

vRealize Automation Cloud Assembly の使用と管 理

2022 年 10 月

vRealize Automation 8.2

最新の技術ドキュメントは、VMware の Web サイト (<https://docs.vmware.com/jp/>)

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

ヴィエムウェア株式会社
〒108-0023 東京都港区芝浦 3-1-1
田町ステーションタワー N 18 階
www.vmware.com/jp

Copyright © 2022 VMware, Inc. All rights reserved. 著作権および商標情報。

目次

1 vRealize Automation Cloud Assembly の概要 7

vRealize Automation Cloud Assembly の機能 8

2 チュートリアル 11

vSphere インフラストラクチャおよび展開のセットアップとテスト 13

本番ワークロードの構成とプロビジョニング 30

マルチクラウド インフラストラクチャと展開 38

パート1: サンプル インフラストラクチャの構成 39

パート2: サンプル プロジェクトの作成 45

パート3: サンプル クラウド テンプレートの設計と展開 46

VMware Cloud on AWS の構成 62

基本的な VMware Cloud on AWS ワークフローの構成 63

VMware Cloud on AWS の隔離されたネットワークの構成 76

Infoblox 用外部 IP アドレス管理統合の構成 80

ダウンロード パッケージの展開前に、Infoblox アプリケーションの必要な拡張属性を追加 82

外部 IP アドレス管理プロバイダ パッケージのダウンロードと展開 83

IP アドレス管理統合ポイント用の実行環境の作成 84

Infoblox 用外部 IP アドレス管理統合の追加 86

既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定 89

外部 IP アドレス管理プロバイダ範囲の割り当てを使用するクラウド テンプレートの定義と展開 92

IP アドレス管理統合における Infoblox 固有のプロパティの使用 94

3 組織での vRealize Automation Cloud Assembly の設定 98

vRealize Automation のユーザー ロールについて 98

組織およびサービスのユーザー ロール 100

カスタム ユーザー ロール 113

使用事例: ユーザー ロールによってアクセスを制御する方法 116

クラウド アカウントの追加 136

クラウド アカウントを使用するために必要な認証情報 136

Microsoft Azure クラウド アカウントの作成 153

Amazon Web Services クラウド アカウントの作成 154

Google Cloud Platform クラウド アカウントの作成 155

vCenter クラウド アカウントの作成 157

NSX-V クラウド アカウントの作成 158

NSX-T クラウド アカウントの作成 160

VMware Cloud on AWS クラウド アカウントの作成 162

VMware Cloud Foundation クラウド アカウントの作成 164

他のアプリケーションとの統合 165

GitLab および GitHub 統合の使用法 165

外部 IP アドレス管理統合の構成方法 171

新しい外部 IP アドレス管理統合パッケージへのアップグレード方法 173

vRealize Automation Cloud Assembly での My VMware 統合の設定 174

Cloud Assembly で vRealize Orchestrator との統合を設定する 174

vRealize Automation Cloud Assembly で Kubernetes を使用する方法 177

vRealize Automation Cloud Assembly の構成管理について 193

vRealize Automation Cloud Assembly で Active Directory 統合を作成する方法 202

VMware SDDC Manager 統合の構成 204

vRealize Operations Manager との統合 205

オンボーディング プランについて 213

選択されたマシンを単一の展開としてオンボーディング 214

ルール フィルタを適用したマシンを個別の展開としてオンボーディング 216

詳細設定 221

インターネット プロキシ サーバの構成 221

NSX-T と複数の vCenter Server のマッピングで可能になる事例 225

NSX クラウド アカウントの関連付けを削除した場合の動作 226

IP アドレス管理 SDK を使用してプロバイダ固有の外部 IP アドレス管理統合パッケージを作成する方法 226

4 リソース インフラストラクチャのビルド 228

クラウド ゾーンを追加する方法 228

クラウド ゾーンの詳細情報 229

フレーバー マッピングを追加する方法 231

フレーバー マッピングの詳細 231

イメージ マッピングを追加する方法 232

イメージ マッピングの詳細 232

ネットワーク プロファイルを追加する方法 235

ネットワーク プロファイルの詳細 236

ネットワーク設定の使用 242

セキュリティ グループ設定の使用 246

ロード バランサの設定の使用 248

外部 IP アドレス管理統合用のオンデマンド ネットワークをサポートするようにネットワーク プロファイルを設定する方法 249

外部 IP アドレス管理統合用の既存ネットワークをサポートするようにネットワーク プロファイルを設定する方法 252

ストレージ プロファイルを追加する方法 252

ストレージ プロファイルの詳細 252

タグを使用する方法 253

タグ付けストラテジの作成 256

vRealize Automation Cloud Assembly での機能タグの使用 257

vRealize Automation Cloud Assembly での制約タグの使用	258
標準タグ	260
vRealize Automation Cloud Assembly のタグの処理方法	261
単純なタグ付け構造を設定する方法	261
リソースを操作する方法	263
コンピューティング リソース	263
ネットワーク リソース	263
セキュリティ リソース	265
ストレージ リソース	267
マシン リソース	267
ボリューム リソース	267
リソースの詳細	268
vRealize Automation を使用したマルチプロバイダ テナント リソースの構成	279
vRealize Automation 用の仮想プライベート ゾーンの作成方法	280
vRealize Automation テナントの VPZ 構成の管理	283

5 プロジェクトの追加と管理 285

開発チームのプロジェクトを追加する方法	285
プロジェクトの詳細	287
プロジェクト タグとカスタム プロパティの使用	287
展開時のプロジェクトの動作	289

6 展開の設計 291

クラウド テンプレートを作成する方法	292
シンプルなクラウド テンプレートをゼロから作成する方法	294
リソースを選択してクラウド テンプレートに追加する方法	295
クラウド テンプレート リソースを接続する方法	295
有効なクラウド テンプレート コードを作成する方法	296
異なるバージョンを保存する方法	298
シンプルなクラウド テンプレートを拡張する方法	300
ユーザー入力によるクラウド テンプレートのカスタマイズ方法	301
リソース フラグによる申請のカスタマイズ方法	306
リソースの展開順序を設定する方法	308
式を使用してクラウド テンプレート コードの汎用性を高める方法	309
クラウド テンプレートでリモート アクセスを有効にする方法	318
設計に高度な機能を追加する方法	321
展開されたリソースの名前をカスタマイズする方法	321
クラウド テンプレートでマシンを自動的に初期化する方法	324
クラウド テンプレートで使用するカスタム リソース タイプを作成する方法	337
Day 2 の変更に対する準備の方法	348
拡張性によりアプリケーションのライフサイクルを延長および自動化する方法	354

