 9 VM이 프로비저닝되고 있는지 확인하려면 호스트 vCenter 및 vSphere Web Client에 로그인하여 프로비저닝된 시스템을 찾아서 DNS 이름과 IP 주소를 확인합니다.
프로비저닝된 VM이 시작된 후에는 *Infoblox_AllocateIP* 확장성 작업에 의해 MAC 주소가 Infoblox에 전파됩니다.
- 10 vRealize Automation Cloud Assembly에서 새 네트워크 레코드를 보려면 **인프라 > 리소스 > 네트워크**를 선택하고 **IP 주소** 탭을 클릭하여 엽니다.
- 11 배포를 삭제하면 배포에서 VM의 IPAM 주소가 해제되어 해당 IP 주소를 외부 IPAM 제공자가 다른 할당에 다시 사용할 수 있습니다. vRealize Automation Cloud Assembly에서 이 이벤트에 대한 확장성 작업은 *Infoblox_Deallocate*입니다.

vRealize Automation에서 IPAM 통합에 대해 Infoblox 특정 속성 및 확장 가능 특성 사용

Infoblox용 외부 IPAM 통합을 포함하는 vRealize Automation 프로젝트에 대해 Infoblox 특정 속성을 사용할 수 있습니다.

다음 Infoblox 속성을 Infoblox IPAM 통합에서 사용할 수 있습니다. Blueprint 배포 중 IP 주소 할당을 추가적으로 제어하기 위해 vRealize Automation에서 사용할 수 있습니다. 이러한 속성의 사용은 선택 사항입니다.

다음과 같은 모든 Infoblox 속성을 vRealize Automation 8.0용 vRA Cloud Infoblox 플러그인 버전 0.1 패키지에서 사용할 수 있지만 Infoblox.IPAM.Network.dnsView 속성은 vRealize Automation 8.0.1용 vRA Cloud Infoblox 플러그인 버전 1.0 및 vRealize Automation 8.1용 vRA Cloud Infoblox 플러그인 버전 1.1 패키지에서만 사용할 수 있습니다.

참고 이러한 속성의 사용은 **제공자별 외부 IPAM 통합 사용 사례** 샘플 워크플로에 포함되어 있지 않습니다.

■ Infoblox.IPAM.createFixedAddress

이 속성을 사용하면 Infoblox 내에서 고정 주소 레코드를 생성할 수 있습니다. 가능한 값은 True 및 False입니다. 기본적으로 호스트 레코드가 생성됩니다. 기본값: False.

■ Infoblox.IPAM.Network.dnsView

이 속성을 사용하면 Infoblox 내에서 호스트 레코드를 생성할 때 DNS 보기를 사용할 수 있습니다. 기본값: default.

■ Infoblox.IPAM.Network.enableDns

Infoblox에서 IP를 할당할 때 이 속성을 사용하면 DNS 레코드도 생성할 수 있습니다. 가능한 값은 True 및 False입니다. 기본값: True.

■ Infoblox.IPAM.Network.dnsSuffix

이 속성을 사용하면 Infoblox 네트워크의 도메인 DHCP 옵션을 새 항목으로 덮어쓸 수 있습니다. 이 기능은 Infoblox 네트워크에 도메인 DHCP 옵션이 설정되어 있지 않거나 도메인 DHCP 옵션을 덮어써야 하는 경우에 유용합니다. 기본값: 없음(빈 문자열).

Infoblox.IPAM.Network.dnsSuffix는 Infoblox.IPAM.Network.enableDns가 True로 설정된 경우에만 적용됩니다.

vRealize Automation Cloud Assembly에서 다음 방법 중 하나를 사용하여 Infoblox 속성을 지정할 수 있습니다.

- **인프라 > 구성 > 프로젝트** 페이지에서 **사용자 지정 속성** 섹션을 사용하여 프로젝트의 속성을 지정할 수 있습니다. 이 방법을 사용하면 지정된 속성이 이 프로젝트의 범위에서 프로비저닝된 모든 시스템에 적용됩니다.

- Blueprint의 각 시스템 구성 요소에서 속성을 지정할 수 있습니다.

Infoblox.IPAM.Network.dnsView 속성의 사용을 설명하는 샘플 Blueprint 코드는 다음과 같습니다.

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      Infoblox.IPAM.Network.dnsView: default
      image: ubuntu
      cpuCount: 1
      totalMemoryMB: 1024
      networks:
        - network: '${resource.Cloud_Network_1.id}'
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      networkType: existing
      constraints:
        - tag: mk-ipam-demo
```

- 확장성 구독을 사용하여 속성을 지정할 수 있습니다.

이 사용 사례와 관련된 Infoblox 확장 가능 특성에 대한 자세한 내용은 [vRealize Automation](#)와 통합하기 위해 [Infoblox](#) 애플리케이션에서 필요한 확장 가능 특성 추가에서 참조하십시오.

vRealize Automation Cloud Assembly 리소스 인프라 구축

4

vRealize Automation Cloud Assembly 리소스 인프라는 클라우드 계정 영역을 Blueprint 및 해당 워크로드를 배포할 수 있는 영역으로 정의하는 곳입니다.

또한 리소스 인프라에는 이미지 및 시스템 크기와 클라우드 계정 지역 또는 데이터 센터 간에 네트워크 및 스토리지 기능을 정의하는 프로파일에 대한 일반적인 매핑이 생성됩니다.

본 장은 다음 항목을 포함합니다.

- vRealize Automation Cloud Assembly 대상 배치 지역 또는 데이터 센터를 정의하는 클라우드 영역을 추가하는 방법
- vRealize Automation Cloud Assembly에서 공통 시스템 크기를 생성하도록 vRealize Automation Cloud Assembly 플레이버 매핑을 추가하는 방법
- 공통 운영 체제를 생성하도록 vRealize Automation Cloud Assembly 이미지 매핑을 추가하는 방법
- vRealize Automation Cloud Assembly 네트워크 프로파일을 추가하는 방법
- 다른 요구 사항을 설명하는 vRealize Automation Cloud Assembly 스토리지 프로파일을 추가하는 방법
- 태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법
- vRealize Automation Cloud Assembly에서 리소스로 작업하는 방법

vRealize Automation Cloud Assembly 대상 배치 지역 또는 데이터 센터를 정의하는 클라우드 영역을 추가하는 방법

vRealize Automation Cloud Assembly 클라우드 영역은 클라우드 계정 유형(예: AWS 또는 vSphere) 내의 리소스 집합입니다.

특정 계정 지역의 클라우드 영역은 Blueprint가 워크로드를 배포하는 위치입니다. 각 클라우드 영역은 vRealize Automation Cloud Assembly 프로젝트와 연결됩니다.

인프라 > 구성 > 클라우드 영역을 선택하고 **새 영역 추가**를 클릭합니다.

vRealize Automation Cloud Assembly 클라우드 영역에 대해 알아보기

vRealize Automation Cloud Assembly 클라우드 영역은 클라우드 계정 유형(예: AWS 또는 vSphere) 내에 있는 섹션입니다. 클라우드 영역은 프로젝트별로 정의됩니다.

추가적인 배치 컨트롤로는 배치 정책 옵션, 기능 태그 및 계산 태그가 포함됩니다.

■ 배치 정책

배치 정책은 지정된 지역 내에서 호스트를 선택하는 데 사용됩니다.

- **default** - 계산 리소스를 임의로 배치합니다.
- **binpack** - 지정된 계산을 실행하기에 충분한 가용 리소스가 있는, 가장 많이 로드된 호스트에 계산 리소스를 배치합니다.
- **spread** - 계산 리소스를 호스트에 균등하게 배치합니다.

■ 기능 태그

Blueprint에는 배포 배치를 결정하는 데 도움이 되는 제약 조건 태그가 포함됩니다. 배포 시 Blueprint 제약 조건 태그가 클라우드 영역의 일치하는 기능 태그에 매핑되어, 계산 리소스 배치에 사용할 수 있는 클라우드 영역이 결정됩니다.

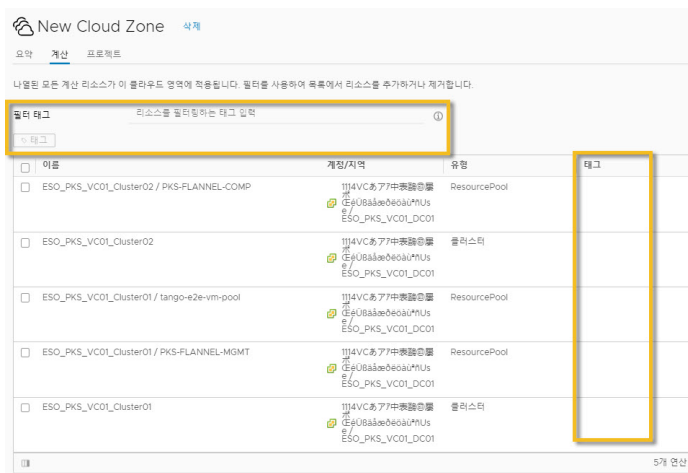
■ 계산

이 클라우드 영역에 프로비저닝할 수 있는 계산 리소스를 보고 관리할 수 있습니다.

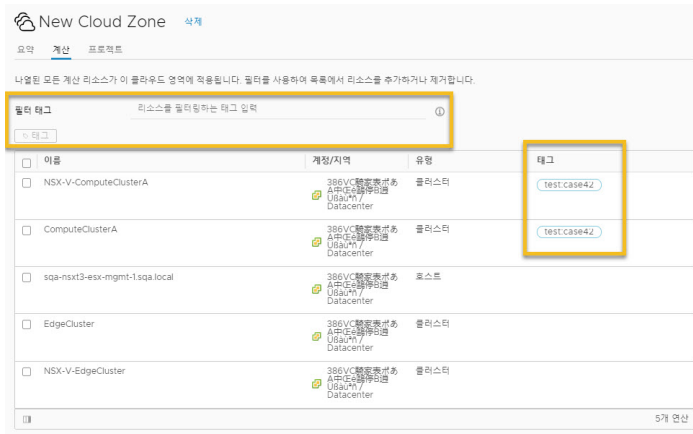
vCenter 계산 클러스터가 DRS를 지원하는 경우 클라우드 영역은 계산 목록에 클러스터만 표시하고 하위 호스트는 표시하지 않습니다. vCenter 계산 클러스터가 DRS를 지원하지 않은 경우 클라우드 영역은 독립형 ESXi 호스트(있는 경우)만 표시합니다.

계산 태그는 배치를 추가적으로 제어하는 데 도움이 됩니다. 다음의 예시와 같이 하나 이상의 태그와 일치하는 계산 리소스만 필터링할 수 있습니다.

- 계산 리소스에 태그가 없고 필터링을 사용하지 않습니다.



- 계산 리소스 2개에 동일한 태그가 있고 필터링을 사용하지 않습니다.



- 계산 리소스 2개에 동일한 태그가 있고, 태그가 계산 리소스 2개에 사용된 태그와 일치합니다.



■ 프로젝트

이 클라우드 영역으로의 워크로드 프로비저닝을 지원하도록 구성된 프로젝트를 볼 수 있습니다.

클라우드 영역을 생성한 후에는 해당 구성을 검증할 수 있습니다.

vRealize Automation Cloud Assembly에서 공통 시스템 크기를 생성하도록 vRealize Automation Cloud Assembly 플레이버 매핑을 추가하는 방법

vRealize Automation Cloud Assembly 플레이버 맵은 자연어를 사용하여 특정 클라우드 계정/지역에 대한 대상 배포 크기를 정의하는 곳입니다.

플레이버 맵은 환경에 적합한 배포 크기를 표현합니다. 한 가지 예는 명명된 데이터 센터의 vCenter 계정에 대한 CPU 1개 메모리 2GB의 경우 "소규모", CPU 2개 메모리 8GB의 경우 "대규모" 및 명명된 지역의 Amazon Web Services 계정에 대한 t2.nano입니다.

인프라 > 구성 > 플레이버 매핑을 선택하고 **새 플레이버 매핑**을 클릭합니다.

vRealize Automation Cloud Assembly의 버전 매핑에 대해 알아보기

버전 매핑은 자연어 이름 지정 방법을 사용하여 vRealize Automation Cloud Assembly에서 특정 클라우드 계정/지역에 대한 대상 배포 크기 집합을 그룹화합니다.

버전 매핑을 사용하면 계정 지역 전체에서 유사한 버전 크기를 포함하는 이름이 지정된 매핑을 생성할 수 있습니다. 예를 들어 `standard_small`이라는 버전 맵에는 프로젝트의 사용 가능한 계정/지역 전체 또는 일부에 사용할 수 있는 유사한 버전 크기(예: CPU 1개, 2GB RAM)가 포함될 수 있습니다. Blueprint를 구축할 때는 필요에 맞는 사용 가능한 버전을 선택합니다.

배포 의도에 따라 프로젝트에 대한 버전 매핑을 구성합니다.

Blueprint 생성을 간소화하기 위해, 새 클라우드 계정을 추가할 때 사전 구성 옵션을 선택할 수 있습니다. 사전 구성 옵션을 선택하면 지정된 지역에 대해 조직에서 가장 널리 사용되는 버전 매핑 및 이미지 매핑이 선택됩니다.

vSphere 리소스가 포함된 Blueprint의 이미지 매핑과 관련하여, vSphere 클라우드 영역에 대해 정의된 버전 매핑이 없으면 Blueprint에서 vSphere 관련 설정을 사용하여 무제한 메모리 및 CPU를 구성할 수 있습니다. vSphere 클라우드 영역에 대해 정의된 버전 매핑이 있으면 버전 매핑은 Blueprint에서 vSphere 관련 구성에 대한 제한으로 작동합니다.

기본 버전 매핑 예시는 [WordPress 사용 사례: 플레이버 매핑 추가 항목을 참조하십시오](#).

공통 운영 체제를 생성하도록 vRealize Automation Cloud Assembly 이미지 매핑을 추가하는 방법

vRealize Automation Cloud Assembly 이미지 맵은 자연어를 사용하여 특정 클라우드 계정/지역에 대한 대상 배포 운영 체제를 정의하는 곳입니다.

인프라 > 구성 > 이미지 매핑을 선택하고 **새 이미지 매핑**을 클릭합니다.

vRealize Automation Cloud Assembly의 이미지 매핑에 대해 알아보기

이미지 매핑은 자연어 이름 지정을 사용하여 vRealize Automation Cloud Assembly에서 특정 클라우드 계정/지역에 대해 미리 정의된 대상 OS 규격 집합을 그룹화합니다.

Microsoft Azure 및 Amazon Web Services 같은 클라우드 벤더 계정은 이미지를 사용하여 OS 및 관련 구성 설정을 포함한 대상 배포 조건 집합을 그룹화합니다. VMware Cloud on AWS를 포함한 vCenter 및 NSX 기반 환경에서도 유사한 그룹화 메커니즘을 사용하여 OS 배포 조건 집합을 정의합니다. Blueprint를 구축한 후 최종적으로 배포하고 반복할 때 필요에 가장 적합한 이미지를 선택합니다.

프로젝트에 대한 이미지 매핑은 유사한 운영 체제, 태그 지정 전략 및 기능적인 배포 의도에 맞게 구성해야 합니다.

기본 이미지 매핑을 정의하는 방법에 대한 예제는 [WordPress 사용 사례: 이미지 매핑 추가 항목을 참조하십시오](#).

Blueprint 생성을 간소화하기 위해, 새 클라우드 계정을 추가할 때 사전 구성 옵션을 선택할 수 있습니다. 사전 구성 옵션을 선택하면 지정된 지역에 대해 조직에서 가장 널리 사용되는 버전 매핑 및 이미지 매핑이 선택됩니다.

Blueprint에 이미지 정보를 추가하는 경우에는 시스템 구성 요소의 properties 섹션에서 image 또는 imageRef 항목 중 하나를 사용합니다. 예를 들어 스냅샷에서 복제하려는 경우에는 imageRef 속성을 사용합니다.

Blueprint 코드의 image 및 imageRef 항목에 대한 예는 [장 6 vRealize Automation Cloud Assembly 배포 설계](#) 항목을 참조하십시오.

컨텐츠 라이브러리에 대한 사용 권한을 할당하려면 관리자가 사용 권한을 사용자에게 글로벌 사용 권한으로 부여해야 합니다. 관련 정보는 [VMware vSphere 설명서](#)의 "vSphere 가상 시스템 관리"에서 [컨텐츠 라이브러리에 대한 사용 권한의 계층적 상속](#)을 참조하십시오.

클라우드 계정/지역에 대한 이미지 동기화

이미지 동기화를 실행하여 **인프라 > 구성 > 이미지 매핑** 페이지에서 지정된 클라우드 계정/지역에 대해 추가하거나 제거할 이미지가 최신 상태인지 확인할 수 있습니다.

- 1 **인프라 > 연결 > 클라우드 계정**을 선택하여 연결된 **클라우드 계정/지역**을 엽니다. 기존 클라우드 계정/지역을 선택합니다.
- 2 **이미지 동기화** 버튼을 클릭하고 작업을 완료합니다.



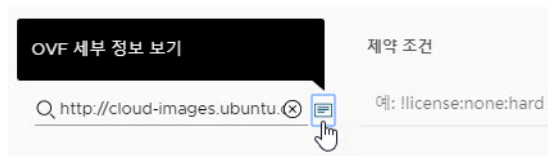
- 3 작업이 완료되면 **인프라 > 구성 > 이미지 매핑**을 클릭합니다. 기존 이미지 매핑을 편집하거나 새로 정의하고 1단계의 클라우드 계정/지역을 선택합니다.
- 4 **이미지 매핑** 페이지에서 이미지 동기화 아이콘을 클릭합니다.



- 5 **이미지 매핑** 페이지에서 지정된 클라우드 계정/지역에 대한 이미지 매핑 설정을 구성합니다.

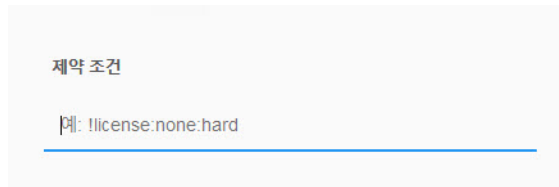
OVF 세부 정보 보기

vRealize Automation Cloud Assembly Blueprint 개체(예: vCenter 시스템 구성 요소 및 이미지 맵)에 OVF 규격을 포함할 수 있습니다. 이미지에 OVF 파일이 포함되면 파일을 열지 않고도 해당 콘텐츠를 검색할 수 있습니다. OVF 위로 마우스를 이동하면 이름과 위치를 비롯한 OVF 세부 정보가 표시됩니다. OVF 파일 형식에 대한 자세한 내용은 [vcenter ovf: 속성](#)을 참조하십시오.



제약 조건 및 태그를 사용하여 이미지 선택 구체화

Blueprint에서 이미지 선택을 더욱 구체화하려면, 제약 조건을 하나 이상 추가하여, 배포할 수 있는 이미지 유형에 대해 태그 기반 제약을 지정하면 됩니다. 이미지 매핑 구성을 생성하거나 편집할 때 표시되는 제공된 **제약 조건**의 예는 `!license:none:hard`입니다. 이 예시는 Blueprint에 `license:none` 태그가 "없음" 경우에만 이미지를 사용할 수 있는 태그 기반 제약을 보여줍니다. `license:88` 및 `license:92`와 같은 태그를 추가하면 Blueprint에 `license:88` 및 `license:92` 태그가 "있는" 경우에만 지정된 이미지를 사용할 수 있습니다.



클라우드 구성 스크립트를 사용하여 배포 제어

이미지 맵, Blueprint 또는 둘 다에 클라우드 구성 스크립트를 사용하여 vRealize Automation Cloud Assembly 배포에 사용할 사용자 지정 OS 특성을 정의할 수 있습니다. 예를 들어 Blueprint를 공용 또는 사설 클라우드 중 어디에 배포하는지에 따라 특정 사용자 사용 권한, OS 사용 권한 또는 기타 조건을 이미지에 적용할 수 있습니다. 클라우드 구성 스크립트는 Linux 기반 이미지에 대해서는 `cloud-init` 형식을 따르고 Windows 기반 이미지에 대해서는 `cloudbase-init` 형식을 따릅니다. vRealize Automation Cloud Assembly는 Linux 시스템에 대해서는 `cloud-init` 도구를 지원하고 Windows에 대해서는 `cloudbase-init` 도구를 지원합니다.

Windows 시스템의 경우에는 `cloudbase-init`에서 지원하는 모든 클라우드 구성 스크립트 형식을 사용할 수 있습니다.

아래의 샘플 Blueprint 코드에 나오는 시스템 리소스는 클라우드 구성 스크립트가 포함된 이미지를 사용하며, 해당 내용은 `image` 항목 아래에 나와 있습니다.

```
resources:
  demo-machine:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      image: MyUbuntu16
      https://cloud-images.ubuntu.com/releases/16.04/release-20170307/ami-ubuntu-16.04-1.10.3-00-15269239.ova
      cloudConfig: |
        ssh_pwauth: yes
        chpasswd:
          list: |
            ${input.username}:${input.password}
          expire: false
        users:
          - default
          - name: ${input.username}
            lock_passwd: false
            sudo: ['ALL=(ALL) NOPASSWD:ALL']
```

```
groups: [wheel, sudo, admin]
shell: '/bin/bash'
runcmd:
  - echo "Defaults:${input.username} !requiretty" >> /etc/sudoers.d/${input.username}
```

이미지 매핑과 Blueprint에 클라우드 구성 스크립트가 포함된 경우 발생하는 결과

클라우드 구성 스크립트가 포함된 Blueprint가 클라우드 구성 스크립트가 포함된 이미지 매핑을 사용할 경우, 두 스크립트가 결합됩니다. 병합 작업은 스크립트가 #cloud-config 형식인지 여부를 고려하면서 이미지 매핑 스크립트의 콘텐츠를 우선 처리하고 다음으로 Blueprint 스크립트의 콘텐츠를 처리합니다.

- #cloud-config 형식의 스크립트에 대해, 병합은 각 모듈(예: runcmd, users 및 write_files)의 콘텐츠를 다음과 같이 결합합니다.
 - 콘텐츠가 목록인 모듈의 경우 두 목록 모두에서 동일한 명령은 제외하고 이미지 매핑의 명령 목록과 Blueprint의 명령 목록이 병합됩니다.
 - 콘텐츠가 사전인 모듈의 경우 명령이 병합되고 그 결과 사전이 결합됩니다. 두 사전에 동일한 키가 있는 경우 이미지 매핑 스크립트 사전의 키는 유지되고 Blueprint 스크립트 사전의 키는 무시됩니다.
 - 콘텐츠가 문자열인 모듈의 경우 이미지 매핑 스크립트의 콘텐츠 값이 유지되고 Blueprint 스크립트의 콘텐츠 값은 무시됩니다.
- 두 스크립트 모두 #cloud-config 이외의 형식이거나 한 스크립트는 #cloud-config 형식이지만 다른 스크립트가 다른 형식인 경우, 두 스크립트는 이미지 매핑 스크립트가 먼저 실행되고 Blueprint 스크립트는 이미지 매핑 스크립트가 완료될 때 실행되는 방식으로 결합됩니다.

관련 정보는 [사용자 데이터 섹션 병합](#)을 참조하십시오.

클라우드 구성 스크립트 구성 및 사용에 대한 자세한 정보

vRealize Automation Cloud Assembly에서 Windows 배포용 cloud-init를 구성하려면 [vRealize Automation](#)에서 [cloud-init](#) 또는 [cloudbase-init](#)를 사용하여 [Windows](#) 템플릿을 설정하는 방법 항목을 참조하십시오.

클라우드 구성 스크립트 사용에 대한 자세한 내용은 [vRealize Automation Cloud Assembly Blueprint](#)에서 시스템을 자동으로 초기화하는 방법 항목 및 VMware 블로그 문서 [Customizing Cloud Assembly Deployments with Cloud-Init](#)을 참조하십시오.

vRealize Automation Cloud Assembly 네트워크 프로파일을 추가하는 방법

vRealize Automation Cloud Assembly 네트워크 프로파일은 배포할 네트워크의 동작을 설명합니다.

예를 들어 네트워크가 인터넷 연결되거나 및 내부 전용이어야 할 수 있습니다. 네트워크 및 해당 프로파일은 클라우드별로 다릅니다.

인프라 > 구성 > 네트워크 프로파일을 선택하고 **새 네트워크 프로파일**을 클릭합니다.

vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기

네트워크 프로파일은 vRealize Automation Cloud Assembly의 특정 지역 또는 데이터 센터의 클라우드 계정에서 사용할 수 있는 네트워크 설정 및 네트워크 그룹을 정의합니다.

일반적으로 네트워크 프로파일은 예를 들어 기존 네트워크에 아웃바운드 액세스만 있는 소규모 테스트 환경이나 일련의 보안 정책이 필요한 대규모의 로드 밸런싱된 운영 환경과 같은 대상 배포 환경을 지원하기 위해 정의합니다. 네트워크 프로파일을 워크로드별 네트워크 특성의 모음으로 생각할 수 있습니다.

네트워크 프로파일의 내용

네트워크 프로파일에는 다음 설정을 포함하여 vRealize Automation Cloud Assembly의 명명된 클라우드 계정 유형 및 지역에 대한 특정 정보가 포함되어 있습니다.

- 네트워크 프로파일에 대한 명명된 클라우드 계정/지역 및 선택적 기능 태그.
- 명명된 기존 네트워크 및 해당 설정.
- 네트워크 프로파일의 주문형 네트워크 및 기타 측면을 정의하는 네트워크 정책.
- 기존 로드 밸런서가 선택적으로 포함됩니다.
- 기존 보안 그룹이 선택적으로 포함됩니다.

네트워크 프로파일을 기반으로 네트워크 IP 관리 기능을 결정합니다.

네트워크 프로파일 기능 태그는 Blueprint의 제약 조건 태그와 일치하기 때문에 네트워크 선택을 제어하는 데 도움을 줍니다. 네트워크 프로파일별로 수집되는 네트워크에 할당된 모든 태그도 Blueprint의 태그와 일치하기 때문에 Blueprint를 배포할 때 네트워크 선택을 제어하는 데 유용합니다.

기능 태그는 선택 사항입니다. 기능 태그는 네트워크 프로파일의 모든 네트워크에 적용되지만 이것은 네트워크가 해당 네트워크 프로파일의 일부로 사용되는 경우로 국한됩니다. 기능 태그가 없는 네트워크 프로파일의 경우 네트워크 태그에서만 태그 일치가 발생합니다. 매칭된 네트워크 프로파일에 정의된 네트워크 및 보안 설정은 Blueprint가 배포될 때 적용됩니다.

고정 IP를 사용하는 경우 주소 범위는 vRealize Automation에서 관리합니다. DHCP의 경우 IP 시작 및 끝 주소는 vRealize Automation이 아닌 독립 DHCP 서버에서 관리합니다. DHCP 또는 혼합 네트워크 주소 할당을 사용하는 경우 네트워크 활용도 값은 0으로 설정됩니다. 주문형 네트워크 할당 범위는 네트워크 프로파일에 지정된 CIDR 및 서브넷 크기를 기반으로 합니다. 배포에서 정적 및 동적 할당을 모두 지원하기 위해 할당된 범위는 정적 할당을 위한 범위 하나와 동적 할당을 위한 범위 하나의 두 개 범위로 나뉩니다.

네트워크

서브넷이라고도 하는 네트워크는 IP 네트워크의 논리적인 한 구획입니다. 네트워크 그룹은 클라우드 계정, IP 주소 또는 범위, 네트워크 태그를 그룹화하여 Blueprint 배포를 프로비저닝하는 방법과 위치를 제어합니다. 프로파일의 네트워크 매개 변수는 배포의 시스템이 IP 레이어 3을 통해 서로 통신할 수 있는 방법을 정의합니다. 네트워크에는 태그가 있을 수 있습니다.

네트워크를 네트워크 프로파일에 추가하고, 네트워크 프로파일에 사용되는 네트워크 측면을 편집하고, 네트워크 프로파일에서 네트워크를 제거할 수 있습니다.

■ 네트워크 도메인 또는 전송 영역

네트워크 도메인이나 전송 영역은 vSphere vNetwork Distributed 포트 그룹(dvPortGroup)에 대한 분산 가상 스위치(dvSwitch)입니다. 전송 영역은 *dvSwitch* 또는 *dvPortGroup*과 같은 용어와 유사한 기존 NSX 개념입니다.

NSX 클라우드 계정을 사용하는 경우 페이지의 요소 이름은 **전송 영역**이고, 그렇지 않으면 **네트워크 도메인**입니다.

표준 스위치의 경우 네트워크 도메인이나 전송 영역은 스위치 자체와 동일합니다. 네트워크 도메인이나 전송 영역은 vCenter 내 서브넷의 경계를 정의합니다.

전송 영역은 NSX 논리적 스위치가 도달할 수 있는 호스트를 제어합니다. 하나 이상의 vSphere 클러스터에 걸쳐있을 수 있습니다. 전송 영역에서는 특정 네트워크 사용에 참여할 수 있는 클러스터 및 가상 시스템을 제어합니다. 동일한 NSX 전송 영역에 속하는 서브넷은 동일한 시스템 호스트에 사용할 수 있습니다.

■ 도메인

대상 가상 시스템의 vCenter Single Sign-On 도메인입니다. 도메인은 vSphere 구성 중에 vCenter 관리자에 의해 구성됩니다. 도메인은 vCenter의 로컬 인증 공간을 결정합니다.

■ IPv4 CIDR 및 IPv4 기본 게이트웨이

Blueprint의 vSphere 클라우드 계정 및 vSphere 시스템 구성 요소는 이중 IPv6 및 IPv4 인터넷 프로토콜 메서드를 지원합니다. 예: 192.168.100.14/24는 IPv4 주소 192.168.100.14 및 연결된 라우팅 접두사 192.168.100.0, 또는 해당 서브넷 마스크 255.255.255.0을 나타냅니다. 여기에는 24개의 선행 1비트가 있습니다. IPv4 블록 192.168.100.0/22는 192.168.100.0에서 192.168.103.255까지의 IP 주소 1024개를 나타냅니다.

■ IPv6 CIDR 및 IPv6 기본 게이트웨이

Blueprint의 vSphere 클라우드 계정 및 vSphere 시스템 구성 요소는 이중 IPv6 및 IPv4 인터넷 프로토콜 메서드를 지원합니다. 예를 들어 2001:db8::/48은 2001:db8:0:0:0:0:0:0에서 2001:db8:0:ffff:ffff:ffff:ffff:ffff까지의 IPv6 주소 블록을 나타냅니다.

IPv6 형식은 주문형 네트워크에 대해 지원되지 않습니다.

■ DNS 서버 및 DNS 검색 도메인

■ 공용 IP 지원

네트워크에 공용으로 플래그 지정하려면 이 옵션을 선택합니다. `network type: public` 속성이 있는 Blueprint의 네트워크 구성 요소가 공용으로 플래그가 지정된 네트워크와 매칭됩니다. 네트워크 선택을 결정하기 위해 Blueprint 배포 중에 추가 매칭이 발생합니다.

■ 영역에 대한 기본값

네트워크에 클라우드 영역의 기본 네트워크로 플래그를 지정하려면 이 옵션을 선택합니다. Blueprint 배포 중 다른 네트워크보다 기본 네트워크가 선호됩니다.

■ 원본

네트워크 소스를 식별합니다.

■ 태그

네트워크에 할당된 하나 이상의 태그를 지정합니다. 태그는 선택 사항입니다. 태그 일치는 Blueprint 배포에 사용할 수 있는 네트워크에 영향을 미칩니다.

네트워크 태그는 네트워크 프로파일과 관계없이 네트워크 항목 자체에 존재합니다. 네트워크 태그는 태그가 추가된 네트워크의 모든 발생 및 해당 네트워크를 포함하는 모든 네트워크 프로파일에 적용됩니다. 네트워크는 임의의 수의 네트워크 프로파일에 인스턴스될 수 있습니다. 네트워크 프로파일 상주 여부와 관계없이, 네트워크 태그는 네트워크가 사용되는 모든 위치에서 네트워크와 연결됩니다.

Blueprint를 배포할 때 Blueprint의 네트워크 구성 요소에 있는 제약 조건 태그가 네트워크 프로파일 기능 태그를 포함한 네트워크 태그에 매칭됩니다. 기능 태그를 포함하는 네트워크 프로파일의 경우 기능 태그가 해당 네트워크 프로파일이 사용할 수 있는 모든 네트워크에 적용됩니다. 매칭된 네트워크 프로파일에 정의된 네트워크 및 보안 설정은 Blueprint가 배포될 때 적용됩니다.

네트워크 정책

연결된 클라우드 계정에 따라 네트워크 정책을 사용하여 outbound, private 및 routed 네트워크 유형 및 주문형 보안 그룹의 설정을 정의할 수 있습니다. 또한 existing 네트워크와 연결된 로드 밸런서가 있을 때 네트워크 정책을 사용하여 해당 네트워크를 제어할 수도 있습니다.

네트워크 유형에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 네트워크 및 네트워크 프로파일 사용](#) 항목을 참조하십시오.

다음과 같은 주문형 선택 사항에 대한 옵션은 **네트워크 프로파일** 화면 도움말에 설명되어 있고 아래에 요약되어 있습니다.

■ 주문형 네트워크 또는 주문형 보안 그룹 생성 안 함

이 옵션은 existing 또는 public 네트워크 유형을 지정할 때 사용할 수 있습니다. outbound, private 또는 routed 네트워크가 필요한 Blueprint는 이 프로파일과 일치하지 않습니다.

■ 주문형 네트워크 생성

이 옵션은 outbound, private 또는 routed 네트워크 유형을 지정할 때 사용할 수 있습니다.

Amazon Web Services, Microsoft Azure, NSX, vSphere 및 VMware Cloud on AWS는 이 옵션을 지원합니다.

■ 주문형 보안 그룹 생성

이 옵션은 outbound 또는 private 네트워크 유형을 지정할 때 사용할 수 있습니다.

네트워크 유형이 outbound 또는 private인 경우 일치하는 Blueprint에 대해 새로운 보안 그룹이 생성됩니다.

Amazon Web Services, Microsoft Azure, NSX 및 VMware Cloud on AWS는 이 옵션을 지원합니다.

네트워크 정책 설정은 클라우드 계정 유형에 따라 다를 수 있습니다. 이러한 설정은 화면 포지판 도움말에 설명되어 있고 아래에 요약되어 있습니다.

■ 네트워크 도메인 또는 전송 영역

네트워크 도메인이나 전송 영역은 vSphere vNetwork Distributed 포트 그룹(dvPortGroup)에 대한 분산 가상 스위치(dvSwitch)입니다. **전송 영역**은 *dvSwitch* 또는 *dvPortGroup*과 같은 용어와 유사한 기존 NSX 개념입니다.

NSX 클라우드 계정을 사용하는 경우 페이지의 요소 이름은 **전송 영역**이고, 그렇지 않으면 **네트워크 도메인**입니다.

표준 스위치의 경우 네트워크 도메인이나 전송 영역은 스위치 자체와 동일합니다. 네트워크 도메인이나 전송 영역은 vCenter 내 서브넷의 경계를 정의합니다.

전송 영역은 NSX 논리적 스위치가 도달할 수 있는 호스트를 제어합니다. 하나 이상의 vSphere 클러스터에 걸쳐있을 수 있습니다. 전송 영역에서는 특정 네트워크 사용에 참여할 수 있는 클러스터 및 가상 시스템을 제어합니다. 동일한 NSX 전송 영역에 속하는 서브넷은 동일한 시스템 호스트에 사용할 수 있습니다.

■ 외부 서브넷

아웃바운드 액세스 권한이 있는 주문형 네트워크에는 아웃바운드 액세스 권한이 있는 외부 서브넷이 필요합니다. 외부 서브넷은 Blueprint에서 요청된 경우 아웃바운드 액세스를 제공하는 데 사용되며 네트워크 배치를 제어하지 않습니다. 예를 들어 외부 서브넷은 전용 네트워크를 배치하는 데 영향을 주지 않습니다.

■ CIDR

CIDR 표기법은 IP 주소 및 연결된 라우팅 접두사를 간단하게 표현한 것입니다. CIDR 값은 프로비저닝 중에 서브넷을 생성하는 데 사용할 네트워크 주소 범위를 지정합니다. **네트워크 정책** 탭의 이 CIDR 설정은 /nn으로 끝나고 0 - 32 사이의 값을 포함하는 IPv4 표기법을 수락합니다.

■ 서브넷 크기

이 옵션은 이 네트워크 프로파일을 사용하는 배포에서 격리된 각 네트워크에 대해 생성하도록 IPv4 표기법을 사용하여 주문형 네트워크의 크기를 지정합니다. 서브넷 크기 설정은 내부 또는 외부 IP 주소 관리에 사용할 수 있습니다.

IPv6 형식은 주문형 네트워크에 대해 지원되지 않습니다.

■ 논리적 분산 라우터

NSX-V 클라우드 계정을 사용하는 경우 주문형 라우팅된 네트워크에 대해 논리적 분산 라우터를 지정해야 합니다.

DLR(논리적 분산 라우터)은 NSX-V의 주문형 라우팅된 네트워크 간에 East/West 트래픽을 라우팅하는 데 사용됩니다. 이 옵션은 네트워크 프로파일의 계정/지역 값이 NSX-V 클라우드 계정에 연결된 경우에만 표시됩니다.

■ IP 범위 할당

이 옵션은 vSphere를 포함하여 NSX 또는 VMware Cloud on AWS를 지원하는 클라우드 계정에 사용할 수 있습니다.

다음 세 옵션 중 하나를 선택하여 배포 네트워크에 대한 IP 범위 할당 유형을 지정할 수 있습니다.

■ 정적 및 DHCP

기본값이고 권장됩니다. 이 혼합 옵션은 할당된 **CIDR 및 서브넷 범위** 설정을 사용하여, DHCP 서버 풀이 DHCP(동적) 메서드를 사용하여 주소 공간 할당의 절반을 지원하고 정적 메서드를 사용하여 IP 주소 공간 할당의 절반을 지원하도록 구성합니다. 주문형 네트워크에 연결된 일부 시스템에는 할당된 고정 IP 주소가 필요하고 일부는 동적 IP 주소가 필요한 경우 이 옵션을 사용합니다. 두 개의 IP 범위가 생성됩니다.

이 옵션은 주문형 네트워크에 연결된 시스템을 사용하는 배포(일부 시스템에는 고정 IP가 할당되고 다른 시스템에는 NSX DHCP 서버에서 동적으로 할당한 IP가 있는 배포) 및 로드 밸런서 VIP가 정적인 배포에 가장 효과적입니다.

■ DHCP(동적)

이 옵션은 할당된 CIDR을 사용하여 DHCP 서버에서 IP 풀을 구성합니다. 이 네트워크의 모든 IP 주소는 동적으로 할당됩니다. 할당된 각 CIDR에 대해 단일 IP 범위가 생성됩니다.

■ 정적

이 옵션은 할당된 CIDR을 사용하여 IP 주소를 정적으로 할당합니다. 이 네트워크에 대해 DHCP 서버를 구성할 필요가 없는 경우 이 옵션을 사용합니다. 할당된 각 CIDR에 대해 단일 IP 범위가 생성됩니다.

■ 네트워크 리소스 - 외부 네트워크

외부 네트워크를 기존 네트워크라고도 합니다. 이러한 네트워크는 데이터가 수집되며 선택이 가능하도록 설정됩니다.

■ 네트워크 리소스 - Tier-0 논리적 라우터

NSX-T는 NSX 배포 외부의 네트워크에 대한 게이트웨이로 Tier-0 논리적 라우터를 사용합니다. Tier-0 논리적 라우터는 주문형 네트워크에 대한 아웃바운드 액세스를 구성합니다.

■ 네트워크 리소스 - Edge 클러스터

지정된 Edge 클러스터는 라우팅 서비스를 제공합니다. Edge 클러스터는 주문형 네트워크 및 로드 밸런서에 대한 아웃바운드 액세스를 구성하는 데 사용됩니다. Edge 장치를 배포할 리소스 풀 또는 Edge 클러스터를 식별합니다.

■ 네트워크 리소스 - Edge 데이터스토어

지정된 Edge 데이터스토어는 Edge 장치를 프로비저닝하는 데 사용됩니다. 이 설정은 NSX-V에만 적용됩니다.

로드 밸런서

네트워크 프로파일에 로드 밸런서를 추가할 수 있습니다. 나열된 로드 밸런서는 소스 클라우드 계정에서 데이터를 수집한 정보를 기반으로 사용할 수 있습니다.

네트워크 프로파일의 로드 밸런서에 있는 태그가 **Blueprint**의 로드 밸런서 구성 요소에 사용되는 태그와 일치하는 경우 로드 밸런서가 배포 중에 고려됩니다. 일치하는 네트워크 프로파일의 로드 밸런서가 **Blueprint**를 배포할 때 사용됩니다.

자세한 내용은 **vRealize Automation Cloud Assembly**에서 **네트워크 프로파일** 및 **Blueprint** 설계의 로드 밸런서 설정 사용 및 **vRealize Automation Cloud Assembly Blueprint**의 네트워크, 보안 및 로드 밸런서 예 항목을 참조하십시오.

보안 그룹

Blueprint를 배포할 때 해당 네트워크 프로파일의 보안 그룹이 프로비저닝된 시스템 NIC에 적용됩니다. **Amazon Web Services** 특정 네트워크 프로파일의 경우 네트워크 프로파일의 보안 그룹은 [네트워크] 탭에 나열된 네트워크와 동일한 네트워크 도메인(VPC)에서 사용할 수 있습니다. 네트워크 프로파일의 [네트워크] 탭에 네트워크가 나열되어 있지 않으면 사용 가능한 모든 보안 그룹이 표시됩니다.

보안 그룹을 사용하여 주문형 private 또는 outbound 네트워크에 대한 격리 설정을 추가로 정의할 수 있습니다. 보안 그룹은 existing 네트워크에도 적용됩니다.

보안 그룹은 네트워크 프로파일과 일치하는 네트워크에 연결된 배포의 모든 시스템에 적용됩니다.

Blueprint에 여러 네트워크가 있을 수 있고 각각 서로 다른 네트워크 프로파일과 일치하므로 네트워크별로 서로 다른 보안 그룹을 사용할 수 있습니다.

기존 보안 그룹에 태그를 추가하면 **Blueprint Cloud.SecurityGroup** 구성 요소에서 보안 그룹을 사용할 수 있습니다. 보안 그룹에는 하나 이상의 태그가 있어야 합니다. 그렇지 않으면 **Blueprint**에서 사용할 수 없습니다. 자세한 내용은 **보안 리소스** 및 **vRealize Automation Cloud Assembly Blueprint**의 네트워크, 보안 및 로드 밸런서 예 항목을 참조하십시오.

네트워크 프로파일, 네트워크, Blueprint 및 태그에 대한 추가 정보

네트워크 프로파일에 대한 자세한 내용은 도움말의 이 섹션에 있는 다른 항목과 **WordPress** 사용 사례: **네트워크 프로파일 추가** 항목을 참조하십시오.

네트워크에 대한 추가 정보는 **네트워크 리소스** 항목을 참조하십시오.

Blueprint의 샘플 네트워크 구성 요소 코드 예는 **vRealize Automation Cloud Assembly Blueprint**의 네트워크, 보안 및 로드 밸런서 예의 내용을 참조하십시오.

샘플 네트워크 자동화 워크플로는 **Cloud Assembly** 및 **NSX**를 사용하여 네트워크 자동화를 참조하십시오.

태그 및 태그 전략에 대한 추가 정보는 태그를 사용하여 **vRealize Automation Cloud Assembly** 리소스 및 배포를 관리하는 방법 항목을 참조하십시오.

vRealize Automation Cloud Assembly의 네트워크 및 네트워크 프로파일에서 IP 주소 사용

네트워크 및 네트워크 프로파일 설정을 사용하면 vRealize Automation Cloud Assembly Blueprint 및 배포에서 네트워크 IP 주소를 사용하는 방법을 제어할 수 있습니다.

네트워크 프로파일을 사용하여 정적, DHCP 또는 정적 및 DHCP IP 주소 설정이 혼합된 기존 네트워크 도메인의 서브넷을 정의할 수 있습니다.

네트워크 정책 탭을 사용하여 서브넷을 정의하고 IP 주소 설정을 지정할 수 있습니다. 자세한 내용은 [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

vRealize Automation Cloud Assembly 네트워크의 IPv4 및 IPv6 지원

vRealize Automation Cloud Assembly 네트워크는 순수 IPv4 또는 이중 스택 IPv4 및 IPv6을 지원합니다. 순수 IPv6은 현재 지원되지 않습니다.

모든 클라우드 계정 및 통합 유형에 대해 순수 IPv4가 지원되지만 이중 스택 IPv4 및 IPv6은 vSphere 클라우드 계정과 해당 끝점에 대해서만 지원됩니다.

IPv6은 현재 로드 밸런서, NSX 주문형 네트워크 또는 외부 타사 IPAM 제공자에 사용할 수 없습니다.

외부 IPAM 제공자 지원

제공된 내부 IPAM 지원 외에도 외부 IPAM 제공자를 사용하여 Blueprint 및 배포에서 네트워크에 대한 IP 주소를 동적 또는 정적으로 할당할 수 있습니다.

Infoblox와 같은 외부 IPAM 제공자에 대한 지원은 벤더별 IPAM 통합 지점에서 가능하며, 이러한 통합 지점은 **인프라 > 연결 > 통합 추가 > IPAM** 메뉴 시퀀스를 사용하여 생성할 수 있습니다.

타사 IPAM 제공자 주소 정보를 정의하는 옵션은 **네트워크 정책 > IPAM IP 범위 추가** 페이지에서 **IPAM IP 범위 추가** 옵션을 통해 사용할 수 있습니다.

외부 IPAM 통합 지점 생성 방법에 대한 자세한 내용은 [vRealize Automation에서 외부 IPAM 통합 지점 구성](#) 항목을 참조하십시오. 특정 IPAM 벤더에 대한 IPAM 통합 지점을 생성하는 방법에 대한 예는 [제공자별 외부 IPAM 통합 사용 사례](#) 항목을 참조하십시오.

vRealize Automation Cloud Assembly에서 네트워크 및 네트워크 프로파일 사용

vRealize Automation Cloud Assembly에서 네트워크 및 네트워크 프로파일을 사용하면 배포에 대한 네트워크 프로비저닝 동작을 정의하는 데 도움이 됩니다.

vRealize Automation Cloud Assembly에서는 클라우드별 네트워크 프로파일을 정의할 수 있습니다. [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#)의 내용을 참조하십시오.

네트워크 유형

Blueprint의 네트워크 구성 요소는 다음 networkType 유형 중 하나로 정의됩니다.

네트워크 유형	정의
existing	기본 클라우드 제공자(예: vCenter, Amazon Web Services 및 Microsoft Azure)에 구성된 기존 네트워크를 선택합니다. 기존 네트워크는 outbound 주문형 네트워크에 필요합니다. 기존 네트워크에서 정적 IP 주소의 범위를 정의할 수 있습니다.
public	공용 네트워크의 시스템은 인터넷에서 액세스할 수 있습니다. IT 관리자는 이러한 네트워크를 정의합니다. public 네트워크에 대한 정의는 공용 네트워크를 따라 네트워크 트래픽이 발생하도록 허용하는 네트워크에 대한 existing 네트워크의 정의와 동일합니다.
private	주문형 네트워크 유형입니다. 배포된 네트워크의 리소스 간에만 네트워크 트래픽이 발생하도록 제한합니다. 인바운드 및 아웃바운드 트래픽을 방지합니다. NSX에서 주문형 NAT 일대다와 같을 수 있습니다.
outbound	주문형 네트워크 유형입니다. 배포의 계산 리소스 간에만 네트워크 트래픽이 발생하도록 제한하되, 단방향 아웃바운드 네트워크 트래픽도 허용합니다. NSX에서 외부 IP를 사용한 주문형 NAT와 같을 수 있습니다.
routed	주문형 네트워크 유형입니다. 라우팅된 네트워크에는 서로 연결된 사용 가능한 서브넷에 걸쳐 나누어진 라우팅 가능 IP 공간이 포함됩니다. 라우팅된 네트워크 프로파일이 동일한 라우팅된 네트워크로 프로비저닝된 가상 시스템은 서로 통신할 수 있으며 기존 네트워크와도 통신할 수 있습니다. 라우팅된 네트워크는 NSX-V 및 NSX-T 네트워크에서 사용할 수 있는 주문형 네트워크 유형입니다. 이 연결은 기본적으로 Microsoft Azure 및 Amazon Web Services에서 제공됩니다. routed 네트워크는 Cloud.NSX.Network 네트워크 구성 요소의 Blueprint 규칙에서만 사용할 수 있습니다.

네트워크 구성 요소 데이터가 포함된 채워진 Blueprint의 예는 [vRealize Automation Cloud Assembly Blueprint의 네트워크, 보안 및 로드 밸런서 예 항목](#)을 참조하십시오.

네트워킹 시나리오

다음과 같은 네트워크 프로파일 구성을 사용하는 Blueprint를 배포할 경우 다음과 같은 동작을 예상할 수 있습니다.

표 4-1. 네트워킹 시나리오

네트워크 유형 또는 시나리오	클라우드 영역에 대해 네트워크 프로파일을 사용할 수 없음	클라우드 영역에 대해 네트워크 프로파일을 사용할 수 있음
네트워크 없음	<p>Blueprint에 네트워크가 지정되지 않은 경우, 계산과 동일한 프로비저닝 영역에서 임의의 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>사용 가능한 프로비저닝 영역에 네트워크가 없으면 프로비저닝이 실패합니다.</p>	<p>일치하는 네트워크 프로파일에서 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>조건을 충족하는 네트워크 프로파일이 없으면 프로비저닝이 실패합니다.</p>
기존 네트워크	<p>Blueprint의 네트워크 구성 요소에 제약 조건 태그가 포함된 경우, 해당 제약 조건은 사용 가능한 네트워크 목록을 필터링하는 데 사용됩니다. Blueprint의 네트워크 구성 요소에 포함된 제약 조건 태그는 네트워크 태그 및 네트워크 프로파일 제약 조건 태그(사용 가능한 경우)와 일치여부가 확인됩니다.</p> <p>필터링된 네트워크 목록에서 계산과 동일한 프로비저닝 영역에 있는 단일 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>제약 조건에 기반하여 필터링한 이후 프로비저닝 영역에 네트워크가 없으면 프로비저닝이 실패합니다.</p>	<p>일치하는 네트워크 프로파일에서 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>조건을 충족하는 네트워크 프로파일이 없으면 프로비저닝이 실패합니다.</p> <p>네트워크 제약 조건을 사용하여, 미리 할당된 태그를 기준으로 프로파일에 있는 기존 네트워크를 필터링할 수 있습니다.</p>
공용 네트워크	<p>네트워크에 제약 조건이 있는 경우, supports public IP 특성이 설정된 사용 가능한 네트워크 목록을 필터링하는 데 해당 제약 조건이 사용됩니다.</p> <p>필터링된 네트워크 목록에서 계산과 동일한 프로비저닝 영역에 있는 임의의 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>제약 조건에 기반하여 필터링한 이후 프로비저닝 영역에 공용 네트워크가 없으면 프로비저닝이 실패합니다.</p>	<p>일치하는 네트워크 프로파일에서 supports public IP 특성을 가진 네트워크가 선택됩니다.</p> <p>기본값으로 레이블이 지정된 네트워크가 기본으로 사용됩니다.</p> <p>네트워크 제약 조건을 사용하여, 미리 할당된 태그를 기준으로 프로파일에 있는 기존 공용 네트워크를 필터링할 수 있습니다.</p>
전용 네트워크	<p>전용 네트워크에 네트워크 프로파일의 정보가 필요하기 때문에 프로비저닝이 실패합니다.</p>	<p>새 네트워크 또는 새 보안 그룹은 일치하는 네트워크 프로파일의 설정을 기반으로 생성됩니다.</p> <p>네트워크 제약 조건을 사용하여 네트워크 프로파일 및 네트워크를 필터링할 수 있습니다.</p>

표 4-1. 네트워킹 시나리오 (계속)

네트워크 유형 또는 시나리오	클라우드 영역에 대해 네트워크 프로파일을 사용할 수 없음	클라우드 영역에 대해 네트워크 프로파일을 사용할 수 있음
아웃바운드 네트워크	아웃바운드 네트워크에 네트워크 프로파일의 정보가 필요하기 때문에 프로비저닝이 실패합니다.	새 네트워크 또는 새 보안 그룹은 일치하는 네트워크 프로파일의 설정을 기반으로 생성됩니다. 네트워크 제약 조건을 사용하여 네트워크 프로파일 및 네트워크를 필터링할 수 있습니다.
주문형 라우팅된 네트워크	라우팅된 네트워크에 네트워크 프로파일의 정보가 필요하기 때문에 프로비저닝이 실패합니다.	NSX-V의 경우 DLR(논리적 분산 라우터)를 선택해야 합니다. NSX-T 및 VMware Cloud on AWS의 경우 전용 및 아웃바운드와 유사한 주문형 설정이 필요합니다.
기존 네트워크 또는 공용 네트워크를 사용하는 Wordpress 사용 사례 예시	기존 네트워크 또는 공용 네트워크에 대한 설명 대로 프로비저닝이 수행됩니다.	기존 네트워크 및 공용 네트워크 동작에 대한 위의 설명을 참조하십시오. Wordpress 사용 사례 항목을 참조하십시오.
기존 또는 공용 네트워크 및 전용 또는 아웃바운드 네트워크를 사용하는 Wordpress 사용 사례 예시	네트워크에 네트워크 프로파일의 정보가 필요하기 때문에 프로비저닝이 실패합니다.	전용 네트워크 및 아웃바운드 네트워크에 대한 위의 설명을 참조하십시오. Wordpress 사용 사례 항목을 참조하십시오.
로드 밸런서를 사용하는 Wordpress 사용 사례 예시	로드 밸런서에 네트워크 프로파일의 정보가 필요하기 때문에 프로비저닝이 실패합니다. 기존 로드 밸런서가 있으면 프로비저닝을 수행할 수 있습니다.	네트워크 프로파일 구성에 기반하여 새 로드 밸런서가 생성됩니다. 네트워크 프로파일에 사용하도록 설정된 기존 로드 밸런서를 지정할 수 있습니다. 기존 로드 밸런서를 요청했지만 네트워크 프로파일의 제약 조건을 충족하는 로드 밸런서가 없으면 프로비저닝이 실패합니다. Wordpress 사용 사례 항목을 참조하십시오.

vRealize Automation Cloud Assembly에서 네트워크 프로파일 및 Blueprint 설계의 로드 밸런서 설정 사용

NSX-V 또는 NSX-T 네트워크 프로파일 구성을 기반으로 vRealize Automation Cloud Assembly Blueprint에서 로드 밸런서를 정의할 수 있습니다.

로드 밸런서 옵션은 Blueprint의 네트워크 프로파일에 있는 설정 및 NSX-T 또는 NSX-V 소스 애플리케이션의 설정에 따라 달라집니다.

NSX-T 네트워크 및 로드 밸런서 옵션

NSX-T는 각 Tier-1 논리적 라우터에 대해 하나의 로드 밸런서 서비스를 지원합니다. 로드 밸런서 옵션은 Blueprint에서 로드 밸런서 구성 요소가 연결되어 있는 네트워크에 따라 다릅니다.

■ 주문형 아웃바운드 네트워크

로드 밸런서 계산이 주문형 아웃바운드 네트워크에 연결되어 있는 경우 주문형 네트워크의 Tier-1 라우터에 대해 로드 밸런서가 생성됩니다.

■ 주문형 전용 네트워크

주문형 네트워크 VIP는 소스 외부 네트워크의 VIP에 연결되어 있어야 합니다.

로드 밸런서 계산이 주문형 전용 네트워크에 연결되어 있는 경우 새로운 Tier-1 라우터가 생성되고 네트워크 프로파일에 지정된 Tier-0 라우터에 연결됩니다. 이후 로드 밸런서는 Tier-1 라우터에 연결됩니다. VIP가 외부 네트워크에 있는 경우 Tier-1 라우터 VIP 알림이 사용되도록 설정됩니다.

■ 기존 네트워크

로드 밸런서가 기존 네트워크에 연결되어 있는 경우 로드 밸런서는 기존 네트워크의 Tier-1 라우터와 함께 생성됩니다. Tier-1 라우터에 연결된 로드 밸런서 서비스가 없는 경우에만 새로운 소형 로드 밸런서 서비스가 생성됩니다. 로드 밸런서 서비스가 이미 있다면 새로운 가상 서버가 여기에 연결됩니다. 기존 네트워크가 Tier-1 라우터에 연결되어 있지 않은 경우 새로운 Tier-1 라우터가 생성되어 네트워크 프로파일에 정의된 Tier-0 라우터에 연결되고 Tier-1 라우터 VIP 알림은 사용되도록 설정되지 않습니다.

■ 새 보안 그룹이 있는 주문형 네트워크

시스템이 기존 네트워크에 연결되고 보안 그룹을 사용하여 격리가 생성되므로 이 옵션은 기존 네트워크에 생성된 로드 밸런서와 유사합니다. 차이점은 데이터 경로를 사용하도록 설정하기 위해 Tier-1 업링크 포트 IP가 격리 보안 그룹에 추가된다는 점입니다.

주문형 네트워크를 위해 생성된 로드 밸런서 서비스는 더 이상 사용되지 않으면 제거됩니다. 로드 밸런서가 제거되면 모니터, 풀, 애플리케이션 프로파일 및 VIP가 제거됩니다. 외부 네트워크에 대해 생성된 로드 밸런서 서비스는 제거되지 않습니다.

Blueprint 구성 요소 패널에서 사용할 수 있는 클라우드 애그노스틱 로드 밸런서 구성 요소를 사용하여 Blueprint에서 NSX-T 로드 밸런서 설정을 지정할 수 있습니다.

NSX-T 네트워크 배포에서의 로드 밸런서 사용에 대해 자세히 알아보려면 VMware 블로그 게시물 [NSX-T를 사용하는 vRA Cloud Assembly 로드 밸런서 심화 연구](#)를 참조하십시오.

NSX-V 네트워크 및 로드 밸런서 옵션

새 네트워크 서브넷과 보안 그룹 중 무엇을 사용하여 격리된 네트워크를 생성했는지에 따라 로드 밸런서를 투암 또는 원암으로 구성할 수 있습니다.

Blueprint 구성 요소 패널의 NSX 섹션에서 사용할 수 있는 로드 밸런서 구성 요소를 사용하여 Blueprint에서 NSX-V 로드 밸런서 설정을 지정할 수 있습니다.

투암 로드 밸런서 생성을 위한 워크플로는 다음과 같습니다.

- 1 서비스 Edge를 생성합니다.
- 2 서비스 Edge의 업링크 인터페이스를 공용 네트워크에 연결합니다.
- 3 다운링크 인터페이스를 격리된(아웃바운드) 네트워크에 연결합니다.
- 4 네트워크 프로파일 정적 IP 범위에서 로드 밸런서에 대한 정적 IP 주소를 할당합니다.
- 5 로드 밸런서를 구성합니다.
- 6 방화벽을 구성합니다.
- 7 기본 게이트웨이를 구성합니다.

원암 로드 밸런서 생성을 위한 워크플로는 다음과 같습니다.

- 1 서비스 Edge를 생성합니다.
- 2 예약에서 선택된 네트워크를 업링크로 연결합니다.
- 3 네트워크 프로파일 정적 IP 범위에서 로드 밸런서에 대한 정적 IP 주소를 할당합니다.
- 4 로드 밸런서를 구성합니다.
- 5 방화벽을 구성합니다.
- 6 기본 게이트웨이를 구성합니다.

다른 요구 사항을 설명하는 vRealize Automation Cloud Assembly 스토리지 프로파일을 추가하는 방법

vRealize Automation Cloud Assembly 스토리지 프로파일은 배포할 스토리지의 종류를 설명합니다.

스토리지는 일반적으로 서비스 수준 또는 비용, 성능 또는 용도(예: 백업)와 같은 특징에 따라 프로파일링됩니다.

인프라 > 구성 > 스토리지 프로파일을 선택하고 **새 스토리지 프로파일**을 클릭합니다.

vRealize Automation Cloud Assembly 스토리지 프로파일에 대해 알아보기

클라우드 계정 지역에는 클라우드 관리자가 해당 지역의 스토리지를 정의할 수 있는 스토리지 프로파일이 포함되어 있습니다.

스토리지 프로파일에는 디스크 사용자 지정 정보 및 기능 태그를 통해 스토리지 유형을 식별할 수 있는 방법이 포함됩니다. 이러한 태그는 배포 시 원하는 스토리지를 생성하기 위해 프로비저닝 서비스 요청 조건과 일치됩니다.

스토리지 프로파일은 클라우드별 지역에 구성됩니다. 클라우드 계정 하나에 여러 지역이 포함될 수 있으며, 각 지역에 스토리지 프로파일이 여러 개 포함될 수 있습니다.

벤더에 관계없이 스토리지 프로파일을 배치할 수도 있습니다. 예를 들어 서로 다른 벤더 계정 3개가 있고 각 계정에 지역이 1개 있다고 가정해 보겠습니다. 각 지역에는 "fast" 라는 기능 태그가 지정된 스토리지 프로파일이 포함되어 있습니다. 프로비저닝 시 하드 "fast" 조건 태그가 포함된 요청은 어느 벤더 클라우드가 리소스를 제공하는지에 관계 없이, 일치하는 "fast" 기능을 찾습니다. 이 일치 항목은 배포된 스토리지 항목을 생성하는 동안 연결된 스토리지 프로파일의 설정을 적용합니다.

참고 클라우드 스토리지마다 성능 특성이 다를 수 있지만 그래도 태그를 지정한 관리자가 제공하는 "fast" 오퍼링으로 간주됩니다.

스토리지 프로파일에 추가하는 기능 태그는 실제 리소스 대상을 식별하는 게 아니라 스토리지 유형을 설명합니다. 실제 리소스에 대한 자세한 내용은 [스토리지 리소스](#) 항목을 참조하십시오.

태그를 사용하여 vRealize Automation Cloud Assembly 리소스 및 배포를 관리하는 방법

태그는 기능 및 제약 조건의 일치를 통해 배포의 배치를 적용하는 vRealize Automation Cloud Assembly의 중요 구성 요소입니다. vRealize Automation Cloud Assembly를 최적화된 방식으로 사용하려면 태그를 이해하고 효과적으로 구현해야 합니다.

기본적으로 태그는 vRealize Automation Cloud Assembly 항목에 추가하는 레이블입니다. 조직과 구현 내용에 적절한 모든 태그를 생성할 수 있습니다. 태그는 vRealize Automation Cloud Assembly가 리소스와 인프라를 사용하여 배포 가능한 서비스를 구축하는 방식과 위치를 제어하므로 레이블 이상의 기능을 합니다. 또한 태그는 Cloud Assembly 내의 거버넌스를 지원합니다.

태그 구조

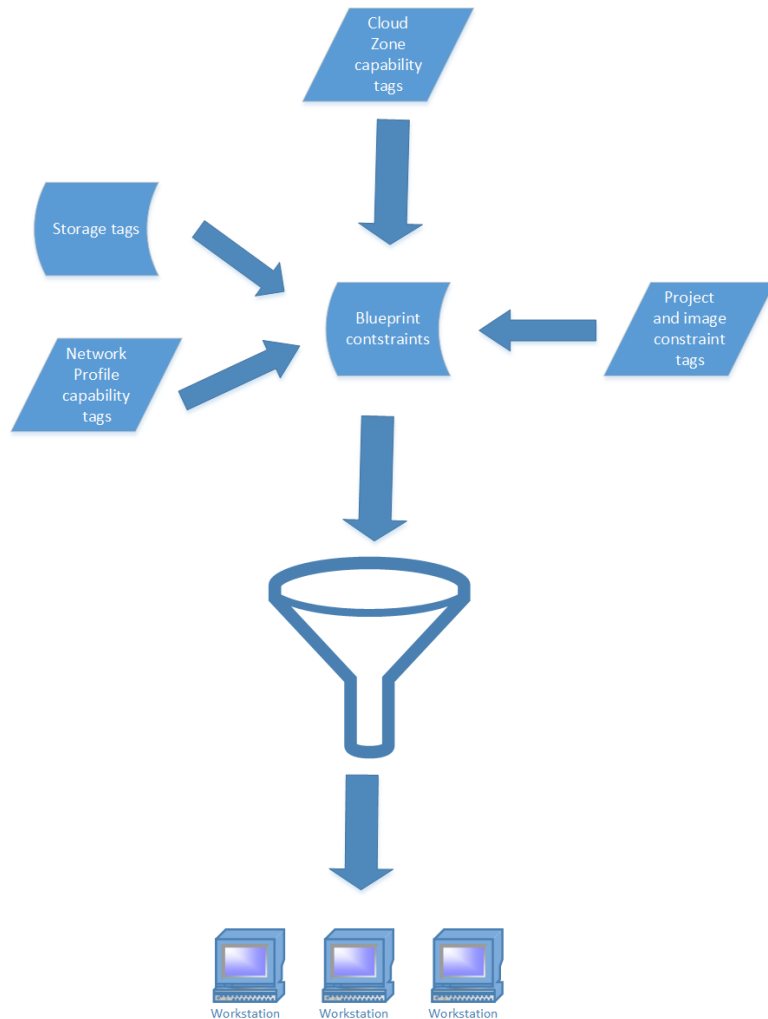
구조적으로 태그는 name:value 쌍 규칙을 따라야 하지만 그렇지 않은 경우 태그의 구조는 대부분 자유 형식입니다. vRealize Automation Cloud Assembly 전체에서 모든 태그는 동일하게 나타나고 태그 기능은 컨텍스트에 의해 결정됩니다.

예를 들어 인프라 리소스의 태그는 주로 기능 태그로 작동합니다. 그 이유는 vRealize Automation Cloud Assembly가 이러한 태그를 사용하여 리소스와 배포를 매칭하기 때문입니다. 부수적으로는 리소스도 식별합니다.

태그 기능

vRealize Automation Cloud Assembly 내에서 태그의 기본 기능은 기능 및 제약 조건을 사용하여 배포를 구성하는 것입니다. 클라우드 영역, 네트워크 및 스토리지 프로파일, 개별 인프라 리소스에 배치된 기능 태그는 배포에 필요한 기능을 정의합니다. 클라우드 관리자가 프로젝트에 배치하는 제약 조건 태그는 해당 프로젝트에 대한 거버넌스의 한 형태를 이행하는 데 사용됩니다. 이러한 제약 조건 태그는 Blueprint에 표시된 다른 제약 조건에 추가됩니다.

프로비저닝 동안, vRealize Automation Cloud Assembly는 이러한 기능을 Blueprint에서 태그로 표시된 제약 조건과 일치시켜 배포 구성을 정의합니다. 이 태그 기반 기능 및 제약 조건 기능은 vRealize Automation Cloud Assembly에서 배포 구성을 위한 기초 역할을 합니다. 예를 들어 태그를 사용하여 특정 지역의 PCI 리소스에서만 인프라를 사용할 수 있게 만들 수 있습니다.



또한 다른 차원에서 태그는 스토리지, 네트워크 항목 및 다른 인프라 리소스의 검색 및 식별을 용이하게 합니다.

예를 들어 클라우드 영역을 설정 중이고 여러 계산 리소스를 사용할 수 있다고 가정합니다. 이러한 계산 리소스에 대해 적절하게 태그를 지정했다면 [클라우드 영역] 페이지의 [계산] 탭에서 검색 기능을 사용하여 특정 클라우드 영역과 연결된 리소스를 필터링할 수 있습니다.

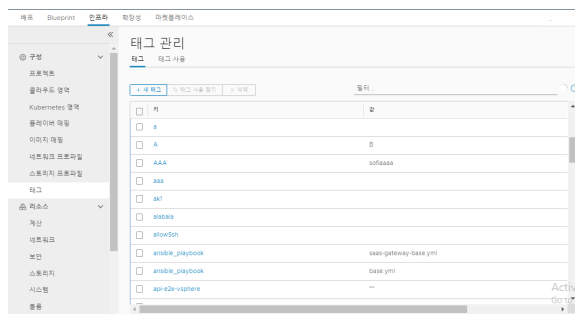
또한 [태그 관리] 페이지 및 리소스 구성 페이지에는 태그 이름으로 항목을 찾을 수 있는 검색 기능이 포함되어 있습니다. 이와 같은 검색 및 식별 기능을 용이하게 하려면 이러한 항목에 대해 논리적이고 사람이 읽을 수 있는 태그를 사용하는 것이 매우 중요합니다.

외부 태그

vRealize Automation Cloud Assembly는 외부 태그를 포함할 수도 있습니다. 이러한 태그는 vRealize Automation Cloud Assembly 인스턴스와 연결한 클라우드 계정에서 자동으로 가져옵니다. 이러한 태그를 vSphere, AWS, Azure 또는 기타 외부 소프트웨어 제품으로부터 가져올 수도 있습니다. 가져온 태그는 사용자가 생성한 태그와 동일한 방식으로 사용할 수 있습니다.

태그 관리

vRealize Automation Cloud Assembly에서 [태그 관리] 페이지를 사용하여 태그 라이브러리를 모니터링하고 관리할 수 있습니다. 이 페이지에서 태그를 생성할 수도 있습니다. 또한 [태그 관리] 페이지는 외부 태그를 보고 식별할 수 있는 유일한 페이지입니다.



태그 전략

혼란을 최소화하려면 vRealize Automation Cloud Assembly에서 태그를 생성하기 전에 적절한 태그 전략 및 태그 지정 규칙을 고안하여 태그를 생성하고 사용하는 모든 사용자가 태그의 의미와 태그가 사용되는 방식을 이해할 수 있도록 해야 합니다. **태그 지정 전략 생성** 항목을 참조하십시오.

태그 지정 전략 생성

Cloud Assembly 기능을 최대한 효과적으로 활용하고 잠재적인 혼란을 최소화하기 위해 조직의 IT 구조와 목표에 따라 적절한 태그 지정 전략을 신중하게 계획하고 구현해야 합니다.

태그 지정에는 몇 가지 공통된 목표가 있지만 조직의 요구 사항, 구조 및 목표에 맞추어 적절한 태그 지정 전략을 생성해야 합니다.

태그 지정 모범 사례

효과적인 태그 전략의 몇 가지 일반적인 특성은 다음과 같습니다.

- 조직의 비즈니스 구조와 연관지어 일관된 태그 지정 계획을 설계 및 구현하고 모든 관련 사용자에게 이 계획을 전달합니다. 계획은 배포 요구 사항을 지원하며 명확하고 가독성 있는 언어를 사용하고 모든 관련 사용자가 이해할 수 있어야 합니다.
- 간단명료하고 알기 쉬운 이름 및 값을 태그에 사용합니다. 예를 들어 스토리지 및 네트워크 항목의 태그 이름이 명확하고 일관되어야 사용자가 배포된 리소스를 선택하거나 배포된 리소스에 대한 태그 할당을 검토할 때 바로 파악할 수 있습니다.

- 값 없이 이름만 사용하여 태그를 생성할 수도 있지만 모범 사례로는 다른 사용자가 태그 용도를 명확히 파악할 수 있도록 각 태그 이름에 관련 값을 지정하는 것이 더 좋습니다.

태그 지정 구현

기본적인 태그 지정 전략을 위한 주요 고려 사항을 확인하십시오. 다음 목록은 전략을 계획할 때 고려해야 할 일반적인 고려 사항입니다. 이러한 고려 사항은 절대적인 기준이 아니라 대표적인 예입니다. 사용 사례와 관련된 다른 고려 사항이 있을 수 있으며, 구체적인 사용 사례에 맞는 전략을 계획해야 합니다.

- 배포할 환경 수. 일반적으로 각 환경을 나타내는 태그를 생성합니다.
- 배포를 지원하기 위해 계산 리소스를 구성하고 사용하는 방식
- 배포할 영역 또는 위치 수. 일반적으로 각 영역 또는 위치를 나타내는 프로파일 수준에서 태그를 생성합니다.
- 배포에 사용할 수 있는 스토리지 옵션 수 및 이러한 옵션을 구분하는 방법. 이러한 옵션은 태그로 나타내야 합니다.
- 네트워크 옵션을 범주화하고 모든 해당 옵션에 대한 태그를 생성합니다.
- 일반적인 배포 변수. 예를 들어 배포할 환경 수를 고려해야 합니다. 일반적으로 많은 조직이 최소한 테스트, 개발 및 운영 환경을 두고 있습니다. 하나 이상의 이러한 환경에 배포를 쉽게 구성할 수 있도록 **Blueprint** 제약 조건 태그와 클라우드 영역 기능 태그를 생성하고 조정하는 것이 좋습니다.
- 사용되는 네트워크 및 스토리지 프로파일의 컨텍스트에 맞게 네트워크 및 스토리지 리소스에 대한 태그를 조정합니다. 리소스 태그를 사용하면 리소스 배포를 더 세부적으로 제어할 수 있습니다.
- 클라우드 영역 및 네트워크 프로파일 기능 태그와 기타 기능 태그와 **Blueprint** 제약 조건 태그를 상호 조정합니다. 일반적으로 관리자가 먼저 클라우드 영역 및 네트워크 프로파일에 대한 기능 태그를 생성하면 다른 사용자가 이러한 기능 태그와 일치하는 제약 조건을 사용하여 **Blueprint**를 설계할 수 있습니다.

조직에 중요한 고려 사항을 파악한 후에 이러한 고려 사항을 해결하는 적절한 태그 이름을 체계적으로 계획할 수 있습니다. 그런 다음 전략의 개요를 작성하여 태그를 생성하고 편집할 수 있는 권한을 가진 모든 사용자에게 제공합니다.

유용한 구현 방식으로, 일단 모든 계산 인프라 리소스에 개별적으로 태그를 지정할 수 있습니다. 앞서 언급한 대로 특정 리소스와 연관된 논리적 범주를 태그 이름에 사용합니다. 예를 들어 스토리지 리소스에 대한 태그를 tier1, tier2 등으로 지정할 수 있습니다. 또한 계산 리소스에 대한 태그는 Windows, Linux 등과 같은 해당 운영 체제를 기반으로 지정할 수 있습니다.

리소스에 태그를 지정한 후에는 요구 사항에 가장 잘 맞는 방식으로 클라우드 영역과 스토리지 및 네트워크 프로파일에 대한 태그를 생성할 수 있습니다.

vRealize Automation Cloud Assembly에서 기능 태그 사용

vRealize Automation Cloud Assembly에서 기능 태그를 사용하면 인프라 구성 요소의 배포에 대한 배치 논리를 정의할 수 있습니다. 기능 태그는 이러한 배치를 하드 코딩하는 보다 강력하고 간결한 옵션입니다.

계산 리소스, 클라우드 영역, 이미지와 이미지 맵, 네트워크와 네트워크 프로파일에 대한 기능 태그를 생성할 수 있습니다. 이러한 리소스를 생성하기 위한 페이지에는 기능 태그 생성을 위한 옵션이 포함되어 있습니다. vRealize Automation Cloud Assembly의 [태그 관리] 페이지를 사용하여 기능 태그를 생성할 수도 있습니다. 클라우드 영역과 네트워크 프로파일의 기능 태그는 해당 영역 또는 프로파일 내의 모든 리소스에 영향을 줍니다. 스토리지 또는 네트워크 구성 요소의 기능 태그는 해당 태그가 적용되는 구성 요소에만 영향을 줍니다.

일반적으로 기능 태그는 계산 리소스의 위치, 네트워크의 어댑터 유형 또는 스토리지 리소스의 계층 수준과 같은 항목을 정의할 수 있습니다. 이 밖에 환경 위치 또는 유형과 기타 비즈니스 고려 사항도 정의할 수 있습니다. 전반적인 태그 지정 전략과 마찬가지로 기능 태그는 논리적 방식으로 구성해야 합니다.

vRealize Automation Cloud Assembly는 배포 시 클라우드 영역 및 Blueprint의 제약 조건과 기능 태그를 일치시킵니다. 따라서 기능 태그를 생성하고 사용할 때에는 일치가 예상대로 수행되도록 적절한 Blueprint 제약 조건의 생성을 이해하고 계획해야 합니다.

예를 들어 Wordpress 예시의 [클라우드 영역 추가] 항목에서 OurCo-AWS-US-East 영역과 OurCo-AWS-US-West 영역에 대한 개발 및 테스트 태그를 생성했습니다. 이것은 OurCo-AWS-US-East 영역이 개발 환경이고 OurCo-AWS-US-West 영역이 테스트 환경임을 나타냅니다. 이러한 기능 태그를 적절한 제약 조건 태그와 함께 쌍으로 구성하면 원하는 환경에 배포할 수 있습니다.

vRealize Automation Cloud Assembly에서 제약 조건 태그 사용

리소스, 클라우드 영역 및 프로파일에 정의된 기능을 일치시켜 적절한 배포를 생성하려면 vRealize Automation Cloud Assembly 내의 Blueprint 및 기타 다양한 구성 요소에 제약 조건 태그를 추가합니다.

vRealize Automation Cloud Assembly에는 제약 조건 태그를 적용할 수 있는 두 개의 주요 영역이 있습니다. 그 첫 번째는 프로젝트 및 이미지의 구성 측면이고, 두 번째는 Blueprint의 배포 측면입니다. 두 영역에 모두 적용되는 제약 조건은 Blueprint에 병합되어 일련의 배포 요구 사항을 구성합니다.

프로젝트에서 제약 조건 태그가 작동하는 방식

Cloud Assembly를 구성할 때 클라우드 관리자는 프로젝트와 이미지 맵에 제약 조건 태그를 적용할 수 있습니다. 이러한 방식으로 클라우드 관리자는 프로젝트 수준에서 거버넌스 제약 조건을 직접 적용할 수 있습니다. 이 수준에서 추가된 모든 제약 조건은 적용 가능한 프로젝트에 대해 요청된 모든 Blueprint에 적용됩니다.

프로젝트의 태그와 Blueprint의 태그가 충돌하는 경우에는 프로젝트 태그가 우선하므로 클라우드 관리자는 거버넌스 규칙을 적용할 수 있습니다. 예를 들어 클라우드 관리자가 프로젝트에서 location:london 태그를 생성했지만 개발자가 Blueprint에 location:boston 태그를 배치한 경우 전자가 후자에 우선하기 때문에 리소스는 location:london 태그를 포함하는 인프라에 배포됩니다.

프로젝트에 최대 3개의 제약 조건을 적용할 수 있습니다. 프로젝트 제약 조건은 경성 또는 연성입니다. 기본적으로 제약 조건은 경성입니다. 경성 제약 조건을 사용하면 배포 제한을 엄격하게 적용할 수 있습니다. 경성 제약 조건이 하나 이상 충족되지 않으면 배포가 실패합니다. 연성 제약 조건은 사용 가능한 경우에 선택되는 기본 설정을 나타내는 방식으로 제시되지만 연성 제약 조건이 충족되지 않아도 배포는 실패하지 않습니다.

Blueprint에서 제약 조건 태그가 작동하는 방식

Blueprint에서는 제약 조건 태그를 리소스에 YAML 코드로 추가하여 클라우드 관리자가 리소스, 클라우드 영역 및 스토리지, 네트워크 프로파일에서 생성한 적절한 기능 태그를 일치시킵니다. 또한 제약 조건 태그를 구현하기 위한 더 복잡한 다른 옵션이 있습니다. 예를 들어 요청 시 변수를 사용하여 하나 이상의 태그를 채울 수 있습니다. 이를 통해 요청 시간에 하나 이상의 태그를 지정할 수 있습니다.

Blueprint YAML 코드에서 tag 레이블을 사용하여 제약 조건 태그를 생성합니다. 프로젝트의 제약 조건 태그가 Blueprint에서 생성된 제약 조건 태그에 추가됩니다.

vRealize Automation Cloud Assembly는 YAML 파일에서 더 쉽게 제약 조건을 사용하도록 간단한 문자열 형식을 지원합니다.

```
[!tag_key[:tag_value][:hard|soft]
```

기본적으로 vRealize Automation Cloud Assembly는 경성 적용이 포함된 엄격한 제약 조건을 생성합니다. 태그 값은 애플리케이션의 나머지 부분에서와 마찬가지로 권장되지만 선택적입니다.

다음 "WordPress with MySQL" 예제는 계산 리소스에 대한 특정 위치 정보를 나타내는 YAML 제약 조건 태그를 보여 줍니다.