リソースのプロパティについて	391
コードの例	391
クラウド テンプレートの vSphere リソースの例	391
確認可能なクラウド テンプレート	395
クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例	402
ユーザー名とパスワードでアクセスできる Puppet 対応のクラウド テンプレート	423
Terraform 構成を含める方法	433
Terraform ランタイム環境の準備	433
Terraform 構成の準備	439
Terraform 構成のデザイン	440
Terraform 構成に関する詳細	443
マーケットプレイスの使用方法	446

7 展開の管理 447

展開を監視する方法	448
vRealize Automation Cloud Assembly の展開に失敗した場合の対処	449
完了した展開のライフサイクルを管理する方法	452
環境で実行できるアクション	454

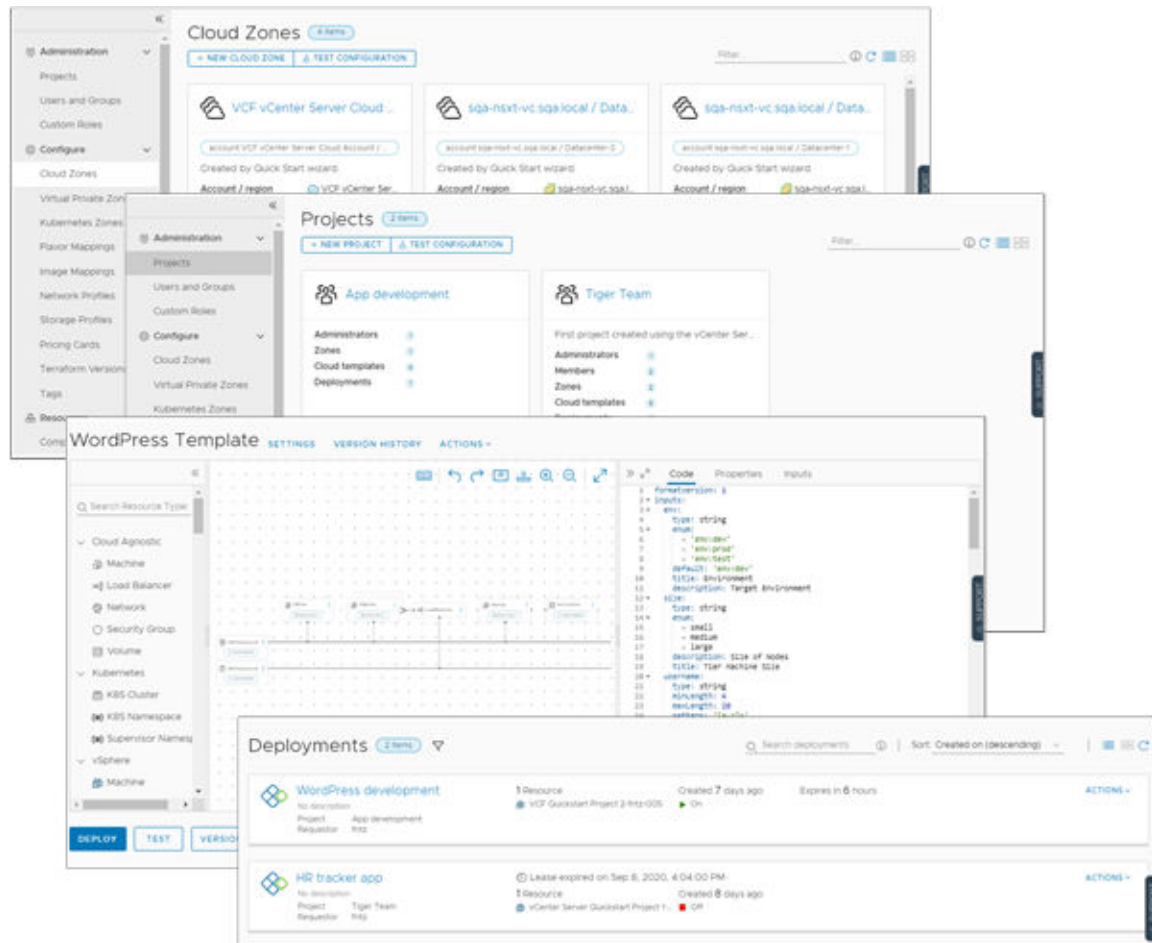
vRealize Automation Cloud Assembly の概要

1

vRealize Automation Cloud Assembly を使用してパブリックおよびプライベートのクラウド プロバイダに接続し、それらのリソースに作成したマシン、アプリケーション、サービスを展開することができます。開発からテスト、本番まで、反復的なワークフローをサポートする環境内で、コードとしてのクラウド テンプレートを開発します。プロビジョニング時には、さまざまなクラウド ベンダーに展開することができます。このサービスは、管理された VMware SaaS および NaaS ベースのフレームワークです。

vRealize Automation Cloud Assembly の概要には、以下の基本機能が含まれています。

- [インフラストラクチャ] タブでは、クラウド ベンダーのリソースとユーザーを追加および編成できます。このタブには、展開されたクラウド テンプレートに関する情報も示されます。
- [マーケットプレイス] タブには、クラウド テンプレート ライブラリのビルドおよびサポートする OVA または OVF へのアクセスに使用される、VMware Solution Exchange のクラウド テンプレートとイメージがあります。
- [デザイン] タブは、開発の拠点となります。キャンバスと YAML エディタを使用して、マシンおよびアプリケーションを開発し、展開します。
- [展開] タブには、プロビジョニングされたリソースの現在のステータスが表示されます。詳細と履歴にアクセスして、展開を管理するために使用できます。