```
name: "wordpressWithMySQL"
components:
  mysql:
    type: "Compute"
    data:
      name: "mysql"
      # ... skipped lines ...
  wordpress:
    type: "Compute"
    data:
      name: "wordpress"
      instanceType: small
      imageType: "ubuntu-server-1604"
      constraints:
        - tag: "!location:eu:hard"
        - tag: "location:us:soft"
        - tag: "!pci"
      # ... skipped lines ...
```

Blueprint 사용 방법에 대한 자세한 내용은 [WordPress 사용 사례: Blueprint 생성 및 확장](#) 항목을 참조하십시오.

프로젝트와 Blueprint에서 경성 및 연성 제약 조건이 작동하는 방식

프로젝트와 Blueprint의 제약 조건은 모두 경성이거나 연성일 수 있습니다. 앞의 코드 조각은 경성 및 연성 제약 조건의 예를 보여 줍니다. 기본적으로 모든 제약 조건은 경성입니다. 경성 제약 조건을 사용하면 배포 제한을 엄격하게 적용할 수 있습니다. 경성 제약 조건이 하나 이상 충족되지 않으면 배포가 실패합니다. 연성 제약 조건은 사용 가능한 경우에 적용되는 기본 설정을 나타내지만 충족되지 않아도 요청이 실패하지 않습니다.

특정 리소스 유형에 대해 일련의 경성 및 연성 제약 조건이 있는 경우 연성 제약 조건이 '결정자' 역할을 할 수도 있습니다. 즉, 여러 리소스가 하나의 경성 제약 조건을 충족하는 경우 배포에 사용될 실제 리소스를 선택하는 데 연성 제약 조건이 사용됩니다.

예를 들어 네트워크, 스토리지 및 확장성 항목의 조합으로 프로젝트에 최대 3개의 제약 조건을 지정할 수 있습니다. 또한 각 제약 조건이 경성인지 아니면 연성인지 선택할 수 있습니다. `location:boston` 태그를 사용하여 경성인 스토리지 제약 조건을 생성했다고 가정합니다. 프로젝트에 이 제약 조건에 일치하는 스토리지가 없다면 관련된 모든 배포가 실패합니다.

참고 프로젝트 및 Blueprint에서 `failOnConstraintMergeConflict` 플래그는 제약 조건의 동작을 수정합니다. 이 플래그가 `true`로 설정된 경우 프로젝트 제약 조건과 Blueprint 제약 조건 간에 충돌이 있으면 요청이 실패합니다. 이 플래그가 없거나 플래그가 `false`로 설정되면 프로젝트 제약 조건이 Blueprint 제약 조건에 우선합니다.

표준 태그

vRealize Automation Cloud Assembly는 배포된 리소스에 대해 분석, 모니터링 및 그룹화를 지원하기 위해 일부 배포에 표준 태그를 적용합니다.

표준 태그는 vRealize Automation Cloud Assembly 내에서 고유합니다. 다른 태그와 달리 표준 태그는 사용자가 배포를 구성하는 동안 사용하지 않으며, 제약 조건도 적용되지 않습니다. 이러한 태그는 AWS, Azure 및 vSphere 배포를 프로비저닝하는 동안 자동으로 적용됩니다. 표준 태그는 시스템 사용자 지정 속성으로 저장되며 프로비저닝 이후에 배포에 추가됩니다.

다음은 표준 태그 목록입니다.

표 4-2. 표준 태그

설명	태그
조직	<code>org:orgID</code>
프로젝트	<code>project:projectID</code>
요청자	<code>requester:username</code>
배포	<code>deployment:deploymentID</code>
Blueprint 참조(해당하는 경우)	<code>Blueprint:blueprintID</code>
Blueprint의 구성 요소 이름	<code>blueprintResourceName:CloudMachine_1</code>
배치 제약 조건: Blueprint 및 요청 매개 변수에 적용 또는 IT 정책을 통해 적용	<code>constraints:key:value:soft</code>
클라우드 계정	<code>cloudAccount:accountID</code>
영역 또는 프로파일(해당하는 경우)	<code>zone:zoneID, networkProfile:profileID, storageProfile:profileID</code>

vRealize Automation Cloud Assembly의 태그 처리 방식

vRealize Automation Cloud Assembly에서 태그는 프로비저닝 과정에서 리소스가 프로비저닝된 배포에 할당되는 방법과 위치를 결정하는 기능 및 제약 조건을 표시합니다.

vRealize Automation Cloud Assembly는 프로비저닝된 배포를 생성할 때 특정 순서와 계층을 사용하여 태그를 확인합니다. 이 프로세스의 기본적인 사항을 이해하면 효율적으로 태그를 구현하여 예측 가능한 배포를 생성할 수 있습니다.

다음 목록에는 기능 및 제약 조건 태그 처리의 개략적인 작업 및 순서가 요약되어 있습니다.

- 클라우드 영역은 가용성 및 프로파일을 비롯한 여러 기준으로 필터링됩니다. 이때 클라우드 영역이 속한 영역의 프로파일에 포함된 태그를 일치시킵니다.
- 영역 및 계산 기능 태그는 고정 제약 조건으로 나머지 클라우드 영역을 필터링하는 데 사용됩니다.
- 필터링된 영역 중에서는 우선 순위에 따라 클라우드 영역이 선택됩니다. 우선 순위가 동일한 여러 클라우드 영역이 있는 경우 클라우드 영역과 계산 기능을 조합한 유동 제약 조건을 일치시켜 이러한 영역이 정렬됩니다.
- 클라우드 영역을 선택한 후에는 Blueprint에 표시된 고정 및 유동 제약 조건을 포함한 일련의 필터와 일치시켜 호스트가 선택됩니다.

간단한 태그 지정 구조를 설정하는 방법

이 항목에서는 논리적인 vRealize Automation Cloud Assembly 태그 지정 전략을 위한 기본적인 접근 방식과 옵션을 설명합니다. 이러한 예를 실제 배포의 시작점으로 활용할 수도 있고 요구 사항에 가장 잘 맞는 다른 전략을 세울 수도 있습니다.

대개 클라우드 관리자가 태그 생성 및 유지 관리를 주로 담당합니다.

이 항목에서는 vRealize Automation Cloud Assembly 설명서의 다른 곳에서 설명한 WordPress 사용 사례를 제시하여 일부 주요 항목에 태그를 추가하는 방법을 설명합니다. 또한 WordPress 사용 사례에 나온 태그 지정 예시에 대한 가능한 대안과 확대 사례를 설명합니다.

WordPress 사용 사례에 대한 자세한 내용은 [WordPress 사용 사례](#) 항목을 참조하십시오.

WordPress 사용 사례에서는 클라우드 영역과 스토리지 및 네트워크 프로파일에 태그를 배치하는 방법을 설명합니다. 이러한 프로파일은 구성된 리소스 패키지로 생각할 수 있습니다. 프로파일에 배치된 태그는 프로파일 내 모든 항목에 적용됩니다. 또한 스토리지 리소스와 개별 네트워크 항목 및 계산 리소스에도 태그를 생성하고 배치할 수 있지만 이러한 태그는 태그가 배치된 특정 리소스에만 적용됩니다. 태그를 설정할 때는 먼저 계산 리소스에 태그를 지정하고 이후에 프로파일과 클라우드 영역에 태그를 추가하는 것이 좋습니다. 또한 이러한 태그를 사용하여 클라우드 영역의 계산 리소스 목록을 필터링할 수 있습니다.

예를 들어, 이 사용 사례에 나온 대로 스토리지 프로파일에 태그를 배치할 수 있지만 개별 스토리지 정책, 데이터스토어 및 스토리지 계정도 태그를 배치할 수 있습니다. 이러한 리소스에 태그를 배치하면 스토리지 리소스의 배포 방식을 보다 세부적으로 제어할 수 있습니다. 배포를 준비하기 위해 처리되는 과정에서 이러한 태그는 프로파일 태그 다음에 처리되는 수준으로 확인됩니다.

일반적인 고객 시나리오를 구성하는 방법의 예로 `region: eastern` 태그를 네트워크 프로파일에 배치할 수 있습니다. 이 태그는 해당 프로파일 내 모든 리소스에 적용됩니다. 그런 다음 해당 프로파일 내 PCI 네트워크 리소스에 `networktype:pci` 태그를 배치할 수 있습니다. `eastern` 및 `pci`의 제약 조건을 가진 Blueprint는 Eastern 영역의 이 PCI 네트워크를 사용하는 배포를 생성합니다.

절차

1 논리적이고 적절한 방식으로 계산 인프라 리소스에 태그를 지정하십시오.

[클라우드 영역 생성] 페이지의 [계산] 탭에서 검색 기능을 사용하여 찾을 수 있도록 논리적인 방식으로 계산 리소스에 태그를 지정하는 것이 특히 중요합니다. 이 검색 기능을 사용하여 특정 클라우드 영역에 연결된 계산 리소스를 빠르게 필터링할 수 있습니다. 프로파일 수준에서 스토리지와 네트워크에 태그를 지정하는 경우 개별 스토리지 및 네트워크 리소스에는 태그를 지정할 필요가 없습니다.

- a **리소스 > 계산**을 선택하여 vRealize Automation Cloud Assembly 인스턴스에 가져온 계산 리소스를 봅니다.
- b 필요한 대로 각 계산 리소스를 선택하고 **태그**를 클릭하여 해당 리소스에 태그를 추가합니다. 필요한 경우 각 리소스에 둘 이상의 태그를 추가할 수 있습니다.
- c 필요에 따라 스토리지 및 네트워크 리소스에 대해 위의 단계를 반복합니다.

2 클라우드 영역 및 네트워크 프로파일 기능 태그를 생성합니다.

동일한 태그를 클라우드 영역과 네트워크 프로파일에 모두 사용하거나, 구현하기에 더 적합한 경우 각 항목에 고유한 태그를 생성할 수 있습니다.

네트워크 프로파일에서는 전체 프로파일과 프로파일 내 서브넷에 태그를 배치할 수 있습니다. 프로파일 수준에서 적용된 태그는 서브넷을 포함하여 해당 프로파일 내 모든 구성 요소에 적용되는 한편, 서브넷에 배치된 태그는 해당 태그가 배치된 특정 서브넷에만 적용됩니다. 태그 처리 시 프로파일 수준의 태그가 서브넷 수준의 태그보다 우선합니다.

클라우드 영역 또는 네트워크 프로파일에 태그를 추가하는 방법에 대한 자세한 내용은 [WordPress 사용 사례: 클라우드 영역 추가](#)와 [WordPress 사용 사례: 네트워크 프로파일 추가](#) 항목을 참조하십시오.

이 예에서는 vRealize Automation Cloud Assembly 클라우드 영역 및 네트워크 프로파일 태그에 대한 사용 사례 설명서 전반에 나오는 간단한 태그 3개를 생성합니다. 이러한 태그는 프로파일 구성 요소의 환경을 식별합니다.

- `zone:test`
- `zone:dev`
- `zone:prod`

3 스토리지 구성 요소에 대한 스토리지 프로파일 태그를 생성합니다.

일반적으로 스토리지 태그는 스토리지 항목의 성능 수준(예: Tier1 또는 Tier2)을 식별하거나 PCI 같은 스토리지 항목의 특성을 식별합니다.

스토리지 프로파일에 태그를 추가하는 방법에 대한 자세한 내용은 [WordPress 사용 사례: 스토리지 프로파일 추가](#) 항목을 참조하십시오.

- `usage:general`
- `usage:fast`

결과

기본적인 태그 지정 구조를 생성한 후에는 이 구조를 사용하여 필요에 따라 태그를 추가하거나 편집하면서 태그 지정 기능을 세분화하고 확장할 수 있습니다.

vRealize Automation Cloud Assembly에서 리소스로 작업하는 방법

클라우드 관리자는 데이터 수집을 통해 노출되는 vRealize Automation Cloud Assembly 리소스를 검토할 수 있습니다. 클라우드 관리자는 리소스에 기능 태그를 지정하여 vRealize Automation Cloud Assembly Blueprint를 배포할 위치에 영향을 줄 수 있습니다.

계산 리소스

클라우드 관리자는 데이터 수집을 통해 노출되는 계산 리소스를 검토할 수 있습니다. 클라우드 관리자는 리소스에 태그를 직접 적용하도록 선택하여 vRealize Automation Cloud Assembly 프로비저닝할 때 일치 기능을 찾도록 레이블을 지정할 수 있습니다.

네트워크 리소스

vRealize Automation Cloud Assembly에서 클라우드 관리자는 프로젝트의 클라우드 계정 및 통합에서 데이터가 수집된 네트워크 리소스를 살펴보고 편집할 수 있습니다.

클라우드 계정을 추가하면 데이터 수집에서 클라우드 계정의 네트워크 및 보안 정보를 검색하고 해당 정보를 네트워크 프로파일 및 기타 옵션에서 사용 가능한 상태로 설정합니다.

네트워크는 사용 가능한 네트워크 도메인 또는 전송 영역의 IP별 구성 요소입니다. Amazon Web Services 또는 Microsoft Azure 사용자인 경우 네트워크를 서브넷으로 간주합니다.

vRealize Automation Cloud Assembly **네트워크** 페이지에는 다음과 같은 정보가 포함되어 있습니다.

- 클라우드 계정의 네트워크 도메인(예: vCenter, NSX-V 또는 Amazon Web Services)에서 외부적으로 정의된 네트워크 및 로드 밸런서.
- 클라우드 관리자가 배포한 네트워크 및 로드 밸런서.
- 클라우드 관리자가 정의 또는 수정한 IP 범위 및 기타 네트워크 특성.
- 제공자별 외부 IPAM 통합의 특정 주소 공간에 대한 외부 IPAM 제공자 IP 범위.

네트워크에 대한 자세한 내용은 다음 정보, [네트워크](#) 페이지의 다양한 설정에 대한 표지판 도움말 및 [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

네트워크

네트워크와 해당 특성을 보고 편집할 수 있습니다. 예를 들어 태그를 추가하거나 공용 IP 액세스에 대한 지원을 제거할 수 있습니다. 또한 사용 가능한 네트워크 내에서 새로운 IP 범위를 정의하거나 기존 IP 범위를 관리할 수 있습니다.

기존 네트워크의 경우, 네트워크의 확인란을 선택하고 **IP 범위 관리** 또는 **태그**를 선택하여 IP 범위와 태그 설정을 변경할 수 있습니다. 또는 네트워크 자체를 선택하여 정보를 편집할 수 있습니다.

태그는 적절한 네트워크와 네트워크 프로파일(선택 사항)을 Blueprint의 네트워크 구성 요소에 일치시키는 방법을 제공합니다. 네트워크 태그는 네트워크가 상주할 수 있는 네트워크 프로파일에 관계없이 해당 네트워크의 모든 인스턴스에 적용됩니다. 네트워크는 임의의 수의 네트워크 프로파일에 인스턴스될 수 있습니다. 네트워크 프로파일 상주 여부와 관계없이, 네트워크 태그는 네트워크가 사용되는 모든 위치에서 네트워크와 연결됩니다. Blueprint가 하나 이상의 네트워크 프로파일과 일치하면 Blueprint의 다른 구성 요소와 네트워크 태그 일치가 발생합니다.

IP 범위

IP 범위를 선택하여 조직의 특정 네트워크에 대한 시작 및 끝 IP 주소를 정의하거나 변경할 수 있습니다.

기본 게이트웨이는 IP 범위에 포함할 수 없습니다. 서브넷 IP 범위에는 서브넷 게이트웨이 값이 포함될 수 없습니다.

특정 IPAM 제공자에 대해 외부 IPAM 통합을 사용하는 경우 IPAM IP 범위를 추가할 수 있습니다. 이 프로세스는 [vRealize Automation Cloud Assembly에서 IPAM 제공자 값을 사용하도록 네트워크 및 네트워크 프로파일 구성에서 전체 외부 IPAM 통합 워크플로와 관련하여 설명되어 있습니다](#).

IP 주소

조직에서 사용하는 정의된 IP 주소의 상태(예: 사용 가능 또는 할당됨)를 표시합니다.

로드 밸런서

조직의 계정/지역 클라우드 계정에 사용 가능한 로드 밸런서에 대한 정보를 표시합니다. 사용 가능한 각 로드 밸런서에 대해 구성된 설정을 열고 표시할 수 있습니다. 로드 밸런서에 대한 태그를 추가 및 제거할 수도 있습니다.

보안 리소스

클라우드 계정을 추가하면 데이터 수집에서 클라우드 계정의 네트워크 및 보안 정보를 검색하고 해당 정보를 네트워크 프로파일 및 기타 옵션에서 사용 가능한 상태로 설정합니다.

보안 그룹 및 방화벽 규칙은 네트워크 격리를 지원합니다.

보안 그룹

vRealize Automation Cloud Assembly에서 생성된 주문형 보안 그룹과 데이터 수집을 통해 노출되는 NSX-T 및 Amazon Web Services와 같은 소스 애플리케이션에서 생성된 기존 보안 그룹을 볼 수 있습니다.

사용 가능한 보안 그룹을 보고 선택한 보안 그룹에 대해 태그를 추가하거나 제거할 수 있습니다. 생성한 주문형 보안 그룹을 편집할 수도 있습니다. 클라우드 관리자는 보안 그룹을 열 수 있고 편집할 수 있습니다. Blueprint 작성자는 시스템 NIC에 하나 이상의 보안 그룹을 할당하여 Blueprint 배포를 위한 네트워크 규칙 및 기타 보안 측면을 제어할 수 있습니다.

보안 그룹 구성 요소의 Blueprint 코드에서 방화벽 규칙을 사용하는 경우,

기본 클라우드 계정 끝점(예: NSX-V, NSX-T 또는 Amazon Web Services 애플리케이션)의 기존 보안 그룹은 vRealize Automation Cloud Assembly가 데이터를 수집합니다. 기존 보안 그룹은 **원본** 열에 Discovered로 표시되고 분류됩니다. vRealize Automation Cloud Assembly에서 생성하고 Blueprint 또는 네트워크 프로파일에 위치하는 주문형 보안 그룹은 **원본** 열에 Managed by Cloud Assembly로 표시되고 분류됩니다. 조직에서 사용할 수 있는 보안 그룹만 표시됩니다.

vRealize Automation Cloud Assembly가 아닌 소스 애플리케이션(예: 소스 NSX 애플리케이션)에서 직접 기존 보안 그룹을 편집하는 경우 vRealize Automation Cloud Assembly 내에서 연결된 클라우드 계정 또는 통합 지점에 대한 데이터 수집을 실행할 때까지 vRealize Automation Cloud Assembly에 업데이트가 표시되지 않습니다.

클라우드 관리자는 Blueprint에서 사용할 수 있도록 기존 보안 그룹에 태그를 하나 이상 할당할 수 있습니다. 보안 그룹에는 하나 이상의 태그가 있어야 합니다. 그렇지 않으면 Blueprint에서 사용할 수 없습니다. Blueprint 작성자는 Blueprint의 Cloud.SecurityGroup 구성 요소를 통해 태그 제약 조건을 사용하여 기존 보안 그룹을 할당할 수 있습니다. Blueprint에서 Cloud.SecurityGroup 보안 그룹 구성 요소를 시스템 NIC에 적용하여 Blueprint 배포의 네트워크 규칙 및 기타 보안 측면을 제어해야 합니다.

관리자가 기존 보안 그룹에 태그를 추가하는 방법 또는 소스 애플리케이션의 보안 그룹에서 태그 데이터를 수집하는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

Blueprint 작성자가 Blueprint 보안 구성 요소에서 태그를 사용하는 방법에 대한 예는 [vRealize Automation Cloud Assembly Blueprint의 네트워크, 보안 및 로드 밸런서 예](#) 항목을 참조하십시오.

Blueprint 코드 샘플은 [vRealize Automation Cloud Assembly Blueprint의 네트워크, 보안 및 로드 밸런서 예](#)에서 "시스템 NIC에 제약 조건 태그가 적용된 기존 보안 그룹" 예를 참조하십시오.

네트워크 프로파일의 보안 그룹 사용에 대한 자세한 내용은 [vRealize Automation Cloud Assembly의 네트워크 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

네트워크 프로파일 및 Blueprint에서 보안 그룹을 정의하는 방법

다음 방법 중 하나로 보안 그룹 기능을 활용할 수 있습니다.

■ 네트워크 프로파일에 지정된 기존 보안 그룹

기존 보안 그룹을 네트워크 프로파일에 추가할 수 있습니다. Blueprint에서 해당 네트워크 프로파일을 사용하는 경우 해당 시스템은 보안 그룹의 멤버로 함께 그룹화됩니다. 이 방법을 사용하는 경우 Blueprint에 보안 그룹 구성 요소를 추가할 필요가 없습니다.

■ Blueprint의 시스템 구성 요소에 연결된 보안 그룹 구성 요소

Blueprint에 보안 그룹 구성 요소를 끌어다 놓고 Blueprint의 기존 보안 그룹 구성 요소 및 데이터 수집된 리소스의 기존 보안 그룹에서 제약 조건 태그를 사용하여 시스템 NIC에 보안 그룹 구성 요소를 바인딩할 수 있습니다.

- NSX-T 애플리케이션에 지정된 NSX-T 태그
- 네트워크 구성 요소가 Blueprint의 시스템 NIC에 연결되어 있는 Blueprint의 네트워크 구성 요소에서 제약 조건으로 지정된 NSX-T 태그를 사용할 수 있습니다. NSX-T 태그를 사용하면 NSX-T 소스 끝점에서 데이터 수집된 미리 정의된 NSX-T 태그를 사용하여 시스템을 동적으로 그룹화할 수 있습니다. NSX-T에서 NSX-T 태그를 생성할 때 논리적 포트를 사용합니다.

스토리지 리소스

클라우드 관리자는 연결된 클라우드 계정의 vRealize Automation Cloud Assembly 데이터 수집을 통해 검색된 스토리지 리소스와 해당 기능을 사용할 수 있습니다.

스토리지 리소스 기능은 일반적으로 소스 클라우드 계정에서 시작되는 태그를 통해 노출됩니다. 클라우드 관리자는 vRealize Automation Cloud Assembly를 사용하여 스토리지 리소스에 추가 태그를 직접 적용하도록 선택할 수 있습니다. 추가 태그는 프로비저닝 시간에 목적 일치를 위한 특정 기능에 레이블을 지정할 수 있습니다.

스토리지 리소스의 기능은 vRealize Automation Cloud Assembly 스토리지 프로파일 정의의 일부로 표시됩니다. [vRealize Automation Cloud Assembly 스토리지 프로파일에 대해 알아보기](#) 항목을 참조하십시오.

시스템 리소스

vRealize Automation에서 모든 사용자는 데이터 수집을 통해 노출되는 시스템 리소스를 검토할 수 있습니다.

프로젝트의 모든 시스템이 [시스템] 목록에 나타납니다.

관리되는 시스템과 마찬가지로, 프로젝트에서 클라우드 계정에 연결되어 있는 관리되지 않는 시스템이 이 목록에 나타납니다. [원본] 열은 시스템 상태를 나타냅니다.

- 검색됨 - 아직 온보딩되지 않은 시스템.
- 배포됨 - vRealize Automation에서 프로비저닝되거나 온보딩된 시스템.

워크로드 온보딩 계획을 사용하여 관리되지 않는 시스템을 vRealize Automation 관리로 가져올 수 있습니다.

온보딩 계획을 사용하여 관리되지 않는 시스템을 vRealize Automation 관리로 가져오는 방법에 대한 자세한 내용은 [vRealize Automation Cloud Assembly의 온보딩 계획이란?](#) 항목을 참조하십시오.

볼륨 리소스

vRealize Automation Cloud Assembly에서 모든 사용자는 볼륨 리소스를 검토할 수 있습니다.

vRealize Automation Cloud Assembly는 다음 두 가지 소스에서 만들어진 볼륨 또는 논리적 드라이브를 표시합니다.

- 소스 클라우드 계정의 데이터 수집을 통해 검색된 볼륨
- vRealize Automation Cloud Assembly에서 프로비저닝한 워크로드와 연결된 볼륨

볼륨 또는 논리적 드라이브에 따라 용량 및 기능을 검토할 수 있습니다. 또한 소스 클라우드 계정에서 시작되었거나 vRealize Automation Cloud Assembly 자체에 추가된 기능 태그가 목록에 표시됩니다.

vRealize Automation Cloud Assembly의 리소스에 대해 알아보기

vRealize Automation Cloud Assembly는 데이터 수집 리소스(예: 비용)에 대한 추가 정보를 노출할 수 있습니다.

vRealize AutomationvRealize Automation Cloud Assembly에서 데이터 수집의 작동 방식

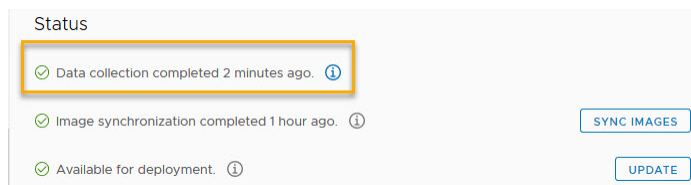
초기 데이터 수집 후에는 리소스 데이터 수집이 10분마다 자동으로 발생합니다. 데이터 수집 간격을 구성할 수 없으며 데이터 수집을 수동으로 시작할 수 없습니다.

해당 페이지의 [상태] 섹션에서 기존 클라우드 계정에 대한 리소스 데이터 수집 및 이미지 동기화 관련 정보를 검색할 수 있습니다. **인프라 > 연결 > 클라우드 계정**을 선택한 다음 선택한 기존 클라우드 계정에서 **열기**를 클릭하면 됩니다.

기존 클라우드 계정을 열고 해당 페이지의 **상태** 섹션에서 연결된 끝점 버전을 볼 수 있습니다. 연결된 끝점이 업그레이드된 경우 새 끝점 버전이 데이터 수집 중에 검색되고 클라우드 계정 페이지의 **상태** 섹션에 반영됩니다.

리소스 데이터 수집

데이터 수집은 10분마다 발생합니다. 각 클라우드 계정은 해당 데이터 수집이 마지막으로 완료된 시간을 표시합니다.

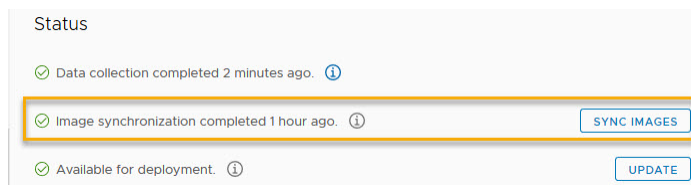


이미지 데이터 수집

이미지 동기화는 24시간마다 발생합니다. 일부 클라우드 계정 유형에 대해 이미지 동기화를 시작할 수 있습니다. 이미지 동기화를 시작하려면 클라우드 계정을 열고(**인프라 > 클라우드 계정**으로 이동한 후 기존 클라우드 계정을 선택하여 열기) **이미지 동기화** 버튼을 클릭합니다. NSX 클라우드 계정에 대한 이미지 동기화 옵션은 없습니다.

참고 이미지는 내부적으로 공용 또는 전용으로 분류됩니다. 공용 이미지는 공유되고 특정 클라우드 구독 또는 조직에 국한되지 않습니다. 전용 이미지는 공유되지 않으며 특정 구독에 국한됩니다. 공용 및 전용 이미지는 24시간마다 자동으로 동기화됩니다. 클라우드 계정 페이지의 옵션을 사용하여 전용 이미지에 대한 동기화를 트리거할 수 있습니다.

클라우드 계정 페이지는 이미지 동기화가 마지막으로 완료된 시간을 표시합니다.



배포에서 Fault Tolerance 및 고가용성을 용이하게 하기 위해 각 NSX-T 데이터 센터 끝점은 NSX Manager 3개의 클러스터를 나타냅니다. 관련 정보는 [vRealize Automation Cloud Assembly에서 NSX-T 클라우드 계정 생성](#) 항목을 참조하십시오.

클라우드 계정 및 온보딩 계획

클라우드 계정을 생성할 경우, 이 클라우드 계정에 연결된 모든 시스템의 데이터가 수집되어 **인프라 > 리소스 > 시스템** 페이지에 표시됩니다. 클라우드 계정에 vRealize Automation Cloud Assembly 외부에 배포된 시스템이 있는 경우에는 온보딩 계획을 사용하여 vRealize Automation Cloud Assembly가 해당 시스템을 관리하도록 할 수 있습니다.

클라우드 계정 추가에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에 클라우드 계정 추가](#)의 내용을 참조하십시오.

관리되지 않는 시스템의 온보딩에 대한 자세한 내용은 [vRealize Automation Cloud Assembly의 온보딩 계획이란?](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly 배포 비용 소개

프로젝트 관리자와 클라우드 관리자는 종종 비용 관리 작업을 수행합니다. vRealize Automation Cloud Assembly 비용 관리를 사용하면 개별 배포의 금전적인 영향을 이해할 수 있기 때문에 리소스를 관리하는데 유용합니다.

비용 산정을 보려면 vRealize Operations의 비용이 vRealize Automation 과 함께 작동하도록 구성하고 사용하도록 설정해야 합니다. vRealize Automation 으로 vRealize Operations을 구성하는 경우에는 두 애플리케이션을 동일한 표준 시간대로 설정해야 합니다. vRealize Operations에서 표준 시간대를 구성하려면 SSH를 사용하도록 설정하고 각 vRealize Operations 노드에 로그인한 후 \$ALIVE_Base/user/conf/analytics/advanced.properties 파일을 편집하고 timeZoneUseInMeteringCalculation = <time zone>을 추가합니다.

시간 경과에 따른 배포 비용은 배포 카드에 월간 누계 비용으로 표시되며 매월 초에 0으로 재설정됩니다. 구성 요소 비용 명세는 배포 세부 정보에서 사용할 수 있습니다. 배포 수준에서 이 정보를 제공하면 클라우드 관리자가 이 내용을 알 수 있으며 멤버 역시 자신들의 작업이 예산 및 장기적인 개발에 미칠 수 있는 영향을 이해할 수 있습니다.

이와 유사한 비즈니스 이유로 전체 프로젝트의 총 비용을 이해해야 할 수도 있습니다. 프로젝트 수준의 비용은 프로젝트 팀의 모든 배포에 대한 총 비용을 반영합니다. 전체 프로젝트 비용을 보려면 **인프라 > 프로젝트 > 프로젝트 선택 > 비용**을 선택합니다. 결과에는 총 비용과 배포별 비용 분석이 표시됩니다.

참고 프로젝트 비용에는 전용 클라우드 워크로드 비용만 포함됩니다. 프로젝트에 공용 클라우드에 속하는 배포가 포함된 경우 이러한 배포 비용은 프로젝트 비용에 포함되지 않습니다.

비용 계산 방법

계산 및 스토리지 리소스에 대한 배포 수준에서 표시되는 초기 비용은 업계 표준 벤치마크 요금에 기반하며 이후 시간 경과에 따라 계산됩니다. 원가는 호스트에 적용되고 서비스는 CPU 및 메모리 요금을 계산합니다. 서버는 24시간마다 비용을 다시 계산합니다.

또한 [vROPs 끝점] 페이지, **인프라 > 통합 > vROPs 끝점 >**에서 언제든지 비용 서버를 수동으로 새로 고칠 수도 있습니다. vCenter Servers 섹션에서 **동기화**를 클릭합니다. **동기화** 옵션을 사용하여 비용 서버를 수동으로 새로 고치면 조직의 모든 프로젝트에 대해 비용이 다시 계산됩니다. 조직의 프로젝트 수에 따라 이 프로세스가 많고 시간이 오래 걸릴 수 있습니다.

지원되는 리소스 목록을 보려면 vRealize Automation Cloud Assembly의 비용이 지정된 구성 요소 유형 목록 항목을 참조하십시오.

계산된 요금을 사용자 지정하는 방법

클라우드 관리자는 일정 기간 동안 벤치마크 값을 사용하다 보면 실제 비용이 할인된 것을 발견하게 될 수 있습니다. 비즈니스 실무가 더 잘 반영되도록 비용 계산에 사용되는 요금을 조정할 수 있습니다.

조정을 수행하면 다음에 서비스가 계산을 실행할 때 계산 변경 내용이 반영됩니다. 서버는 24시간마다 값을 다시 계산합니다.

vRealize Automation Cloud Assembly의 비용이 지정된 구성 요소 유형 목록

vRealize Automation Cloud Assembly에는 다음과 같은 Blueprint 구성 요소 유형에 대한 벤치마크 비용 정보가 제공됩니다.

표 4-3. 비용 계산된 구성 요소 유형


Blueprint 구성 요소 유형	서비스 이름/개체 유형	Blueprint 리소스 유형	설명
클라우드 애그노스틱	시스템	Cloud.Machine	애그노스틱 시스템이 vSphere로 구성된 경우 배포 비용을 볼 수 있습니다.
	디스크	Cloud.Volume	애그노스틱 디스크가 vSphere로 구성된 가상 시스템에 연결되어 있는 경우 배포 비용을 볼 수 있습니다.
vSphere	vSphere 시스템	Cloud.vSphere.Machine	클라우드 특정 Blueprint를 사용하여 배포됨.
	vSphere 디스크	Cloud.vSphere.Disk	가상 시스템에 연결된 클라우드별 Blueprint를 사용하여 배포됩니다.

배포 비용을 예측하는 방법


카탈로그 항목을 배포하기 전에 선행 비용을 배포에 대한 비용 예상으로 사용할 수 있습니다.

Daily Cost Estimate

×


vSphere machine with disk

\$2.14


Cloud_vSphere_Machine_1

\$2.11

Compute


Storage

Additional charges

\$1.97

\$0.03


\$0.11


Cloud_vSphere_Disk_1

\$0.03

Storage

\$0.03


Cloud_vSphere_Network_1

Cost for this resource is not supported.

CLOSE

배포에 대한 선행 비용은 지정된 카탈로그 항목(배포되기 전)에 대한 리소스 할당을 기반으로 하는 일별 비용 예상입니다. 카탈로그 항목이 배포된 후에는 **배포 및 인프라 > 프로젝트** 탭에서 선행 비용의 집계로 월간 누계 비용을 볼 수 있습니다. 선행 비용 산정은 vSphere 시스템 및 vSphere 디스크와 같은 사설 클라우드 리소스, Cloud Assembly 카탈로그 항목 및 사설 클라우드에 대해 구성된 vCenter를 사용하는 클라우드를 애그노스틱 항목에 대해 지원됩니다.

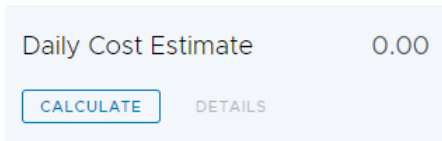
참고 공용 클라우드 리소스 또는 비 vSphere 시스템 또는 디스크 사설 클라우드 리소스에는 선행 비용 산정이 지원되지 않습니다.

사전 요구 사항

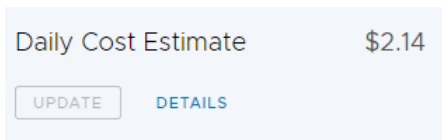
vRealize Automation Cloud Assembly에서 선행 비용을 보려면 비용 산정이 가능하도록 구성되고 통화가 미리 설정된 vRealize Operations 통합 끝점이 있어야 합니다.

절차

- 1 [카탈로그]에서 카탈로그 항목을 선택하고 **요청**을 클릭합니다.



- 2 카탈로그 항목 요청에 대한 세부 정보를 입력하고 **계산**을 클릭합니다.



- 3 (선택 사항) **세부 정보**를 클릭하면 [일별 비용 예상] 창에서 비용 분석을 확인할 수 있습니다.

다음에 수행할 작업

일별 비용 예상이 타당하면 **제출**을 클릭하여 배포 요청을 계속 진행합니다.

배포 비용이 높게 표시되는 이유를 확인하는 방법

비용을 관리하는 동안 프로젝트 관리자와 클라우드 관리자에게 배포 비용이 높아 보일 수 있습니다.

배포 비용은 해당 리소스를 기준으로 계산됩니다. 경우에 따라 단일 배포의 리소스를 결합하면 비용이 증가하여 비즈니스 요구에 따른 배포 비용이 높아질 수 있습니다. 따라서 프로젝트에 비용이 높은 배포가 하나 이상 포함되면 집계 프로젝트 비용도 증가합니다.

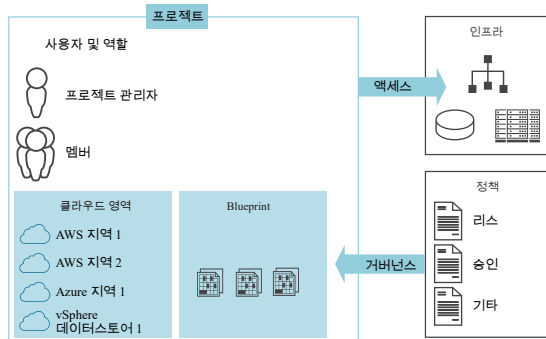
배포 비용이 높은지 판단하는 것은 비즈니스 요구 사항에 따라 다릅니다. 배포에 대한 비용 분석은 배포의 세부 정보를 열고 **비용**을 클릭하여 볼 수 있습니다. 배포 비용을 수정하려면 해당 Blueprint의 리소스 구성을 편집하고 다시 배포해야 합니다.

vRealize Automation Cloud Assembly 프로젝트 추가 및 관리

5

프로젝트는 vRealize Automation Cloud Assembly Blueprint에 액세스할 수 있는 사용자 및 Blueprint가 배포되는 위치를 제어합니다. 프로젝트를 사용하면 사용자가 수행할 수 있는 작업과 클라우드 인프라에서 Blueprint를 배포할 수 있는 클라우드 영역을 구성하고 제어할 수 있습니다.

클라우드 관리자는 사용자 및 클라우드 영역을 추가할 수 있는 프로젝트를 설정합니다. Blueprint를 생성하고 배포하는 사용자는 프로젝트 하나 이상의 멤버여야 합니다.



본 장은 다음 항목을 포함합니다.

- vRealize Automation Cloud Assembly 개발 팀용 프로젝트를 추가하는 방법
- vRealize Automation Cloud Assembly 프로젝트에 대해 알아보기

vRealize Automation Cloud Assembly 개발 팀용 프로젝트를 추가하는 방법

프로젝트를 생성하여 멤버와 클라우드 영역을 추가하면 프로젝트 멤버가 Blueprint를 연결된 영역에 배포할 수 있습니다. vRealize Automation Cloud Assembly 관리자 권한으로 개발 팀을 위한 프로젝트를 생성합니다. 그런 다음 프로젝트 관리자를 할당하거나 프로젝트 관리자 권한으로 작업할 수 있습니다.

Blueprint 생성 시, Blueprint를 연결할 프로젝트를 먼저 선택합니다. 프로젝트는 Blueprint를 생성하기 전에 존재해야 합니다.

프로젝트가 개발 팀의 비즈니스 요구 사항을 지원하는지 확인합니다.

- 프로젝트에서 팀의 목표를 지원하는 리소스를 제공합니까? 인프라 리소스와 프로젝트가 Blueprint를 지원하는 방식에 대한 예는 [WordPress 사용 사례](#) 항목을 참조하십시오.

이 절차는 기본 구성만 포함된 초기 프로젝트 생성을 기반으로 합니다. 개발 팀에서 Blueprint를 생성하고 배포하면 프로젝트를 수정할 수 있습니다. 제약 조건, 사용자 지정 속성 및 기타 옵션을 추가하여 배포 효율성을 향상시킬 수 있습니다. [vRealize Automation Cloud Assembly 프로젝트에 대해 알아보기](#)에 있는 문서를 참조하십시오.

사전 요구 사항

- 클라우드 영역을 구성했는지 확인합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#) 항목을 참조하십시오.
- 이 프로젝트의 클라우드 영역으로 포함된 영역에 대한 매핑 및 프로파일을 구성했는지 확인합니다. [장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축](#) 항목을 참조하십시오.
- 이 작업을 수행하는 데 필요한 사용 권한이 있는지 확인합니다. [vRealize Automation Cloud Assembly 사용자 역할이란?](#)의 내용을 참조하십시오.
- 프로젝트 관리자로 지정할 사용자를 결정합니다. 프로젝트 관리자가 vRealize Automation Cloud Assembly에서 수행할 수 있는 작업을 이해하려면 [vRealize Automation Cloud Assembly 사용자 역할이란?](#) 항목을 참조하십시오.
- 프로젝트에 Active Directory 그룹을 추가하는 경우 조직에 대한 Active Directory 그룹을 구성했는지 확인합니다. "vRealize Automation 관리"에 있는 [vRealize Automation에서 그룹 역할 할당 편집](#)을 참조하십시오. 그룹이 동기화되지 않은 경우 프로젝트에 추가하려고 할 때 사용할 수 없습니다.

절차

- 1 **인프라 > 구성 > 프로젝트**를 선택하고 **새 프로젝트**를 클릭합니다.
- 2 프로젝트 이름을 입력합니다.
- 3 **사용자** 탭을 클릭합니다.
 - a 프로젝트 멤버의 배포에 소유자만 액세스할 수 있게 하려면 **배포 공유**를 해제합니다.
 - b 할당된 역할이 있는 사용자를 추가합니다.
- 4 **프로비저닝** 탭을 클릭하고 하나 이상의 클라우드 영역을 추가합니다.

클라우드 영역에는 사용자가 배포한 Blueprint를 지원하는 리소스가 포함되어야 합니다.
- 5 **생성**을 클릭합니다.
- 6 프로젝트 클라우드 영역을 사용하여 프로젝트를 테스트하려면 [프로젝트] 페이지에서 **구성 테스트**를 클릭합니다.

시뮬레이션은 프로젝트 클라우드 영역 리소스에 대해 표준화된 가상의 배포 테스트를 실행합니다. 실패하는 경우 세부 정보를 검토하고 리소스 구성을 수정할 수 있습니다.

다음에 수행할 작업

Blueprint를 시작합니다. [장 6 vRealize Automation Cloud Assembly 배포 설계](#) 항목을 참조하십시오.

vRealize Automation Cloud Assembly 프로젝트에 대해 알아보기

프로젝트는 Blueprint와 리소스 간의 컨텍터입니다. 작동 방식 및 활용하는 방법을 더 많이 이해할수록 vRealize Automation Cloud Assembly 개발 및 배포 프로세스의 효율성을 높일 수 있습니다.

vRealize Automation Cloud Assembly 프로젝트 태그 및 사용자 지정 속성 사용

관리자는 프로젝트 요구 사항이 vRealize Automation Cloud Assembly Blueprint와 다른 경우 프로젝트 수준 거버넌스 제약 조건 또는 사용자 지정 속성을 추가할 수 있습니다. 제약 조건 태그 외에, 프로비저닝 프로세스 중 배포된 리소스에 추가된 리소스 태그를 추가하여 리소스를 관리할 수 있습니다.

프로젝트 리소스 태그 소개

프로젝트 리소스 태그는 배포된 리소스를 관리하고 규정 준수를 보장하는 데 사용할 수 있는 표준화된 식별 태그로 작동합니다.

프로젝트에 정의된 리소스 태그는 해당 프로젝트의 일부로 배포된 모든 구성 요소 리소스에 추가됩니다. 그런 다음 표준 태그 지정을 통해 다른 애플리케이션을 사용하여 리소스를 관리할 수 있습니다.

예를 들어 클라우드 관리자는 CloudHealth와 같은 애플리케이션을 사용하여 비용을 관리하려고 합니다. 유럽연합 휴먼 리소스 도구 개발 전용 프로젝트에 `costCenter:eu-cc-1234` 태그를 추가합니다. 프로젝트 팀이 이 프로젝트에서 배포하면, 해당 태그가 배포된 리소스에 추가됩니다. 그런 다음 이 태그가 포함된 리소스를 식별하고 관리하도록 비용 산정 도구를 구성합니다. 다른 비용 센터가 있는 다른 프로젝트에는 키와 함께 사용할 대체 값이 있습니다.

프로젝트 제약 조건 태그 소개

프로젝트 제약 조건은 거버넌스 정의로 사용됩니다. 제약 조건은 프로젝트 클라우드 영역에서 배포 요청에 사용하거나 사용하지 않을 리소스를 정의하는 `key:value` 태그입니다.

배포 프로세스는 네트워크 및 스토리지에 대해 프로젝트 제약 조건과 일치하는 태그를 찾고, 일치하는 태그를 기반으로 배포를 수행합니다.

확장성 제약 조건은 확장성 워크플로에 사용할 vRealize Orchestrator 통합 인스턴스를 지정하는 데 사용됩니다.

프로젝트 제약 조건을 구성할 때 다음과 같은 형식을 고려하십시오.

- **key:value** 및 **key:value:hard**. 이 태그는 Blueprint를 일치하는 기능 태그를 가진 리소스에 배포해야 하는 경우에 두 가지 형식 중 하나로 사용합니다. 일치하는 태그를 찾을 수 없으면 배포 프로세스가 실패합니다. 예를 들어 프로젝트의 멤버가 배포한 Blueprint를 PCI 준수 네트워크에 프로비저닝해야 합니다. `security:pci`를 사용합니다. 프로젝트 클라우드 영역에서 네트워크를 찾을 수 없으면 안전하지 않은 배포가 없도록 하기 위해 배포가 실패합니다.

- **key:value:soft.** 이 태그는 일치하는 리소스를 원하지만 배포 프로세스가 실패 없이 진행되도록 태그가 일치하지 않는 리소스도 허용할 수 있는 경우에 사용됩니다. 예를 들어 프로젝트 멤버가 Blueprint를 더 저렴한 스토리지에 배포하는 것을 선호하지만 스토리지 가용성이 배포 가능성에 방해가 되는 것을 원하지 않습니다. tier:silver:soft를 사용합니다. 그러면 프로젝트 클라우드 영역에 tier:silver 태그가 지정된 스토리지가 없는 경우에도 Blueprint가 다른 스토리지 리소스에 배포됩니다.
- **!key:value.** 일치하는 태그를 가진 리소스에 배포하지 않으려면 **hard** 또는 **soft** 옵션과 함께 이 태그를 사용합니다.

중요한 점은 프로젝트 제약 조건 태그는 Blueprint 제약 조건 태그보다 우선 순위가 높기 때문에 배포 시 Blueprint 제약 조건 태그를 재정의합니다. 이러한 재정의가 발생하면 안 되는 Blueprint가 있는 경우에는 Blueprint에 failOnConstraintMergeConflict:true를 사용할 수 있습니다. 예를 들어 프로젝트에 네트워크 loc:london 제약 조건이 있지만 Blueprint의 제약 조건이 loc:mumbai인 경우 프로젝트 위치를 우선적으로 사용하는 대신 제약 조건 충돌 메시지와 함께 배포가 실패하도록 설정하려면 다음 샘플과 유사한 속성을 추가합니다.