この章には、次のトピックが含まれています。

- [vRealize Automation Cloud Assembly の機能](#)

vRealize Automation Cloud Assembly の機能

vRealize Automation Cloud Assembly は、クラウド テンプレートの開発および展開サービスです。ユーザーとチームは、このサービスを使用してクラウド ベンダーのリソースにマシン、アプリケーション、およびサービスを展開します。

Cloud Assembly 管理者（一般的にはクラウド管理者と呼ばれる）として、プロビジョニング インフラストラクチャを設定し、ユーザーとリソースをグループ化するプロジェクトを作成します。

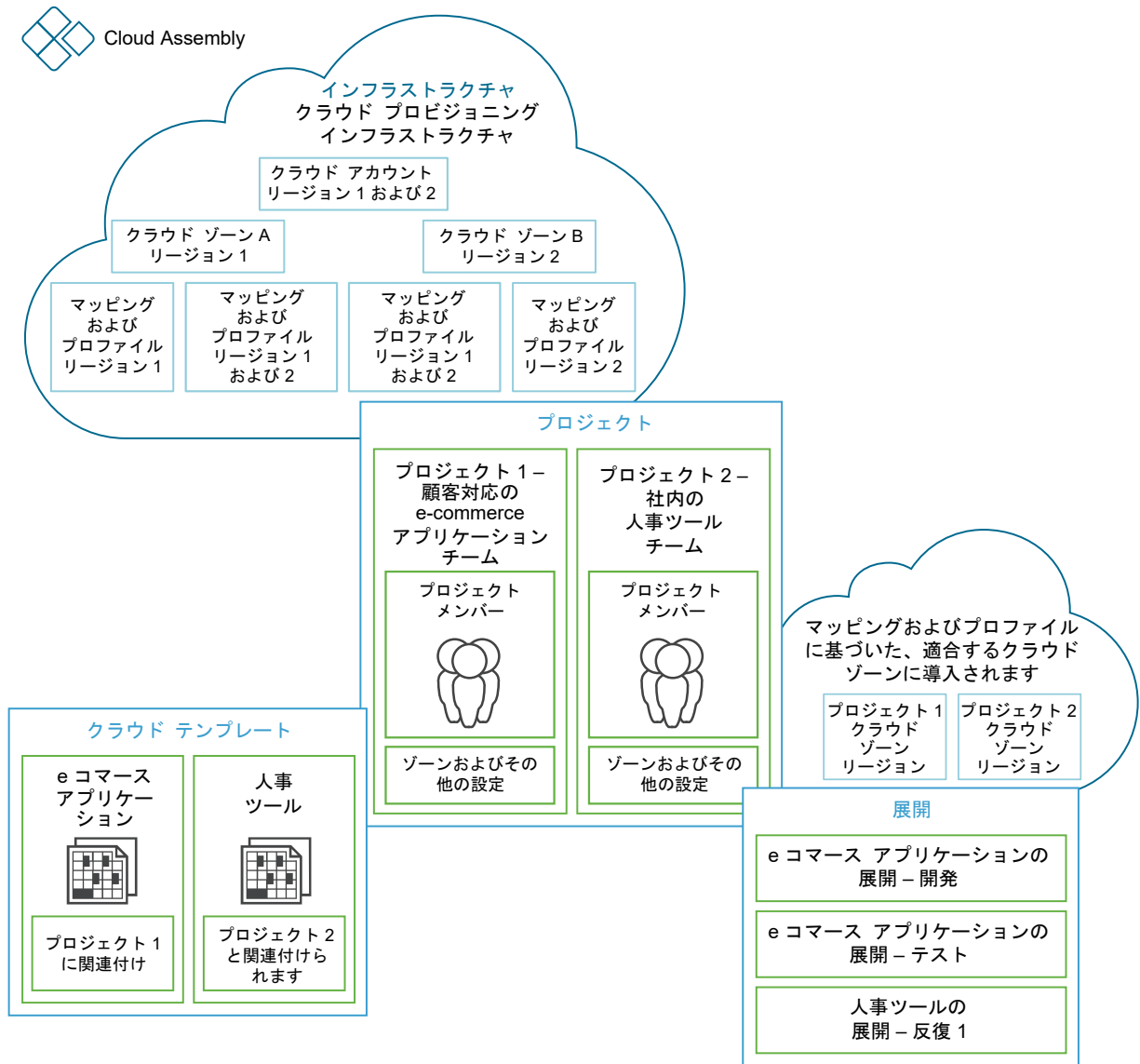
- クラウド ベンダーのアカウントを追加します。 [vRealize Automation Cloud Assembly へのクラウド アカウントの追加](#)を参照してください。
- どの領域またはデータストアが、開発者が展開するクラウド ゾーンであるかを判断します。 [vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報](#)を参照してください。
- クラウド ゾーンを定義するポリシーを作成します。 [4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。

- クラウド ゾーンで開発者をグループ化するプロジェクトを作成します。[vRealize Automation Cloud Assembly のプロジェクト タグとカスタム プロパティの使用](#) を参照してください。

クラウド テンプレート開発者は、1 つまたは複数のプロジェクトのメンバーです。テンプレートを作成して、プロジェクトの 1 つに関連付けられているクラウド ゾーンに展開します。

- キャンバスを使用して、プロジェクトのクラウド テンプレートを開発します。プロジェクト管理者は、マーケットプレイスを使用して、VMware Solution Exchange からテンプレートおよびサポートするイメージをダウンロードできます。[6 章 vRealize Automation Cloud Assembly 展開の設計](#)および [vRealize Automation Cloud Assembly マーケットプレイスの使用方法](#) を参照してください。
- ポリシーと制約に基づいて、クラウド テンプレートをプロジェクトのクラウド ゾーンに展開します。
- 使用されていないアプリケーションの削除など、展開を管理します。[7 章 vRealize Automation Cloud Assembly 展開の管理](#)を参照してください。

vRealize Automation Cloud Assembly へようこそ。インフラストラクチャを定義してクラウド テンプレートを作成し、展開する方法の例が必要な場合は、[チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。



Cloud Assembly チュートリアル

2

このチュートリアルでは、vRealize Automation Cloud Assembly に熟練するのに役立つ一般的なタスクの実行方法を示します。

はじめに、このガイドには、このチュートリアルの手順に加えて、それ以外の情報も記載されていることにご注意ください。関連するトピックへのリンクが用意されています。

ユーザー サポートへのアクセス

同様に重要なのは、アプリケーション全体を通じてユーザー サポートが提供されていることです。ユーザー サポートはユーザーが機能について理解できるようサポートし、テキスト ボックスの入力方法を決定するのに役立つ情報を提供します。外部のドキュメントには、より詳細なサンプル コードやユースケースが記載されています。

サポートの種類	サポートの利用方法	例
フィールドレベルの Signpost ヘルプ	フィールドの横にある [情報] アイコン (i) をクリックします。	
状況に応じたサポート パネルのヘルプ	名前と組織の横にあるヘルプ アイコン (?) をクリックします。	
外部ドキュメントへのアクセス	[Docs] ラベルが付いている記事のタイトルをクリックするか、[VMware Docs で詳細を表示] をクリックします。	

この章には、次のトピックが含まれています。

- チュートリアル：vRealize Automation Cloud Assembly での vSphere インフラストラクチャおよび展開のセットアップとテスト
- チュートリアル：本番ワークロードをプロビジョニングするための vRealize Automation Cloud Assembly の構成
- チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト
- チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成
- チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成

チュートリアル：vRealize Automation Cloud Assembly での vSphere インフラストラクチャおよび展開のセットアップとテスト

vRealize Automation を初めて使用する場合、またはリフレッシャー コースを受講したい場合は、このチュートリアルに沿って vRealize Automation Cloud Assembly 構成プロセスを行います。vSphere リソース タイプに基づく VMware Cloud Templates を使用して、クラウド vSphere アカウント エンドポイントの追加、インフラストラクチャの定義、プロジェクトへのユーザーの追加、およびワークロードの設計と展開を行いながら、全体的なプロセスについて学習します。

このチュートリアルは単なる入門用ですが、複数のパブリック クラウドとプライベート クラウドにわたってセルフサービスを自動化する方法や、開発を反復的に行う方法について確認できます。このチュートリアルでは、VMware vCenter Server と NSX-T について重点的に説明しています。このワークフローが完了すると、学習内容を適用して、より多くのタイプのクラウド アカウントを追加したり、より高度なクラウド テンプレートを提供したりできるようになります。

この手順の進行に伴い、データの例が示されます。これらの例は、環境内で有効な値に置き換えてください。

このチュートリアルのすべての手順は vRealize Automation Cloud Assembly で実行します。

このチュートリアルでは、必要な各コンポーネントを構成する方法について説明します。

- **手順 1：vCenter Server と NSX クラウド アカウントを追加する。** クラウド アカウントとは、クラウド ベンダーのエンドポイントに vRealize Automation Cloud Assembly を接続するための認証情報です。
- **手順 2：クラウド ゾーンのコンピューティング リソースを定義する。** クラウド ゾーンとは、プロジェクトのニーズやコンプライアンスとコストの管理目標に基づいてさまざまなプロジェクトに割り当てる、アカウント/リージョン内の特定のコンピューティング リソースです。
- **手順 3：アカウント/リージョンで使用可能なリソースを構成する。** インフラストラクチャ リソースとは、クラウド テンプレートで使用されるアカウント/リージョンに関連付けられたコンピューティング リソース、ストレージ リソース、ネットワーク リソース、およびその他のリソースの定義です。
- **手順 4：プロジェクトを作成する。** プロジェクトとは、プロジェクトのアプリケーション開発目標に基づいて、ユーザーにクラウド ゾーンへのアクセス権を付与する方法です。
- **手順 5：基本的なクラウド テンプレートを設計して展開する。** クラウド テンプレートとは、繰り返し開発および展開されるアプリケーション ワークロードの定義です。

この構成プロセスは Cloud Assembly 開発環境の基盤となります。インフラストラクチャを構築し、クラウド テンプレート開発スキルを習得する際は、このワークフローを繰り返し行って、発展させます。

開始する前に

- Cloud Assembly 管理者ロールが割り当てられていることを確認します。vRealize Automation の組織およびサービスのユーザー ロールを参照してください。
- vRealize Automation コンソールで VMware vCenter Server または VMware Cloud Foundation のクイックスタート ウィザードを使用していない場合は、ここで使用できます。

これらのウィザードベースのワークフローに、このチュートリアルに記載されている構成がほとんど含まれています。

このチュートリアルは、作業用インフラストラクチャを構築してワークロードを展開する方法について理解を深めるための実践的なガイドです。

『スタート ガイド』の [Cloud Assembly をセットアップする方法](#)を参照してください。

- vRealize Automation Cloud Assembly に用意されているガイド付きセットアップをまだ使用していない場合は、ここで使用できます。ガイド付きセットアップでは、このチュートリアルで実行する手順の大部分を実行できます。ガイド付きセットアップを開くには、タブ バーの右側にある [ガイド付きセットアップ] をクリックします。
- vCenter Server と NSX の認証情報があることを確認します。認証情報に必要な権限の詳細については、[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。他のユーザーをプロジェクトに追加する予定がある場合は、これらのユーザーが vRealize Automation Cloud Assembly サービスのメンバーであることを確認します。

手順 1 : vCenter Server と NSX クラウド アカウントを追加する

クラウド アカウントは、vCenter Server および関連付けられた NSX サーバに接続するために vRealize Automation で使用される認証情報を提供します。

- 1 vCenter Server クラウド アカウントを追加します。

vCenter Server クラウド アカウントは、リソースの検出やクラウド テンプレートの展開を行うために vRealize Automation Cloud Assembly で使用される vCenter Server の認証情報を提供します。

vCenter Server クラウド アカウントの詳細については、[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。

- a [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択します。
- b [クラウド アカウントの追加] をクリックして、[vCenter Server] を選択します。
- c 値を入力します。

New Cloud Account

Name * vCenter Server Account

Description

vCenter Server Credentials

vCenter IP address / FQDN * sc2vc05.cmbu.local ⓘ

Username * mgmt@cmbu.local

Password *

VALIDATE ✓ Credentials validated successfully. ✕

Configuration

Allow provisioning to these datacenters * ☒ wld01-DC

☒ Create a cloud zone for the selected datacenters

NSX cloud account 🔍 Search for cloud accounts

Capabilities

Capability tags Enter capability tags ⓘ

ADD **CANCEL**

これらの値は単なる例であることに注意してください。値は環境によって異なります。

設定	サンプルの値
名前	vCenter Server のアカウント
vCenter Server の IP アドレス/FQDN	your-dev-vcenter.company.com
ユーザー名とパスワード	vCenterCredentials@yourCompany.com