```
constraints:
  - tag: 'loc:mumbai'
  failOnConstraintMergeConflict:true
```

사용자 지정 속성을 사용하는 방법

보고를 위한 프로젝트 사용자 지정 속성을 사용하여 확장성 작업 및 워크플로를 트리거하여 채우고 Blueprint 수준 속성을 재정의할 수 있습니다.

배포에 사용자 지정 속성을 추가하면 해당 값을 사용자 인터페이스에 사용하거나, 보고서를 생성할 수 있도록 API를 사용하여 해당 값을 검색할 수 있습니다.

확장성의 경우, 확장성 구독을 위해 사용자 지정 속성을 사용할 수도 있습니다.

프로젝트에 대해 변경할 특정 속성 값이 Blueprint에 포함될 수 있습니다. 다른 이름 및 값을 사용자 지정 속성으로 제공할 수 있습니다.

배포 시간에 vRealize Automation Cloud Assembly 프로젝트가 작동하는 방식

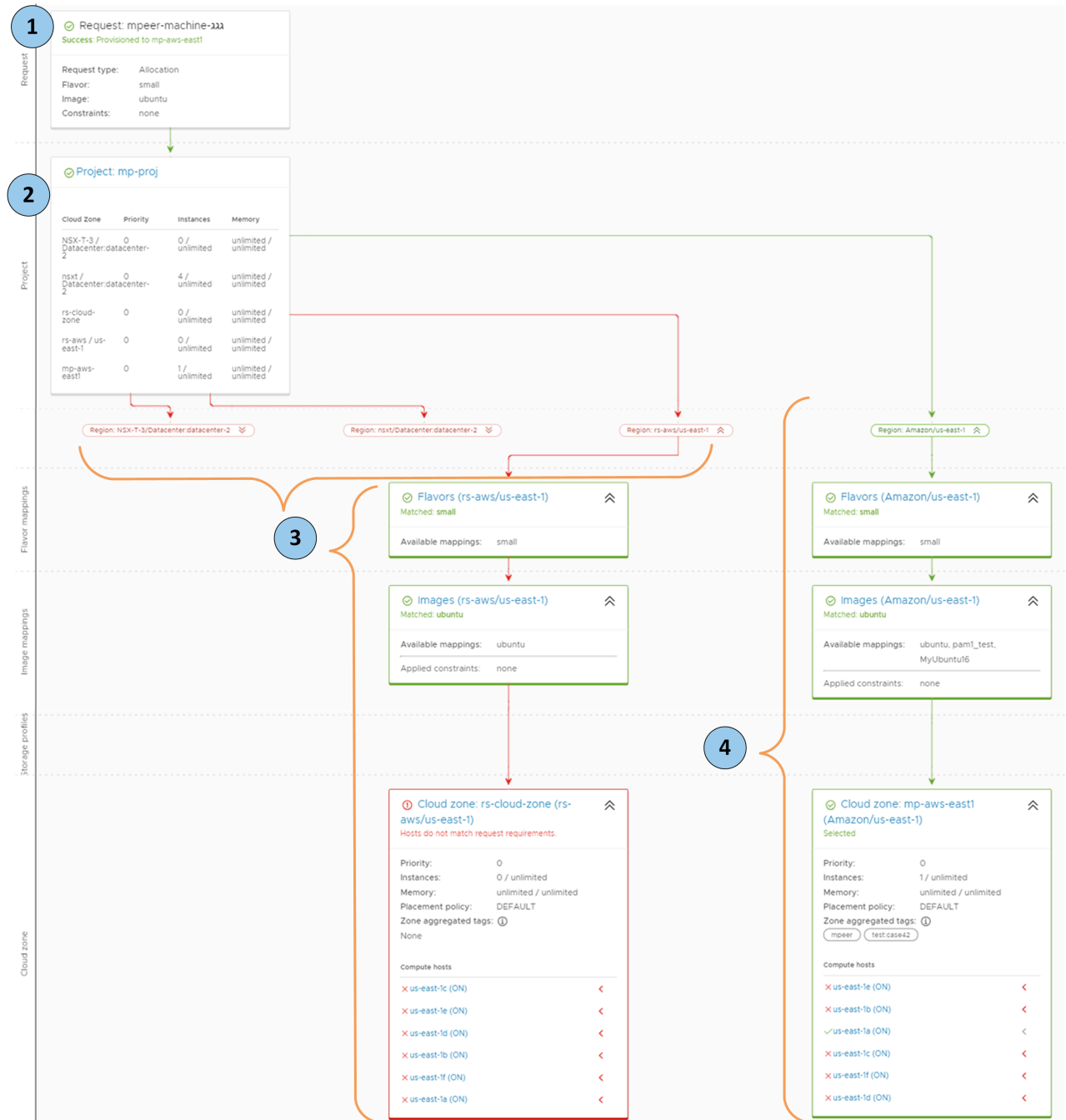
프로젝트는 프로비저닝된 리소스의 사용자 소유권 및 클라우드 영역에 대한 사용자 액세스를 제어합니다. 클라우드 관리자든 Blueprint 개발자든 배포를 관리하고 문제를 해결할 수 있으려면 배포 시간에 프로젝트가 어떻게 작동하는지 이해해야 합니다.

다양한 팀의 프로젝트를 설정하는 클라우드 관리자는 프로젝트에서 Blueprint 구성 요소가 배포되는 위치를 결정하는 방법을 이해해야 합니다. 이러한 이해는 Blueprint 개발자를 지원하는 프로젝트를 생성하고 실패한 배포 문제를 해결하는 데 도움을 줍니다.

Blueprint를 생성할 때 먼저 Blueprint를 프로젝트와 연결합니다. 배포 시, 프로젝트 클라우드 영역에 대해 Blueprint 요구 사항을 평가하여 최적의 배포 위치를 찾습니다.

다음 워크플로는 해당 프로세스를 설명합니다.

- 1 Blueprint 배포 요청을 제출합니다.
- 2 프로젝트는 Blueprint 및 프로젝트 요구 사항(예: 플레이버, 이미지, 제약 조건 태그)을 평가합니다. 요구 사항을 프로젝트 클라우드 영역과 비교하여 요구 사항이 지원되는 영역을 찾습니다.
- 3 이러한 영역에는 요청을 지원하는 리소스가 없습니다.
- 4 이 클라우드 영역은 요청 요구 사항을 지원합니다. Blueprint가 이 클라우드 영역 계정 지역에 배포됩니다.



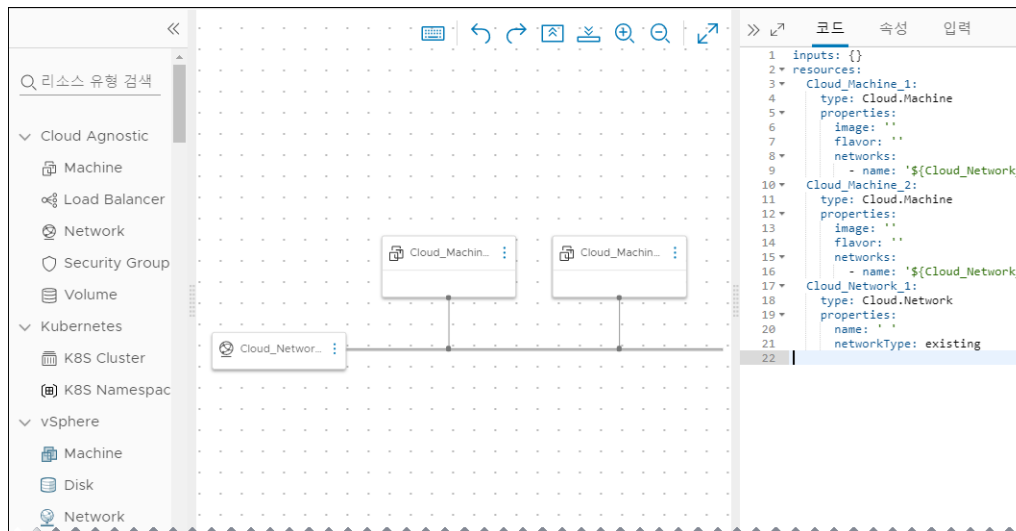
vRealize Automation Cloud Assembly 배포 설계

6

배포는 vRealize Automation Cloud Assembly를 통해 클라우드 리소스에 생성하는 시스템, 애플리케이션 및 서비스를 정의하는 규격인 **Blueprint**로 시작됩니다.

Blueprint 개발자는 특정 클라우드 벤더를 대상으로 하는 Blueprint를 설계하거나, 클라우드 애그노스틱 Blueprint를 설계할 수 있습니다. 프로젝트에 할당된 클라우드 영역에 따라 어느 접근 방법을 사용할지 결정됩니다. 클라우드 영역을 구성하는 리소스 종류를 이해하려면 클라우드 관리자에게 문의하십시오.

vRealize Automation Cloud Assembly Blueprint를 생성하는 것은 인프라를 코드로 구현하는 프로세스입니다. 먼저 설계 캔버스에서 구성 요소를 추가하고 연결합니다. 그런 다음 캔버스의 오른쪽에 있는 코드 편집기를 사용하여 세부 정보를 완성합니다. 코드 편집기를 사용하면 코드를 직접 입력하거나 속성 값을 양식에 입력할 수 있습니다.



본 장은 다음 항목을 포함합니다.

- Blueprint를 생성하기 전에
- Blueprint를 생성하는 방법
- 간단한 vRealize Automation Cloud Assembly Blueprint를 처음부터 생성하는 방법
- 간단한 vRealize Automation Cloud Assembly Blueprint를 향상시키는 방법

- 여러 버전의 vRealize Automation Cloud Assembly Blueprint를 저장하는 방법
- vRealize Automation Cloud Assembly를 사용하여 배포된 리소스의 이름을 사용자 지정하는 방법
- vRealize Automation 리소스 속성 소개
- 일부 Blueprint 코드 예시
- vRealize Automation Cloud Assembly 마켓플레이스를 사용하는 방법
- 확장성을 사용하여 애플리케이션 수명 주기를 연장 및 자동화하는 방법

Blueprint를 생성하기 전에

vRealize Automation Cloud Assembly Blueprint는 언제든지 생성할 수 있지만, Blueprint를 배포하려면 클라우드 리소스 인프라부터 정의해야 합니다.

- **장 4 vRealize Automation Cloud Assembly 리소스 인프라 구축**

또한 이러한 인프라 리소스를 클라우드 영역으로 포함하는 vRealize Automation Cloud Assembly 프로젝트도 생성해야 합니다.

- **vRealize Automation Cloud Assembly 프로젝트 태그 및 사용자 지정 속성 사용**

Blueprint를 생성하는 방법

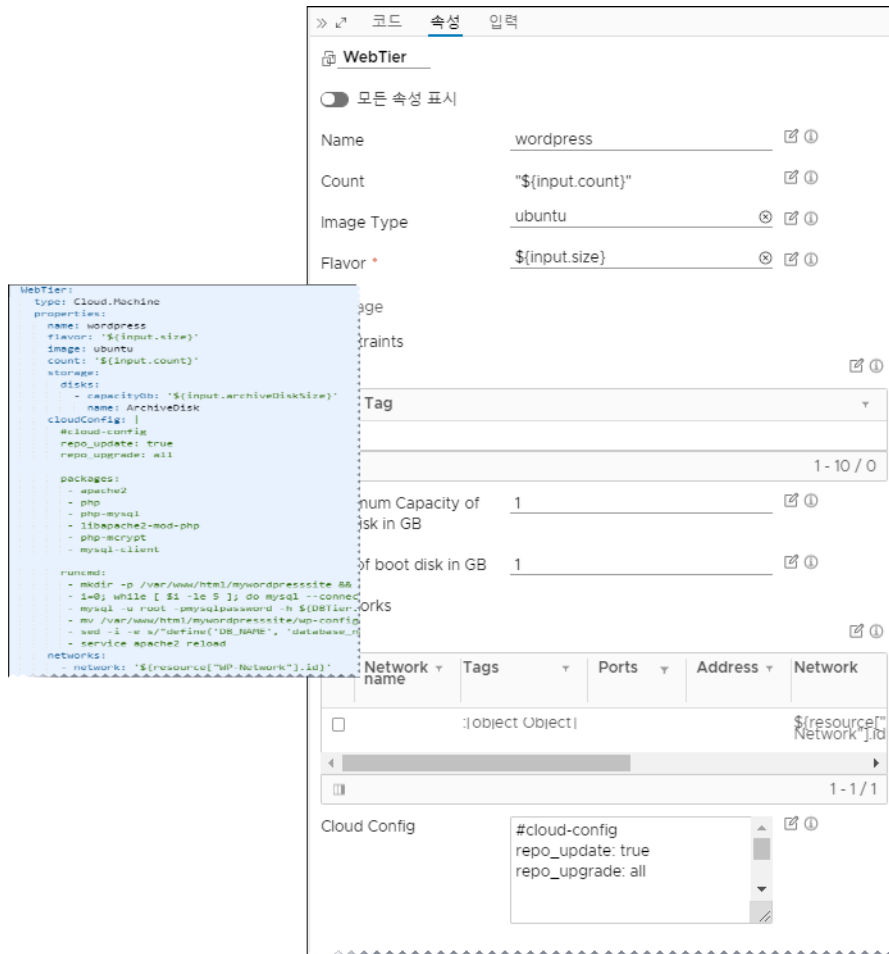
vRealize Automation Cloud Assembly는 Blueprint를 코드로 생성하고 저장하기 때문에 Blueprint를 설계하고 재사용하기가 쉽습니다.

빈 캔버스에서 Blueprint를 구축하거나 기존 코드를 활용할 수 있습니다.

vRealize Automation Cloud Assembly Blueprint 설계 페이지

Blueprint를 처음부터 생성하려면 **Blueprint**로 이동하여 **새로 만들기**를 클릭합니다. 구성 요소를 캔버스에 끌어 놓고 연결한 다음 코드 편집기에서 구성을 완료합니다.

코드 편집기를 사용하면 코드를 직접 입력하고, 잘라내고, 복사하고 붙여 넣을 수 있습니다. 코드 편집이 불편하면 설계 캔버스에서 리소스를 선택하고 코드 편집기 **속성** 탭을 클릭한 다음 그 곳에 값을 입력하면 됩니다. 입력한 속성 값이 마치 직접 입력한 것처럼 코드에 나타납니다.



한 Blueprint에서 다른 Blueprint로 코드를 복사하여 붙여 넣을 수 있습니다.

Blueprint 복제

Blueprint를 복제하려면 **Blueprint**로 이동하여 소스를 선택하고 **복제**를 클릭합니다. Blueprint를 복제하여 소스의 복사본을 생성한 후 해당 클론을 새 프로젝트에 할당하거나, 새 애플리케이션의 시작 코드로 사용합니다.

업로드 및 다운로드

vRealize Automation Cloud Assembly 마켓플레이스에는 점프스타트를 위해 마무리된 Blueprint가 제공됩니다. [vRealize Automation Cloud Assembly 마켓플레이스를 사용하는 방법](#) 항목을 참조하십시오.

또한 사이트에 적합한 방식으로 Blueprint YAML 코드를 업로드, 다운로드 및 공유할 수 있습니다. 외부 편집기와 개발 환경을 사용하여 Blueprint 코드를 수정할 수도 있습니다.

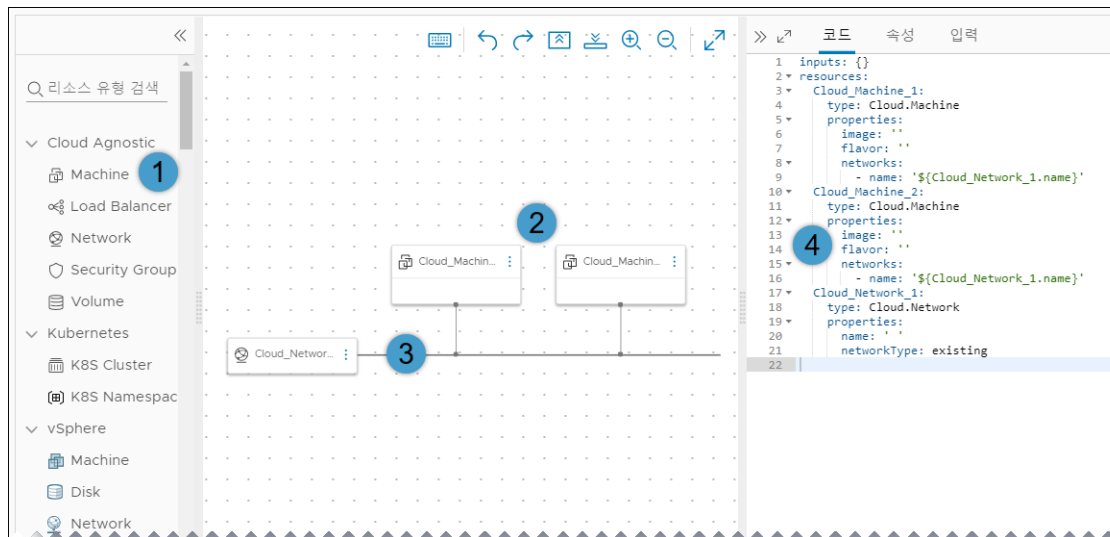
참고 공유 Blueprint 코드를 검증하는 유용한 방법은 Blueprint 설계 페이지의 vRealize Automation Cloud Assembly 코드 편집기에서 코드를 검사하는 것입니다.

Blueprint 249개 항목					
<div> <div>+ 새로 만들기</div> <div>+ 업로드</div> <div>+ 저장소 동기화</div> <div>+ 삭제</div> <div>+ 복사</div> <div>+ 백업</div> <div>+ 다운로드</div> <div>+ 설정</div> </div> <div>🔍 필터...</div>					
<input type="checkbox"/>	이름	소스 제어	프로젝트	마지막 업데이트	업데이트한 사람
<input checked="" type="checkbox"/>	vSphere-With-Disk-Attached		test-AD-project	2020년 1월 21일 오후 12:48:57	sestervii@vmware.com
<input type="checkbox"/>	DB		0709-AWS-w2용가상시스템구축	2020년 1월 21일 오전 11:34:36	pmartini@vmware.com
<input type="checkbox"/>	WordPress-BP		0709-AWS-w2용가상시스템구축	2020년 1월 20일 오후 3:25:36	canl@vmware.com
<input type="checkbox"/>	git BPP		Azure Project용가상시스템구축	2020년 1월 20일 오후 3:18:40	canl@vmware.com
<input type="checkbox"/>	mp-nxsv		0717VSPHERE_PROJECTVSPHERE용가상시스템구축	2020년 1월 20일 오전 10:03:51	pmartini@vmware.com
<input type="checkbox"/>	> WP - PORI		wordpress project	2020년 1월 19일 오후 7:12:37	pmartini@vmware.com

간단한 vRealize Automation Cloud Assembly Blueprint를 처음부터 생성하는 방법

설계 페이지를 사용하면 프로비저닝할 시스템 또는 애플리케이션에 대한 vRealize Automation Cloud Assembly Blueprint 규칙을 생성할 수 있습니다.

- 1 구성 요소를 찾습니다.
- 2 구성 요소를 캔버스로 끌어 옵니다.
- 3 구성 요소를 연결합니다.
- 4 Blueprint 코드를 편집하여 구성 요소를 구성합니다.



설계 페이지에서 Blueprint 이름 또는 버전을 변경하거나, 특정 버전으로 되돌리거나, Blueprint를 복제 또는 배포할 수도 있습니다.

Blueprint에 vRealize Automation Cloud Assembly 구성 요소를 선택하고 추가하는 방법

vRealize Automation Cloud Assembly 구성 요소는 Blueprint 구축 블록입니다. 설계 페이지에서는 클라우드 애그노스틱 구성 요소 또는 클라우드 벤더별 구성 요소를 사용할 수 있습니다.

구성 요소는 설계 페이지의 왼쪽에 선택할 수 있게 표시됩니다.

클라우드 애그노스틱 구성 요소

클라우드 애그노스틱 구성 요소는 모든 클라우드 벤더에 배포할 수 있습니다. 프로비저닝 시 일치하는 클라우드별 리소스가 배포에 사용됩니다. 예를 들어 Blueprint를 AWS 및 vSphere 클라우드 영역 둘 모두에 배포하려면 클라우드 애그노스틱 구성 요소를 사용합니다.

클라우드 벤더 구성 요소

Amazon Web Services, Microsoft Azure 및 VMware vSphere 구성 요소는 일치하는 AWS, Azure 또는 vSphere 클라우드 영역에만 배포될 수 있습니다.

클라우드 애그노스틱 구성 요소를 특정 벤더를 위한 클라우드별 구성 요소가 포함된 Blueprint에 추가할 수 있습니다. 단, 프로젝트 클라우드 영역에서 벤더별로 지원되는 사항을 잘 알고 있어야 합니다.

구성 관리 구성 요소

구성 관리 구성 요소는 통합된 애플리케이션에 따라 다릅니다. 예를 들어 Puppet 구성 요소는 다른 구성 요소의 구성을 모니터링하고 적용할 수 있습니다.

vRealize Automation Cloud Assembly에서 Blueprint 리소스를 연결하는 방법

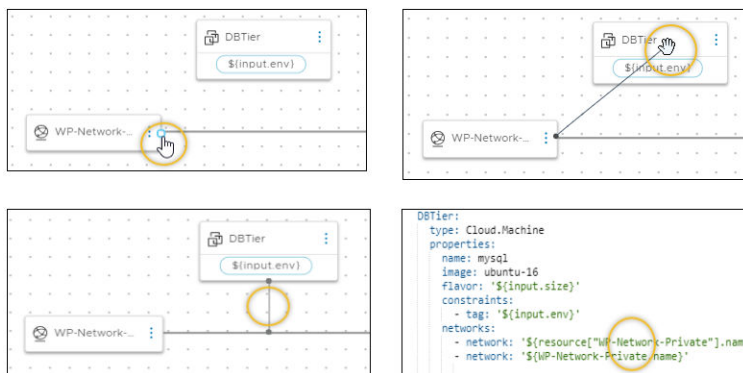
그래픽 설계 캔버스를 사용하여 vRealize Automation Cloud Assembly Blueprint 리소스를 연결합니다.

리소스 연결이 호환되는 경우에 리소스를 연결할 수 있습니다. 예를 들면 다음과 같습니다.

- 시스템 클러스터에 로드 밸런서 연결
- 네트워크에 시스템 연결
- 시스템에 외부 스토리지 연결

리소스를 연결하려면 리소스 가장자리 위로 마우스를 이동하여 연결 버블을 표시합니다. 그런 다음 버블을 클릭한 채로 대상 리소스로 끌고 가서 마우스 버튼을 놓습니다.

코드 편집기에서 보면 소스 리소스에 대한 추가 코드가 대상 리소스 코드에 표시됩니다.



리소스 사이의 실선은 리소스가 동일한 위치에서 끝나야 함을 나타냅니다. 캔버스에 연결을 추가할 수 있지만 충돌하는 배치 제약 조건 태그가 있는 경우 배포가 실패합니다. 예를 들어, 하나는 테스트 us-west-1 클라우드 영역으로 하드 제한되고 다른 하나는 운영 us-east-1 클라우드 영역으로 하드 제한된 리소스를 연결하는 경우 배포가 실패합니다.

올바른 vRealize Automation Cloud Assembly Blueprint 코드를 생성하는 방법

vRealize Automation Cloud Assembly 구성 요소를 추가하고 캔버스에서 이러한 구성 요소를 연결하면 시작 코드만 생성됩니다. 이러한 구성 요소를 완전하게 구성하려면 코드를 편집해야 합니다.

코드 편집기를 사용하면 코드를 직접 입력하거나 속성 값을 양식에 입력할 수 있습니다. 직접 코드 생성을 지원하기 위해 vRealize Automation Cloud Assembly 편집기에는 구문 완성 및 오류 검사 기능이 포함되어 있습니다.

편집기 힌트 예

사용 가능한 값

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15
16     networks:
17       - name: small flavor
18       - name: large flavor
19   type: Cloud.Network
20   properties:
21     name: ''
22     networkType: existing
  
```

허용되는 속성

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15
16     tags: array
17
18     storage: object
19
20     remoteAccess: object
21
22     name: string
23
24     imageRef: string
25
26     count: integer
27
28     constraints: array
29
30     cloudConfig: string
  
```


하위 속성

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15     constraints:
16       -
17         tag: string
18   type: Cloud.Network
19   properties:
20     name: ''
  
```

편집기 힌트 예

구문 오류

 Please correct errors in YAML editor before editing in canvas: row: 14, column: 17

```

10 ▾ Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: 'small'
15     constraints:
16     networks:
17       - name: '${Cloud_Network_1.name}'
18 ▾ Cloud_Network_1:
19   type: Cloud.Network
20   properties:
21     name: ''
22     networkType: existing

```

검색

(Ctrl+F)

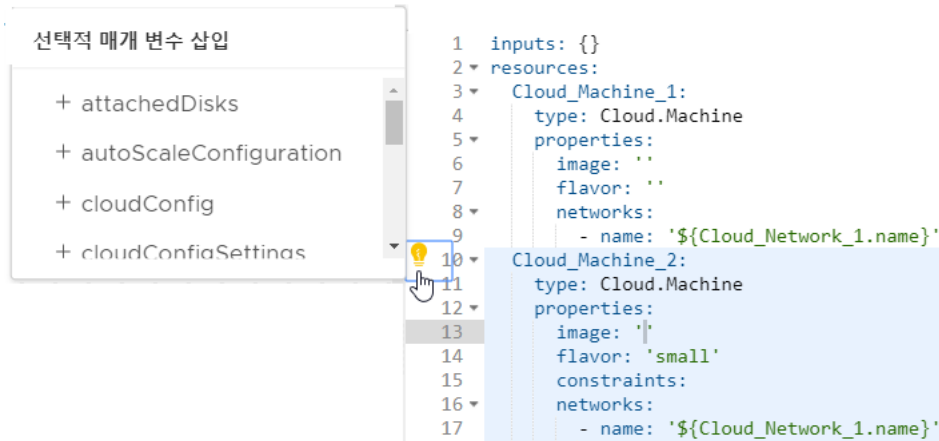
```

1  inputs: {}
2  ▾ resources:
3  ▾ Cloud_Machine_1:
4     type: Cloud.Machine
5     properties:
6     image: ''
7     flavor: ''
8     networks:
9       - name: '${Cloud_Network_1.name}'
10 ▾ Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: 'small'
15     constraints:
16     networks:
17       - name: '${Cloud_Network_1.name}'

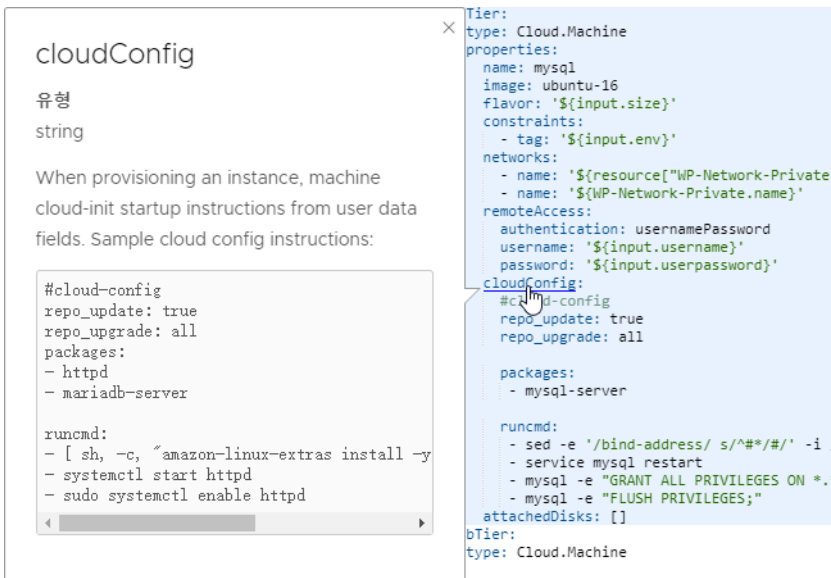
```

편집기 힌트 예

선택적 매개
변수



스키마 도움 말 모든 사용자 지정 속성에 대해서는 VMware (code) 사이트의 통합 리소스 스키마를 참조할 수도 있습니다.



간단한 vRealize Automation Cloud Assembly Blueprint를 향상시키는 방법

간단한 Blueprint를 한 차원 높일 수 있는 고급 vRealize Automation Cloud Assembly Blueprint 코드를 구현할 수 있습니다.

여기에 설명된 기술을 파악하려면 인프라 코드에 익숙해야 합니다. 다행히 vRealize Automation Cloud Assembly 코드는 사람이 읽을 수 있고 쉽게 따라 할 수 있습니다.

사용자 입력으로 vRealize Automation Cloud Assembly Blueprint를 사용자 지정하는 방법

Blueprint 개발자는 입력 매개 변수를 사용하여 사용자가 요청 시 사용자 지정 선택이 가능하도록 할 수 있습니다.

사용자가 입력을 제공할 때, 약간 다른 Blueprint 복사본을 여러 개 저장할 필요가 없습니다. 카탈로그 사용자가 배포를 업데이트할 때마다 새로운 입력을 요구하는 메시지가 표시되기 때문에 입력으로 2일차 작업에 대한 Blueprint를 준비할 수 있습니다.

경고 일부 속성을 변경하면 리소스가 다시 생성됩니다. 예를 들어 `Cloud.Service.Azure.App.Service` 아래에서 `connection_string.name`을 변경하면 기존 리소스가 삭제되고 새 리소스가 생성됩니다.

2일차 변경을 지원하는 입력을 설계할 때 리소스를 삭제하고 다시 생성하는 입력을 허용할지 여부를 결정합니다. 리소스를 다시 생성하는 속성에 대해 알아보려면 [VMware {code}의 vRealize Automation Blueprint 스키마](#)에서 통합된 리소스 스키마를 참조하십시오.

다음 입력은 MySQL 데이터베이스 서버에 대해 하나의 Blueprint를 생성하고, 이 Blueprint를 사용자가 여러 클라우드 리소스 환경에 배포하여 매번 다른 용량과 자격 증명을 적용할 수 있는 방법을 보여줍니다.

Environment	env.dev	▼ ⓘ
Database Tier Size *	small	▼ ⓘ
Database Username *	ouradmin	
Database Password *	•••••	
MySQL Data Disk Size	4	⬆️ ⓘ

Blueprint 입력 매개 변수를 정의하는 방법

선택 가능한 값을 설정하는 `inputs` 섹션을 Blueprint 코드에 추가합니다.

다음 예에서는 시스템 크기, 운영 체제 및 클러스터링된 서버의 수를 선택할 수 있습니다.

```
inputs:
  wp-size:
    type: string
    enum:
      - small
      - medium
    description: Size of Nodes
    title: Node Size
  wp-image:
    type: string
    enum:
      - coreos
```

```

- ubuntu
  title: Select Image/OS
wp-count:
  type: integer
  default: 2
  maximum: 5
  minimum: 2
  title: Wordpress Cluster Size
  description: Wordpress Cluster Size (Number of nodes)

```

코드 편집이 불편하면 코드 편집기 **입력** 탭을 클릭하고 여기에 설정을 입력할 수 있습니다. 다음 예는 앞에서 언급한 MySQL 데이터베이스에 대한 일부 입력을 보여줍니다.

The screenshot shows the 'Inputs' tab in the vRealize Automation Cloud Assembly interface. It displays a table of blueprint inputs and an 'Edit Blueprint Input' dialog for the 'size' input.

<input type="checkbox"/>	Name	Title	Type	Default Value
<input type="checkbox"/>	size	Tier Machine Size	string	
<input type="checkbox"/>	username	Database Username	string	
<input type="checkbox"/>	userpassword	Database Password	string	****
<input type="checkbox"/>	databaseDiskSize	MySQL Data Disk Size	number	4

Edit Blueprint Input: size

Name *

Title

Description

Type

Encrypted ☐

Blueprint 입력 매개 변수를 참조하는 방법

다음으로 resources 섹션에서 `${input.property-name}` 구문을 사용하여 입력 매개 변수를 참조합니다.

속성 이름에 공백이 포함되어 있는 경우 점 표기법을 사용하는 대신 대괄호와 큰따옴표로 구분합니다. `{input["속성 이름"]}`

중요 Blueprint 코드에서는 입력 매개 변수를 나타내는 경우를 제외하고는 `input`이라는 단어를 사용할 수 없습니다.

```
resources:
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      flavor: '${input.wp-size}'
      image: '${input.wp-image}'
      count: '${input.wp-count}'
```

입력 속성 목록

속성	설명
const	oneOf와 함께 사용됩니다. 친숙한 제목과 연결된 실제 값입니다.
default	입력에 대해 미리 채워진 값입니다. 기본값은 올바른 유형이어야 합니다. 정수의 기본값으로 단어를 입력하지 마십시오.
description	입력에 대한 사용자 도움말 텍스트입니다.
encrypted	사용자가 입력한 내용을 암호화할지 여부를 true 또는 false 로 지정합니다. 암호는 대개 암호화됩니다.
enum	허용되는 값의 드롭다운 메뉴입니다. 다음 예시를 형식 가이드로 사용하십시오. <pre>enum: - value 1 - value 2</pre>
format	입력에 필요한 형식을 설정합니다. 예를 들어 (25/04/19)는 날짜-시간을 지원합니다. vRealize Automation Service Broker 사용자 지정 양식에서 날짜 선택을 사용할 수 있습니다.
items	어레이 내의 항목을 선언합니다. 숫자, 정수, 문자열, 부울 또는 개체를 지원합니다.
maxItems	어레이 내에서 선택 가능한 최대 항목 수입니다.
maxLength	문자열에 허용되는 최대 문자 수입니다. 예를 들어 필드를 25자로 제한하려면 <code>maxLength: 25</code> 를 입력합니다.

속성	설명
maximum	숫자나 정수에 허용되는 최대값입니다.
minItems	어레이 내에서 선택 가능한 최소 항목 수입니다.
minLength	문자열에 허용되는 최소 문자 수입니다.
minimum	숫자나 정수에 허용되는 가장 작은 값입니다.
oneOf	사용자 입력 양식에서 친숙하지 않은 값(const)에 대한 친숙한 이름(title)을 표시할 수 있습니다. 기본값을 설정하는 경우 title 이 아니라 const를 설정합니다. 문자열, 정수 및 숫자 형식과 함께 사용할 수 있습니다.
pattern	정규식 구문에서 문자열 입력에 허용되는 문자입니다. 예: '[a-z]+' 또는 '[a-z0-9A-Z@#&]+'
속성	개체에 대한 key:value 속성 블록을 선언합니다.
readOnly	양식 레이블만 제공하는 데 사용됩니다.
title	oneOf와 함께 사용됩니다. const 값에 대한 친숙한 이름입니다. title은 배포 시 사용자 입력 양식에 표시됩니다.
type	숫자, 정수, 문자열, 부울 또는 개체의 데이터 유형입니다.
writeOnly	양식에서 별표 뒤에 키 입력을 숨깁니다. enum과 함께 사용할 수 없습니다. vRealize Automation Service Broker 사용자 지정 양식에서 암호 필드로 표시됩니다.

추가 예시

열거형이 포함된 문자열

```
image:
  type: string
  title: Operating System
  description: The operating system version to use.
  enum:
    - ubuntu 16.04
    - ubuntu 18.04
  default: ubuntu 16.04

shell:
  type: string
  title: Default shell
  description: The default shell that will be configured for the created user.
  enum:
    - /bin/bash
    - /bin/sh
```

최소 및 최대 정수

```
count:
  type: integer
  title: Machine Count
  description: The number of machines that you want to deploy.
  maximum: 5
  minimum: 1
  default: 1
```

개체 어레이

```
tags:
  type: array
  title: Tags
  description: Tags that you want applied to the machines.
  items:
    type: object
    properties:
      key:
        type: string
        title: Key
      value:
        type: string
        title: Value
```

친숙한 이름의 문자열

```
platform:
  type: string
  oneOf:
    - title: AWS
      const: platform:aws
    - title: Azure
      const: platform:azure
    - title: vSphere
      const: platform:vsphere
  default: platform:aws
```

패턴 유효성 검사가 포함된 문자열

```
username:
  type: string
  title: Username
  description: The name for the user that will be created when the machine is provisioned.
  pattern: ^[a-zA-Z]+$
```

문자열을 암호로

```
password:
  type: string
  title: Password
  description: The initial password that will be required to logon to the machine.
  Configured to reset on first login.
  writeOnly: true
```

문자열을 텍스트 영역으로

```
ssh_public_key:
  type: string
  title: SSH public key
  maxLength: 256
```

부울

```
public_ip:
  type: boolean
  title: Assign public IP address
  description: Choose whether your machine should be internet facing.
  default: false
```

vRealize Automation Cloud Assembly에서 구성 요소 배포 순서를 설정하는 방법

vRealize Automation Cloud Assembly Blueprint를 배포하는 경우 일부 구성 요소를 사용하기 위해서는 먼저 다른 구성 요소가 사용 가능해야 할 수 있습니다.

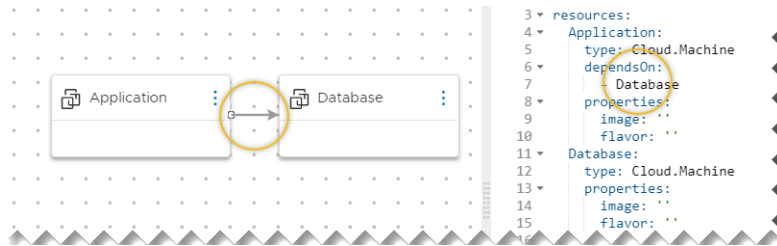
명시적 종속성을 생성하는 방법

때로는 다른 구성 요소가 먼저 배포되어야 하는 구성 요소가 있습니다. 예를 들어 애플리케이션 서버를 생성하여 데이터베이스에 액세스하도록 구성하려면 그 전에 먼저 데이터베이스 서버가 있어야 합니다.

명시적 종속성은 배포 시 빌드 순서를 설정하거나, 축소 또는 확장 작업에 대한 빌드 순서를 설정합니다. 그 래픽 설계 캔버스 또는 코드 편집기를 사용하여 명시적 종속성을 추가할 수 있습니다.

- 설계 캔버스 옵션—종속 구성 요소에서 시작하여 먼저 배포할 구성 요소에서 끝나는 연결을 그립니다.
- 코드 편집기 옵션—종속 구성 요소에 dependsOn 속성을 추가하고 먼저 배포할 구성 요소를 식별합니다.

명시적 종속성은 캔버스에 실선 화살표를 생성합니다.



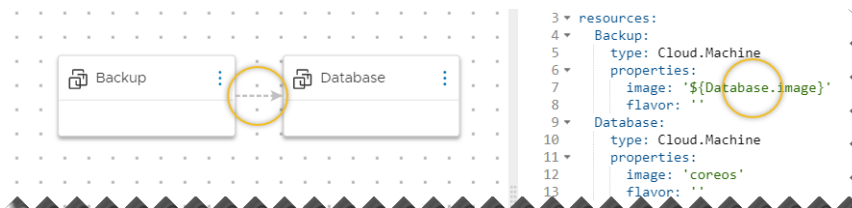
암시적 종속성 또는 속성 바인딩을 생성하는 방법

일부 구성 요소에는 다른 구성 요소의 속성에 있는 값이 필요한 경우가 있습니다. 예를 들어 백업 서버에는 백업 중인 데이터베이스 서버의 운영 체제 이미지가 필요할 수 있습니다. 따라서 데이터베이스 서버가 먼저 있어야 합니다.

속성 바인딩이라고도 하는 암시적 종속성은 종속 구성 요소를 배포하기 전에 필요한 속성을 사용할 수 있을 때까지 대기하여 빌드 순서를 제어합니다. 암시적 종속성은 코드 편집기를 사용하여 추가합니다.

- 종속 구성 요소를 편집하고 먼저 있어야 하는 구성 요소와 속성을 식별하는 속성을 추가합니다.

암시적 종속성 또는 속성 바인딩은 캔버스에 파선 화살표를 생성합니다.



표현식을 사용하여 vRealize Automation Cloud Assembly Blueprint 코드를 보다 다양하게 만드는 방법

유연성을 높이기 위해 vRealize Automation Cloud Assembly Blueprint 코드에 표현식을 추가할 수 있습니다.

표현식은 다음 예시와 같이 `${expression}` 구문을 사용합니다.

예시는 중요한 라인만 표시하도록 정리되어 있습니다. 편집되지 않은 전체 Blueprint는 끝에 표시되어 있습니다.

예

배포 시 사용자가 원격 액세스에 필요한 암호화된 키를 붙여 넣을 수 있도록 허용:

```

inputs:
  sshKey:
    type: string
    maxLength: 500
resources:
  frontend:
    type: Cloud.Machine

```

```
properties:
  remoteAccess:
    authentication: publicPrivateKey
    sshKey: '${input.sshKey}'
```

VMware Cloud on AWS에 배포하려면 폴더 이름을 "워크로드"의 필수 이름으로 설정합니다.

```
inputs:
  environment:
    type: string
    enum:
      - AWS
      - vSphere
      - Azure
      - VMC
      - GCP
    default: vSphere
  resources:
    frontend:
      type: Cloud.Machine
      properties:
        folderName: '${input.environment == "VMC" ? "Workload" : ""}'
```

배포시 선택한 환경과 일치하는 "env" (모두 소문자) 태그로 시스템에 태그를 지정합니다.

```
inputs:
  environment:
    type: string
    enum:
      - AWS
      - vSphere
      - Azure
      - VMC
      - GCP
    default: vSphere
  resources:
    frontend:
      type: Cloud.Machine
      properties:
        constraints:
          - tag: '${"env:" + to_lower(input.environment)}'
```

프런트 엔드 클러스터의 시스템 수를 1(소형) 또는 2(대형)로 설정합니다. 대규모 클러스터는 제거 프로세스에 의해 설정됩니다.

```
inputs:
  envsize:
    type: string
    enum:
      - Small
      - Large
  resources:
```

```
frontend:
  type: Cloud.Machine
  properties:
    count: '${input.envsize == "Small" ? 1 : 2}'
```

네트워크 리소스에 있는 속성에 바인딩하여 시스템을 동일한 "기본" 네트워크에 연결합니다.

```
resources:
  frontend:
    type: Cloud.Machine
    properties:
      networks:
        - name: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - name: '${resource.Cloud_Network_1.name}'
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      name: Default
      networkType: existing
```

API에 제출된 액세스 자격 증명 암호화:

```
resources:
  apitier:
    type: Cloud.Machine
    properties:
      cloudConfig: |
        #cloud-config
      runcmd:
        - export apikey=${base64_encode(input.username:input.password)}
        - curl -i -H 'Accept:application/json' -H 'Authorization:Basic :$apikey' http://
example.com
```

API 시스템의 주소를 확인합니다.

```
resources:
  frontend:
    type: Cloud.Machine
    properties:
      cloudConfig: |
        runcmd:
          - echo ${resource.apitier.networks[0].address}
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - name: '${resource.Cloud_Network_1.name}'
```

전체 Blueprint

```

inputs:
  environment:
    type: string
    enum:
      - AWS
      - vSphere
      - Azure
      - VMC
      - GCP
    default: vSphere
  sshKey:
    type: string
    maxLength: 500
  envsize:
    type: string
    enum:
      - Small
      - Large
resources:
  frontend:
    type: Cloud.Machine
    properties:
      folderName: '${input.environment == "VMC" ? "Workload" : ""}'
      image: ubuntu
      flavor: medium
      count: '${input.envsize == "Small" ? 1 : 2}'
      remoteAccess:
        authentication: publicPrivateKey
        sshKey: '${input.sshKey}'
      cloudConfig: |
        packages:
          - nginx
        runcmd:
          - echo ${resource.apitier.networks[0].address}
      constraints:
        - tag: '${"env:" + to_lower(input.environment)}'
      networks:
        - name: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      folderName: '${input.environment == "VMC" ? "Workload" : ""}'
      image: ubuntu
      flavor: small
      cloudConfig: |
        #cloud-config
        runcmd:
          - export apikey=$(base64_encode(input.username:input.password))
          - curl -i -H 'Accept:application/json' -H 'Authorization:Basic :$apikey' http://
example.com
      remoteAccess:
        authentication: publicPrivateKey
        sshKey: '${input.sshKey}'

```



```

constraints:
  - tag: '${"env:" + to_lower(input.environment)}'
networks:
  - name: '${resource.Cloud_Network_1.name}'
Cloud_Network_1:
  type: Cloud.Network
  properties:
    name: Default
    networkType: existing
  constraints:
    - tag: '${"env:" + to_lower(input.environment)}'

```

vRealize Automation Cloud Assembly Blueprint 표현식 구문

vRealize Automation Cloud Assembly Blueprint 표현식 구문은 표현식의 모든 기능을 노출합니다.

표현식을 사용하여 vRealize Automation Cloud Assembly Blueprint 코드를 보다 다양하게 만드는 방법에 표시된 예시에서 구문은 일부만 표시됩니다.

리터럴

다음과 같은 리터럴이 지원됩니다.

- 부울(true 또는 false)
- 정수
- 부동 소수점
- String

백슬래시는 큰따옴표, 작은따옴표 및 백슬래시 자체를 이스케이프합니다.

"는 \"로 이스케이프됩니다.

'는 \'로 이스케이프됩니다.

\는 \\로 이스케이프됩니다.

다음 예제와 같이 따옴표는 동일한 유형의 따옴표로 묶인 문자열 내에서만 이스케이프해야 합니다.

```
"I am a \"double quoted\" string inside \"double quotes\"."
```

- null

환경 변수

환경 이름:

- orgId
- projectId
- projectName
- deploymentId

- deploymentName
- blueprintId
- blueprintVersion
- blueprintName
- requestedBy(사용자)
- requestedAt(시간)

구문:

```
env.ENV_NAME
```

예:

```
${env.blueprintId}
```

리소스 변수

리소스 변수를 사용하면 다른 리소스의 리소스 속성에 바인딩할 수 있습니다.

구문:

```
resource.RESOURCE_NAME.PROPERTY_NAME
```

예:

- \${resource.db.id}
- \${resource.db.networks[0].address}
- \${resource.app.id}(count가 지정되지 않은 클러스터링되지 않은 리소스에 대해 문자열을 반환합니다. 클러스터링된 리소스에 대해서는 어레이를 반환합니다.)
- \${resource.app[0].id}(클러스터링된 리소스에 대한 첫 번째 항목을 반환합니다.)

리소스 자체 변수

리소스 자체 변수는 할당 단계를 지원하는 리소스에만 허용됩니다. 리소스 자체 변수는 할당 단계가 완료된 후에만 사용 가능하거나 값 집합을 가집니다.

구문:

```
self.property_name
```

예:

```
${self.address}(할당 단계 중 할당된 주소를 반환합니다.)
```

resource_x, self.property_name 및 resource.resource_x.property_name이라는 이름의 리소스는 동일하며, 둘 다 자체 참조로 고려됩니다.

클러스터 수 인덱스

구문:

count.index

예:

`${count.index == 0 ? "primary" : "secondary"}`(클러스터링된 리소스에 대한 노드 유형을 반환합니다.)

제한 사항:

리소스 할당에 대한 count.index 사용은 지원되지 않습니다. 예를 들어 입력 시간에 생성된 디스크의 어레이 내에서 위치를 참조할 때 다음 용량 표현식이 실패합니다.

```
inputs:
  disks:
    type: array
    minItems: 0
    maxItems: 12
    items:
      type: object
      properties:
        size:
          type: integer
          title: Size (GB)
          minSize: 1
          maxSize: 2048
resources:
  Cloud_vSphere_Disk_1:
    type: Cloud.vSphere.Disk
    properties:
      capacityGb: '${input.disks[count.index].size}'
      count: '${length(input.disks)}'
```

조건

구문:

- 같음 연산자는 == 및 !=입니다.
- 관계형 연산자는 < > <= 및 >=입니다.
- 논리 연산자는 && || 및 !입니다.
- 조건부 패턴 사용:

condition-expression ? true-expression : false-expression

예:

`${input.count < 5 && input.size == 'small'}`

`${input.count < 2 ? "small" : "large"}`

산술 연산자

구문:

연산자는 + - / * 및 %입니다.

예:

```
${(input.count + 5) * 2}
```

문자열 연결

구문:

```
${'ABC' + 'DEF'}는 ABCDEF로 평가됩니다.
```

[] 및 . 연산자

[] 및 . 연산자에 대한 처리를 통일하기 위해 표현식은 ECMAScript를 따릅니다.

따라서 `expr.identifier`는 `expr["identifier"]`에 해당합니다. 식별자는 값이 식별자인 리터럴을 구성하는 데 사용되며 [] 연산자가 해당 값과 함께 사용됩니다.

예:

```
${resource.app.networks[0].address}
```

또한 속성에 공백이 포함되어 있는 경우 점 표기법을 사용하는 대신 대괄호와 큰따옴표로 구분합니다.

틀림:

```
input.operating system
```

맞음:

```
input["operating system"]
```

맵 구성

구문:

```
${{'key1':'value1', 'key2':input.key2}}
```

어레이 구성

구문:

```
${['key1','key2']}
```

예:

```
${[1,2,3]}
```

함수

구문:

```
${ 함수(인수... ) }
```

예:

```
${to_lower(resource.app.name) }
```

표 6-1. 함수

함수	설명
abs(숫자)	절대 숫자 값
floor(숫자)	인수보다 작거나 같고 수학 정수와 동일한 최대(양의 무한대에 가장 가까움) 값을 반환
ceil(숫자)	인수보다 크거나 같고 수학 정수와 동일한 최소(음의 무한대에 가장 가까움) 값을 반환
to_lower(문자열)	문자열을 소문자로 변환
to_upper(문자열)	문자열을 대문자로 변환
contains(어레이, 값)	어레이에 값이 포함되어 있는지 확인
contains(문자열, 값)	문자열에 값이 포함되어 있는지 확인
join(어레이, 구분 기호)	구분 기호로 문자열 어레이를 연결하고 문자열을 반환
split(문자열, 분 기호)	구분 기호로 문자열을 분할하고 문자열 어레이를 반환
slice(어레이, 시작, 끝)	시작 인덱스에서 끝 인덱스까지 어레이 조각을 반환
reverse(어레이)	어레이 항목의 방향을 반전
starts_with(주체, 접두사)	주체 문자열이 접두사 문자열로 시작하는지 확인
ends_with(주체, 접미사)	주체 문자열이 접미사 문자열로 끝나는지 확인
replace(문자열, 대상, 교체)	대상 문자열을 포함하는 문자열을 대상 문자열로 교체
substring(문자열, 시작, 끝)	시작 인덱스에서 끝 인덱스까지 문자열의 하위 문자열을 반환
format(형식, 값 ...)	Java Class Formatter 형식 및 값을 사용하여 형식이 지정된 문자열을 반환
keys(맵)	맵의 키를 반환
values(맵)	맵의 값을 반환
merge(맵, 맵)	병합된 맵을 반환
length(문자열)	문자열 길이를 반환
length(어레이)	어레이 길이를 반환
max(어레이)	숫자 어레이에서 최대값을 반환
min(어레이)	숫자 어레이에서 최소값을 반환
sum(어레이)	숫자 어레이에서 모든 값의 합계를 반환
avg(어레이)	숫자 어레이에서 모든 값의 평균을 반환
digest(값, 유형)	지원되는 유형(md5, sha1, sha256, sha384, sha512)을 사용하여 값의 다이제스트를 반환

표 6-1. 함수 (계속)

함수	설명
to_string(값)	값의 문자열 표현을 반환
to_number(문자열)	문자열을 숫자로 구문 분석
not_null(어레이)	null이 아닌 첫 번째 항목을 반환
base64_encode(문자열)	base64로 인코딩된 값을 반환
base64_decode(문자열)	디코딩된 base64 값을 반환
now()	현재 시간을 ISO-8601형식으로 반환
uuid()	임의로 생성된 UUID를 반환
from_json(문자열)	json 문자열을 구문 분석
to_json(값)	값을 json 문자열로 직렬화
json_path(값, 경로)	XPath for JSON을 사용하여 값에 대해 경로를 평가
matches(문자열, 정규식)	문자열이 정규 표현식과 일치하는지 확인
url_encode(문자열)	URL 인코딩 규칙을 사용하여 문자열을 인코딩
trim(문자열)	앞뒤 공백 제거

vRealize Automation Cloud Assembly Blueprint에서 시스템을 자동으로 초기화하는 방법

vSphere 사용자 지정 규격을 통해 또는 명령을 직접 실행하여 vRealize Automation Cloud Assembly에서 시스템 초기화를 적용할 수 있습니다.

Blueprint 코드의 속성은 이름으로 vSphere 사용자 지정 규격을 참조할 수 있습니다. 또는 특정 명령을 삽입하는 Blueprint에 cloudConfig 섹션을 추가할 수 있습니다.

삽입된 명령과 사용자 지정 규격 초기화를 결합하려는 경우에는 주의해야 합니다. 공식적으로 호환되지 않으며 함께 사용하면 일관성이 없거나 원치 않는 결과가 발생할 수 있습니다.

사용자 지정 규격이 cloudConfig 섹션의 명령을 방해하는 방법에 대한 예는 [정적 IP 주소를 사용하여 Linux 시스템을 배포하는 방법](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly Blueprint의 vSphere 사용자 지정 규격

사용자 지정 규격을 사용하면 vSphere 기반 클라우드 영역에 배포하는 경우, 배포 시에 게스트 운영 체제 설정을 적용할 수 있습니다.

사용자 지정 규격은 배포 대상인 vSphere에 존재해야 합니다.

Blueprint 코드를 직접 편집합니다. 다음 예는 vSphere의 WordPress 호스트에 대한 cloud-assembly-linux 사용자 지정 규격을 가리킵니다.

```
resources:
  WebTier:
    type: Cloud.vSphere.Machine
    properties:
      name: wordpress
      cpuCount: 2
      totalMemoryMB: 1024
      imageRef: 'Template: ubuntu-18.04'
      customizationSpec: 'cloud-assembly-linux'
      resourceGroupName: '/Datacenters/Datacenter/vm/deployments'
```

사용자 지정 규격 및 초기화 명령

프로비저닝 환경이 현재 vSphere에서 수행 중인 작업과 일치되도록 하려면 사용자 지정 규격을 계속 사용하는 것이 가장 좋은 방법일 수 있습니다. 그러나 하이브리드 또는 다중 클라우드 프로비저닝으로 확장하기 위한 더 중립적인 접근 방식은 Blueprint의 cloudConfig 섹션에 초기화 명령을 삽입하는 것입니다.

Blueprint의 cloudConfig 섹션에 대한 자세한 내용은 [vRealize Automation Cloud Assembly Blueprint의 구성 명령](#)의 내용을 참조하십시오.

vRealize Automation Cloud Assembly Blueprint의 구성 명령

vRealize Automation Cloud Assembly Blueprint 코드에 cloudConfig 섹션을 추가하여, 배포 시 실행되는 시스템 초기화 명령을 추가할 수 있습니다.

- Linux—초기화 명령은 개방형 [cloud-init](#) 표준을 따릅니다.
- Windows—초기화 명령은 [Cloudbase-init](#)를 사용합니다.

참고 Linux [cloud-init](#) 및 Windows [Cloudbase-init](#)는 동일한 구문을 공유하지 않습니다. 한 운영 체제의 cloudConfig 섹션이 다른 운영 체제의 시스템 이미지에서 작동하지 않습니다.

초기화 명령을 사용하여 인스턴스 생성 시 데이터 또는 설정 적용을 자동화하면 사용자, 권한, 설치 또는 기타 명령 기반 작업을 사용자 지정할 수 있습니다. 예는 다음과 같습니다.

- 호스트 이름 설정
- SSH 개인 키 생성 및 설정
- 패키지 설치

vRealize Automation Cloud Assembly에서는 인프라를 구성할 때 시스템 이미지에 초기화 명령을 추가할 수도 있습니다. 소스 이미지를 참조하는 모든 **Blueprint**는 동일한 초기화를 가져옵니다.

중요 이미지 맵과 **Blueprint** 둘 다에 초기화 명령이 포함되어 있을 수 있습니다. 배포 시 명령이 병합되고, vRealize Automation Cloud Assembly가 통합된 명령을 실행합니다.

동일한 명령이 두 곳 모두에 있고 매개 변수만 다른 경우에는 이미지 맵 명령만 실행됩니다.

자세한 내용은 [vRealize Automation Cloud Assembly의 이미지 매핑에 대해 알아보기](#)의 내용을 참조하십시오.

다음은 Linux 기반 MySQL 서버에 대한 **WordPress 사용 사례: 기본 Blueprint 생성** Blueprint 코드에서 가져온 **cloudConfig** 섹션 예시입니다.

명령이 제대로 해석되도록 하려면 아래와 같이 파이프 문자(**cloudConfig: |**)를 항상 포함해야 합니다.

cloud-init 스크립트가 예기치 않게 작동하는 경우 문제를 해결할 때 **/var/log/cloud-init-output.log**에서 캡처된 콘솔 출력을 확인하십시오. **cloud-init**에 대한 자세한 내용은 [cloud-init 설명서](#)를 참조하십시오.

```
cloudConfig: |
#cloud-config
repo_update: true
repo_upgrade: all
packages:
- apache2
- php
- php-mysql
- libapache2-mod-php
- php-mcrypt
- mysql-client
runcmd:
- mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
- i=0; while [ $i -le 5 ]; do mysql --connect-timeout=3 -h $
{DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
- mysql -u root -pmysqlpassword -h ${DBTier.networks[0].address} -e "create database
wordpress_blog;"
- mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
- sed -i -e s/"define( 'DB_NAME', 'database_name_here' );"/"define( 'DB_NAME',
'wordpress_blog' );"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define( 'DB_USER', 'username_here' );"/"define( 'DB_USER', 'root' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_PASSWORD',
'password_here' );"/"define( 'DB_PASSWORD', 'mysqlpassword' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_HOST',
'localhost' );"/"define( 'DB_HOST', '${DBTier.networks[0].address}' );"/ /var/www/html/
mywordpresssite/wp-config.php
- service apache2 reload
```


정적 IP 주소를 사용하여 Linux 시스템을 배포하는 방법

vSphere에 배포할 때 정적 IP 주소를 사용하려면 vRealize Automation Cloud Assembly가 vSphere 사용자 지정 규격을 생성해야 하지만 이는 cloud-init 명령을 방해할 수 있습니다.

문제

- vRealize Automation Cloud Assembly Blueprint에는 vSphere 가상 시스템에 정적 IP 주소를 적용하기 위한 `assignment: static`이 포함되어 있습니다.
- 또한 Blueprint에는 cloud-init를 사용하여 실행되는 초기화 명령을 포함하는 `cloudConfig` 섹션도 포함되어 있습니다.
- 가상 시스템에 정적 IP를 제공하기 위해 vRealize Automation Cloud Assembly는 적용할 vSphere 사용자 지정 규격을 동적으로 생성합니다.
- 사용자 지정 규격을 적용할 때마다 마지막 작업이 가상 시스템을 다시 시작합니다.
- 사용자 지정 규격에서는 cloud-init 명령이 실행 중임을 인식하지 못하므로 다시 시작으로 인해 명령이 중단됩니다.
- cloud-init 명령은 처음 부팅할 때만 실행되고 중단되면 자동으로 복구되지 않습니다.
- 따라서 가상 시스템은 일부만 구성된 상태로 유지됩니다.

해결 방법

지정된 시간 동안 cloud-init를 비활성화하는 시스템 템플릿을 생성합니다. 그런 다음 사용자 지정 규격 및 다시 시작이 cloud-init 전에 발생할 수 있도록 템플릿을 기반으로 시스템을 배포합니다.

절차 예 - Ubuntu 18.04

다음 단계는 Ubuntu 18.04에 적용됩니다. 다른 Linux 버전 또는 오퍼링에 대해서는 여기에 제시된 내용을 조정하여 적용해야 할 수 있습니다.

- 1 가상 시스템을 생성하고 원하는 버전 업데이트 및 패키지를 사용하여 최신 상태로 설정합니다.
 다른 Linux 오퍼링에는 cloud-init가 미리 설치되어 있지 않을 수 있지만 Ubuntu 18.04에는 미리 설치되어 있습니다.
- 2 cloud-init를 재구성하여 데이터 소스를 OVF로 설정합니다.

```
sudo dpkg-reconfigure cloud-init
```
- 3 다음 파일을 편집합니다.

```
/etc/cloud/cloud.cfg
```

 - a 다음 줄을 추가하여 기존 GOSC(게스트 운영 체제 사용자 지정)를 사용하도록 설정합니다.

```
disable_vmware_customization: true
```

- b 네트워크 구성이 사용되도록 설정되어 있는지 확인합니다. 사용 안 함 설정이 있으면 삭제하거나 주석 처리합니다.

```
network:
  # config: disabled
```

또는 다음 디렉토리에 있는 모든 구성 파일을 검사합니다.

```
/etc/cloud/cloud.cfg.d/*
```

network: {config: disabled} 설정이 포함된 파일을 모두 삭제합니다.

- 4 다음 파일을 편집합니다.

```
/usr/lib/tmpfiles.d/tmp.conf
```

- 설정을 주석 처리하여 임시 디렉토리가 지워지지 않도록 합니다.

```
# D /tmp 1777 root root -
```

- 5 다음 파일을 편집합니다.

```
/lib/systemd/system/open-vmtools.service
```

- 다음 줄을 [Unit] 섹션 아래에 추가하여 open-vmtools가 dbus.service 후에 시작되도록 구성합니다.

```
After=dbus.service
```

- 6 cloud-init를 사용하지 않도록 설정하는 비어 있는 새 파일을 생성합니다.

```
sudo touch /etc/cloud/cloud-init.disabled
```

- 7 re_init.sh 스크립트를 생성합니다. 사용자 지정 규격에 대해 일시 중지되는 cron 작업 지연 후 스크립트는 cloud-init를 다시 사용하도록 설정하고 초기화합니다.

```
sudo rm -rf /etc/cloud/cloud-init.disabled
sudo cloud-init init
sleep 20
sudo cloud-init modules --mode config
sleep 20
sudo cloud-init modules --mode final
```

- 8 스크립트에 대한 실행 권한을 추가합니다.

```
sudo chmod +x re_init.sh
```

- 9 시작 시 90초의 대기 후 실행되는 cron 작업을 생성합니다. crontab -e를 입력하고 다음을 입력합니다.

```
@reboot ( sleep 90 ; sh /script_path/delay_init.sh )
```

사용자 지정 규격 및 다시 시작이 완료되는 데 시간이 더 오래 걸린다면 90초를 초과하여 적용할 수 있습니다.

- 10 템플릿을 정리하는 `cleaner.sh` 스크립트를 생성합니다. `cloudadmin`을 운영 체제 설치 중에 설정한 사용자로 바꿉니다.

샘플 스크립트는 Ubuntu에만 적용됩니다. 다른 Linux 오퍼링을 위한 스크립트를 생성하려면 강조 표시된 필수 섹션을 포함해야 합니다.

```
#!/bin/bash

# Add usernames to add to /etc/sudoers for passwordless sudo
users=("ubuntu" "cloudadmin")

for user in "${users[@]}"
do
cat /etc/sudoers | grep ^$user
RC=$?
if [ $RC != 0 ]; then
bash -c "echo \"$user ALL=(ALL) NOPASSWD:ALL\" >> /etc/sudoers"
fi
done

#grab Ubuntu Codename
codename="$(lsb_release -c | awk {'print $2'})"

#Stop services for cleanup
service rsyslog stop

#clear audit logs
if [ -f /var/log/audit/audit.log ]; then
cat /dev/null > /var/log/audit/audit.log
fi
if [ -f /var/log/wtmp ]; then
cat /dev/null > /var/log/wtmp
fi
if [ -f /var/log/lastlog ]; then
cat /dev/null > /var/log/lastlog
fi

#cleanup persistent udev rules
if [ -f /etc/udev/rules.d/70-persistent-net.rules ]; then
rm /etc/udev/rules.d/70-persistent-net.rules
fi

#cleanup /tmp directories
rm -rf /tmp/*
rm -rf /var/tmp/*

#cleanup current ssh keys
#rm -f /etc/ssh/ssh_host_*

#cat /dev/null > /etc/hostname

#cleanup apt
apt-get clean
```

```
#Clean Machine ID

truncate -s 0 /etc/machine-id
rm /var/lib/dbus/machine-id
ln -s /etc/machine-id /var/lib/dbus/machine-id

#Clean Cloud-init
cloud-init clean --logs --seed

#cleanup shell history
history -w
history -c
```

- 11 템플릿 정리 스크립트에 대한 실행 권한을 추가합니다.

```
sudo chmod +x cleaner.sh
```

- 12 Ubuntu 18.04에서 정리 스크립트를 사용하려면 루트 권한이 필요합니다. 다음 파일을 편집합니다.

```
/etc/ssh/sshd_config
```

- a 루트 사용자로 전환할 수 있는지 확인합니다.

```
PermitRootLogin yes
```

- b 루트에 대한 암호를 설정합니다.

```
sudo passwd root
```

- 13 정리 스크립트를 실행합니다.

```
sudo ./script_path/cleaner.sh
```

- 14 (선택 사항) 보안 목적으로 추가적인 루트 로그인을 방지하려면 12단계를 되돌립니다.

- 15 가상 시스템을 종료하고 vSphere을 사용하여 템플릿으로 변환합니다.

템플릿 업데이트

cron 작업은 템플릿을 업데이트할 때마다 실행됩니다. 업데이트에 걸리는 시간이 지연 시간(예: 90초)을 초과하는 경우 템플릿을 종료하기 전에 /etc/cloud/cloud-init.disabled 파일을 다시 추가하고 정리 스크립트를 다시 실행해야 합니다. 그렇지 않으면 처음 부팅할 때 cloud-init가 사용되지 않도록 설정되지 않고 사용자 지정 규격을 다시 시작하면 다시 cloud-init 명령이 중단됩니다.

문제 해결

vSphere 사용자 지정 규격이 cloud-init 완료를 방해하는 것으로 의심되는 경우에는 사용자 지정 규격을 일시적으로 사용하지 않도록 설정하고 cloud-init가 예상 대로 완료될 수 있는지 확인합니다. 사용자 지정 규격을 일시적으로 사용하지 않도록 설정하려면 `customizeGuestOS: false` 속성을 사용합니다.

```
properties:
  image: ubuntu
  cpuCount: 1
  totalMemoryMB: 8192
  customizeGuestOS: false
```

vRealize Automation Cloud Assembly 배포를 초기화 대기 상태로 만드는 방법

vRealize Automation Cloud Assembly 배포를 진행하기 전에 가상 시스템이 완전히 시작되어야 하는 경우가 있습니다.

예를 들어 아직 패키지를 설치 중인 시스템을 배포하고 웹 서버를 시작하면, 급한 사용자가 애플리케이션 사용이 가능해지기 전에 연결을 시도하는 상황이 발생할 수 있습니다.

이 기능을 사용할 때는 다음과 같은 고려 사항에 유의해야 합니다.

- 이 기능은 `cloud-init phone_home` 모듈을 사용하며 Linux 시스템을 배포할 때 사용할 수 있습니다.
- `phone_home`은 `Cloudbase-init` 제한 때문에 Windows에서 사용할 수 없습니다.
- `phone_home`은 명시적 종속성과 같은 배포 순서에 영향을 줄 수 있지만 타이밍 및 처리 옵션에 대한 유연성이 더 높습니다.

[vRealize Automation Cloud Assembly에서 구성 요소 배포 순서를 설정하는 방법](#) 항목을 참조하십시오.

- `phone_home`을 사용하려면 Blueprint에 `cloudConfig` 명령이 필요합니다.
- 사용자의 창의력이 한 가지 요소입니다. `cloudConfig` 섹션에는 작업 간에 내장된 대기 시간을 포함할 수 있으며, 이것을 `phone_home`과 공동으로 사용할 수 있습니다.
- 시스템 템플릿에 `phone_home` 모듈 설정이 이미 포함된 경우에는 Blueprint 기반 `phone_home`이 작동하지 않습니다.
- 시스템에서 vRealize Automation Cloud Assembly로 다시 아웃바운드 통신 액세스가 가능해야 합니다.

vRealize Automation Cloud Assembly에서 `phone_home`을 사용하여 시스템 초기화를 대기하려면 Blueprint에 `cloudConfigSettings` 섹션을 추가합니다.

```
cloudConfigSettings:
  phoneHomeShouldWait: true
  phoneHomeTimeoutSeconds: 600
  phoneHomeFailOnTimeout: true
```

속성	설명
phoneHomeShouldWait	초기화를 대기할지 여부이며, true 또는 false입니다.
phoneHomeTimeoutSeconds	초기화가 아직 실행 중이어도 배포를 계속할지 여부를 결정하는 시기입니다. 기본값은 10분입니다.
phoneHomeFailOnTimeout	시간 초과 후 배포를 계속할지 여부이며, true 또는 false입니다. 계속 진행하더라도 별도의 이유로 인해 배포가 계속 실패할 수 있습니다.

vRealize Automation Cloud Assembly Blueprint에서 원격 액세스를 사용하도록 설정하는 방법

vRealize Automation Cloud Assembly가 배포된 시스템에 원격으로 액세스하려면 배포 전에 해당 시스템의 Blueprint에 속성을 추가합니다.

원격 액세스의 경우 다음 인증 옵션 중 하나를 구성할 수 있습니다.

참고 키를 복사해야 하는 경우에는 프로비저닝 시 키를 자동으로 복사하기 위해 Blueprint에 cloudConfig 섹션을 생성할 수도 있습니다. 구체적인 내용은 여기에 설명되어 있지 않지만 [vRealize Automation Cloud Assembly Blueprint에서 시스템을 자동으로 초기화하는 방법](#)에 cloudConfig에 대한 일반 정보가 제공됩니다.

vRealize Automation Cloud Assembly 프로비저닝 시 키 쌍 생성

원격 액세스 인증을 위한 자체적인 공용-개인 키 쌍이 없는 경우 vRealize Automation Cloud Assembly에서 키 쌍을 생성할 수 있습니다.

다음 코드를 지침으로 사용합니다.

- 1 vRealize Automation Cloud Assembly에서 프로비저닝하기 전에 예시에 표시된 대로 Blueprint에 remoteAccess 속성을 추가합니다.

사용자 이름은 선택 사항입니다. 이 옵션을 생략하면 시스템에서 임의의 ID가 사용자 이름으로 생성됩니다.

예:

```
type: Cloud.Machine
properties:
  name: our-vm2
  image: Linux18
  flavor: small
  remoteAccess: authentication: generatedPublicPrivatekey username: testuser
```

- 2 vRealize Automation Cloud Assembly에서 시스템을 해당 Blueprint에서 프로비저닝하고 이를 시작됨 상태로 전환합니다.

프로비저닝 프로세스를 통해 키가 생성됩니다.

3 **배포 > 토폴로지** 속성에서 키 이름을 찾습니다.

4 vSphere 클라이언트와 같은 클라우드 제공자 인터페이스를 사용하여 프로비저닝된 시스템 명령줄에 액세스합니다.

5 개인 키에 읽기 권한을 부여합니다.

```
chmod 600 키-이름
```

6 vRealize Automation Cloud Assembly 배포로 이동하여 시스템을 선택한 후 **작업 > 개인 키 얻기**를 클릭합니다.

7 개인 키 파일을 로컬 시스템에 복사합니다.

일반적인 로컬 파일 경로는 `/home/username/.ssh/키-이름`입니다.

8 원격 SSH 세션을 열고 프로비저닝된 시스템에 연결합니다.

```
ssh -i 키-이름 사용자-이름@시스템-ip
```

vRealize Automation Cloud Assembly에 자체 공용-개인 키 쌍 제공

많은 기업이 인증을 위해 자체 공용-개인 키 쌍을 생성하고 배포합니다.

다음 코드를 지침으로 사용합니다.

1 로컬 환경에서 공개-개인 키 쌍을 얻거나 생성합니다.

지금은 로컬에서 키를 생성하고 저장합니다.

2 vRealize Automation Cloud Assembly에서 프로비저닝하기 전에 예시에 표시된 대로 Blueprint에 `remoteAccess` 속성을 추가합니다.

`sshKey`에는 공개 키 파일 `key-name.pub`에서 발견된 긴 영숫자가 포함됩니다.

사용자 이름은 선택 사항이며 로그인에 사용할 수 있도록 생성됩니다. 이 옵션을 생략하면 시스템에서 임의의 ID가 사용자 이름으로 생성됩니다.

예:

```
type: Cloud.Machine
properties:
  name: our-vm1
  image: Linux18
  flavor: small
  remoteAccess:
    authentication: publicPrivateKey
    sshKey: ssh-rsa Iq+5aQgBP3ZNT4o1baP5Ii+dstIcowRRkyobbfpA1mj9ts1f
    qGxvU66PX9IeZax5hZvNWFgjw6ag+Z1zndOLhVdVoW49f274/mIRId7UW...
    username: testuser
```

3 vRealize Automation Cloud Assembly에서 시스템을 해당 Blueprint에서 프로비저닝하고 이를 시작됨 상태로 전환합니다.

4 클라우드 벤더 클라이언트를 사용하여 프로비저닝된 시스템에 액세스합니다.

- 5 시스템의 home 폴더에 공용 키 파일을 추가합니다. `remoteAccess.sshKey`에 지정한 키를 사용합니다.

- 6 개인 키 파일 해당 항목이 로컬 시스템에 있는지 확인합니다.

이 키는 일반적으로 `.pub` 확장명이 없는 `/home/username/.ssh/key-name`입니다.

- 7 원격 SSH 세션을 열고 프로비저닝된 시스템에 연결합니다.

```
ssh -i 키-이름 사용자-이름@시스템-ip
```

vRealize Automation Cloud Assembly에 AWS 키 쌍 제공

AWS 키 쌍 이름을 Blueprint에 추가하면 vRealize Automation Cloud Assembly가 AWS에 배포하는 시스템에 원격으로 액세스할 수 있습니다.

AWS 키 쌍은 지역별로 다르다는 점에 유의하십시오. 워크로드를 `us-east-1`로 프로비저닝하는 경우, `us-east-1`에 키 쌍이 존재해야 합니다.

다음 코드를 지침으로 사용합니다. 이 옵션은 AWS 클라우드 영역에만 적용됩니다.

```
type: Cloud.Machine
properties:
  image: Ubuntu
  flavor: small
  remoteAccess: authentication: keyPairName keyPair: cas-test
constraints:
  - tag: 'cloud:aws'
```

vRealize Automation Cloud Assembly에 사용자 이름 및 암호 제공

Blueprint에 사용자 이름과 암호를 추가하여 vRealize Automation Cloud Assembly가 배포하는 시스템에 간단하게 원격 액세스할 수 있습니다.

보안 수준은 떨어지지만 사용자 이름과 암호를 사용하여 원격으로 로그인하는 것이 상황에 따라 필요할 수 있습니다. 일부 클라우드 벤더 또는 구성에서는 이 보안 수준이 낮은 옵션을 지원하지 않을 수 있습니다.

- 1 vRealize Automation Cloud Assembly에서 프로비저닝하기 전에 예시에 표시된 대로 Blueprint에 `remoteAccess` 속성을 추가합니다.

로그인하는 데 사용할 계정으로 사용자 이름 및 암호를 설정합니다.

예:

```
type: Cloud.Machine
properties:
  name: our-vm3
  image: Linux18
  flavor: small
  remoteAccess: authentication: usernamePassword username: testuser password: admin123
```

- 2 vRealize Automation Cloud Assembly에서 시스템을 해당 Blueprint에서 프로비저닝하고 이를 시작된 상태로 전환합니다.

- 클라우드 벤더의 인터페이스로 이동하여 프로비저닝된 시스템에 액세스합니다.
- 프로비저닝된 시스템에서 계정을 생성하거나 사용하도록 설정합니다.
- 로컬 시스템에서 프로비저닝된 시스템 IP 주소 또는 FQDN에 대한 원격 세션을 열고 평소와 같이 사용자 이름과 암호를 사용하여 로그인합니다.

여러 버전의 vRealize Automation Cloud Assembly Blueprint를 저장하는 방법

Blueprint 개발자는 위험한 추가 변경을 수행하기 전에 제대로 작동하는 Blueprint의 스냅샷을 안전하게 캡처할 수 있습니다.

배포 시, 배포할 버전을 선택할 수 있습니다.

Blueprint 버전을 캡처하는 방법

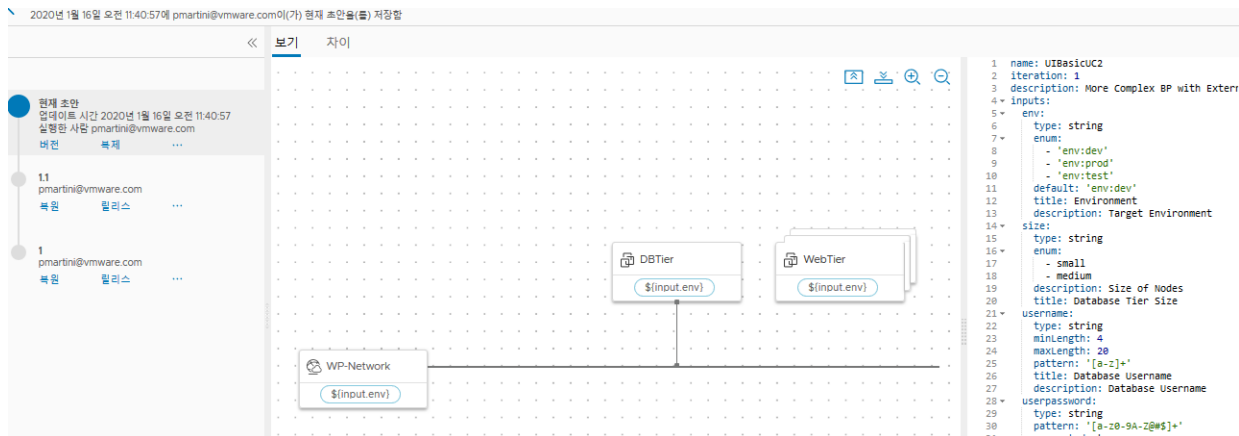
설계 페이지에서 **버전**을 클릭하고 이름을 제공합니다.

이름은 영숫자와 특수 문자(마침표, 하이픈 및 밑줄만 허용)를 사용해야 하고 공백은 사용할 수 없습니다.

이전 버전을 복원하는 방법

설계 페이지에서 **버전 기록**을 클릭합니다.

왼쪽에서 이전 버전을 선택하여 캔버스 및 코드 편집기에서 검사합니다. 원하는 버전을 찾으면 **복원**을 클릭합니다. 복원을 수행하면 명명된 버전을 제거하지 않고 현재 초안을 덮어씹습니다.



vRealize Automation Service Broker 사용자에게 버전을 릴리스하는 방법

설계 페이지에서 **버전 기록**을 클릭합니다.

왼쪽에서 버전을 선택하고 **릴리스**를 클릭합니다. 현재 초안은 버전으로 저장하기 전에는 릴리스할 수 없습니다.

Blueprint 버전이 둘 이상 릴리스되면 vRealize Automation Service Broker에서 최신 버전이 사용됩니다.

Blueprint 버전을 비교하는 방법

변경 내용과 버전이 누적되면 그 차이를 식별하는 것이 좋습니다.

[버전 기록] 보기에서 버전을 선택하고 **차이**를 클릭합니다. 그런 다음, **비교 대상** 드롭다운에서 비교할 다른 버전을 선택합니다.

코드 차이 검토 또는 시각적 토폴로지 차이 검토 사이를 전환할 수 있습니다.

그림 6-1. 코드 차이

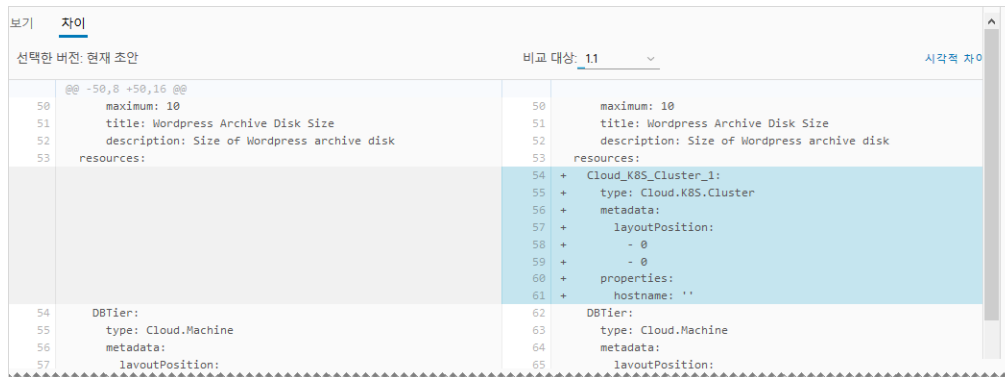
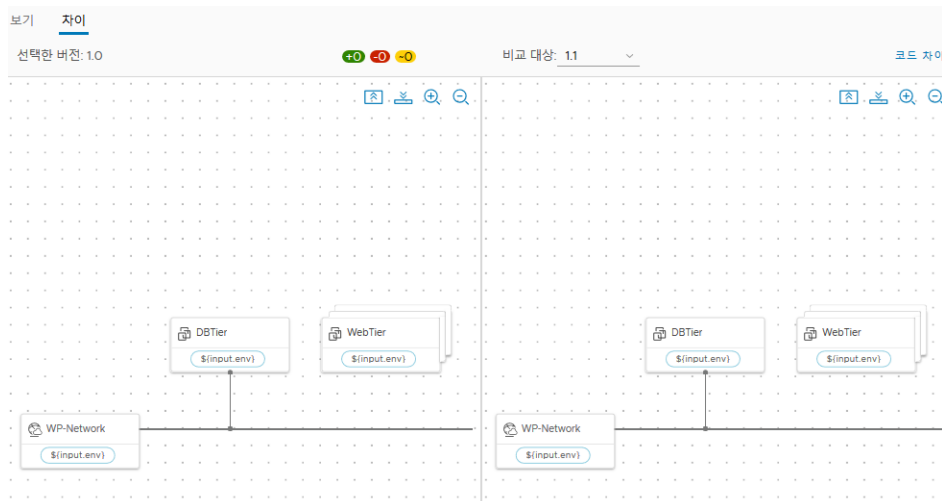


그림 6-2. 시각적 토폴로지 차이



Blueprint를 복제하는 방법

버전을 저장하는 것과 같지는 않지만 설계 페이지에서 **작업 > 복제**를 선택하면 다른 개발을 위해 현재 Blueprint의 복사본이 만들어집니다.

vRealize Automation Cloud Assembly를 사용하여 배포된 리소스의 이름을 사용자 지정하는 방법

클라우드 또는 프로젝트 관리자는 환경의 리소스에 대해 규정된 이름 지정 규칙을 사용하며, 배포된 리소스가 사용자 상호 작용 없이 해당 규칙을 따르도록 설정할 수 있습니다. vRealize Automation Cloud Assembly 프로젝트에서 모든 배포에 대한 이름 지정 템플릿을 생성할 수 있습니다.

예를 들어 호스트 이름 지정 규칙은 리소스에 *projectname-sitecode-costcenter-whereDeployed-identifier*를 접두사로 지정하는 것입니다. 각 프로젝트의 시스템에 대한 사용자에게 맞게 이름 지정 템플릿을 구성합니다. 템플릿 변수 중 일부는 배포될 때 시스템에서 가져오고, 다른 변수는 프로젝트 사용자 지정 속성에 기반합니다.

모든 리소스 이름은 고유해야 합니다. 증분 숫자 속성을 사용하면 고유성을 보장할 수 있습니다. vRealize Automation Cloud Assembly에서 이름이 지정된 배포를 포함하여 모든 배포에 대해 숫자가 증가합니다. 시스템이 더욱 견고해지면서 번호 매기기가 무작위로 표시될 수 있지만 고유성은 여전히 보장됩니다.

여기에 제공된 예제 외에도 사용자 이름, 사용되는 이미지, 기타 기본 제공 옵션 및 간단한 문자열을 추가할 수도 있습니다. 템플릿을 구축할 때 가능한 옵션에 관한 힌트가 제공됩니다.

여기에 나와 있는 일부 값은 사용 사례 예시일 뿐입니다. 사용자 환경에서 이러한 값을 그대로 사용해서는 안 됩니다. 클라우드 인프라 및 배포 관리 요구 사항에 맞게 고유한 값으로 대체하거나 예시의 값에서 추론할 수 있는 부분에 대해 생각해 보십시오.

사전 요구 사항

- 프로젝트에서 배포에 사용할 이름 지정 규칙을 알고 있는지 확인합니다.
- 이 절차에서는 사용자 지정 호스트 접두사 이름 지정을 테스트하는 데 사용할 간단한 Blueprint가 있거나 생성할 수 있다고 가정합니다.

절차

- 1 **인프라 > 프로젝트**를 선택합니다.
- 2 기존 프로젝트를 선택하거나 새로 생성합니다.
- 3 **프로비저닝** 탭에서 [사용자 지정 속성] 섹션을 찾아서 사이트 코드 및 비용 센터 값에 대한 속성을 생성합니다.

여기에 표시된 값을 사용자 환경과 관련된 값으로 바꿀 수 있습니다.

사용자 지정 속성
이 프로젝트의 모든 요청에 추가해야 하는 사용자 지정 속성을 지정합니다. ①

사용자 지정 속성 정의	이름	값
	siteCode	BGL
	costCenter	IT-research

사용자 지정 명명
이 프로젝트에서 프로비저닝 시스템, 네트워크, 보안 그룹 및 디스크에 사용할 이름 지정 템플릿을 지정합니다.

템플릿 `${project.name}-${resource.siteCode}-${resource` ①
Hint: Avoid conflicting names by generating digits in names. `#{#####}`

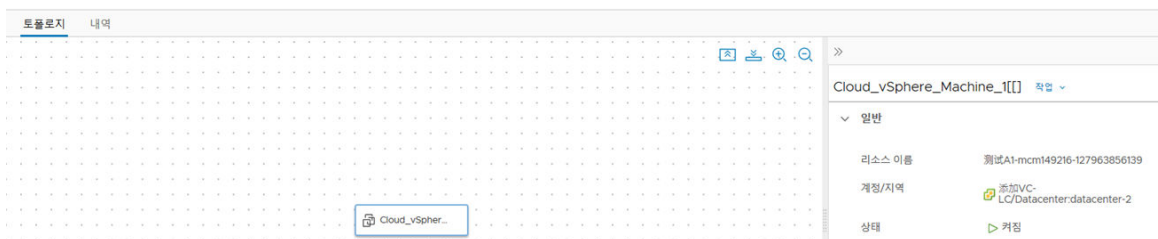
- a 이름이 **siteCode**이고 값이 **BGL**인 사용자 지정 속성을 생성합니다.
 - b 이름이 **costCenter**이고 값이 **IT-research**인 또 다른 사용자 지정 속성을 추가합니다.
- 4 [사용자에 맞게 이름 지정] 섹션을 찾아서 다음 템플릿을 추가합니다.

```
${project.name}-${resource.siteCode}-${resource.costCenter}-${endpoint.name}-${#####}
```

문자열로 복사할 수 있지만, 첫 번째 이름 지정 템플릿인 경우에는 템플릿을 빌드할 때 힌트 텍스트와 빠른 선택을 사용하는 것이 좋습니다.

- 5 프로젝트와 연결된 Blueprint를 배포하여 사용자 지정 이름이 리소스에 적용 되었는지 확인합니다.
 - a **Blueprint** 탭을 클릭한 다음 프로젝트와 연결된 Blueprint를 클릭합니다.
 - b Blueprint를 배포합니다.

배포 탭이 열리고 진행 중인 배포가 표시됩니다.
 - c 배포가 완료되면 배포 이름을 클릭합니다.
 - d **토폴로지** 탭에서 사용자 지정 이름이 오른쪽 창에 있는 리소스 이름인지 확인합니다.



- 6 이름 지정 규칙을 확인하기 위해 테스트 Blueprint를 배포한 경우에는 배포를 삭제할 수 있습니다.

다음에 수행할 작업

다른 프로젝트에 대한 사용자에게 맞게 이름 지정 템플릿을 생성합니다.

vRealize Automation 리소스 속성 소개

vRealize Automation IaC(Infrastructure-as-Code) 편집기에서 마우스로 클릭하거나 가리켜서 구문 및 코드 완성에 필요한 도움을 받을 수 있습니다. 종종 사용자 지정 속성이라고도 하는 전체 Blueprint 리소스 속성 집합을 보려면 통합된 리소스 스키마를 참조하십시오.

이 스키마는 VMware {code} 사이트에서 사용할 수 있습니다. 링크를 따르고 **모델**을 클릭하여 Blueprint에 사용할 수 있는 리소스 개체를 나열합니다.

- [VMware {code}에 대한 vRealize Automation 리소스 유형 스키마](#)

일부 Blueprint 코드 예시

vRealize Automation Cloud Assembly의 Blueprint 코드는 조합 및 응용 가능성에 제한이 거의 없습니다.

종종 성공적인 코드 예시는 더 많은 개발을 위한 최고의 출발점이 됩니다. 예시를 따를 때에는 리소스 이름, 값 등의 측면에서 사용자 사이트 설정을 적용하기 위해 대체가 필요합니다.

vRealize Automation Cloud Assembly Blueprint의 vSphere 시스템 예

이러한 기본 예시는 vRealize Automation Cloud Assembly Blueprint 내에서 vSphere 리소스를 정의합니다.

리소스	Blueprint 예시
CPU, 메모리 및 운영 체제가 포함된 vSphere 가상 시스템	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine cpuCount: 1 totalMemoryMB: 1024 image: ubuntu</pre>
데이터스토어 리소스가 포함된 vSphere 시스템	<pre>resources: demo-vsphere-disk-001: type: Cloud.vSphere.Disk properties: name: DISK_001 type: 'HDD' capacityGb: 10 dataStore: 'datastore-01' provisioningType: thick</pre>

리소스	Blueprint 예시
연결된 디스크가 포함된 vSphere 시스템	<pre> resources: demo-vsphere-disk-001: type: Cloud.vSphere.Disk properties: name: DISK_001 type: HDD capacityGb: 10 dataStore: 'datastore-01' provisioningType: thin demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine cpuCount: 2 totalMemoryMB: 2048 imageRef: >- https://bintray.com/vmware/photon/ download_file?file_path=2.0%2FRC%2Fova%2Fphoton- custom-hw11-2.0-31bb961.ova attachedDisks: - source: '\${demo-vsphere-disk-001.id}' </pre>
연결된 복제 이미지의 vSphere 시스템(슬래시 및 스냅샷 이름 추가)	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: imageRef: 'demo-machine/snapshot-01' cpuCount: 1 totalMemoryMB: 1024 </pre>
vCenter의 특정 폴더에 있는 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine cpuCount: 2 totalMemoryMB: 1024 imageRef: ubuntu resourceGroupName: 'myFolder' </pre>

리소스	Blueprint 예시
여러 NIC가 포함된 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.Machine properties: image: ubuntu flavor: small networks: - name: '\${network-01.name}' deviceIndex: 0 - name: '\${network-02.name}' deviceIndex: 1 network-01: type: Cloud.vSphere.Network properties: name: network-01 network-02: type: Cloud.vSphere.Network properties: name: network-02 </pre>
스냅샷 이미지에 있는 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: imageRef: 'demo-machine/snapshot-01' cpuCount: 1 totalMemoryMB: 1024 </pre>
vCenter의 연결된 태그가 포함된 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.Machine properties: flavor: small image: ubuntu tags: - key: env value: demo </pre>
사용자 지정 규격이 포함된 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine image: ubuntu flavor: small customizationSpec: Linux </pre>

리소스	Blueprint 예시
vSphere 네트워크 구성 요소 및 정적 IP 주소가 포함된 vSphere 시스템	<pre> resources: demo-network: type: Cloud.vSphere.Network properties: name: demo-network demo-machine: type: Cloud.vSphere.Machine properties: image: ubuntu flavor: small networks: - name: demo-network assignment: static </pre>
원격 액세스가 포함된 vSphere 시스템	<pre> inputs: username: type: string title: Username description: Username default: testUser password: type: string title: Password default: VMware@123 encrypted: true description: Password for the given username resources: demo-machine: type: Cloud.vSphere.Machine properties: flavor: small imageRef: >- https://cloud-images.ubuntu.com/ releases/16.04/release-20170307/ubuntu-16.04- server-cloudimg-amd64.ova cloudConfig: ssh_pwauth: yes chpasswd: list: \${input.username}:\${input.password} expire: false users: - default - name: \${input.username} lock_passwd: false sudo: ['ALL=(ALL) NOPASSWD:ALL'] groups: [wheel, sudo, admin] shell: '/bin/bash' runcmd: - echo "Defaults:\${input.username} ! requiretty" >> /etc/sudoers.d/\${input.username} </pre>

문서화된 vRealize Automation Cloud Assembly Blueprint 예

이 예에서는 상세한 주석 집합을 포함하여 vRealize Automation Cloud Assembly Blueprint에 있는 섹션의 구조 및 용도를 검토할 수 있습니다.

```
# *****
#
# This WordPress blueprint is enhanced with comments to explain its
# parameters.
#
# Try cloning it and experimenting with its YAML code. If you're new to
# YAML, visit yaml.org for general information.
#
# The blueprint deploys a minimum of 3 virtual machines and runs scripts
# to install packages.
#
# *****
#
# -----
# Blueprints need a descriptive name and version if
# source controlled in git.
# -----
name: WordPress Blueprint with Comments
formatVersion: 1
version: 1
#
# -----
# Inputs create user selections that appear at deployment time. Inputs
# can set placement decisions and configurations, and are referenced
# later, by the resources section.
# -----
inputs:
#
# -----
# Choose a cloud endpoint. 'Title' is the visible
# option text (oneOf allows for the friendly title). 'Const' is the
# tag that identifies the endpoint, which was set up earlier, under the
# Cloud Assembly Infrastructure tab.
# -----
platform:
  type: string
  title: Deploy to
  oneOf:
    - title: AWS
      const: aws
    - title: Azure
      const: azure
    - title: vSphere
      const: vsphere
  default: vsphere
#
# -----
# Choose the operating system. Note that the Cloud Assembly
# Infrastructure must also have an AWS, Azure, and vSphere Ubuntu image
```

```

# mapped. In this case, enum sets the option that you see, meaning there's
# no friendly title feature this time. Also, only Ubuntu is available
# here, but having this input stubbed in lets you add more operating
# systems later.
# -----
osimage:
  type: string
  title: Operating System
  description: Which OS to use
  enum:
    - Ubuntu
#
# -----
# Set the number of machines in the database cluster. Small and large
# correspond to 1 or 2 machines, respectively, which you see later,
# down in the resources section.
# -----
dbenvsize:
  type: string
  title: Database cluster size
  enum:
    - Small
    - Large
#
# -----
# Dynamically tag the machines that will be created. The
# 'array' of objects means you can create as many key-value pairs as
# needed. To see how array input looks when it's collected,
# open the blueprint and click TEST.
# -----
Mtags:
  type: array
  title: Tags
  description: Tags to apply to machines
  items:
    type: object
    properties:
      key:
        type: string
        title: Key
      value:
        type: string
        title: Value
#
# -----
# Create machine credentials. These credentials are needed in
# remote access configuration later, in the resources section.
# -----
username:
  type: string
  minLength: 4
  maxLength: 20
  pattern: '[a-z]+'
  title: Database Username
  description: Database Username

```

```

userpassword:
  type: string
  pattern: '[a-z0-9A-Z@#\$]+'
  encrypted: true
  title: Database Password
  description: Database Password
#
# -----
# Set the database storage disk size.
# -----
databaseDiskSize:
  type: number
  default: 4
  maximum: 10
  title: MySQL Data Disk Size
  description: Size of database disk
#
# -----
# Set the number of machines in the web cluster. Small, medium, and large
# correspond to 2, 3, and 4 machines, respectively, which you see later,
# in the WebTier part of the resources section.
# -----
clusterSize:
  type: string
  enum:
    - small
    - medium
    - large
  title: Wordpress Cluster Size
  description: Wordpress Cluster Size
#
# -----
# Set the archive storage disk size.
# -----
archiveDiskSize:
  type: number
  default: 4
  maximum: 10
  title: Wordpress Archive Disk Size
  description: Size of Wordpress archive disk
#
# -----
# The resources section configures the deployment of machines, disks,
# networks, and other objects. In several places, the code pulls from
# the preceding interactive user inputs.
# -----
resources:
#
# -----
# Create the database server. Choose a cloud agnostic machine 'type' so
# that it can deploy to AWS, Azure, or vSphere. Then enter its property
# settings.
# -----
DBTier:
  type: Cloud.Machine

```

```

    properties:
#
# -----
# Descriptive name for the virtual machine. Does not become the hostname
# upon deployment.
# -----
        name: mysql
#
# -----
# Hard-coded operating system image to use. To pull from user input above,
# enter the following instead.
# image: '${input.osimage}'
# -----
        image: Ubuntu
#
# -----
# Hard-coded capacity to use. Note that the Cloud Assembly
# Infrastructure must also have AWS, Azure, and vSphere flavors
# such as small, medium, and large mapped.
# -----
        flavor: small
#
# -----
# Tag the database machine to deploy to the cloud vendor chosen from the
# user input. Tags are case-sensitive, so 'to_lower' forces the tag to
# lowercase to ensure a match with a site's tagging convention. It's
# important if platform input were to contain any upper case characters.
# -----
        constraints:
            - tag: '${"env:" + to_lower(input.platform)}'
#
# -----
# Also tag the database machine with any free-form tags that were created
# during user input.
# -----
        tags: '${input.Mtags}'
#
# -----
# Set the database cluster size by referencing the dbenvsize user
# input. Small is one machine, and large defaults to two.
# -----
        count: '${input.dbenvsize == "Small" ? 1 : 2}'
#
# -----
# Add a variable to connect the machine to a network resource based on
# a property binding to another resource. In this case, it's the
# 'WP_Network' network that gets defined further below.
# -----
        networks:
            - name: '${resource.WP_Network.name}'
              network: '${resource.WP_Network.id}'
#
# -----
# Enable remote access to the database server. Reference the credentials
# from the user input.