- d 認証情報を確認するには、[検証] をクリックします。
 - e [これらのデータセンターへのプロビジョニングを許可] を有効にするには、1 つ以上のデータセンターを選択します。
 - f NSX クラウド アカウントをスキップします。後で構成して、vCenter Server アカウントを NSX クラウド アカウントにリンクします。
 - g [追加] をクリックします。
- 2 関連付けられた NSX クラウド アカウントを追加します。

NSX-T クラウド アカウントは、ネットワーク リソースの検出や、クラウド テンプレートを使用したネットワークの展開を行うために vRealize Automation Cloud Assembly で使用される NSX-T 認証情報を提供します。

NSX-T クラウド アカウントの詳細については、[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。

- [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択します。
- [クラウド アカウントの追加] をクリックして、NSX-T または NSX-V を選択します。このチュートリアルでは [NSX-T] を使用します。
- 値を入力します。

New Cloud Account

Name * NSX-T Account

Description

NSX-T Credentials

NSX-T IP address / FQDN * sc2vc05-vip-nsx-mgmt.cmbu.local ⓘ

Username * mgmt@cmbu.local

Password *

NSX mode Policy ⓘ

VALIDATE ✔ Credentials validated successfully. ✕

Associations

vCenter cloud accounts + ADD ✕ REMOVE

<input type="checkbox"/>	Name	Status	Identifier	Type
<input type="checkbox"/>	vCenter Server Account	✔ OK	sc2vc05.cmbu.local	vCenter

1 - 1 of 1 cloud accounts

Capabilities

Capability tags Enter capability tags ⓘ

ADD CANCEL

これらの値は単なる例です。値は環境によって異なります。

設定	サンプルの値
名前	NSX-T アカウント
vCenter Server の IP アドレス/FQDN	your-dev-NSX-vcenter.company.com
ユーザー名とパスワード	NSXCredentials@yourCompany.com
NSX モード	何を選択すればよいかわからない場合は、製品内ヘルプを使用することをお勧めします。フィールドの右側にある情報アイコンをクリックします。フィールドレベルのヘルプには、オプションの構成に役立つ情報が含まれています。この例では、[ポリシー] を選択します。

- 認証情報を確認するには、[検証] をクリックします。
- 前の手順で作成した vCenter Server クラウド アカウントを関連付けるには、[追加] をクリックして、[vCenter Server アカウント] を選択します。

この vCenter Server クラウド アカウントの関連付けにより、ネットワークのセキュリティが確保されます。

- f NSX クラウド アカウント画面で、[追加] をクリックします。

手順 2：クラウド ゾーンのコンピューティング リソースを定義する

クラウド ゾーンとは、プロジェクトで使用できるようになるアカウント/リージョン内のコンピューティング リソースのグループです。プロジェクト メンバーは割り当てられたクラウド ゾーンのリソースを使用して、クラウド テンプレートを展開します。プロジェクト クラウド テンプレートの展開場所をきめ細かく制御できるようにするには、それぞれ異なるコンピューティング リソースを持つ複数のクラウド ゾーンを作成します。

アカウント/リージョンとは、クラウド ペンダーが分離されたリージョンまたはデータストアにリソースを関連付ける方法のことです。アカウントはクラウド アカウントのタイプを示し、リージョンはリージョンまたはデータストアを示します。vCenter Server はデータストアを使用し、選択したクラスタとリソース プールがプロビジョニング リソースになります。

このチュートリアルでは、プロジェクト開発チームの目標をサポートするリソースおよび予算と管理の要件を、クラウド ゾーンに確実に含める必要があります。

クラウド ゾーンの詳細については、[vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報](#)を参照してください。

- 1 [インフラストラクチャ] - [構成] - [クラウド ゾーン] を選択します。
- 2 vCenter Server インスタンスに追加されたクラウド ゾーンをクリックして、値を入力します。

vCenter Account Cloud Zone
DELETE

Summary
Compute
Projects

A cloud zone defines a set of compute resources that can be used for provisioning.

Account / region *
vCenter Account / wld01-DC

Name *
vCenter Account Cloud Zone

Description

Placement policy *
DEFAULT

Folder
Select folder

Capabilities

Capability tags are effectively applied to all compute resources in this cloud zone, but only in the context of this cloud zone.