```

```

# -----
    remoteAccess:
      authentication: usernamePassword
      username: '${input.username}'
      password: '${input.userpassword}'
#
# -----
# You are free to add custom properties, which might be used to initiate
# an extensibility subscription, for example.
# -----
    ABC-Company-ID: 9393
#
# -----
# Run OS commands or scripts to further configure the database machine,
# via operations such as setting a hostname, generating SSH private keys,
# or installing packages.
# -----
    cloudConfig: |
      #cloud-config
      repo_update: true
      repo_upgrade: all
      packages:
        - mysql-server
      runcmd:
        - sed -e '/bind-address/ s/^#*#/' -i /etc/mysql/mysql.conf.d/mysqld.cnf
        - service mysql restart
        - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';"
        - mysql -e "FLUSH PRIVILEGES;"
    attachedDisks: []
#
# -----
# Create the web server. Choose a cloud agnostic machine 'type' so that it
# can deploy to AWS, Azure, or vSphere. Then enter its property settings.
# -----
    WebTier:
      type: Cloud.Machine
      properties:
#
# -----
# Descriptive name for the virtual machine. Does not become the hostname
# upon deployment.
# -----
    name: wordpress
#
# -----
# Hard-coded operating system image to use. To pull from user input above,
# enter the following instead:
# image: '${input.osimage}'
# -----
    image: Ubuntu
#
# -----
# Hard-coded capacity to use. Note that the Cloud Assembly
# Infrastructure must also have AWS, Azure, and vSphere flavors
# such as small, medium, and large mapped.

```

```

# -----
#     flavor: small
#
# -----
# Set the web server cluster size by referencing the clusterSize user
# input. Small is 2 machines, medium is 3, and large defaults to 4.
# -----
#     count: '${input.clusterSize== "small" ? 2 : (input.clusterSize == "medium" ? 3 : 4)}'
#
# -----
# Set an environment variable to display object information under the
# Properties tab, post-deployment. Another example might be
# {env.blueprintID}
# -----
#     tags:
#       - key: cas.requestedBy
#         value: '${env.requestedBy}'
#
# -----
# You are free to add custom properties, which might be used to initiate
# an extensibility subscription, for example.
# -----
#     ABC-Company-ID: 9393
#
# -----
# Tag the web server to deploy to the cloud vendor chosen from the
# user input. Tags are case-sensitive, so 'to_lower' forces the tag to
# lowercase to ensure a match with your site's tagging convention. It's
# important if platform input were to contain any upper case characters.
# -----
#     constraints:
#       - tag: '${"env:" + to_lower(input.platform)}'
#
# -----
# Add a variable to connect the machine to a network resource based on
# a property binding to another resource. In this case, it's the
# 'WP_Network' network that gets defined further below.
# -----
#     networks:
#       - name: '${resource.WP_Network.name}'
#         network: '${resource.WP_Network.id}'
#
# -----
# Run OS commands or scripts to further configure the web server,
# with operations such as setting a hostname, generating SSH private keys,
# or installing packages.
# -----
#     cloudConfig: |
#       #cloud-config
#       repo_update: true
#       repo_upgrade: all
#       packages:
#         - apache2
#         - php
#         - php-mysql

```

```

- libapache2-mod-php
- php-mcrypt
- mysql-client
runcmd:
- mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
- i=0; while [ $i -le 5 ]; do mysql --connect-timeout=3 -h $
{DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
- mysql -u root -pmysqlpassword -h ${resource.DBTier.networks[0].address} -e
"create database wordpress_blog;"
- mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
- sed -i -e s/"define('DB_NAME', 'database_name_here');"/"define('DB_NAME',
'wordpress_blog');"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define('DB_USER', 'username_here');"/"define('DB_USER', 'root');"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define('DB_PASSWORD',
'password_here');"/"define('DB_PASSWORD', 'mysqlpassword');"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define('DB_HOST',
'localhost');"/"define('DB_HOST', '${resource.DBTier.networks[0].address}');"/ /var/www/html/
mywordpresssite/wp-config.php
- service apache2 reload
#
# -----
# Create the network that the database and web servers connect to.
# Choose a cloud agnostic network 'type' so that it can deploy to AWS,
# Azure, or vSphere. Then enter its property settings.
# -----
WP_Network:
  type: Cloud.Network
  properties:
#
# -----
# Descriptive name for the network. Does not become the network name
# upon deployment.
# -----
name: WP_Network
#
# -----
# Set the networkType to an existing network. You could also use a
# constraint tag to target a specific, like-tagged network.
# The other network types are private or public.
# -----
networkType: existing
#
# *****
#
# VMware hopes that you found this commented blueprint useful. Note that
# you can also access an API to create blueprints, or query for input
# schema that you intend to request. See the following Swagger
# documentation.

```

```
#
# www.mgmt.cloud.vmware.com/blueprint/api/swagger/swagger-ui.html
#
# *****
```

vRealize Automation Cloud Assembly Blueprint의 네트워크, 보안 및 로드 밸런서 예

이러한 Blueprint 코드 예제에서는 몇 가지 기본 네트워크, 보안 및 로드 밸런서 구성을 보여줍니다.

세부적인 네트워크 및 보안 구현 시나리오의 예는 다음과 같은 VMware 블로그를 참조하십시오.

- [NSX-T를 사용하는 vRealize Automation Cloud Assembly 로드 밸런서 심화 연구](#)
- [Cloud Assembly 및 NSX를 통한 네트워크 자동화 - 1부\(NSX-T, vCenter 클라우드 계정 및 네트워크 CIDR 포함\)](#)
- [Cloud Assembly 및 NSX를 통한 네트워크 자동화 - 2부\(기본 및 아웃바운드 네트워크 유형 사용 포함\)](#)
- [Cloud Assembly 및 NSX를 통한 네트워크 자동화 - 3부\(기본 및 주문형 보안 그룹 사용 포함\)](#)
- [Cloud Assembly 및 NSX를 통한 네트워크 자동화 - 4부\(기본 및 주문형 로드 밸런서 사용 포함\)](#)

모든 Blueprint 스키마 옵션에 대한 전체 요약은 [vRealize Automation 리소스 유형 스키마](#)를 참조하십시오.

네트워크 유형에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 네트워크 및 네트워크 프로파일 사용 항목](#)을 참조하십시오.

리소스 시나리오	Blueprint 코드 예
여러 NIC가 포함된 vSphere 시스템	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: image: ubuntu flavor: small networks: - name: '\${network-01.id}' deviceIndex: 0 - name: '\${network-02.id}' deviceIndex: 1 Cloud_vSphere_Network_1: type: Cloud.vSphere.Network properties: networkType: existing name: network-01 Cloud_vSphere_Network_2: type: Cloud.NSX.Network properties: networkType: existing name: network-02 </pre>
공용 IP 대신 내부 IP를 사용하는 공용 클라우드 시스템	<pre> resources: wf_proxy: type: Cloud.Machine properties: image: ubuntu 16.04 flavor: small constraints: - tag: 'platform:vsphere' networks: - name: '\${resource.wf_net.id}' assignPublicIpAddress: false </pre>
NSX 네트워크 구성 요소 유형을 사용하여 NSX-V 또는 NSX-T에 대해 라우팅된 네트워크	<pre> Cloud_NSX_Network_1: type: Cloud.NSX.Network properties: networkType: routed </pre>
아웃바운드 네트워크에 대한 NSX 논리적 스위치 태그 지정 이 시나리오에 대한 자세한 내용은 커뮤니티 블로그 게시물 Creating Tags in NSX with Cloud Assembly 를 참조하십시오.	<pre> Cloud_NSX_Network_1: type: Cloud.NSX.Network properties: networkType: outbound tags: - key: app value: opencart </pre>

리소스 시나리오	Blueprint 코드 예
<p>시스템 NIC에 제약 조건 태그가 적용된 기존 보안 그룹</p> <p>기존 보안 그룹을 사용하려면 Cloud.SecurityGroup 구성 요소의 securityGroupType 속성에 <i>existing</i>을 입력합니다.</p> <p>주문형 보안 그룹을 생성하려면 Cloud.SecurityGroup 구성 요소의 securityGroupType 속성에 <i>new</i>를 입력합니다.</p> <p>태그 제약 조건을 사용하여 기존 보안 그룹을 할당하기 위해 Cloud.SecurityGroup 구성 요소에 태그를 할당할 수 있습니다. 태그가 포함되지 않은 보안 그룹은 Blueprint에서 사용할 수 없습니다.</p>	<pre>formatVersion: 1 inputs: {} resources: allowSsh_sg: type: Cloud.SecurityGroup properties: securityGroupType: existing constraints: - tag: allowSsh compute: type: Cloud.Machine properties: image: centos flavor: small networks: - network: '\${resource.prod-net.id}' securityGroups: - '\${resource.allowSsh_sg.id}' prod-net: type: Cloud.Network properties: networkType: existing</pre>

리소스 시나리오	Blueprint 코드 예
원 암(one arm) 로드 밸런서가 있는 주문형 네트워크	<pre> inputs: {} resources: mp-existing: type: Cloud.Network properties: name: mp-existing networkType: existing mp-wordpress: type: Cloud.vSphere.Machine properties: name: wordpress count: 2 flavor: small image: tiny customizationSpec: Linux networks: - network: '\${resource["mp-private"].id}' mp-private: type: Cloud.NSX.Network properties: name: mp-private networkType: private constraints: - tag: nsxt mp-wordpress-lb: type: Cloud.LoadBalancer properties: name: wordpress-lb internetFacing: false network: '\${resource.mp-existing.id}' instances: '\${resource["mp-wordpress"].id}' routes: - protocol: HTTP port: '80' instanceProtocol: HTTP instancePort: '80' healthCheckConfiguration: protocol: HTTP port: '80' urlPath: /index.pl intervalSeconds: 60 timeoutSeconds: 30 unhealthyThreshold: 5 healthyThreshold: 2 </pre>
로드 밸런서가 있는 기존 네트워크	<pre> formatVersion: 1 inputs: count: type: integer default: 1 resources: ubuntu-vm: type: Cloud.Machine properties: name: ubuntu flavor: small image: tiny count: '\${input.count}' networks: </pre>

리소스 시나리오	Blueprint 코드 예
	<pre> - network: '\$ {resource.Cloud_NSX_Network_1.id}' Provider_LoadBalancer_1: type: Cloud.LoadBalancer properties: name: OC-LB routes: - protocol: HTTP port: '80' instanceProtocol: HTTP instancePort: '80' healthCheckConfiguration: protocol: HTTP port: '80' urlPath: /index.html intervalSeconds: 60 timeoutSeconds: 5 unhealthyThreshold: 5 healthyThreshold: 2 network: '\$ {resource.Cloud_NSX_Network_1.id}' internetFacing: false instances: '\${resource["ubuntu-vm"].id}' Cloud_NSX_Network_1: type: Cloud.NSX.Network properties: networkType: existing constraints: - tag: nsxt24prod </pre>

사용자 이름 및 암호 액세스를 사용하는 Puppet 지원 Blueprint

이 예에서는 사용자 이름 및 암호 액세스를 사용하여 Puppet 구성 관리를 vCenter 계산 리소스에 배포된 Blueprint에 추가합니다.

이 절차는 사용자 이름 및 암호 인증이 필요한 배포 가능한 Puppet 지원 리소스를 생성할 수 있는 방법에 대한 하나의 예를 보여 줍니다. 사용자 이름 및 암호 액세스란 사용자가 계산 리소스에서 Puppet 기본 시스템으로 수동으로 로그인해야 Puppet 구성 관리를 호출할 수 있음을 의미합니다.

선택적으로, 계산 리소스가 Puppet 기본 시스템을 사용하여 인증을 처리하도록 Blueprint에서 구성 관리를 설정하는 원격 액세스 인증을 구성할 수 있습니다. 원격 액세스를 사용하도록 설정하면 계산 리소스가 암호 인증을 만족하는 키를 자동으로 생성합니다. 이 경우에도 올바른 사용자 이름은 필요합니다.

vRealize Automation Cloud Assembly Blueprint에서 서로 다른 Puppet 시나리오를 구성할 수 있는 방법에 대한 더 많은 예를 보려면 [AWS Puppet 구성 관리 Blueprint 예제](#) 및 [vCenter Puppet 구성 Blueprint](#) 예 항목을 참조하십시오.

사전 요구 사항

- 올바른 네트워크에 Puppet Enterprise 인스턴스를 설정합니다.
- 통합 기능을 사용하여 Puppet Enterprise 인스턴스를 vRealize Automation Cloud Assembly에 추가합니다. [vRealize Automation Cloud Assembly에서 Puppet Enterprise 통합 구성](#) 항목을 참조하십시오.

- vSphere 계정 및 vCenter 계산 리소스를 설정합니다.

절차

- 1 Puppet 구성 관리 구성 요소를 원하는 Blueprint 캔버스의 vSphere 계산 리소스에 추가합니다.
 - a 인프라 > 관리 > 통합을 선택합니다.
 - b 통합 추가를 클릭하고 Puppet을 선택합니다.
 - c [Puppet 구성] 페이지에 적절한 정보를 입력합니다.

구성	설명	예시 값
호스트 이름	Puppet 기본 시스템의 호스트 이름 또는 IP 주소	Puppet-Ubuntu
SSH 포트	vRealize Automation Cloud Assembly와 Puppet 기본 시스템 간 통신을 위한 SSH 포트입니다. (선택 사항)	해당 없음
Autosign 암호	Autosign 인증서 요청 지원을 위해 노드에서 제공해야 하는 Puppet 기본 시스템에서 구성된 공유 암호입니다.	사용자 특정
위치	Puppet 기본 시스템이 전용 클라우드에 있는지 아니면 공용 클라우드에 있는지 나타냅니다. 참고 교차 클라우드 배포는 배포 계산 리소스와 Puppet 기본 시스템 간에 연결이 있는 경우에만 지원됩니다.	
Cloud proxy	Microsoft Azure 또는 Amazon Web Services와 같은 공용 클라우드 계정에는 필요하지 않습니다. vCenter 기반 클라우드 계정을 사용 중이라면 계정에 적절한 cloud proxy를 선택합니다.	해당 없음
사용자 이름	Puppet 기본 시스템에 대한 SSH 및 RBAC 사용자 이름입니다.	사용자 특정. YAML 값은 '\$ {input.username}'입니다.
암호	Puppet 기본 시스템에 대한 SSH 및 RBAC 암호입니다.	User 특정. YAML 값은 '\$ {input.password}'입니다.
이 사용자에게 대해 sudo 명령 사용	procidd에 대해 sudo 명령을 사용하려면 선택합니다.	true
이름	Puppet 기본 시스템 이름입니다.	PEMasterOnPrem
설명		

- 2 다음 예에 나와 있는 것과 같이 Puppet YAML에 사용자 이름 및 암호를 추가합니다.
- 3 아래 예에 나와 있는 것과 같이 Puppet YAML에 대한 remoteAccess 속성 값이 authentication: username and password로 설정되어 있는지 확인합니다.

예제: vCenter 사용자 이름 및 암호 YAML 코드

다음 예는 vCenter 계산 리소스에서 사용자 이름 및 암호 인증을 추가하기 위한 대표 YAML 코드를 보여줍니다.

```
inputs:
  username:
    type: string
    title: Username
    description: Username to use to install Puppet agent
    default: puppet
  password:
    type: string
    title: Password
    default: VMware@123
    encrypted: true
    description: Password for the given username to install Puppet agent
resources:
  Puppet-Ubuntu:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      imageRef: >-
        https://cloud-images.ubuntu.com/releases/16.04/release-20170307/ubuntu-16.04-server-
        cloudimg-amd64.ova
      remoteAccess:
        authentication: usernamePassword
        username: '${input.username}'
        password: '${input.password}'
  Puppet_Agent:
    type: Cloud.Puppet
    properties:
      provider: PEMasterOnPrem
      environment: production
      role: 'role::linux_webserver'
      username: '${input.username}'
      password: '${input.password}'
      host: '${Puppet-Ubuntu.*}'
      useSudo: true
      agentConfiguration:
        certName: '${Puppet-Ubuntu.address}'
```

AWS Puppet 구성 관리 Blueprint 예제

AWS 계산 리소스에서 Puppet 기반 구성 관리를 지원하도록 Blueprint를 구성하는 방법에는 여러 가지가 있습니다.

사용자 이름 및 암호를 사용하여 AWS에서 Puppet 관리

예시...	샘플 Blueprint YAML
지원되는 모든 Amazon 시스템 이미지의 클라우드 구성 인증	<pre> inputs: username: type: string title: Username default: puppet password: type: string title: Password encrypted: true default: VMware@123 resources: Webserver: type: Cloud.AWS.EC2.Instance properties: flavor: small image: centos cloudConfig: #cloud-config ssh_pwauth: yes chpasswd: list: \${input.username}:\${input.password} expire: false users: - default - name: \${input.username} lock_passwd: false sudo: ['ALL=(ALL) NOPASSWD:ALL'] groups: [wheel, sudo, admin] shell: '/bin/bash' ssh-authorized-keys: - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDytVL+Q6/+vGbmKXoRpX dmettem@dmettem-m01.vmware.com runcmd: - echo "Defaults:\${input.username} !requiretty" >> /etc/sudoers.d/\${input.username} Puppet_Agent: type: Cloud.Puppet properties: provider: PEOonAWS environment: production role: 'role::linux_webserver' host: '\${Webserver.*}' osType: linux username: '\${input.username}' password: '\${input.password}' useSudo: true </pre>
기존 사용자를 사용하여 사용자 지정 Amazon 시스템 이미지의 클라우드 구성 인증	<pre> inputs: username: type: string title: Username default: puppet password: type: string title: Password encrypted: true default: VMware@123 </pre>

예시...	샘플 Blueprint YAML
	<pre> resources: Webserver: type: Cloud.AWS.EC2.Instance properties: flavor: small image: centos cloudConfig: #cloud-config runcmd: - sudo sed -e 's/. *PasswordAuthentication no.*/ PasswordAuthentication yes/' -i /etc/ssh/sshd_config - sudo service sshd restart Puppet_Agent: type: Cloud.Puppet properties: provider: PEOAWS environment: production role: 'role::linux_webserver' host: '\${Webserver.*}' osType: linux username: '\${input.username}' password: '\${input.password}' useSudo: true </pre>

생성된 PublicPrivateKey를 사용하여 AWS에서 Puppet 관리

예제...	샘플 Blueprint YAML
<p>generatedPublicPrivateKey 액세 스를 사용하여 AWS에서 remoteAccess.authentication 인증</p>	<pre> inputs: {} resources: Machine: type: Cloud.AWS.EC2.Instance properties: flavor: small imageRef: ami-a4dc46db remoteAccess: authentication: generatedPublicPrivateKey Puppet_Agent: type: Cloud.Puppet properties: provider: puppet-BlueprintProvisioningITSuite environment: production role: 'role::linux_webserver' host: '\${Machine.*}' osType: linux username: ubuntu useSudo: true agentConfiguration: runInterval: 15m certName: '\${Machine.address}' useSudo: true </pre>

vCenter Puppet 구성 Blueprint 예

vCenter 계산 리소스에서 Puppet 기반 구성 관리를 지원하는 Blueprint를 구성하기 위한 몇 가지 옵션이 있습니다.

사용자 이름 및 암호 인증을 사용하는 vSphere의 Puppet

다음 예제는 사용자 이름 및 암호 인증을 사용하는 vSphere OVA의 Puppet에 대한 YAML 코드 예를 보여줍니다.

표 6-2.

예제...	샘플 Blueprint YAML
<p>사용자 이름 및 암호 인증을 사용하는 vSphere OVA의 Puppet에 대한 YAML 코드</p>	<pre>inputs: username: type: string title: Username default: puppet password: type: string title: Password encrypted: true default: VMware@123 resources: Puppet_Agent: type: Cloud.Puppet properties: provider: PEonAWS environment: dev role: 'role::linux_webserver' username: '\${input.username}' password: '\${input.password}' useSudo: true host: '\${Webserver.*}' osType: linux agentConfiguration: runInterval: 15m certName: '\${Machine.address}' Webserver: type: Cloud.vSphere.Machine properties: cpuCount: 1 totalMemoryMB: 1024 imageRef: >- https://cloud-images.ubuntu.com/releases/16.04/ release-20170307/ubuntu-16.04-server-cloudimg-amd64.ova cloudConfig: #cloud-config ssh_pwauth: yes chpasswd: list: \${input.username}:\${input.password} expire: false users: - default - name: \${input.username} lock_passwd: false sudo: ['ALL=(ALL) NOPASSWD:ALL'] groups: [wheel, sudo, admin] shell: '/bin/bash' ssh-authorized-keys: - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDytVL+Q6+vGbmKXoRpX dmettem@dmettem-m01.vmware.com runcmd: - echo "Defaults:\${input.username}"</pre>
<p>계산 리소스에서 사용자 이름 및 암호 인증을 사용하는 vSphere OVA의 Puppet에 대한 YAML 코드</p>	<pre>inputs: username: type: string title: Username default: puppet</pre>

표 6-2. (계속)

예제...	샘플 Blueprint YAML
	<pre> password: type: string title: Password encrypted: true default: VMware@123 resources: Puppet_Agent: type: Cloud.Puppet properties: provider: PEonAWS environment: dev role: 'role::linux_webserver' username: '\${input.username}' password: '\${input.password}' useSudo: true host: '\${Webserver.*}' osType: linux agentConfiguration: runInterval: 15m certName: '\${Machine.address}' Webserver: type: Cloud.vSphere.Machine properties: cpuCount: 1 totalMemoryMB: 1024 imageRef: >= https://cloud-images.ubuntu.com/releases/16.04/ release-20170307/ubuntu-16.04-server-cloudimg-amd64.ova cloudConfig: #cloud-config ssh_pwauth: yes chpasswd: list: \${input.username}:\${input.password} expire: false users: - default - name: \${input.username} lock_passwd: false sudo: ['ALL=(ALL) NOPASSWD:ALL'] groups: [wheel, sudo, admin] shell: '/bin/bash' ssh-authorized-keys: - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDytVL+Q6+vGbmKXoRpX dmettem@dmettem-m01.vmware.com runcmd: - echo "Defaults:\${input.username} </pre>
<p>제산 리소스에서 원격 액세스 지원 암호 인증을 사용하는 vCenter의 Puppet에 대한 YAML 코드</p>	<pre> inputs: username: type: string title: Username description: Username to use to install Puppet agent default: puppet password: type: string title: Password default: VMware@123 encrypted: true </pre>

표 6-2. (계속)

예제...

샘플 Blueprint YAML

```

    description: Password for the given username to install
Puppet agent
resources:
  Puppet-Ubuntu:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      imageRef: >-
        https://cloud-images.ubuntu.com/releases/16.04/
release-20170307/ubuntu-16.04-server-cloudimg-amd64.ova
      remoteAccess:
        authentication: usernamePassword
        username: '${input.username}'
        password: '${input.password}'
  Puppet_Agent:
    type: Cloud.Puppet
    properties:
      provider: PEMasterOnPrem
      environment: production
      role: 'role::linux_webserver'
      username: '${input.username}'
      password: '${input.password}'
      host: '${Puppet-Ubuntu.*}'
      useSudo: true
      agentConfiguration:
        certName: '${Puppet-Ubuntu.address}'

```

생성된 PublicPrivateKey 인증을 사용하는 vSphere의 Puppet

표 6-3.

예제...	샘플 Blueprint YAML
계산 리소스에서 생성된 PublicPrivateKey 인증을 사용하는 vSphere OVA의 Puppet에 대한 YAML 코드.	<pre> inputs: {} resources: Machine: type: Cloud.vSphere.Machine properties: flavor: small imageRef: >- https://cloud-images.ubuntu.com/releases/16.04/ release-20170307/ubuntu-16.04-server-cloudimg-amd64.ova remoteAccess: authentication: generatedPublicPrivateKey Puppet_Agent: type: Cloud.Puppet properties: provider: puppet-BlueprintProvisioningITSuite environment: production role: 'role::linux_webserver' host: '\${Machine.*}' osType: linux username: ubuntu useSudo: true agentConfiguration: runInterval: 15m certName: '\${Machine.address}' - echo "Defaults:\${input.username} </pre>

vRealize Automation Cloud Assembly 마켓플레이스를 사용하는 방법

리소스 라이브러리를 점프스타트하려면 vRealize Automation Cloud Assembly 마켓플레이스에서 파일을 다운로드합니다.

마켓플레이스에는 [VMware Solution Exchange](#)에서 관리되는 마무리된 Blueprint 및 오픈 가상화 이미지가 제공됩니다. cloud assembly로 태그가 지정된 Solution Exchange 파일은 vRealize Automation Cloud Assembly 마켓플레이스 탭 아래에 표시됩니다.

마켓플레이스에 액세스하는 방법

vRealize Automation Cloud Assembly에서 **인프라 > 연결 > 통합**을 선택합니다. **통합 추가**를 클릭하고 **My VMware**를 클릭한 후 My VMware 계정 자격 증명을 제공합니다.

마켓플레이스 Blueprint 파일을 다운로드하고 사용하는 방법

마켓플레이스 탭에서 **가져오기**를 클릭하고 Blueprint EULA에 동의합니다. 그런 다음, Blueprint를 vRealize Automation Cloud Assembly 프로젝트에 추가하거나 간단히 다운로드할 수 있습니다.

Blueprint 탭에서 Blueprint를 업로드할 수 있습니다.

프로젝트 기반 예제에서는 빅데이터 작업의 프로젝트 관리자라고 가정합니다. 팀을 지원하기 위해, 팀 프로젝트에 추가할 마켓플레이스 Hadoop Blueprint를 찾습니다. 그리고 리소스 환경에 맞게 Blueprint를 사용자 지정하고 릴리스합니다. 그런 다음, 팀에서 배포할 수 있도록 Blueprint를 vRealize Automation Service Broker 카탈로그로 가져옵니다.

마켓플레이스 이미지 파일을 다운로드하고 사용하는 방법

마켓플레이스 탭에서 **가져오기**를 클릭하고 OVF 또는 OVA 이미지 EULA에 동의합니다. 나중에, OVF 또는 OVA 이미지를 다운로드하여 Blueprint 코드에서 참조할 수 있습니다.

이전 예시를 계속 진행하다 보면, 팀에 Hadoop 자체 버전에 대한 액세스 권한이 필요할 수 있습니다. Hadoop OVF를 다운로드하여 vCenter Server 콘텐츠 라이브러리와 같은 클라우드 계정 리소스에 추가합니다. 그런 다음, OVF 이미지를 가리켜야 하는 Blueprint 코드를 업데이트합니다.

확장성을 사용하여 애플리케이션 수명 주기를 연장 및 자동화하는 방법

확장성 작업 또는 vRealize Orchestrator 워크플로를 확장성 구독과 함께 사용하여 애플리케이션 수명 주기를 연장할 수 있습니다.

vRealize Automation Cloud Assembly 확장성을 사용하면 구독을 통해 확장성 작업 또는 vRealize Orchestrator 워크플로를 이벤트에 할당할 수 있습니다. 이벤트가 발생하면 구독은 실행할 작업 또는 워크플로를 시작하고, 모든 구독자에게 알립니다.

확장성 작업

확장성 작업은 작업 및 작업 수행 방법을 지정하는 데 사용되는 작은 경량 코드 스크립트입니다. 미리 정의된 vRealize Automation Cloud Assembly 작업 템플릿에서 또는 ZIP 파일에서 확장성 작업을 가져올 수 있습니다. 작업 편집기를 사용하여 확장성 작업에 대한 사용자 지정 스크립트를 생성할 수도 있습니다. 여러 작업 스크립트가 하나의 스크립트에 연결되어 있으면 작업 흐름을 생성합니다. 작업 흐름을 사용하여 순서가 지정된 작업을 생성할 수 있습니다. 작업 흐름 사용에 대한 자세한 내용은 [작업 흐름이란?](#) 항목을 참조하십시오.

vRealize Orchestrator 워크플로

vRealize Automation Cloud Assembly를 기존 vRealize Orchestrator 환경에 통합하여 확장성 구독에서 워크플로를 사용할 수 있습니다.

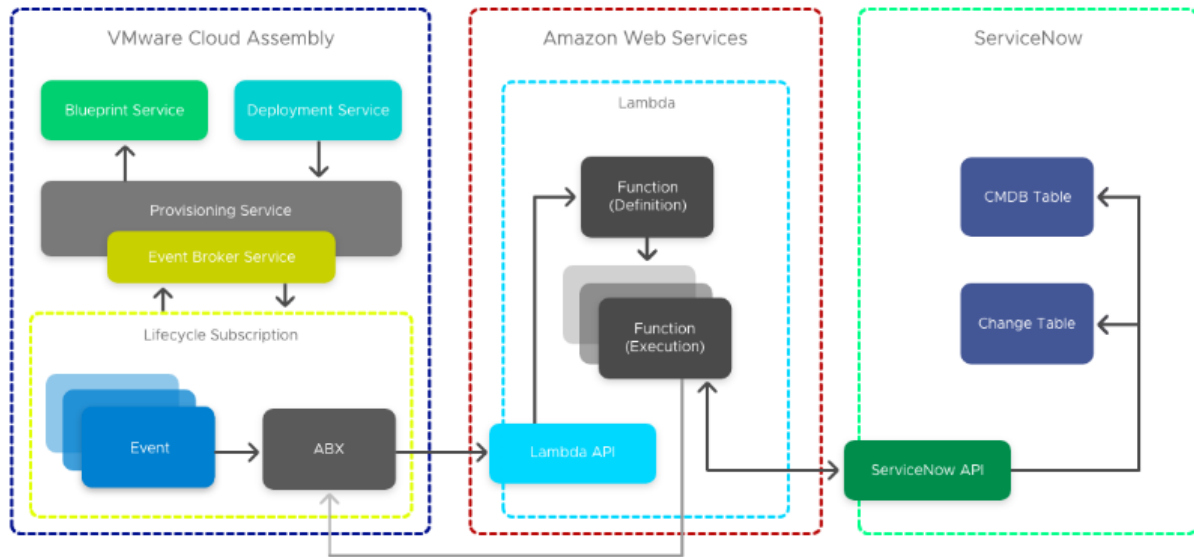
확장성 작업 구독

vRealize Automation Cloud Assembly 구독에 확장성 작업을 할당하여 애플리케이션 수명 주기를 연장할 수 있습니다.

참고 다음 구독은 사용 사례 예시이며 모든 확장성 작업 기능을 다루지는 않습니다.

확장성 작업을 사용하여 Cloud Assembly와 ServiceNow를 통합하는 방법

확장성 작업을 사용하면 vRealize Automation Cloud Assembly를 ServiceNow 같은 Enterprise ITSM과 통합할 수 있습니다.

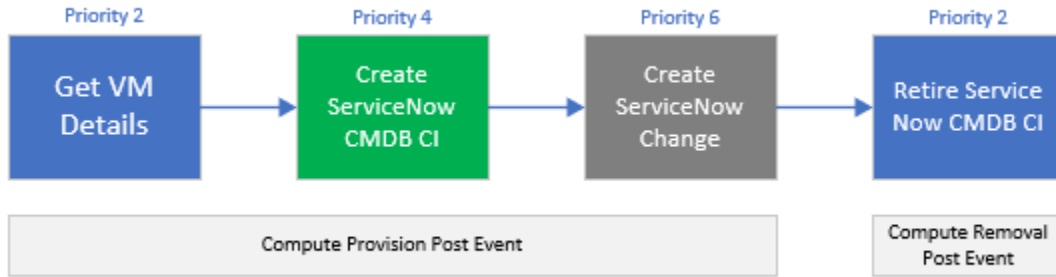


엔터프라이즈 사용자는 규정 준수를 위해 일반적으로 Cloud Management Platform을 ITSM(IT 서비스 관리) 및 CMDB(구성 관리 데이터베이스) 플랫폼과 통합합니다. 이 예시를 사용하면 확장성 작업 스크립트를 사용하여 CMDB 및 ITSM을 위해 vRealize Automation Cloud Assembly를 ServiceNow와 통합할 수 있습니다.

참고 또한 vRealize Orchestrator 워크플로를 사용하여 ServiceNow를 vRealize Automation Cloud Assembly와 통합할 수도 있습니다. 워크플로를 사용하여 ServiceNow를 통합하는 방법에 대한 자세한 내용은 vRealize Orchestrator 워크플로를 사용하여 ITSM용 Cloud Assembly와 ServiceNow를 통합하는 방법 항목을 참조하십시오.

이 통합을 생성하기 위해 4개의 확장성 작업 스크립트를 사용합니다. 처음 3개의 스크립트는 프로비저닝 중에 순차적으로 시작됩니다(프로비저닝 후 계산 이벤트). 4번째 스크립트는 제거 후 계산 이벤트 시점에 트리거됩니다.

이벤트 항목에 대한 자세한 내용은 vRealize Automation Cloud Assembly에서 제공되는 이벤트 항목 항목을 참조하십시오.



VM 세부 정보 가져오기

VM 세부 정보 가져오기 스크립트는 CI 생성에 필요한 페이로드 세부 정보 및 Amazon Web Services Systems Manager(SSM) Parameter Store에 저장되어 있는 ID 토큰을 가져옵니다. 또한 이 스크립트는 나중에 사용하기 위해 추가적인 속성으로 `customProperties`를 업데이트합니다.

ServiceNow CMDB CI 생성

ServiceNow CMDB CI 생성 스크립트는 ServiceNow 인스턴스 URL을 입력으로 전달하고, 보안 요구 사항을 충족하도록 SSM에 인스턴스를 저장합니다. 이 스크립트는 ServiceNow CMDB 고유 레코드 ID 응답(`sys_id`)도 읽습니다. 이 스크립트는 이를 출력으로 전달하고 생성 시 사용자 지정 속성 `serviceNowSysId`를 씁니다. 이 값은 인스턴스가 삭제될 때 CI를 회수된 것으로 표시하는 데 사용됩니다.

참고 Lambda가 SSM Parameter Store에 액세스할 수 있도록 vRealize Automation services Amazon Web Services 역할에 추가적인 사용 권한을 할당해야 할 수도 있습니다.

ServiceNow 변경 사항 생성

이 스크립트는 ServiceNow 인스턴스 URL을 입력으로 전달하고 보안 요구 사항을 충족하기 위해 ServiceNow 자격 증명을 SSM으로 저장하여 ITSM 통합을 완료합니다.

ServiceNow 변경 사항 생성

ServiceNow CMDB CI 회수 스크립트는 ServiceNow 중지 메시지를 표시하고, 생성 스크립트에 생성된 사용자 지정 속성 `serviceNowSysId`에 기반하여 CI를 회수된 것으로 표시합니다.

사전 요구 사항

- 이 통합을 구성하기 전에 조건부 blueprint 속성 `event.data["customProperties"]["enable_servicenow"] == "true"`를 사용하여 모든 이벤트 구독을 필터링합니다.

참고 이 속성은 ServiceNow 통합이 필요한 Blueprint에 있습니다.

- Python 애플리케이션이 설치되어 있습니다.

구독 필터링에 대한 자세한 내용은 [확장성 구독 생성](#) 항목을 참조하십시오.

절차

- 1 가상 시스템에서 명령줄 프롬프트를 엽니다.
- 2 VM 세부 정보 가져오기 스크립트를 실행합니다.

```
from botocore.vendored import requests
import json
import boto3
client = boto3.client('ssm','ap-southeast-2')

def handler(context, inputs):
    baseUri = inputs['url']
    casToken = client.get_parameter(Name="casToken",WithDecryption=True)

    url = baseUri + "/iaas/login"
    headers = {"Accept":"application/json","Content-Type":"application/json"}
    payload = {"refreshToken":casToken['Parameter']['Value']}

    results = requests.post(url,json=payload,headers=headers)

    bearer = "Bearer "
    bearer = bearer + results.json()["token"]

    deploymentId = inputs['deploymentId']
    resourceId = inputs['resourceIds'][0]

    print("deploymentId: " + deploymentId)
    print("resourceId:" + resourceId)

    machineUri = baseUri + "/iaas/machines/" + resourceId
    headers = {"Accept":"application/json","Content-Type":"application/json",
    "Authorization":bearer }
    resultMachine = requests.get(machineUri,headers=headers)
    print("machine: " + resultMachine.text)

    print( "serviceNowCPUCount: " + json.loads(resultMachine.text)["customProperties"]
    ["cpuCount"] )
    print( "serviceNowMemoryInMB: " + json.loads(resultMachine.text)["customProperties"]
    ["memoryInMB"] )

    #update customProperties
    outputs = {}
    outputs['customProperties'] = inputs['customProperties']
    outputs['customProperties']['serviceNowCPUCount'] = int(json.loads(resultMachine.text)
    ["customProperties"]["cpuCount"])
    outputs['customProperties']['serviceNowMemoryInMB'] = json.loads(resultMachine.text)
    ["customProperties"]["memoryInMB"]
    return outputs
```

- 3 CMDB 구성 항목 생성 작업을 실행합니다.

```
from botocore.vendored import requests
import json
import boto3
```

```

client = boto3.client('ssm','ap-southeast-2')

def handler(context, inputs):

    snowUser = client.get_parameter(Name="serviceNowUserName",WithDecryption=False)
    snowPass = client.get_parameter(Name="serviceNowPassword",WithDecryption=True)
    table_name = "cmdb_ci_vmware_instance"
    url = "https://" + inputs['instanceUrl'] + "/api/now/table/{0}".format(table_name)
    headers = {'Content-type': 'application/json', 'Accept': 'application/json'}
    payload = {
        'name': inputs['customProperties']['serviceNowHostname'],
        'cpus': int(inputs['customProperties']['serviceNowCPUCount']),
        'memory': inputs['customProperties']['serviceNowMemoryInMB'],
        'correlation_id': inputs['deploymentId'],
        'disks_size': int(inputs['customProperties']['provisionGB']),
        'location': "Sydney",
        'vcenter_uuid': inputs['customProperties']['vcUuid'],
        'state': 'On',
        'sys_created_by': inputs['__metadata']['userName'],
        'owned_by': inputs['__metadata']['userName']
    }
    results = requests.post(
        url,
        json=payload,
        headers=headers,
        auth=(snowUser['Parameter']['Value'], snowPass['Parameter']['Value'])
    )
    print(results.text)

    #parse response for the sys_id of CMDB CI reference
    if json.loads(results.text)['result']:
        serviceNowResponse = json.loads(results.text)['result']
        serviceNowSysId = serviceNowResponse['sys_id']
        print(serviceNowSysId)

        #update the serviceNowSysId customProperty
        outputs = {}
        outputs['customProperties'] = inputs['customProperties']
        outputs['customProperties']['serviceNowSysId'] = serviceNowSysId;
        return outputs

```

4 생성 작업 스크립트를 실행합니다.

```

from botocore.vendored import requests
import json
import boto3
client = boto3.client('ssm','ap-southeast-2')

def handler(context, inputs):
    snowUser = client.get_parameter(Name="serviceNowUserName",WithDecryption=False)
    snowPass = client.get_parameter(Name="serviceNowPassword",WithDecryption=True)
    table_name = "change_request"
    url = "https://" + inputs['instanceUrl'] + "/api/now/table/{0}".format(table_name)
    headers = {'Content-type': 'application/json', 'Accept': 'application/json'}
    payload = {

```

```

        'short_description': 'Provision CAS VM Instance'
    }
    results = requests.post(
        url,
        json=payload,
        headers=headers,
        auth=(snowUser['Parameter']['Value'], snowPass['Parameter']['Value'])
    )
    print(results.text)

```

결과

vRealize Automation Cloud Assembly가 ITSM ServiceNow와 통합되었습니다.

다음에 수행할 작업

필요한 경우, CMDB 구성 항목 회수 작업을 사용하여 CI를 회수할 수 있습니다.

```

from botocore.vendored import requests
import json
import boto3
client = boto3.client('ssm', 'ap-southeast-2')

def handler(context, inputs):
    snowUser = client.get_parameter(Name="serviceNowUserName", WithDecryption=False)
    snowPass = client.get_parameter(Name="serviceNowPassword", WithDecryption=True)
    tableName = "cmdb_ci_vmware_instance"
    sys_id = inputs['customProperties']['serviceNowSysId']
    url = "https://" + inputs['instanceUrl'] + "/api/now/" + tableName + "/" + sys_id
    headers = {'Content-type': 'application/json', 'Accept': 'application/json'}
    payload = {
        'state': 'Retired'
    }

    results = requests.put(
        url,
        json=payload,
        headers=headers,
        auth=(inputs['username'], inputs['password'])
    )
    print(results.text)

```

확장성 작업을 사용하여 vRealize Automation Cloud Assembly에 ServiceNow를 통합할 수 있는 방법에 대한 자세한 내용은 [ServiceNow](#) 통합을 위해 작업 기반 확장성을 사용하여 [Cloud Assembly](#) 확장을 참조하십시오.

프로비저닝하는 동안 확장성 작업을 사용하여 가상 시스템에 태그를 지정하는 방법

확장성 작업을 구독과 함께 사용하여 VM 태그 지정을 자동화하고 간소화할 수 있습니다.

클라우드 관리자는 확장성 작업 및 확장성 구독을 사용하여, 지정된 입력 및 출력으로 태그가 자동으로 지정되는 배포를 생성할 수 있습니다. VM 태그 지정 구독이 포함된 프로젝트에 대해 새 배포를 생성하면 배포 이벤트가 VM 태그 지정 스크립트의 실행을 트리거하고, 태그가 자동으로 적용됩니다. 이를 통해 시간을 절약하고 효율성을 높이는 동시에 배포를 더 쉽게 관리할 수 있습니다.

사전 요구 사항

- 클라우드 관리자 자격 증명에 대한 액세스.
- Lambda 함수에 대한 Amazon Web Services 역할.

절차

- 1 **확장성 > 라이브러리 > 작업 > 새 작업**으로 이동하고 다음 매개 변수를 사용하여 작업을 생성합니다.

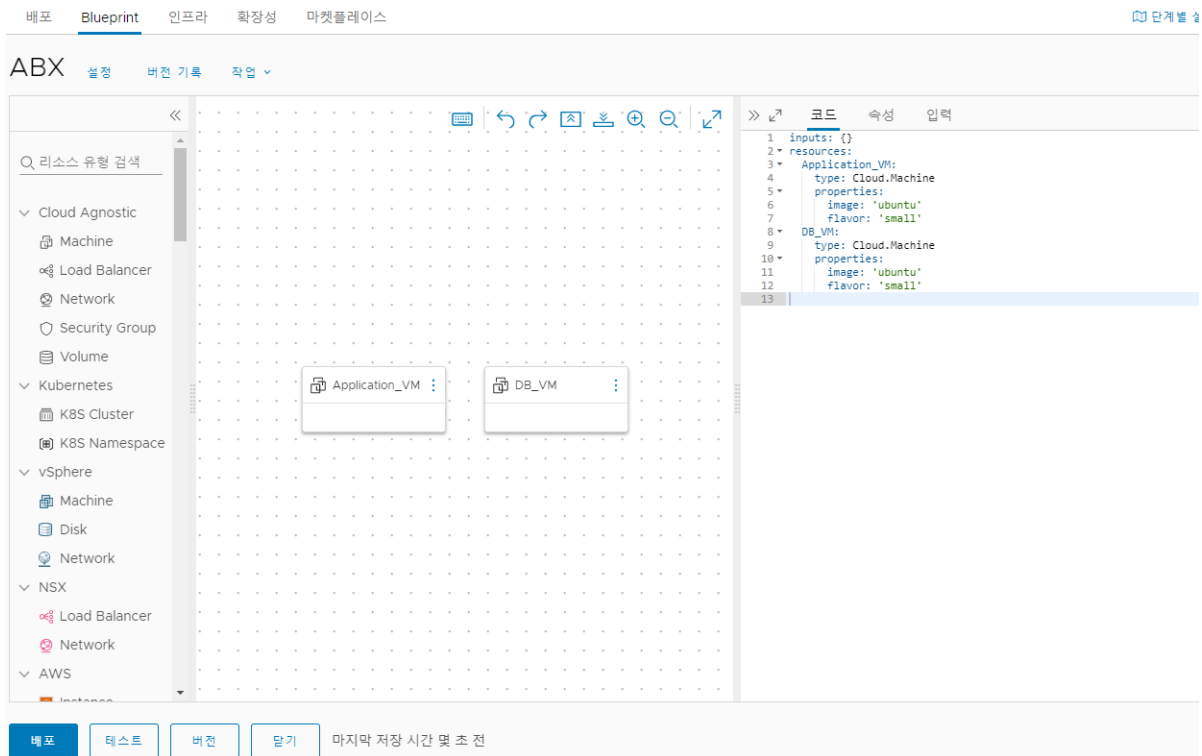
매개 변수	설명
작업 이름	확장성 작업 이름이며, 'TagVM'을 접두사 또는 접미사로 사용하는 것이 좋습니다.
프로젝트	확장성 작업을 테스트할 대상 프로젝트입니다.
작업 템플릿	VM 태그 지정
런타임	Python
스크립트 소스	스크립트 작성

- 2 **기본 함수**으로 **Handler**를 입력합니다.
- 3 확장성 작업을 테스트하기 위한 태그 지정 입력을 추가합니다.
예: `resourceNames = ["DB_VM"], target = world`
- 4 작업을 저장하려면 **저장**을 클릭합니다.
- 5 작업을 테스트하려면 **테스트**를 클릭합니다.
- 6 작업 편집기를 종료하려면 **닫기**를 클릭합니다.
- 7 **확장성 > 구독**으로 이동합니다.
- 8 **새 구독**을 클릭합니다.
- 9 다음과 같은 구독 세부 정보를 입력합니다.