Capability tags
Enter capability tags

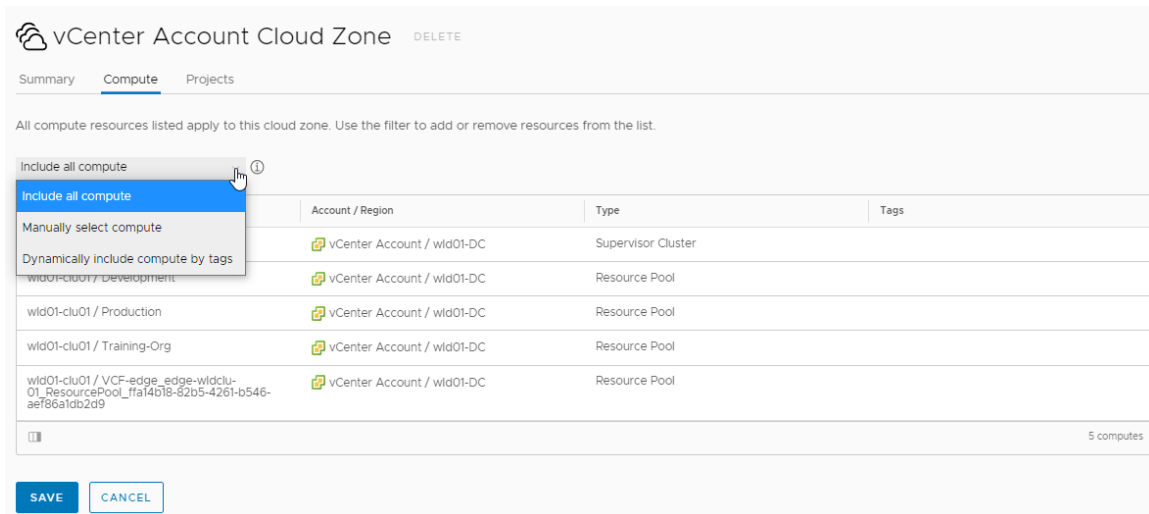
SAVE
CANCEL

設定	サンプルの値
アカウント/リージョン	vCenter Server アカウント/データセンターの名前
名前	vCenter Server クラウド ゾーン 作成後にこの値を変更することはできません。vCenter Server ごとに異なるデータセンターを構成する場合は、新しいクラウド ゾーンを作成して、アカウント/リージョンを選択できるようにする必要があります。
説明	開発用のすべての vCenter Server コンピューティング リソース。
ポリシー	デフォルト フィールド値に関して不明な点がある場合は、ヘルプを参照してください。

すべての値は単なる例であることに注意してください。ゾーンの詳細は環境によって異なります。

- 3 [コンピューティング] タブをクリックして、コンピュート リソースがすべて存在することを確認します。

特定のコンピューティング リソースを除外する必要がある場合は、[コンピューティングの手動選択] に切り替えて、クラウド ゾーンに含めるコンピューティング リソースのみを追加します。



4 [保存] をクリックします。

5 追加のクラウド ゾーンにこのプロセスを繰り返しますが、ゾーン名は一意にする必要があります。

手順 3 : アカウント/リージョンで使用可能なリソースを構成する

アカウント/リージョンをクラウド ゾーンに追加しました。ここで、クラウド アカウントの使用可能なマシン サイズ (フレーバー マッピング)、イメージ マッピング、ネットワーク プロファイル、およびストレージ プロファイルを定義します。クラウド テンプレートを展開するときにはマッピングとプロファイルの定義が評価されて、一致するか確認されるため、適切なマシン サイズ (フレーバー)、イメージ、ネットワーク、およびストレージが確実にワークロードに含まれるようになります。

1 アカウント/リージョンのフレーバー マッピングを構成します。

フレーバーは、「T シャツのサイズ決め」と呼ばれることもあります。クラウド テンプレートの構成方法によっては、適用されるフレーバー マッピングによって CPU の数とメモリのサイズが決まります。

フレーバー マッピングの詳細については、[vRealize Automation でのフレーバー マッピングの詳細](#)を参照してください。

a [インフラストラクチャ] - [構成] - [フレーバー マッピング] の順に選択します。

b [新しいフレーバー マッピング] をクリックして、小規模、中規模、大規模のマシンを定義する値を入力します。

これらはサンプル値であることに注意してください。関連するアカウント/リージョンを選択し、サイズを定義する必要があります。

The screenshot shows a configuration form for a flavor named 'small'. The form includes a 'Flavor name' field with the value 'small' and a 'Configuration' section with two columns: 'Account / Region' and 'Value'. The 'Account / Region' column contains 'vCenter Account / wld01-DC' and the 'Value' column contains '2'. There is also a '1' in the 'Value' column and a 'GB' unit selector.

設定	サンプルの値
フレーバー名	small
アカウント/リージョン	vCenter Server アカウント/データセンター
CPU 値	2
メモリ値	1 GB

- c [作成] をクリックします。
- d その他のサイズを作成するには、アカウント/リージョンに対して中規模および大規模のフレーバー マッピングを構成します。

設定	サンプルの値
フレーバー名	medium
アカウント/リージョン	vCenter Server アカウント/データセンター
CPU 値	4
メモリ値	2 GB
フレーバー名	large
アカウント/リージョン	vCenter Server アカウント/データセンター
CPU 値	8
メモリ値	4 GB

2 アカウント/リージョンのイメージ マッピングを構成します。

イメージは、クラウド テンプレート内のマシンのオペレーティング システムです。vCenter Server イメージを使用している場合は、vCenter Server テンプレートを選択します。

イメージ マッピングの詳細については、[vRealize Automation でのイメージ マッピングの詳細](#)を参照してください。

- a [インフラストラクチャ] - [構成] - [イメージ マッピング] の順に選択します。
- b [新しいイメージ マッピング] をクリックして、アカウント/リージョンのイメージを検索します。
- これらはサンプル値であることに注意してください。アカウント/リージョンで検出された関連イメージを選択する必要があります。

centos DELETE

Allows you to define images or machine templates by name in a cloud-agnostic way. ①

Image name * centos

Configuration * Account / Region Image Constraints Cloud Configuration

Q vCenter Account / wldf Q centos7 Example: license:none: ① + ADD

設定	サンプルの値
イメージ名	CentOS
アカウント/リージョン	vCenter Server アカウント
イメージ	centos7


- c [作成] をクリックします。
- d この手順を繰り返して、追加のイメージ マッピングを作成します。たとえば、アカウント/リージョンの ubuntu マッピングを作成します。

3 ネットワーク プロファイルを構成します。

ネットワーク プロファイルは、アカウント/リージョンで使用可能なネットワークおよびネットワーク設定を定義します。プロファイルは、ターゲットの展開環境をサポートする必要があります。


このタスクは、成功するために必要な最小限の構成情報を提供します。ネットワーク プロファイルの詳細については、[vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください。

- a [インフラストラクチャ] - [構成] - [ネットワーク プロファイル] を選択します。
- b [新しいネットワーク プロファイル] をクリックして、vCenter Server アカウント/データセンター アカウント/リージョンのプロファイルを作成します。

 **Network Profile** [DELETE](#)

Summary **Networks** Network Policies Load Balancers Security Groups

A network profile defines a group of networks and network settings used when machines are provisioned.

Account / region  vCenter Account / wld01-DC

Name * Network Profile


Description Networks for development teams.

Capabilities
Capability tags listed here are matched to constraint tags in the cloud template.