세부 정보	설정
이벤트 항목	VM의 태그 지정 단계와 관련된 이벤트 항목을 선택합니다. 예를 들어 계산 할당 이벤트 항목을 선택합니다. 참고 태그는 구독 스키마의 일부여야 합니다.
차단	구독에 대한 시간 초과를 1분으로 설정합니다.

세부 정보	설정
Runnable 항목 유형	확장성 작업 Runnable 유형을 선택합니다.
Runnable ID	사용자 지정 확장성 작업을 선택합니다.

- 10 사용자 지정 확장성 작업 구독을 저장하려면 **생성**을 클릭합니다.
- 11 가상 시스템 2개(애플리케이션 VM 및 DB VM)가 포함된 **새 Blueprint**를 생성합니다.



- 12 VM을 배포하려면 **배포**를 클릭합니다.
- 13 배포 중에 이벤트가 시작되고 확장성 작업이 실행되는지 확인합니다.
- 14 태그가 올바르게 적용되었는지 확인하려면 **인프라 > 리소스 > 시스템**으로 이동합니다.

확장성 작업에 대해 알아보기

작업 기반 확장성은 vRealize Automation Cloud Assembly 내에서 간소화된 코드 스크립트를 사용하여 확장성 작업을 자동화합니다.

작업 기반 확장성은 스크립트 가능한 소규모 작업을 정의하고 EBS(Event Broker Service)에서 제공하는 특정 이벤트에 대해 시작하도록 작업을 구성할 수 있는 유연한 경량 런타임 엔진 인터페이스를 제공합니다.

vRealize Automation Cloud Assembly 내에서 이러한 코드의 확장성 작업 스크립트를 생성하여 구독에 할당할 수 있습니다. 워크플로와 마찬가지로, 확장성 작업 스크립트는 확장성 구독에 포함된 이벤트가 발생하면 트리거됩니다. 확장성 작업 스크립트는 작업 및 단계를 더 경량으로 간단하게 자동화하는 데 사용됩니다. 또한 vRealize Orchestrator 서버를 사용하여 온-프레미스에 호스팅되는 워크플로와 달리 클라우드에 호스팅됩니다. vRealize Automation Cloud Assembly를 vRealize Orchestrator 서버와 통합하는 방법에 대한 자세한 내용은 [Cloud Assembly에서 vRealize Orchestrator 통합 구성](#) 항목을 참조하십시오.

작업 기반 확장성은...

- 작고 재사용할 수 있는 스크립트 가능 작업을 통해 경량 통합 및 사용자 지정을 가능하게 함으로써 vRealize Orchestrator 워크플로를 대체할 수 있는 대안을 제공합니다.

- 재사용할 수 있는 매개 변수화된 작업이 포함된 작업 템플릿을 재사용할 수 있는 방법을 제공합니다.

확장성 작업은 사용자 정의 작업 스크립트 코드를 작성하거나 미리 정의된 스크립트 코드를 .ZIP 패키지로 가져오는 방법으로 생성할 수 있습니다. 작업 기반 확장성은 Node.js 및 Python 런타임 환경 둘 모두 지원하며 Amazon Web Services Lambda를 기반으로 합니다. 따라서 Amazon Web Services IAM(Identity and Access Management)이 포함된 활성 구독이 있어야 하며 vRealize Automation Cloud Assembly에서 Amazon Web Services를 끝점으로 구성해야 합니다. Amazon Web Services Lambda 사용 방법에 대한 자세한 내용은 [ABX: Serverless Extensibility of Cloud Assembly Services](#)를 참조하십시오.

참고 확장성 작업은 프로젝트별로 다릅니다.

확장성 작업을 생성하는 방법

vRealize Automation Cloud Assembly를 통해 확장성 구독에 사용할 확장성 작업을 생성할 수 있습니다.

확장성 작업을 사용하면 다양하게 사용자 지정할 수 있는 경량의 유연한 방법으로 사용자 정의된 스크립트 코드와 작업 템플릿을 사용하여 애플리케이션의 수명 주기를 연장할 수 있습니다. 작업 템플릿에는 확장성 작업의 기초를 설정하기 위한 미리 정의된 매개 변수가 포함됩니다.

확장성 작업은 두 가지 방법으로 생성할 수 있습니다.

- 확장성 작업 스크립트의 사용자 정의 코드를 작성합니다.

참고 확장성 작업 편집기에서 사용자 정의 코드를 작성하려면 활성 인터넷 연결이 필요할 수 있습니다.

- 배포 패키지를 확장성 작업을 위한 ZIP 패키지로 가져옵니다. 확장성 작업을 위한 ZIP 패키지 생성에 대한 자세한 내용은 [Python 런타임 확장성 작업을 위한 ZIP 패키지 생성](#) 또는 [Node.js 런타임 확장성 작업을 위한 ZIP 패키지 생성](#) 항목을 참조하십시오.

다음 단계는 Amazon Web Services를 FaaS 제공자로 사용하는 확장성 작업을 생성하는 절차를 설명합니다.

사전 요구 사항

- 활성 및 유효한 프로젝트의 멤버 자격이 필요합니다.

- Lambda 함수에 대해 구성된 Amazon Web Services 역할이 있어야 합니다. 예를 들어 AWSLambdaBasicExecutionRole가 표시됩니다.
- 클라우드 관리자 역할 또는 iam:PassRole 사용 권한이 설정되어 있어야 합니다.

절차

- 1 확장성 > 라이브러리 > 작업을 선택합니다.
- 2 새 작업을 클릭합니다.
- 3 작업의 이름을 입력하고 위치를 선택합니다.
- 4 다음을 클릭합니다.
- 5 작업 템플릿을 검색하고 선택합니다.

참고 작업 템플릿을 사용하지 않고 사용자 지정 작업을 생성하려면 **사용자 지정 스크립트**를 선택합니다.

구성 가능한 새 매개 변수가 나타납니다.

- 6 스크립트 작성 또는 패키지 가져오기를 선택합니다.
- 7 작업 런타임을 선택합니다.
- 8 작업 진입점에 대한 기본 함수 이름을 입력합니다.

참고 ZIP 패키지에서 가져온 작업의 경우, 기본 함수에는 진입점이 들어 있는 스크립트 파일의 이름도 포함되어야 합니다. 예를 들어 기본 스크립트 파일의 제목이 main.py이고 진입점이 handler (context, inputs)인 경우 기본 함수의 이름은 *main.handler*여야 합니다.

- 9 작업의 입력 및 출력 매개 변수를 정의합니다.
- 10 (선택 사항) 작업에 애플리케이션 종속성을 추가합니다.

참고 ZIP 패키지에서 가져온 작업의 경우 애플리케이션 종속성이 자동으로 추가됩니다.

- 11 시간 초과 및 메모리 제한을 정의하려면 **사용자 지정 시간 초과 및 제한 설정** 옵션을 사용하도록 설정합니다.
- 12 작업을 테스트하려면 **저장**을 클릭한 다음 **테스트**를 클릭합니다.

다음에 수행할 작업

확장성 작업을 생성하고 확인한 후에는 구독에 할당할 수 있습니다.

참고 확장성 구독은 확장성 작업의 최신 릴리스 버전을 사용합니다. 새 버전의 작업을 생성한 후 편집기 창 오른쪽 상단에 있는 **버전**을 클릭합니다. 구독에 사용할 작업 버전을 릴리스하려면 **릴리스**를 클릭합니다.

확장성 내보내기 및 가져오기 작업

vRealize Automation Cloud Assembly를 통해 다양한 프로젝트에서 사용할 확장성 작업을 내보내고 가져올 수 있습니다.

사전 요구 사항

기존 확장성 작업.

절차

1 확장성 작업을 내보냅니다.

a **확장성 > 라이브러리 > 작업**으로 이동합니다.

b 확장성 작업을 선택하고 **내보내기**를 클릭합니다.

작업 스크립트 및 해당 종속성이 로컬 환경에 ZIP 파일로 저장됩니다.

2 확장성 작업을 가져옵니다.

a **확장성 > 라이브러리 > 작업**으로 이동합니다.

b **가져오기**를 클릭합니다.

c 내보낸 확장성 작업을 선택하여 프로젝트에 할당합니다.

d **가져오기**를 클릭합니다.

참고 가져온 확장성 작업이 지정된 프로젝트에 이미 할당되어 있으면, 충돌 해결 정책을 선택하라는 메시지가 표시됩니다.

대안 작업 편집기에서 **패키지 가져오기**를 직접 선택하여 작업 스크립트를 가져올 수도 있습니다.

Python 런타임 확장성 작업을 위한 ZIP 패키지 생성

vRealize Automation Cloud Assembly 확장성 작업에서 사용되는 Python 스크립트와 종속성을 포함하는 ZIP 패키지를 생성할 수 있습니다.

확장성 작업을 위한 스크립트를 구축하는 방법에는 두 가지가 있습니다.

- vRealize Automation Cloud Assembly의 확장성 작업 편집기에서 스크립트를 직접 작성합니다.
- 로컬 환경에서 스크립트를 생성하고 이를 모든 관련 종속성과 함께 ZIP 패키지에 추가합니다.

ZIP 패키지를 사용하면 확장성 작업에 사용하기 위해 vRealize Automation Cloud Assembly로 가져올 수 있는 작업 스크립트 및 종속성의 미리 구성된 사용자 지정 템플릿을 생성할 수 있습니다.

또한 사용자 환경에서 인터넷 액세스가 부족한 경우와 같이 작업 스크립트의 종속성과 연결된 모듈을 vRealize Automation Cloud Assembly 서비스에서 확인할 수 없는 시나리오에서 ZIP 패키지를 사용할 수 있습니다.

또한 ZIP 패키지를 사용하여 여러 Python 스크립트 파일이 포함된 확장성 작업을 생성할 수 있습니다. 여러 스크립트 파일을 사용하면 확장성 작업 코드의 구조를 구성하는 데 유용할 수 있습니다.

사전 요구 사항

Python 3.3 이하 버전을 사용 중인 경우 PIP 패키지 설치 관리자를 다운로드하고 구성합니다. [Python 패키지 색인](#)을 참조하십시오.

절차

- 1 로컬 시스템에서 작업 스크립트 및 종속성에 대한 폴더를 생성합니다.

예: /home/user1/zip-action

- 2 기본 Python 작업 스크립트를 폴더에 추가합니다.

예: /home/user1/zip-action/main.py

- 3 (선택 사항) Python 스크립트에 대한 종속성을 폴더에 추가합니다.

a 종속성이 포함된 requirements.txt 파일을 생성합니다. [요구 사항 파일](#)을 참조하십시오.

b Linux 셸을 엽니다.

참고 vRealize Automation Cloud Assembly의 작업 기반 확장성의 런타임은 Linux 기반입니다. 따라서 Python 종속성이 Windows 환경에서 컴파일된 경우 생성된 ZIP 패키지는 확장성 작업을 생성하는 데 사용하지 못할 수 있습니다. 따라서 Linux 셸을 사용해야 합니다.

- c 다음 명령을 실행하여 스크립트 폴더에 requirements.txt 파일을 설치합니다.

```
pip install -r requirements.txt --target=home/user1/zip-action
```

- 4 할당된 폴더에서 스크립트 요소와 requirements.txt 파일(해당하는 경우)을 선택하고 이를 ZIP 패키지에 압축합니다.

참고 스크립트 요소와 종속성 요소는 모두 ZIP 패키지의 루트 수준에 저장해야 합니다. Linux 환경에서 ZIP 패키지를 생성할 때 패키지 콘텐츠가 루트 수준에 저장되지 않는 문제가 발생할 수 있습니다. 이 문제가 발생하는 경우 명령줄 셸에서 `zip -r` 명령을 실행하여 패키지를 생성합니다.

```
cd your_script_and_dependencies_folder
zip -r ../your_action_ZIP.zip *
```

다음에 수행할 작업

ZIP 패키지를 사용하여 확장성 작업 스크립트를 생성합니다. [확장성 작업을 생성하는 방법](#)의 내용을 참조하십시오.

Node.js 런타임 확장성 작업을 위한 ZIP 패키지 생성

vRealize Automation Cloud Assembly 확장성 작업에서 사용되는 Node.js 스크립트와 종속성을 포함하는 ZIP 패키지를 생성할 수 있습니다.

확장성 작업을 위한 스크립트를 구축하는 방법에는 두 가지가 있습니다.

- vRealize Automation Cloud Assembly의 확장성 작업 편집기에서 스크립트를 직접 작성합니다.

- 로컬 환경에서 스크립트를 생성하고 이를 모든 관련 종속성과 함께 ZIP 패키지에 추가합니다.

ZIP 패키지를 사용하면 확장성 작업에 사용하기 위해 vRealize Automation Cloud Assembly로 가져올 수 있는 작업 스크립트 및 종속성의 미리 구성된 사용자 지정 템플릿을 생성할 수 있습니다.

또한 사용자 환경에서 인터넷 액세스가 부족한 경우와 같이 작업 스크립트의 종속성과 연결된 모듈을 vRealize Automation Cloud Assembly 서비스에서 확인할 수 없는 시나리오에서 ZIP 패키지를 사용할 수 있습니다.

또한 패키지를 사용하여 여러 Node.js 스크립트 파일이 포함된 확장성 작업을 생성할 수 있습니다. 여러 스크립트 파일을 사용하면 확장성 작업 코드의 구조를 구성하는 데 유용할 수 있습니다.

절차

- 1 로컬 시스템에서 작업 스크립트 및 종속성에 대한 폴더를 생성합니다.

예: /home/user1/zip-action

- 2 기본 Node.js 작업 스크립트를 폴더에 추가합니다.

예: /home/user1/zip-action/main.js

- 3 (선택 사항) Node.js 스크립트에 대한 종속성을 폴더에 추가합니다.

- a 스크립트 폴더에 종속성이 있는 package.json 파일을 생성합니다. [package.json 파일 생성 및 package.json 파일에서 dependencies 및 devDependencies 지정을 참조하십시오.](#)

- b 명령줄 셸을 엽니다.

- c 작업 스크립트 및 종속성에 대해 생성한 폴더로 이동합니다.

```
cd /home/user1/zip-action
```

- d 다음 명령을 실행하여 스크립트 폴더에 package.json 파일을 설치합니다.

```
npm install --production
```

참고 이 명령은 폴더에 node_modules 디렉토리를 생성합니다.

- 4 할당된 폴더에서 스크립트 요소와 node_modules 디렉토리(해당하는 경우)를 선택하고 이를 ZIP 패키지에 압축합니다.

참고 스크립트 요소와 종속성 요소는 모두 ZIP 패키지의 루트 수준에 저장해야 합니다. Linux 환경에서 ZIP 패키지를 생성할 때 패키지 콘텐츠가 루트 수준에 저장되지 않는 문제가 발생할 수 있습니다. 이 문제가 발생하는 경우 명령줄 셸에서 `zip -r` 명령을 실행하여 패키지를 생성합니다.

```
cd your_script_and_dependencies_folder
zip -r ../your_action_ZIP.zip *
```

다음에 수행할 작업

ZIP 패키지를 사용하여 확장성 작업 스크립트를 생성합니다. **확장성 작업을 생성하는 방법**의 내용을 참조하십시오.

클라우드별 확장성 작업 구성

클라우드 계정을 사용할 수 있도록 확장성 작업을 구성할 수 있습니다.

확장성 작업을 생성할 때 작업을 구성하고 다음과 같은 다양한 클라우드 기반 계정에 연결할 수 있습니다.

- Microsoft Azure
- Amazon Web Services

사전 요구 사항

올바른 클라우드 계정이 필요합니다.

절차

- 1 **확장성 > 라이브러리 > 작업**을 선택합니다.
- 2 **새 작업**을 클릭합니다.
- 3 필요에 따라 작업 매개 변수를 입력합니다.
- 4 **FaaS 제공자** 드롭다운 메뉴에서 클라우드 계정 제공자를 선택하거나 **자동**을 선택합니다.

참고 **자동**을 선택하는 경우 작업에서 FaaS 제공자를 자동으로 정의합니다.

- 5 **저장**을 클릭합니다.

결과

구성된 클라우드 계정에서 사용할 수 있도록 확장성 작업이 연결됩니다.

온-프레미스 확장성 작업 구성

Amazon Web Services나 Microsoft Azure 클라우드 계정 대신 온-프레미스 FaaS 제공자를 사용하도록 확장성 작업을 구성할 수 있습니다.

확장성 작업에 대해 온-프레미스 FaaS 제공자를 사용하면 vRealize Automation Cloud Assembly 확장성 구독에서 LDAP, CMDB 또는 vCenter 데이터 센터와 같은 온-프레미스 서비스를 사용할 수 있습니다.

절차

- 1 **확장성 > 라이브러리 > 작업**을 선택합니다.
- 2 **새 작업**을 클릭합니다.
- 3 확장성 작업의 이름과 프로젝트를 입력합니다.
- 4 (선택 사항) 확장성 작업에 대한 설명을 입력합니다.
- 5 **다음**을 클릭합니다.
- 6 확장성 작업 스크립트를 생성하거나 가져옵니다.

7 **FaaS 제공자** 드롭다운 메뉴를 클릭하고 **온-프레미스**를 선택합니다.

8 새 확장성 작업을 저장하려면 **저장**을 클릭합니다.

다음에 수행할 작업

생성된 확장성 작업을 vRealize Automation Cloud Assembly 확장성 구독에서 사용합니다.

작업 흐름이란?

작업 흐름은 수명 주기 및 자동화를 더 확장하는 데 사용되는 확장성 작업 스크립트의 집합입니다.

모든 작업 흐름은 flow_start로 시작하고 flow_end로 끝납니다. 다음과 같은 작업 흐름 요소를 사용하여 여러 확장성 작업 스크립트를 함께 연결할 수 있습니다.

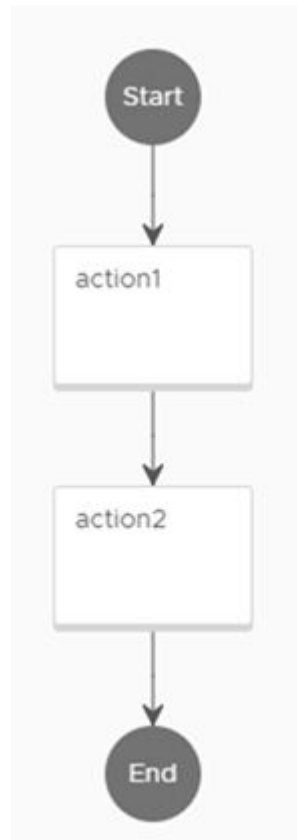
- **순차적 작업 흐름** - 순차적으로 실행되는 여러 확장성 작업 스크립트입니다.
- **분기 작업 흐름** - 경로를 분할하여 동일한 결과를 제공하는 여러 확장성 작업 스크립트 또는 흐름입니다.
- **참여 작업 흐름** - 서로 조인되어 동일한 결과를 제공하는 여러 확장성 작업 스크립트 또는 흐름입니다.
- **조건부 작업 흐름** - 조건이 충족되었을 때 실행되는 여러 확장성 작업 스크립트 또는 흐름입니다.

순차적 작업 흐름

순차적으로 실행되는 여러 확장성 작업 스크립트입니다.

```
version: "1"
flow:
  flow_start:
    next: action1
  action1:
    action: <action_name>
    next: action2
  action2:
    action: <action_name>
    next: flow_end
```

참고 이전 작업을 next: 작업으로 할당하여 이전 작업으로 루프백할 수 있습니다. 예를 들어 이 예에서는 next: flow_end 대신 next: action1을 입력하여 action1을 다시 실행하고 작업 시퀀스를 다시 시작할 수 있습니다.

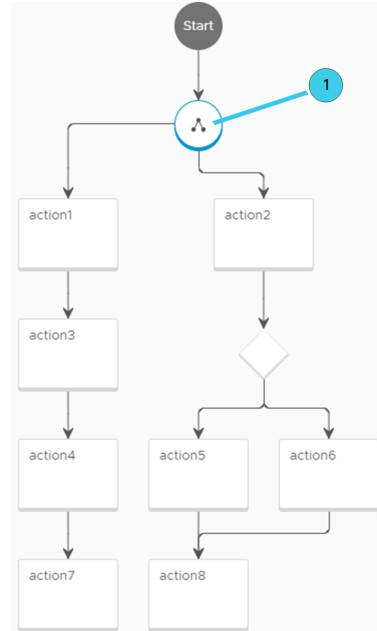


분기 작업 흐름

경로를 분할하여 동일한 출력을 제공하는 여러 확장성 작업 스크립트 또는 흐름입니다.

```
version: "1"
flow:
  flow_start:
    next: forkAction
  forkAction:
    fork:
      next: [action1, action2]
  action1:
    action: <action_name>
    next: action3
  action3:
    action: <action_name>
    next: action4
  action4:
    action: <action_name>
    next: action7
  action7:
    action: <action_name>
  action2:
    action: <action_name>
```

참고 이전 작업을 `next: 작업으로 할당하여` 이전 작업으로 루프백할 수 있습니다. 예를 들어 `next: flow_end`를 사용하여 작업 흐름을 종료하는 대신 `next: action1`을 입력하면 `action1`을 다시 실행하여 작업 시퀀스를 다시 시작할 수 있습니다.



❶ 분기 요소

참여 작업 흐름

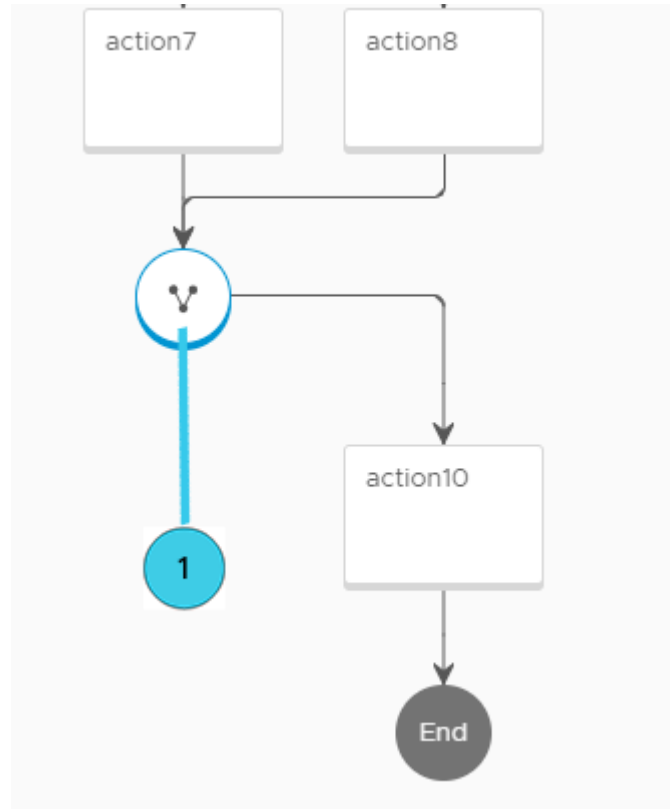
경로를 함께 조인하여 동일한 출력을 제공하는 여러 확장성 작업 스크립트 또는 흐름입니다.

```

version: "1"
action7:
  action: <action_name>
  next: joinElement
action8:
  action: <action_name>
  next: joinElement
joinElement:
  join:
    type: all
    next: action10
action10:
  action: <action_name>
  next: flow_end

```

참고 이전 작업을 `next:` 작업으로 할당하여 이전 작업으로 루프백할 수 있습니다. 예를 들어 이 예에서는 `next: flow_end` 대신 `next: action1`을 입력하여 **action1**을 다시 실행하고 작업 시퀀스를 다시 시작할 수 있습니다.



① 참여 요소

조건부 작업 흐름

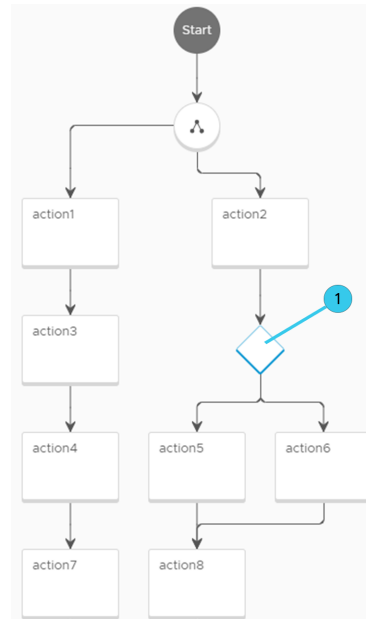
스위치 요소를 사용하여 조건이 충족되면 실행되는 여러 확장성 작업 스크립트 또는 흐름입니다.

경우에 따라서는 조건이 `true`와 같아야 작업이 실행됩니다. 그 외의 경우에는 이 예시에 나와 있듯이 매개 변수 값이 충족되어야 작업이 실행될 수 있습니다. 조건이 충족되지 않으면 작업 흐름이 실패합니다.

```

version: 1
id: 1234
name: Test
inputs: ...
outputs: ...
flow:
  flow_start:
    next: forkAction
  forkAction:
    fork:
      next: [action1, action2]
  action1:
    action: <action_name>
    next: action3
  action3:
    action: <action_name>
    next: action4
  action4:
    action: <action_name>
    next: action7
  action7:
    action: <action_name>
    next: joinElement
  action2:
    action: <action_name>
    next: switchAction
  switchAction:
    switch:
      "${1 == 1}": action5
      "${1 != 1}": action6
  action5:
    action: <action_name>
    next: action8
  action6:
    action: <action_name>
    next: action8
  action8:
    action: <action_name>

```



① 스위치 요소

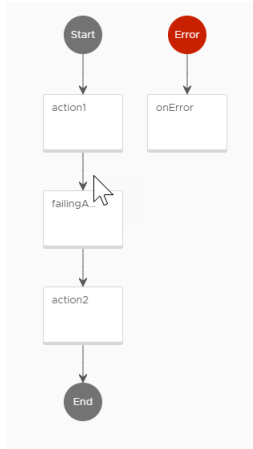
참고 이전 작업을 `next: 작업으로 할당하여` 이전 작업으로 루프백할 수 있습니다. 예를 들어 `next: flow_end`를 사용하여 작업 흐름을 종료하는 대신 `next: action1`을 입력하면 `action1`을 다시 실행하여 작업 시퀀스를 다시 시작할 수 있습니다.

작업 흐름에서 오류 처리기를 사용하는 방법

오류 처리기 요소를 사용하여 지정된 흐름 단계에서 오류를 생성하도록 작업 흐름을 구성할 수 있습니다.

오류 처리기 요소에는 두 가지 입력이 필요합니다.

- 실패한 작업의 지정된 오류 메시지.
- 작업 흐름 입력.



흐름의 작업이 실패하고 작업 흐름에 오류 처리기 요소가 포함되어 있는 경우 오류 메시지가 생성되면서 작업 실패를 경고합니다. 오류 처리기는 자체 작업입니다. 다음 스크립트는 작업 흐름에서 사용될 수 있는 오류 처리기의 예입니다.

```
def handler(context, inputs):

    errorMsg = inputs["errorMsg"]
    flowInputs = inputs["flowInputs"]

    print("Flow execution failed with error {0}".format(errorMsg))
    print("Flow inputs were: {0}".format(flowInputs))

    outputs = {
        "errorMsg": errorMsg,
        "flowInputs": flowInputs
    }

    return outputs
```

[작업 실행] 창에서 성공한 실행과 실패한 실행을 볼 수 있습니다.

선택	상태	작업	작업 ID
<input type="checkbox"/>	완료됨	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/>	실패	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/>	완료됨	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/>	완료됨	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6

이 예에서 오류 처리기 요소가 들어 있는 flow-with-handler 작업 흐름이 성공적으로 실행되었습니다. 하지만 흐름의 작업 중 하나가 실패하여 오류 처리기가 시작되었고 오류가 생성되었습니다.

작업 실행을 추적하는 방법

작업 실행 탭에는 구독에서 트리거된 확장성 작업 및 해당 상태에 대한 로그가 표시됩니다.

확장성 > 작업 > 작업 실행을 사용하여 작업 실행 로그를 볼 수 있습니다. 또한 하나 이상의 속성을 한 번에 사용하여 작업 실행 목록을 필터링할 수도 있습니다. 실행 ID를 클릭하면 개별 작업 실행에 대한 추가적인 세부 정보를 볼 수 있습니다.

실패한 확장성 작업 실행 문제 해결

확장성 작업 실행이 실패하면 문제 해결 단계를 수행하여 문제를 해결할 수 있습니다.

작업 실행이 실패하면 오류 메시지, 실패 상태 및 실패 로그가 수신될 수 있습니다. 작업 실행이 실패했다면 그 원인은 배포 실패이거나 코드 실패입니다.

문제	해결 방법
배포 실패	이러한 실패는 클라우드 계정 구성, 작업 배포 또는 작업의 배포를 방해할 수 있는 기타 종속성과 관련된 문제의 결과입니다. 사용한 프로젝트가 구성된 클라우드 계정 내에 정의되어 있는지 그리고 기능을 실행할 수 있는 사용 권한이 있는지 확인합니다. 작업을 다시 시작하기 전에 작업의 세부 정보 페이지 내에서 특정 프로젝트를 대상으로 작업을 테스트할 수 있습니다.
코드 실패	이러한 실패는 잘못된 스크립트 또는 코드의 결과입니다. [작업 실행] 로그를 사용하여 문제를 해결하고 잘못된 스크립트를 수정합니다.

확장성 워크플로 구독

vRealize Orchestrator에 호스팅된 워크플로를 vRealize Automation Cloud Assembly와 함께 사용하여 애플리케이션 수명 주기를 연장할 수 있습니다.

vRealize Orchestrator 워크플로 구독을 사용하여 가상 시스템 속성을 수정하는 방법

기존 vRealize Orchestrator 워크플로를 사용하여 가상 시스템 속성을 수정하고 가상 시스템을 Active Directory에 추가할 수 있습니다.

구독 스키마는 EBS(Event Broker Service) 메시지에 대한 페이로드의 형식을 정의합니다. 워크플로 내에서 EBS 메시지 페이로드를 받아서 사용하려면 "inputProperties" 워크플로 입력 매개 변수를 정의해야 합니다.

사전 요구 사항

- 클라우드 관리자 사용자 역할이 있어야 합니다.
- 기존 vRealize Orchestrator 온 프레미스 워크플로가 있어야 합니다.
- 성공적인 통합 및 vRealize Orchestrator 클라이언트 서버에 대한 연결이 있어야 합니다.

절차

- 1 **확장성 > 구독**을 선택합니다.

2 새 구독을 클릭합니다.

3 다음 매개 변수를 사용하여 구독을 생성합니다.

매개 변수	값
이름	RenameVM
이벤트 항목	원하는 vRealize Orchestrator 통합에 적합한 이벤트 항목을 선택합니다. 예를 들어 계산 할당 이벤트 항목을 선택합니다.
차단/비차단	비차단
Runnable 항목	vRealize Orchestrator Runnable 유형을 선택합니다.
Runnable ID	원하는 워크플로를 선택합니다. 예: Set VM name

4 구독을 저장하려면 **생성**을 클릭합니다.

5 Blueprint를 생성하거나 기존 Blueprint를 배포하여 구독을 할당하고 활성화합니다.

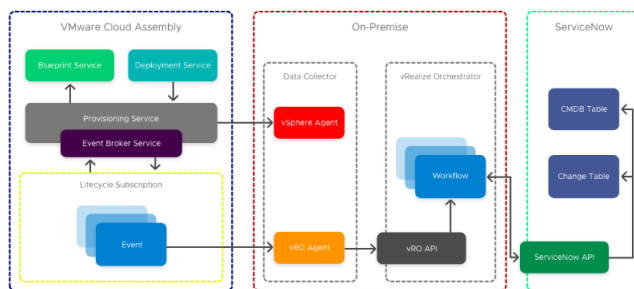
다음에 수행할 작업

다음 방법 중 하나를 사용하여 워크플로가 성공적으로 시작되었는지 확인합니다.

- 워크플로 실행이 기록되는지 확인합니다(**확장성 > 작업 > 워크플로 실행**).
- vRealize Orchestrator 클라이언트를 열고 워크플로로 이동한 후 상태를 확인하거나, 해당하는 로그 탭을 열어서 워크플로 상태를 확인합니다.

vRealize Orchestrator 워크플로를 사용하여 ITSM용 Cloud Assembly와 ServiceNow를 통합하는 방법

vRealize Orchestrator 호스팅된 워크플로를 사용하여 ITSM 규정 준수를 위해 vRealize Automation Cloud Assembly와 ServiceNow를 통합할 수 있습니다.



엔터프라이즈 사용자는 규정 준수를 위해 일반적으로 Cloud Management Platform을 ITSM(IT 서비스 관리) 및 CMDB(구성 관리 데이터베이스) 플랫폼과 통합합니다. 이 예시를 사용하면 vRealize Orchestrator 호스팅된 워크플로를 사용하여 CMDB 및 ITSM을 위해 vRealize Automation Cloud Assembly를 ServiceNow와 통합할 수 있습니다. vRealize Orchestrator 통합 및 워크플로를 사용할 때 기능 태그는 서로 다른 환경에 대해 인스턴스가 여러 개 있는 경우에 특히 유용합니다. 기능 태그에 대한 자세한 내용은 [vRealize Automation Cloud Assembly에서 기능 태그 사용](#) 항목을 참조하십시오.

참고 확장성 작업 스크립트를 사용하여 ServiceNow를 vRealize Automation Cloud Assembly와 통합할 수도 있습니다. 확장성 작업 스크립트를 사용한 ServiceNow 통합에 대한 자세한 내용은 [확장성 작업을 사용하여 Cloud Assembly와 ServiceNow를 통합하는 방법](#) 항목을 참조하십시오.

이 예시에서 ServiceNow 통합은 3개의 최상위 수준 워크플로로 구성되며, 각 워크플로에는 자체 구독이 포함되어 있어 각 구성 요소를 개별적으로 업데이트하고 반복할 수 있습니다.

- 이벤트 구독 진입점 - 기본 로깅을 사용하며, 요청하는 사용자 및 vCenter VM(해당하는 경우)을 식별합니다.
- 통합 워크플로 - 개체 및 피드 입력을 기술 워크플로로 분리하고 로깅, 속성 및 출력 업데이트를 처리합니다.
- 기술 워크플로 - 페이로드 이외의 추가적인 가상 시스템 속성을 사용하여 CMDB CI, CR 및 CAS IaaS API를 생성하기 위한 ServiceNow API용 다운스트림 시스템 통합입니다.

사전 요구 사항

- 독립형 또는 클러스터링된 vRealize Orchestrator 환경.
- vRealize Automation Cloud Assembly의 vRealize Orchestrator 통합. vRealize Automation Cloud Assembly와 독립형 vRealize Orchestrator 통합에 대한 자세한 내용은 [Cloud Assembly에서 vRealize Orchestrator 통합 구성](#) 항목을 참조하십시오.

절차

- 1 여러 워크플로에 사용되는 공통 구성이 포함된 구성 파일을 vRealize Orchestrator에서 생성하고 저장합니다.
- 2 CAS API 토큰을 1단계의 구성 파일과 동일한 위치에 저장합니다.

참고 CAS API 토큰에는 만료 기간이 있습니다.

- 3 제공된 스크립트 요소를 사용하여 vRealize Orchestrator에서 워크플로를 생성합니다. 이 스크립트는 REST 호스트를 참조하고 해당 위치를 찾습니다. 또한 추가적인 인증 헤더로 추가되는 토큰의 선택적 매개 변수를 사용하는 REST 작업을 표준화합니다.

```
var configPath = "CS"
var configName = "environmentConfig"
var attributeName = "CASRestHost"

//get REST Host from configuration element
```

```

var restHost =
System.getModule("au.com.cs.example").getRestHostFromConfig(configPath, configName, attribute
Name)

var ConfigurationElement =
System.getModule("au.com.cs.example").getConfigurationElementByName(configName, configPath);
System.debug("ConfigurationElement:" + ConfigurationElement);
var casToken = ConfigurationElement.getAttributeWithKey("CASToken")["value"]
if(!casToken){
    throw "no CAS Token";
}
//REST Template
var opName = "casLogin";
var opTemplate = "/iaas/login";
var opMethod = "POST";

// create the REST operation:
var opLogin =
System.getModule("au.com.cs.example").createOp(restHost, opName, opMethod, opTemplate);

//cas API Token
var contentObject = {"refreshToken":casToken}
postContent = JSON.stringify(contentObject);

var loginResponse =
System.getModule("au.com.cs.example").executeOp(opLogin, null, postContent, null) ;

try{
    var tokenResponse = JSON.parse(loginResponse)['token']
    System.debug("token: " + tokenResponse);
} catch (ex) {
    throw ex + " No valid token";
}

//REST Template Machine Details
var opName = "machineDetails";
var opTemplate = "/iaas/machines/" + resourceId;
var opMethod = "GET";

var bearer = "Bearer " + tokenResponse;

var opMachine =
System.getModule("au.com.cs.example").createOp(restHost, opName, opMethod, opTemplate);

// (Rest Operation, Params, Content, Auth Token)
var vmResponse =
System.getModule("au.com.cs.example").executeOp(opMachine, null, "", bearer) ;

try{
    var vm = JSON.parse(vmResponse);
} catch (ex) {
    throw ex + " failed to parse vm details"
}

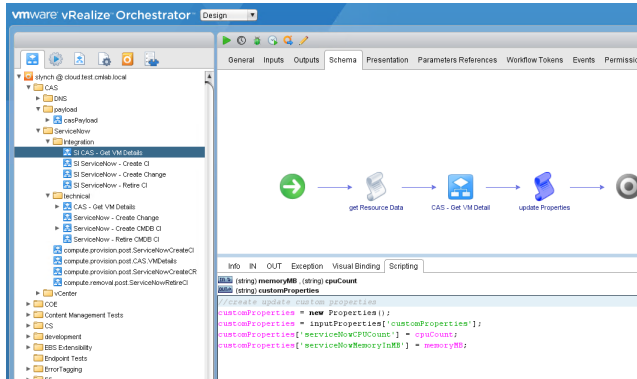
System.log("cpuCount: " + vm["customProperties"]["cpuCount"]);

```

```
System.log("memoryInMB: " + vm["customProperties"]["memoryInMB"]);

cpuCount = vm["customProperties"]["cpuCount"];
memoryMB = vm["customProperties"]["memoryInMB"];
```

이 스크립트는 cpuCount 및 memoryMB 출력을 상위 워크플로에 보내고 기존 customProperties 속성을 업데이트합니다. 이러한 값은 CMDB를 생성할 때 이후 워크플로에서 사용할 수 있습니다.



- 4 ServiceNow CMDB CI 생성 스크립트 요소를 워크플로에 추가합니다. 이 요소는 구성 항목을 사용하여 ServiceNow REST 호스트의 위치를 찾고, cmdb_ci_vmware_instance 표의 REST 작업을 생성하며, 사후 데이터를 위해 워크플로 입력에 기반한 콘텐츠 개체 문자열을 생성한 후, 반환되는 sys_id를 출력합니다.

```
var configPath = "CS"
var configName = "environmentConfig"
var attributeName = "serviceNowRestHost"
var tableName = "cmdb_ci_vmware_instance"

//get REST Host from configuration element
var restHost =
System.getModule("au.com.cs.example").getRestHostFromConfig(configPath,configName,attribute
Name)

//REST Template
var opName = "serviceNowCreatCI";
var opTemplate = "/api/now/table/" + tableName;
var opMethod = "POST";

// create the REST operation:
var opCI =
System.getModule("au.com.cs.example").createOp(restHost,opName,opMethod,opTemplate);

//cmdb_ci_vm_vmware table content to post;
var contentObject = {};
contentObject["name"] = hostname;
contentObject["cpus"] = cpuTotalCount;
contentObject["memory"] = MemoryInMB;
contentObject["correlation_id"] = deploymentId
contentObject["disks_size"] = diskProvisionGB
contentObject["location"] = "Sydney";
```

```

contentObject["vcenter_uuid"] = vcUuid;
contentObject["state"] = "On";
contentObject["owned_by"] = owner;

postContent = JSON.stringify(contentObject);
System.log("JSON: " + postContent);

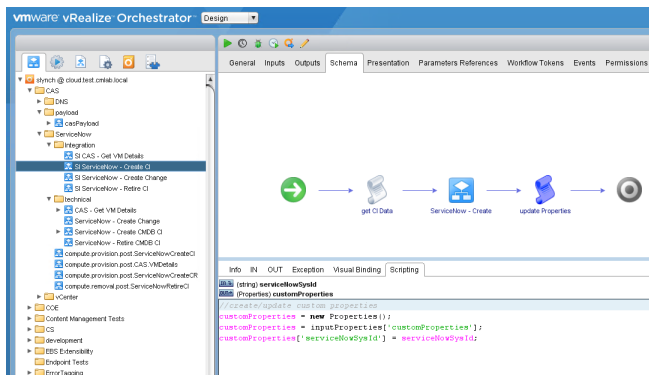
// (Rest Operation, Params, Content, Auth Token)
var ciResponse =
System.getModule("au.com.cs.example").executeOp(opCI,null,postContent,null) ;

try{
    var cmdbCI = JSON.parse(ciResponse);
} catch (ex) {
    throw ex + " failed to parse ServiceNow CMDB response";
}

serviceNowSysId = cmdbCI['result']['sys_id'];

```

- 5 하위 워크플로의 출력을 사용하여 기존 customProperties를 사용하는 속성 개체를 생성하고 ServiceNow의 값으로 serviceNowSysId 속성을 덮어씁니다. 이 고유 ID는 삭제 시 CMDB에서 인스턴스를 회수된 것으로 표시하기 위해 사용됩니다.



결과

vRealize Automation Cloud Assembly가 ITSM ServiceNow와 통합되었습니다. 워크플로를 사용하여 vRealize Automation Cloud Assembly에 ServiceNow를 통합할 수 있는 방법에 대한 자세한 내용은 [ServiceNow 통합을 위해 vRealize Orchestrator를 사용하여 Cloud Assembly 확장을 참조하십시오.](#)

워크플로 구독에 대해 알아보기

vRealize Orchestrator와 vRealize Automation Cloud Assembly를 통합을 사용하면 워크플로를 사용하여 애플리케이션의 수명 주기를 확장할 수 있습니다.

vRealize Automation Cloud Assembly에는 기존의 온-프레미스 워크플로를 구독에 가져오고 연결할 수 있도록 vRealize Orchestrator가 통합되어 있습니다. 이러한 워크플로는 vRealize Orchestrator 서버에 유지됩니다. vRealize Orchestrator 클라이언트에서만 워크플로를 생성, 수정 및 삭제할 수 있습니다.

vRealize Orchestrator 워크플로 생성 모범 사례

워크플로 구독은 특정 항목 스키마를 기반으로 합니다. 구독에서 vRealize Orchestrator 워크플로를 시작할 수 있으려면 이벤트 데이터를 사용하도록 올바른 입력 매개 변수로 워크플로를 구성해야 합니다.

워크플로 입력 매개 변수

사용자 지정 워크플로에는 모든 매개 변수 또는 페이로드의 모든 데이터를 사용하는 단일 매개 변수가 포함될 수 있습니다.

단일 매개 변수를 사용하려면 유형이 Properties이고 이름이 inputProperties인 하나의 매개 변수를 구성합니다.

워크플로 출력 매개 변수

사용자 지정 워크플로에는 회신 이벤트 항목 유형에 필요한 후속 이벤트와 관련된 출력 매개 변수가 포함될 수 있습니다.

이벤트 항목에 회신이 필요한 경우 워크플로 출력 매개 변수가 회신 스키마와 일치해야 합니다.