Capability tags Enter capability tags [?](#)

設定	サンプルの値
アカウント/リージョン	vCenter Server アカウント/データセンター
名前	ネットワーク プロファイル
説明	開発チーム向けのネットワーク。







- c [ネットワーク] タブをクリックし、[ネットワークの追加] をクリックします。

 **Network Profile** [DELETE](#)

Summary **Networks** Network Policies Load Balancers Security Groups

Networks listed here are used when provisioning to existing, on-demand, or public networks. [?](#)

[+ ADD NETWORK](#) [TAGS](#) [MANAGE IP RANGES](#) [REMOVE](#)

<input type="checkbox"/>	Name	Account / Region	Zone	Network Domain	CIDR	Support Public IP	Default for Zone	Origin	Tags
<input type="checkbox"/>	DevProject-004	 NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	192.168.1.64/27	--	--	 Deployed	
<input type="checkbox"/>	External-mcm13/3520-150877845350	 NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	172.16.1.64/28	--	--	 Discovered	
<input type="checkbox"/>	seg-domain-c8e2a5589de-2772-43f5-9eaa-eddc05e35996-vmware-system-nsx-0	 NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	10.244.0.0/28	--	--	 Discovered	external_id.8... ncp/project_u... ncp/cluster.d... ncp/version.1... ncp/project.v...

1 - 3 of 3 networks

- d アプリケーション開発チームが使用できるようにする NSX ネットワークを選択します。
- この例では、DevProject-004 という名前の NSX-T ネットワークがありました。
- e [ネットワーク ポリシー] タブをクリックして、ポリシーを作成します。

New Network Profile

Summary | Networks | **Network Policies** | Load Balancers | Security Groups

Use these settings when creating outbound, private and routed networks. ⓘ

Isolation policy: None ⓘ

Network Resources
Provide on-demand network resources.

Tier-0 logical router: Q TO ⓘ

Edge cluster: Q edge-widcu-0 ⓘ

CREATE **CANCEL**

設定	サンプルの値
隔離ポリシー	なし
Tier-0 論理ルーター	Tier-0-router
Edge クラスター	EdgeCluster

f [作成] をクリックします。

4 ストレージ プロファイルを構成します。

ストレージ プロファイルは、アカウント/リージョンのディスクを定義します。プロファイルは、ターゲットの展開環境をサポートする必要があります。

ストレージ プロファイルの詳細については、[vRealize Automation でのストレージ プロファイルの詳細](#) を参照してください。

a [インフラストラクチャ] - [構成] - [ストレージ プロファイル] の順に選択します。

b [新規ストレージ プロファイル] をクリックして、vCenter Server/データセンター アカウント/リージョンのプロファイルを作成します。

表で指定されていない場合は、デフォルト値のままにします。

設定	サンプルの値
アカウント/リージョン	vCenter Server アカウント/データセンター
名前	ストレージ プロファイル
データストア/クラスター	十分な容量があり、すべてのホストからアクセス可能なデータストアが選択されました。
このリージョンの優先ストレージ	チェック ボックスを選択します。

c [作成] をクリックします。

手順 4：プロジェクトを作成する

ここでは、プロジェクトの目標について実際に考えてみます。

- アプリケーション クラウド テンプレートを作成して展開するためにコンピューティング リソースにアクセスする必要があるのは、どんなユーザーでしょうか。さまざまなプロジェクト ロールで表示および実行できる内容の詳細については、[vRealize Automation の組織およびサービスのユーザー ロール](#)を参照してください。
- プロジェクトのメンバーは、開発から本番までの行程でアプリケーションを作成しますか。必要なリソースは何ですか。
- 必要なクラウド ゾーンは何ですか。プロジェクトの各ゾーンに、どのような優先順位および制限を設定する必要がありますか。

このチュートリアルでは、社内ソフトウェア アプリケーションを開発して拡張する開発チームをサポートしています。

このタスクは、成功するために必要な最小限の構成情報を提供します。プロジェクトの詳細については、[vRealize Automation Cloud Assembly プロジェクトの詳細](#)を参照してください。

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] の順に選択します。
- 2 [新しいプロジェクト] をクリックして、名前に「**開発プロジェクト**」と入力します。
- 3 [ユーザー] タブをクリックしてから、[ユーザーの追加] をクリックします。

この時点でユーザーを追加する必要はありません。ただし、他のユーザーがクラウド テンプレートを使用するようにしたい場合は、プロジェクトのメンバーである必要があります。

- 4 E メール アドレスを入力して、各ユーザーに割り当てられた権限に応じてプロジェクト メンバーまたは管理者としてユーザーを追加します。

- 5 [プロビジョニング] をクリックして、[ゾーンの追加] - [クラウド ゾーン] の順にクリックします。
- 6 ユーザーが展開できるクラウド ゾーンを追加します。

プロジェクトのクラウド ゾーンにリソースの制限を設定することもできます。後で、他のプロジェクトに異なる制限を設定できます。

プロジェクトのクラウド ゾーンの設定	サンプルの値
クラウド ゾーン	vCenter Server アカウントのクラウド ゾーン
プロビジョニングの優先順位	1
インスタンスの制限	5

- 7 クラウド ゾーンをプロジェクトに追加します。
- 8 [作成] をクリックします。

- 9 プロジェクトがクラウド ゾーンに追加されたことを確認するには、[インフラストラクチャ] - [構成] - [クラウド ゾーン] の順に選択し、vCenter Server アカウント ゾーンのクラウド ゾーン カードを開いて [プロジェクト] タブを確認できるようにします。開発プロジェクトが表示されます。

手順 5：基本的なクラウド テンプレートを設計して展開する

クラウド テンプレートを設計および展開して、そのテンプレートをサポートするようにインフラストラクチャが適切に構成されていることを確認します。後で、プロジェクトのニーズを満たすアプリケーションを作成するときに、テンプレート上に構築できます。

クラウド テンプレートを構築する最善の方法は、コンポーネント単位で構築し、変更するたびにコンポーネントが展開されることを確認することです。このチュートリアルでは、単純なマシンから始めて、リソースの追加を繰り返します。

この手順の例では、YAML コード エディタを使用します。こうすると、コード スニペットを簡単に提供できるようになります。ただし、ダイアログ ボックススペースのユーザー インターフェイスを使用する場合は、**入力** をクリックします。