워크플로 실행을 추적하는 방법

워크플로 실행 탭은 구독에서 트리거한 워크플로와 해당 상태에 대한 로그를 보여줍니다.

확장성 > 작업 > 워크플로 실행을 사용하여 워크플로 실행에 대한 로그를 볼 수 있습니다.

실패한 워크플로 구독 문제 해결

워크플로 구독이 실패한 경우 문제 해결 단계를 수행하여 문제를 해결할 수 있습니다.

워크플로 실행이 실패하면 워크플로 구독이 시작되지 않거나 완료되지 않을 수 있습니다. 워크플로 실행 실패는 몇 가지 일반적인 문제로 인해 발생할 수 있습니다.

문제	원인	해결 방법
vRealize Orchestrator 워크플로 구독이 시작되지 않거나 완료되지 않습니다.	이벤트 메시지가 수신되면 사용자 지정 워크플로를 실행하도록 워크플로 구독을 구성했지만 워크플로가 실행되지 않거나 완료되지 않습니다.	<ol style="list-style-type: none"> 1 워크플로 구독이 올바르게 저장되었는지 확인합니다. 2 워크플로 구독 조건이 올바르게 구성되었는지 확인합니다. 3 vRealize Orchestrator에 지정된 워크플로가 포함되어 있는지 확인합니다. 4 vRealize Orchestrator에 워크플로가 올바르게 구성되었는지 확인합니다.
승인 요청 vRealize Orchestrator 워크플로 구독이 실행되지 않습니다.	vRealize Orchestrator 워크플로를 실행하도록 사전 승인 또는 사후 승인 워크플로 구독을 구성했습니다. 서비스 카탈로그에서 정의된 조건과 일치하는 시스템이 요청될 때 워크플로가 실행되지 않습니다.	<p>승인 워크플로 구독을 성공적으로 실행하려면 모든 구성 요소가 올바르게 구성되어 있는지 확인해야 합니다.</p> <ol style="list-style-type: none"> 1 승인 정책이 활성화되고 올바르게 적용되었는지 확인합니다. 2 워크플로 구독이 올바르게 구성 및 저장되었는지 확인합니다. 3 이벤트 로그에서 승인 관련 메시지를 검토합니다.
승인 요청 vRealize Orchestrator 워크플로 구독이 거부됩니다.	지정된 vRealize Orchestrator 워크플로를 실행하는 사전 승인 또는 사후 승인 워크플로 구독을 구성했지만 요청이 외부 승인 수준에서 거부됩니다. 한 가지 가능한 원인은 vRealize Orchestrator의 내부 워크플로 실행 오류입니다. 예를 들어 워크플로가 누락되었거나 vRealize Orchestrator 서버가 실행 중이지 않을 수 있습니다.	<ol style="list-style-type: none"> 1 로그에서 승인 관련 메시지를 검토합니다. 2 vRealize Orchestrator 서버가 실행 중인지 확인합니다. 3 vRealize Orchestrator에 지정된 워크플로가 포함되어 있는지 확인합니다.

확장성 구독에 대해 알아보기

확장성 작업 또는 vRealize Orchestrator에 호스팅된 워크플로를 확장성 구독과 함께 사용하여 애플리케이션 수명 주기를 연장할 수 있습니다.

환경에서 트리거 이벤트가 발생하면 구독이 시작되고 지정된 워크플로 또는 확장성 작업이 실행됩니다. 시스템 이벤트는 이벤트 로그에서 보고, 워크플로 실행은 워크플로 실행 창에서 보고 작업 실행은 작업 실행 창에서 볼 수 있습니다. 구독은 프로젝트별로 다릅니다. 즉, 지정된 프로젝트를 통해 Blueprint 및 배포에 연결됩니다.

확장성 용어

vRealize Automation Cloud Assembly에서 확장성 및 구독을 사용할 때 구독과 이벤트 브로커 서비스에만 해당하는 용어가 나올 수 있습니다.

표 6-4. 확장성 용어

용어	설명
이벤트 항목	동일한 논리적 의도와 동일한 구조를 갖는 일련의 이벤트를 설명합니다. 모든 이벤트는 이벤트 항목의 인스턴스입니다. 특정 이벤트 항목에 차단 매개 변수를 할당할 수 있습니다. 자세한 내용은 차단 이벤트 항목 항목을 참조하십시오.
이벤트	생산자 또는 생산자가 관리하는 엔티티의 상태 변경을 나타냅니다. 이벤트는 이벤트 발생에 대한 정보를 기록하는 엔티티입니다.
이벤트 브로커 서비스	생산자가 구독 소비자에게 게시하는 메시지를 디스패치하는 서비스입니다.
페이로드	해당 이벤트 항목에 관련된 모든 관련 속성이 포함된 이벤트 데이터입니다.
구독	이벤트 항목을 구독하고 알림을 트리거하는 조건을 정의하여 구독자가 이벤트 관련 알림을 받는 것에 관심이 있음을 나타냅니다. 구독은 애플리케이션 수명 주기의 일부를 자동화하는 데 사용되는 트리거 이벤트에 확장성 작업 또는 워크플로를 연결합니다.
구독자	구독 정의를 기반으로 이벤트 브로커 서비스에 게시된 이벤트를 통해 알림을 받는 사용자입니다. 구독자를 소비자라고 할 수도 있습니다.
시스템 관리자	vRealize Automation Cloud Assembly를 사용하여 테넌트 워크플로 구독 및 시스템 워크플로 구독을 생성하고, 읽고, 업데이트하고, 삭제할 수 있는 권한을 가진 사용자입니다.
워크플로 구독	vRealize Orchestrator 워크플로를 트리거하는 이벤트 항목 및 조건을 지정합니다.
작업 구독	확장성 작업 실행을 트리거하는 이벤트 항목 및 조건을 지정합니다.
워크플로	vRealize Automation Cloud Assembly 내에 통합된 vRealize Orchestrator 워크플로입니다. 이러한 워크플로를 구독 내의 이벤트에 연결할 수 있습니다.
확장성 작업	구독 내에서 트리거 이벤트 이후 실행될 수 있는 효율적인 코드 스크립트입니다. 확장성 작업은 워크플로와 비슷하지만 더 간단합니다. 확장성 작업은 vRealize Automation Cloud Assembly 내에서 사용자 지정할 수 있습니다.
작업 실행	작업 실행 탭을 통해 액세스할 수 있습니다. 작업 실행은 트리거 이벤트에 대한 응답으로 실행된 확장성 작업의 세부 로그입니다.

차단 이벤트 항목

일부 이벤트 항목은 차단 이벤트를 지원합니다. 확장성 구독의 동작은 항목이 이러한 이벤트 유형을 지원하는지 여부와 구독을 구성하는 방법에 따라 다릅니다.

vRealize Automation Cloud Assembly 확장성 구독은 비차단 및 차단 이벤트 항목이라는 두 가지의 광범위한 이벤트 항목 유형을 사용할 수 있습니다. 이벤트 항목 유형은 확장성 구독의 동작을 정의합니다.

비차단 이벤트 항목

비차단 이벤트 항목은 비차단 구독만 생성할 수 있도록 허용합니다. 비차단 구독은 비동기식으로 트리거되고 구독이 트리거되는 순서에 의존할 수 없습니다.

차단 이벤트 항목

일부 이벤트 항목은 차단을 지원합니다. 구독이 차단으로 표시되는 경우 설정된 조건을 충족하는 모든 메시지는 차단 구독의 **Runnable** 항목이 실행될 때까지 일치하는 조건이 있는 다른 구독에서 수신되지 않습니다.

차단 구독은 우선 순위에 따라 실행됩니다. 가장 높은 우선 순위 값은 0입니다. 동일한 이벤트 항목에 대해 우선 순위 수준이 같은 둘 이상의 차단 구독이 있는 경우 구독은 구독 이름을 기반으로 알파벳 역순으로 실행됩니다. 모든 차단 구독이 처리되고 나면 메시지가 모든 비차단 구독에 동시 전송됩니다. 차단 구독은 동기식으로 실행되기 때문에 후속 구독에 알림이 표시될 때 변경된 이벤트 페이로드에 업데이트된 이벤트가 포함됩니다.

차단 이벤트 항목을 사용하여 서로 종속 관계에 있는 여러 구독을 관리할 수 있습니다.

예를 들어 두 번째 구독이 첫 번째 구독의 결과에 따라 달라지는 두 개의 프로비저닝 워크플로 구독을 가질 수 있다고 가정하면 첫 번째 구독은 프로비저닝 중에 속성을 변경하고 두 번째 구독은 시스템 이름과 같은 새 속성을 파일 시스템에 기록합니다. **ChangeProperty** 구독은 0으로 우선 순위가 지정되고

RecordProperty는 두 번째 구독이 첫 번째 구독의 결과를 사용하므로 1로 우선 순위가 지정됩니다. 시스템이 프로비저닝되면 **ChangeProperty** 구독이 실행을 시작합니다. **RecordProperty** 구독 조건은 사후 프로비저닝 조건을 기반으로 하기 때문에 이벤트가 **RecordProperty** 구독을 트리거합니다. 하지만

ChangeProperty 워크플로는 차단 워크플로이므로 완료되기 전까지 이벤트가 수신되지 않습니다. 시스템 이름이 변경되고 첫 번째 워크플로 구독이 완료되면 두 번째 워크플로 구독이 실행되면서 파일 시스템에 이름을 기록합니다.

복구 Runnable 항목

차단 이벤트 항목의 경우 복구 **Runnable** 항목을 구독에 추가할 수 있습니다. 구독의 복구 **Runnable** 항목은 기본 **Runnable** 항목이 실패하는 경우 실행됩니다. 예를 들어 기본 **Runnable** 항목이 **ServiceNow**와 같은 CMDB 시스템에서 기록을 생성하는 워크플로인 워크플로 구독을 생성할 수 있습니다. 워크플로 구독이 실패하더라도 일부 기록이 CMDB 시스템에서 생성될 수 있습니다. 이 시나리오에서는 복구 **Runnable** 항목을 사용하여 실패한 **Runnable** 항목으로 인해 CMDB 시스템에 남아 있는 기록을 정리할 수 있습니다.

서로 종속 관계에 있는 여러 개의 구독이 포함된 사용 사례의 경우 `ebs.recover.continuation` 속성을 복구 **Runnable** 항목에 추가할 수 있습니다. 이 속성을 사용하면 현재 구독이 실패하는 경우 체인의 다음 구독으로 확장성 서비스를 계속해야 할지 지시할 수 있습니다.

vRealize Automation Cloud Assembly에서 제공되는 이벤트 항목

vRealize Automation Cloud Assembly에는 미리 정의된 이벤트 항목이 포함되어 있습니다.

이벤트 항목

이벤트 항목은 비슷한 이벤트를 그룹화하는 범주입니다. 구독에 할당된 이벤트 항목은 구독을 트리거할 이벤트를 정의합니다. 다음의 이벤트 항목은 vRealize Automation Cloud Assembly에 기본적으로 제공됩니다. 모든 항목은 리소스의 사용자 지정 속성 또는 태그를 추가하거나 업데이트하는 데 사용될 수 있습니다. vRealize Orchestrator 워크플로 또는 확장성 작업이 실패하면 해당하는 작업도 실패합니다.

표 6-5. Cloud Assembly 이벤트 항목

이벤트 항목	차단 가능	설명
Blueprint.configuration	아니요	Blueprint 생성 또는 삭제와 같은 Blueprint 구성 이벤트가 발생할 때 생성됩니다. 외부 시스템에 이러한 이벤트를 알리는 데 유용할 수 있습니다.
Blueprint.version.configuration	아니요	버전 생성, 릴리스, 릴리스 취소 또는 복원과 같은 새로운 Blueprint 버전 관리 이벤트가 발생할 때 생성됩니다. 이 이벤트 항목은 타사 버전 제어 시스템 통합에 유용할 수 있습니다.
Compute allocation	예	resourcenames 및 hostselections가 할당되기 전에 생성되는 이벤트입니다. 이러한 속성 둘 모두 이 단계에서 수정할 수 있습니다.
Compute post provision	예	리소스 프로비저닝이 완료된 이후 생성되는 이벤트입니다.
Compute post removal	예	계산 리소스가 제거된 이후 생성되는 이벤트입니다.
Compute provision	예	하이퍼바이저에서 리소스가 프로비저닝되기 전에 생성되는 이벤트입니다. 참고 할당된 IP 주소를 변경할 수 있습니다.
Compute removal	예	리소스가 제거되기 전에 생성되는 이벤트입니다.
Compute reservation	예	예약 시 생성되는 이벤트입니다. 참고 배치 순서를 변경할 수 있습니다.
Deployment action completed	예	배포 작업의 완료 후에 생성됩니다.
Deployment action requested	예	배포 작업의 완료 전에 생성됩니다.
Deployment completed	예	Blueprint 또는 카탈로그 요청의 배포 후에 생성됩니다.
Deployment onboarded	아니요	새 배포가 온보딩될 때 생성됩니다.
Deployment requested	예	Blueprint 또는 카탈로그 요청의 배포 전에 생성됩니다.

표 6-5. Cloud Assembly 이벤트 항목 (계속)

이벤트 항목	차단 가능	설명
Deployment resource action completed	예	리소스 작업의 배포 후에 생성됩니다.
Deployment resource action requested	예	리소스 작업의 배포 전에 생성됩니다.
Deployment resource completed	예	배포 리소스의 프로비저닝 후에 생성됩니다.
Deployment resource requested	예	배포 리소스의 프로비저닝 전에 생성됩니다.
Disk allocation	예	디스크 리소스의 사전 할당을 위해 생성됩니다.
Disk post removal	예	디스크 리소스의 삭제 후에 생성됩니다.
Disk post resize	예	디스크 리소스의 크기 조정 후에 생성됩니다.
EventLog	예	관련 이벤트를 기록합니다.
Kubernetes cluster allocation	예	Kubernetes 클러스터에 대한 리소스 사전 할당에 대해 생성됩니다.
Kubernetes cluster post provision	예	Kubernetes 클러스터의 프로비저닝 후에 생성됩니다.
Kubernetes cluster post removal	예	Kubernetes 클러스터의 삭제 후에 생성됩니다.
Kubernetes cluster provision	예	Kubernetes 클러스터의 프로비저닝 전에 생성됩니다.
Kubernetes cluster removal	예	Kubernetes 클러스터 삭제 프로세스의 시작 전에 생성됩니다.
Load balancer post provision	예	로드 밸런서의 프로비저닝 후에 생성됩니다.
Network Configure	예	계산을 할당하는 동안 네트워크가 구성되면 생성되는 이벤트입니다. 참고 네트워크 구성 항목은 여러 IP 주소/NIC를 지원합니다.
Project Lifecycle	아니오	프로젝트가 생성, 업데이트 또는 삭제되면 생성되는 이벤트입니다.

이벤트 스키마

이벤트 항목을 추가한 후 이벤트 스키마를 볼 수 있습니다. 이 스키마는 이벤트의 페이로드, 또는 `inputProperties`의 구조를 정의합니다.

확장성 이벤트 로그

확장성 이벤트 탭에는 환경 안에서 발생한 모든 이벤트의 목록이 표시됩니다.

확장성 > 이벤트를 사용하여 확장성 이벤트 로그를 볼 수 있습니다. 또한 하나 이상의 속성을 한 번에 사용하여 이벤트 목록을 필터링할 수도 있습니다. 이벤트 ID를 클릭하면 개별 이벤트에 대한 추가적인 세부 정보를 볼 수 있습니다.

배포 Blueprint 인프라 **확장성** 마켓플레이스 단계별 설정

이벤트 3442개 항목

필터...

ID	타임 스탬프	이벤트 항목	사용자 이름	대상 ID	설명
d4b40f20-9792-4c48-aec0-0a16ce7eaabe	20. 01. 14. 오후 6:14	Blueprint configuration	pmartini@vmware.com		
1d86cdf8-9555-d312-1836-00d8b2a262e6	20. 01. 14. 오후 4:46	Project Lifecycle Event Topic	project-service	1d0b482d-8396-46a1-a493-54ed2d299ec8	update
dfcf544c-d263-c30c-3b58-23b31be25f6a	20. 01. 14. 오후 4:31	EventLog	vro-gateway	77ef8b8b-3180-44d0-a69a-503f27edc42c	Workflow run [77ef8b8b-3180-44d0-a69a-503f27edc42c] status changed to [FAILED], workflowid:[f246b7b5-fe89-4da5-a640-36ffc6874069] eventId:[33c01131-c9f9-3255-a686-a67671f44bcb]
16e0c6ed-40e4-4cb8-9be6-f7543b9c6a7c	20. 01. 14. 오후 3:55	Deployment completed	pmartini@vmware.com		
b754b4c6-f255-4869-b789-252924b32c4d	20. 01. 14. 오후 3:55	Deployment requested	pmartini@vmware.com		

확장성 구독 생성

vRealize Automation Cloud Assembly와 함께 확장성 작업을 사용하거나 vRealize Orchestrator 통합을 사용하여 애플리케이션을 확장하는 구독을 생성할 수 있습니다.

확장성 구독을 사용하면 특정 수명주기 이벤트에서 워크플로 또는 작업을 트리거하여 애플리케이션을 확장할 수 있습니다. 또한 구독에 필터를 적용하여 지정된 이벤트에 대해 부울 조건을 설정할 수도 있습니다. 예를 들어 부울 식이 'true'인 경우에만 이벤트 및 워크플로/작업이 트리거됩니다. 이것은 이벤트 및 작업이 트리거되는 시점을 제어하려는 경우 유용합니다.

팁 [항목] 텍스트 상자의 필터 이벤트에서 "Alt + Space" 키(Windows의 경우) 또는 "Option + Space" 키(Mac의 경우)를 사용하여 필터 옵션을 표시합니다.

사전 요구 사항

- 클라우드 관리자 사용자 역할이 있어야 합니다.
- vRealize Orchestrator 워크플로를 사용하는 경우:
 - 내장된 vRealize Orchestrator 클라이언트의 라이브러리 또는 통합된 외부 vRealize Orchestrator 인스턴스의 라이브러리.

■ 확장성 작업을 사용하는 경우:

- 기존 확장성 작업 스크립트가 있어야 합니다. 자세한 내용은 [확장성 작업을 생성하는 방법](#) 항목을 참조하십시오.

절차

- 1 **확장성 > 구독**을 선택합니다.
- 2 **새 통합**을 클릭합니다.
- 3 구독의 세부 정보를 입력합니다.
- 4 **이벤트 항목**을 선택합니다.
- 5 (선택 사항) 이벤트 항목에 대한 조건을 설정합니다.
- 6 (선택 사항) 해당하는 경우 이벤트 항목에 대한 차단 동작을 구성합니다.
- 7 **Runnable 항목**을 클릭하고 드롭다운 메뉴에서 **vRO 워크플로** 또는 **ABX 작업**을 선택합니다.
- 8 구독에서 실행할 워크플로 또는 확장성 작업을 선택합니다.
- 9 (선택 사항) 확장성 구독에 대한 프로젝트 범위를 정의하려면 **모든 프로젝트**를 사용하지 않도록 설정하고 **프로젝트 추가**를 클릭합니다.
- 10 **생성**을 클릭하여 구독을 저장합니다.

결과

구독이 생성되었습니다. 선택된 이벤트 항목별로 분류된 이벤트가 발생하면 연결된 vRealize Orchestrator 워크플로 또는 확장성 작업이 시작되고 모든 구독자에게 알림이 제공됩니다.

다음에 수행할 작업

구독을 생성한 후에는 Blueprint를 생성 또는 배포하여 구독을 연결하고 사용할 수 있습니다. 또한 vRealize Automation Cloud Assembly 내의 **확장성** 탭에서 워크플로 실행 상태를 확인할 수 있습니다. vRealize Orchestrator 워크플로를 포함하는 구독의 경우 vRealize Orchestrator 클라이언트에서 실행 및 워크플로 상태를 모니터링할 수도 있습니다.

확장성 구독 문제 해결

확장성 구독 실패 문제를 해결합니다.

구독이 실패하는 경우에는 일반적으로 워크플로 또는 확장성 작업 스크립트의 오류가 원인일 수 있습니다.

항목 매개 변수 및 페이로드 보기

덤프 구독 항목 매개 변수 스크립트를 사용하여 지정된 이벤트 단계에서 가상 시스템의 특정 매개 변수 및 페이로드를 볼 수 있습니다.

기본적으로 이 스크립트는 vRealize Orchestrator 워크플로에 사용 가능한 입력을 디버깅하고 확인하는데 유용합니다. 가상 시스템의 모든 매개 변수를 보려면 워크플로에 다음 스크립트를 사용합니다.

```
function dumpProperties(props, lvl) {
    var keys = props.keys;
    var prefix = ""
    for (var i=0; i<lvl; i++){
        prefix = prefix + "";
    }
    for (k in keys){
        var key = keys[k];
        var value = props.get(keys[k])
        if ("Properties" == System.getObjectType(value)){
            System.log(Prefix + key + "[")
            dumpProperties(value, (lvl+2));
            System.log(prefix+ "]")
        } else{
            System.log( prefix + key + ":" + value)
        }
    }
}

dumpProperties(inputProperties, 0)

customProps = inputProperties.get("customProperties")
```

구독 버전 기록

구독이 실패하는 경우 버전 기록을 볼 수 있습니다.

구독 버전 기록 보기

[버전 기록] 탭은 사용자 및 변경 날짜와 함께 구독의 변경 기록을 보여줄 수 있습니다. 구독이 실패하거나 잘못 실행 중인 경우 버전 기록을 확인하면 원인을 식별하는 데 도움이 될 수 있습니다.

The screenshot shows the vRealize Automation console interface. On the left, a sidebar contains navigation items: '배포' (Blueprints), 'Blueprint', '인프라' (Infrastructure), '확장성' (Scalability), and '마켓플레이스' (Marketplace). Below these is a '구독' (Subscriptions) section with a list of items including '라이브러리' (Library), '이벤트 항목' (Event Item), '작업' (Job), '워크플로' (Workflow), '작업 실행' (Job Execution), and '워크플로 실행' (Workflow Execution). The main content area is titled 'Test subscription - 버전 기록' (Test subscription - Version History). It displays a table of subscription events. The first event is selected, showing its details in a JSON format. The JSON includes fields like 'id', 'type', 'eventTopicId', 'name', 'orgId', 'ownerId', 'subscriberId', 'blocking', 'description', 'criteria', 'constraints', 'projectId', 'timeout', and 'broadcast'. Numbered callouts are present: '1' points to the '구독' menu item in the sidebar, '2' points to the '버전 기록' tab in the main area, and '3' points to the selected event row in the table.

구독 탭에서 구독을 엽니다.

2

버전 기록을 보려면 **버전 기록**을 클릭합니다.

3

각 변경 항목을 클릭하면 변경과 연결된 해당 구독 코드를 볼 수 있습니다.

vRealize Automation Cloud Assembly 배포 관리

7

vRealize Automation Cloud Assembly Blueprint 개발자는 [배포] 탭을 사용하여 배포를 관리합니다. 실패한 프로비저닝 프로세스의 문제를 해결하고, 내용을 변경하고, 사용되지 않는 배포를 제거할 수 있습니다.

배포는 프로비저닝된 Blueprint의 인스턴스입니다. [배포] 탭에는 성공한 배포와 실패한 배포가 나열됩니다. 이 페이지를 사용하여 성공한 배포를 관리하거나 실패한 요청에 대한 문제 해결을 시작할 수 있습니다.

배포 카드 사용

카드 목록을 사용하여 배포를 찾고 관리할 수 있습니다. 특정 배포를 필터링하거나 검색한 후 해당 배포에 대해 작업을 실행할 수 있습니다.

- 1 특성을 기반으로 요청을 필터링합니다.
- 2 키워드 또는 요청자를 기반으로 배포를 검색합니다.
- 3 시간 또는 이름별로 목록을 정렬합니다.
- 4 사용하지 않는 배포를 삭제하여 리소스를 회수하는 작업을 포함하여, 배포에 대해 배포 수준의 작업을 실행합니다.

배포 비용, 만료 날짜 및 상태도 볼 수 있습니다.



본 장은 다음 항목을 포함합니다.

- vRealize Automation Cloud Assembly에서 활성 배포를 모니터링하는 방법
- vRealize Automation Cloud Assembly 배포 실패 시 수행할 수 있는 작업
- 완료된 vRealize Automation Cloud Assembly 배포의 수명주기를 관리하는 방법
- vRealize Automation Cloud Assembly 배포에서 실행할 수 있는 작업

vRealize Automation Cloud Assembly에서 활성 배포를 모니터링하는 방법

vRealize Automation Cloud Assembly Blueprint를 배포한 후, 요청을 모니터링하여 리소스가 프로비저닝되고 실행 중인지 확인할 수 있습니다. 배포 카드부터 리소스 프로비저닝을 확인할 수 있습니다. 다음으로, 배포 세부 정보를 검토할 수 있습니다.

절차

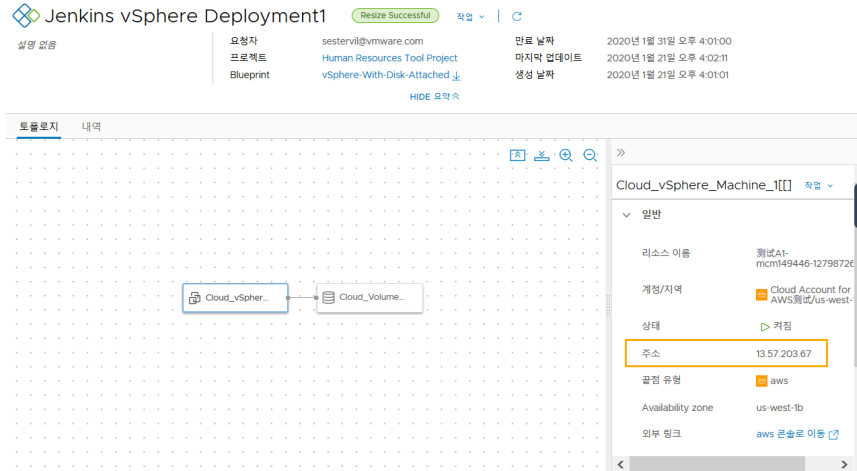
- 1 배포를 클릭하고 필요한 경우 필터 및 검색을 사용하여 처리 중인 배포 카드를 찾습니다.
- 2 카드 상태를 검토합니다.

배포가 진행 중인 경우 프로세스 표시줄에 남은 작업 수가 표시됩니다. 배포가 성공적으로 완료되면 카드에 배포에 대한 기본 세부 정보가 표시됩니다.



- 3 리소스가 배포된 위치를 확인하려면 배포 이름을 클릭하고 [토폴로지] 페이지에서 세부 정보를 검토합니다.

기본 구성 요소의 IP 주소가 필요할 가능성이 높습니다. 각 구성 요소를 클릭하면 해당 구성 요소와 관련된 정보가 제공됩니다. 이 예에서는 IP 주소가 강조 표시됩니다.



외부 링크를 사용할 수 있는지 여부는 클라우드 제공자에 따라 다릅니다. 외부 링크를 사용할 수 있는 경우에는 해당 제공자의 자격 증명이 있어야 구성 요소에 액세스할 수 있습니다.

다음에 수행할 작업

- 배포를 변경할 수 있습니다. 완료된 vRealize Automation Cloud Assembly 배포의 수명주기를 관리하는 방법의 내용을 참조하십시오.
- 배포에 실패하는 경우 vRealize Automation Cloud Assembly 배포 실패 시 수행할 수 있는 작업 항목을 참조하십시오.

vRealize Automation Cloud Assembly 배포 실패 시 수행할 수 있는 작업

배포 요청은 여러 가지 이유로 실패할 수 있습니다. 네트워크 트래픽 문제, 대상 클라우드 제공자의 리소스 부족 또는 배포 규격 결함 때문일 수 있습니다. 또는 배포가 성공했지만 작동하지 않는 것처럼 보일 수도 있습니다. vRealize Automation Cloud Assembly를 사용하면 배포를 검사하고, 오류 메시지를 검토하여 문제가 환경 때문인지, 요청된 워크로드 규격 때문인지 아니면 다른 이유 때문인지 파악할 수 있습니다.

워크플로를 사용하여 조사를 시작합니다. 진행 결과 일시적인 환경 문제가 실패의 원인일 수 있습니다. 이러한 유형의 문제는 상태가 개선된 것을 확인한 이후에 요청을 다시 배포하여 해결할 수 있습니다. 그 이외의 경우에는 조사할 때 다른 부분을 더 세부적으로 검토해야 할 수 있습니다.

프로젝트 멤버는 vRealize Automation Cloud Assembly에서 요청 세부 정보를 검토할 수 있습니다.

절차

- 요청이 실패했는지 확인하려면 **배포** 탭을 클릭하고 배포 카드를 찾습니다.

실패한 배포는 카드에 표시됩니다.

- 오류 메시지를 검토합니다.
- 자세한 내용을 보려면 배포 이름을 클릭하여 배포 세부 정보를 표시합니다.

- 배포 세부 정보 페이지에서 **기록** 탭을 클릭합니다.

- 이벤트 트리를 검토하여 프로비저닝 프로세스가 실패한 위치를 확인합니다. 이 트리는 배포를 수정했지만 변경이 실패한 경우에 유용합니다.
이 트리는 배포 작업을 실행할 때도 표시됩니다. 트리를 사용하여 실패한 변경 사항 문제를 해결할 수 있습니다.
- 세부 정보**에는 오류 메시지의 자세한 버전이 제공됩니다.
- 요청된 항목이 vRealize Automation Cloud Assembly Blueprint인 경우 메시지 오른쪽의 링크를 클릭하면 **요청 세부 정보**를 볼 수 있는 vRealize Automation Cloud Assembly가 열립니다.

- 3 **요청 세부 정보**에는 문제를 조사할 수 있도록 실패한 구성 요소에 대한 프로비저닝 워크플로가 제공됩니다.



- a 오류 메시지를 검토합니다.
 - b **개발 모드**를 켜면 간단한 프로비저닝 워크플로와 더 자세한 순서도 간에 전환할 수 있습니다.
 - c 카드를 클릭하여 배포 스크립트를 검토합니다.
- 4 오류를 해결하고 Blueprint를 다시 배포합니다.

Blueprint 구조에 오류가 있거나 인프라 구성 방식과 관련이 있을 수 있습니다.

다음에 수행할 작업

오류가 해결되고 Blueprint가 배포되면 [요청 세부 정보]에서 다음 예와 유사한 정보를 볼 수 있습니다. 요청 세부 정보를 보려면 **인프라 > 활동 > 요청**을 선택하십시오.



완료된 vRealize Automation Cloud Assembly 배포의 수명주기를 관리하는 방법

배포를 프로비저닝하고 실행한 후 배포를 관리하기 위해 실행할 수 있는 몇 가지 작업이 있습니다. 수명주기 관리에는 전원 켜기 또는 끄기, 배포 크기 조정 및 삭제 등이 포함될 수 있습니다. 개별 구성 요소에 대해 다양한 작업을 실행하여 관리할 수도 있습니다.

절차

- 1 배포를 클릭하고 배포를 찾습니다.
- 2 배포 세부 정보에 액세스하려면 배포 이름을 클릭합니다.

[토폴로지] 탭을 사용하면 배포 구조와 리소스를 시각화할 수 있습니다.

[기록] 탭에는 모든 프로비저닝 이벤트 및 요청된 항목이 배포된 이후에 실행하는 작업과 관련된 모든 이벤트가 포함됩니다. 프로비저닝 프로세스에 문제가 있는 경우, [기록] 탭에 있는 이벤트가 장애를 해결하는 데 도움이 됩니다.

[비용] 탭은 배포된 이후 일부 구성 요소의 현재 비용 정보를 제공합니다.

The screenshot displays the vRealize Automation Cloud Assembly interface. At the top, the blueprint 'EC2 with EBS Attached' is shown with a 'Create Successful' status. Below this, the 'Topology' (토폴로지) and 'History' (내역) tabs are visible. The 'History' tab is active, showing a table of operations. A 'CREATE' button is highlighted, and a modal window displays the 'Create' operation log.

EC2 with EBS Attached Create Successful 작업 |

설명 없음

요청자: pmartini@vmware.com
프로젝트: test-AD-project
Blueprint: vSphere-With-Disk-Attached

만료 날짜: 2020년 1월 31일 오전 9:30:00
마지막 업데이트: 2020년 1월 21일 오전 9:48:54
생성 날짜: 2020년 1월 21일 오전 9:30:54

HIDE 요약

토폴로지 내역

Cloud_vSphere_Machine_1[] 작업

일반

리소스 이름: 测试A1-mcm149216-127963
계정/지역: 添加VC-LC/Datacenter:datacen
상태: > 커짐

모든 요청 (1)

닫기

2020년 1월 21일 오전 9:30:54

CREATE pmartini@vmwa...

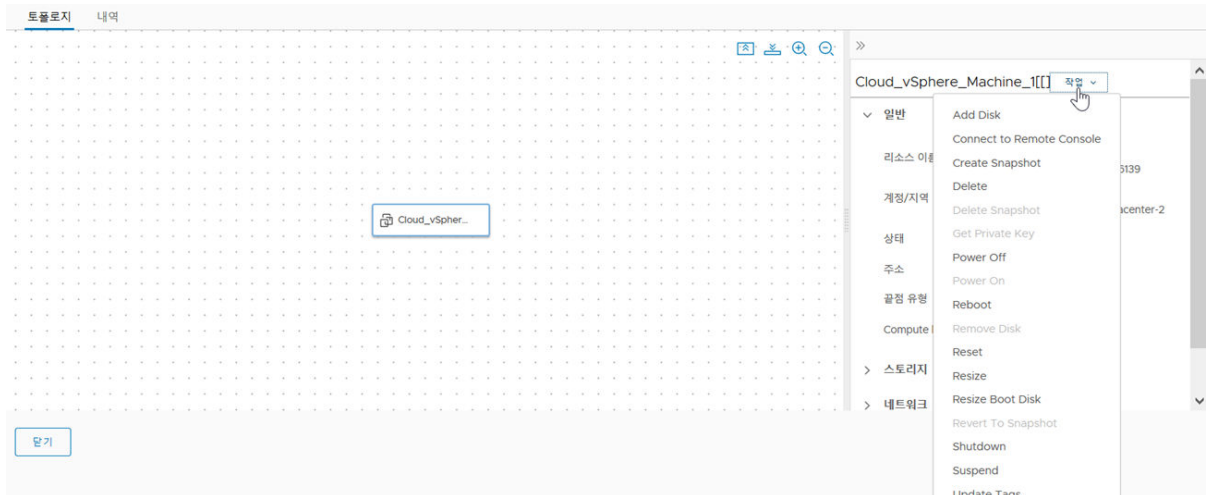
Create Successful 요청한 사람: pmartini@vmware.com 프로비저닝 다이어그램

타임 스탬프	상태	리소스 유형	리소스 이름	세부 정보
2020년 1월 21일 오전 9:41:32	REQUEST_FINISHED			
2020년 1월 21일 오전 9:41:31	COMPLETION_FINISHED			
2020년 1월 21일 오전 9:41:14	COMPLETION_IN_PROGRESS			
2020년 1월 21일 오전 9:40:51	CREATE_FINISHED	Cloud.Machine	Cloud_vSphere_Machine_1[]	
2020년 1월 21일 오전 9:33:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[]	Request is in stage STARTED and substage RE
2020년 1월 21일 오전 9:31:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[]	
2020년 1월 21일 오전 9:30:57	ALLOCATE_FINISHED	Cloud.Machine	Cloud_vSphere_Machine_1[]	
2020년 1월 21일 오전 9:30:55	ALLOCATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[]	
2020년 1월 21일 오전 9:30:55	INITIALIZATION_FINISHED			

닫기

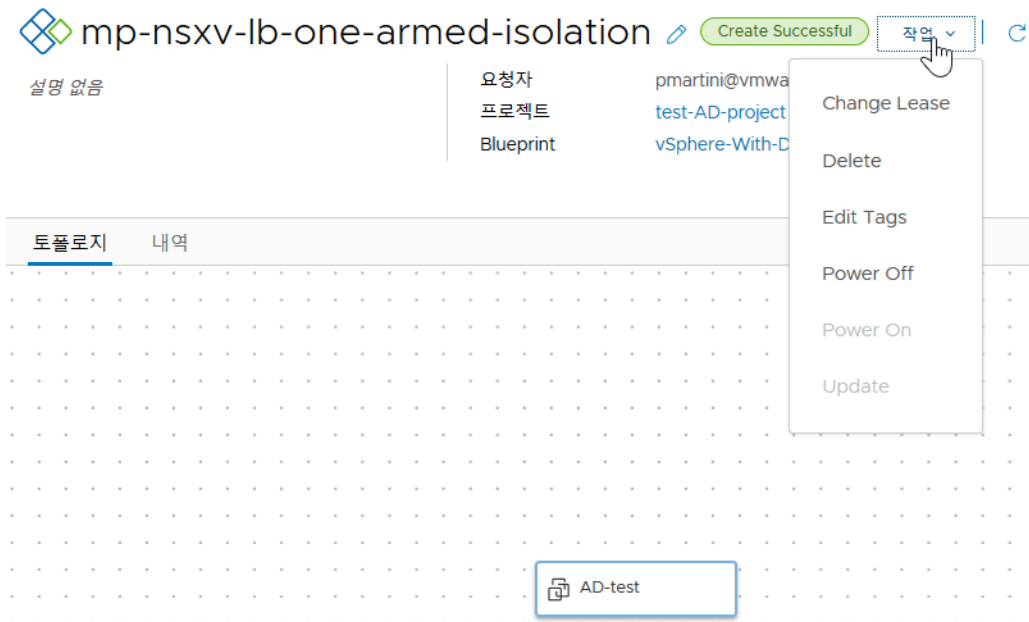
- 3 현재 구성에서 배포 비용이 너무 많이 들어서 구성 요소의 크기를 조정하려는 경우, [토폴로지] 페이지에서 구성 요소를 선택한 다음 구성 요소 페이지에서 **작업 > 크기 조정**을 선택합니다.

사용 가능한 작업은 구성 요소, 클라우드 계정 및 사용 권한에 따라 다릅니다.



- 4 개발 수명주기의 일부로 배포 중 하나가 더 이상 필요하지 않습니다. 배포를 제거하고 리소스를 회수하려면 **작업 > 삭제**를 선택합니다.

사용 가능한 작업은 배포 상태에 따라 다릅니다.



다음에 수행할 작업

가능한 작업에 대한 자세한 내용은 [vRealize Automation Cloud Assembly 배포에서 실행할 수 있는 작업 항목](#)을 참조하십시오.

vRealize Automation Cloud Assembly 배포에서 실행할 수 있는 작업

Blueprint를 배포한 후 vRealize Automation Cloud Assembly에서 작업을 실행하여 리소스를 관리할 수 있습니다. 사용 가능한 작업은 리소스 유형 그리고 해당 작업이 특정 클라우드 계정 또는 통합 플랫폼에서 지원되는지 여부에 따라 달라집니다.

가능한 작업도 관리자가 권한을 부여하는 대상에 따라 달라집니다.

관리자나 프로젝트 관리자는 vRealize Automation Service Broker에서 2일차 작업 정책을 설정할 수 있습니다. [소비자에게 Service Broker 2일차 작업 정책에 대한 권한을 부여하는 방법](#)을 참조하십시오.

표 7-1. 가능한 작업 목록

작업	적용 대상 리소스 유형	대상 클라우드 계정 또는 통합	설명
디스크 추가	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	기존 가상 시스템에 디스크를 추가합니다.
리스 변경	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	<p>리스 만료 날짜 및 시간을 변경합니다.</p> <p>리스가 만료되면 배포가 삭제되고 리소스가 회수됩니다.</p> <p>리스 정책은 vRealize Automation Service Broker에 설정됩니다.</p>
원격 콘솔에 연결	시스템	<ul style="list-style-type: none"> ■ VMware vSphere 	<p>선택한 시스템에서 원격 세션을 엽니다.</p> <p>성공적인 연결을 위해 다음 요구 사항을 검토합니다.</p> <ul style="list-style-type: none"> ■ 배포 소비자는, 프로비저닝된 시스템의 전원이 켜져 있는지 확인합니다.
스냅샷 생성	시스템	<ul style="list-style-type: none"> ■ Google Cloud Platform ■ VMware vSphere 	<p>가상 시스템의 스냅샷을 생성합니다.</p> <p>vSphere에서 2개의 스냅샷만 허용되고 이미 스냅샷이 2개 있는 경우, 스냅샷을 삭제할 때까지 이 명령을 사용할 수 없습니다.</p>
삭제	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	<p>배포를 제거합니다.</p> <p>모든 리소스가 삭제되고 회수됩니다.</p> <p>삭제가 실패하면 배포에 대한 삭제 작업을 두 번째로 실행할 수 있습니다. 두 번째 시도 중에 삭제 실패 무시를 선택할 수 있습니다. 이 옵션을 선택하면 배포가 삭제되지만 리소스가 회수되지 않을 수 있습니다. 모든 리소스가 제거되었는지 확인하려면 배포가 프로비저닝된 시스템을 확인해야 합니다. 그렇지 않은 경우 해당 시스템의 나머지 리소스를 수동으로 삭제해야 합니다.</p>
	시스템 및 로드 밸런서	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	배포에서 시스템 또는 로드 밸런서를 삭제합니다. 이 작업을 수행하면 배포가 불가능해질 수 있습니다.
스냅샷 삭제	시스템	<ul style="list-style-type: none"> ■ VMware vSphere ■ Google Cloud Platform 	가상 시스템의 스냅샷을 삭제합니다.

표 7-1. 가능한 작업 목록 (계속)

작업	적용 대상 리소스 유형	대상 클라우드 계정 또는 통합	설명
태그 편집	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	개별 배포 리소스에 적용되는 리소스 태그를 추가하거나 수정합니다.
전원 끄기	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	게스트 운영 체제를 종료하지 않고 배포 전원을 끕니다.
	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	게스트 운영 체제를 종료하지 않고 시스템 전원을 끕니다.
전원 켜기	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	배포 전원을 켭니다. 리소스가 일시 중단된 경우, 리소스가 일시 중단된 지점에서 정상 작업이 재개됩니다.
	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	시스템 전원을 켭니다. 시스템이 일시 중단된 경우, 시스템이 일시 중단된 지점에서 정상 작업이 재개됩니다.
재부팅	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ VMware vSphere 	가상 시스템에서 게스트 운영 체제를 재부팅합니다. vSphere 시스템의 경우, 이 작업을 사용하려면 해당 시스템에 VMware Tools가 설치되어 있어야 합니다.
재구성	로드 밸런서	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ Google Cloud Platform ■ VMware vSphere 	로드 밸런서 프로토콜, 포트, 상태 구성 및 구성원 풀 설정을 변경합니다.
디스크 제거	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	기존 가상 시스템에서 디스크를 제거합니다.
재설정	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ VMware vSphere 	게스트 운영 체제를 종료하지 않고 가상 시스템을 강제로 다시 시작합니다.
크기 조정	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ Google Cloud Platform ■ VMware vSphere 	가상 시스템의 CPU 및 메모리를 늘리거나 줄입니다.
부팅 디스크 크기 조정	시스템	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	부팅 디스크 매체의 크기를 늘리거나 줄입니다.

표 7-1. 가능한 작업 목록 (계속)

작업	적용 대상 리소스 유형	대상 클라우드 계정 또는 통합	설명
디스크 크기 조정	스토리지 디스크	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform 	스토리지 디스크의 용량을 늘립니다.
다시 시작	시스템	<ul style="list-style-type: none"> ■ Microsoft Azure 	실행 중인 시스템을 종료하고 다시 시작합니다.
스냅샷으로 되돌리기	시스템	<ul style="list-style-type: none"> ■ Google Cloud Platform ■ VMware vSphere 	시스템의 이전 스냅샷으로 되돌립니다. 이 작업을 사용하려면 기존 스냅샷이 있어야 합니다.
Puppet 작업 실행	관리되는 리소스	<ul style="list-style-type: none"> ■ Puppet Enterprise 	배포 내의 시스템에서 선택한 작업을 실행합니다. 작업은 Puppet 인스턴스에 정의됩니다. 작업을 식별하고 입력 매개 변수를 제공할 수 있어야 합니다.
종료	시스템	<ul style="list-style-type: none"> ■ VMware vSphere 	게스트 운영 체제를 종료한 후 시스템 전원을 끕니다. 이 작업을 사용하려면 시스템에 VMware Tools가 설치되어 있어야 합니다.
일시 중단	시스템	<ul style="list-style-type: none"> ■ Microsoft Azure ■ VMware vSphere 	시스템이 사용될 수 없고 현재 사용 중인 스토리지 이외의 시스템 리소스를 사용하지 않도록 시스템을 일시 중지합니다.
업데이트	배포	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	입력 매개 변수에 따라 배포를 변경합니다.
태그 업데이트	시스템 및 디스크	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	개별 리소스에 적용된 태그를 추가, 수정 또는 삭제합니다.