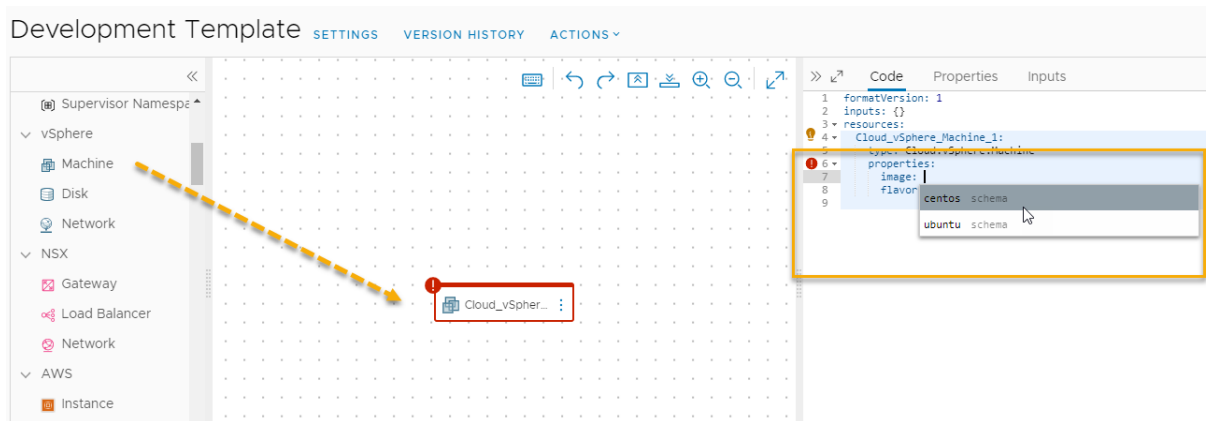
クラウド テンプレートでは、このチュートリアルに記載されている内容よりもはるかに多くの操作を実行できます。詳細については、[6 章 vRealize Automation Cloud Assembly 展開の設計](#)を参照してください。

このチュートリアルでは、vSphere と NSX のリソース タイプを使用します。これらのリソース タイプは vCenter Server クラウド アカウント エンドポイントにのみ展開できます。クラウドに依存しないリソース タイプを使用して、任意のエンドポイントに展開できるクラウド テンプレートを作成することもできます。インフラストラクチャを構成し、任意のエンドポイントのテンプレートをデザインする例については、[チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。



この手順の基本的な手順を示すビデオについては、[基本的なクラウド テンプレートを設計して展開する方法](#)を参照してください。

- 1 [デザイン] - [クラウド テンプレート] の順に選択します。
- 2 [新規作成元] - [空白のキャンバス] の順に選択します。
- 3 [名前]**開発テンプレート** に入力し、[プロジェクト]**開発プロジェクト** の順に選択して、[作成] をクリックします。
- 4 vSphere マシンをデザイン キャンバスに追加し、テストして、展開します。



- a [リソース タイプ] ペインで、[vSphere マシン] をキャンバスにドラッグします。

[コード] ペインにマシンの YAML が表示され、イメージと事前定義された CPU およびメモリのプロパティの値が空になることに注意してください。このテンプレートを使用可能にすることで、柔軟なサイジングが可能になります。

- b イメージ値を選択するには、image の単一引用符の間にポインタを置き、構成したイメージのリストから [centos] を選択します。

これらはサンプル値であることに注意してください。centos イメージを構成しなかった場合は、構成したイメージを選択します。

- c イメージ プロパティの下に行を作成し、flavor を入力または選択し、リストから small を選択します。
d cpuCount および totalMemory を削除します。

YAML は次の例のようになります。

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: centos
      flavor: small
```

- e [テスト] をクリックします。

テストでは、クラウド テンプレートの構文と配置を検証できます。テストが成功しても、テンプレートをエラーなしで展開できるとは限りません。

Test Result for Development Template



Successful

This simulation only tests syntax, placement and basic validity

1 Info Provisioning Diagram

Cloud_vSphere_Machine_1



LINE 4

テストが失敗した場合は、[プロビジョニング図] をクリックして、障害ポイントを確認します。この図を使用してトラブルシューティングを行う方法については、[基本的なクラウド テンプレートのテスト](#)を参照してください。

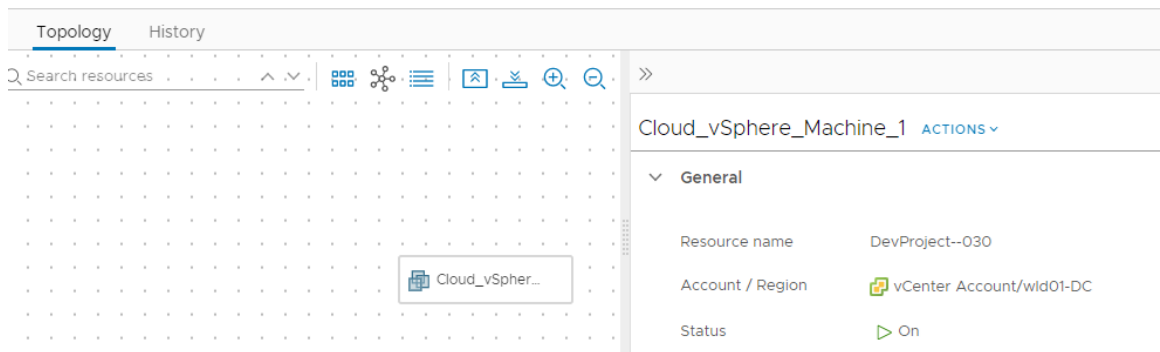
f [[展開]] をクリックします。

g [展開名] に **DevTemplate - machine** と入力して、[展開] をクリックします。

展開の進行状況は、DevTemplate の展開の詳細画面または [展開] タブで追跡できます。

展開に失敗した場合は、問題をトラブルシューティングして、テンプレートを変更できます。[vRealize Automation Cloud Assembly の展開に失敗した場合の対処](#)を参照してください。

展開が成功すると、[展開] タブに次の例のように表示されます。



5 テンプレートのバージョンを作成し、ネットワークを追加します。

クラウド テンプレートのバージョン管理は、Service Broker カタログで利用できるようにするために必要ですが、開発時に元に戻すのに最適なバージョンがあると便利です。

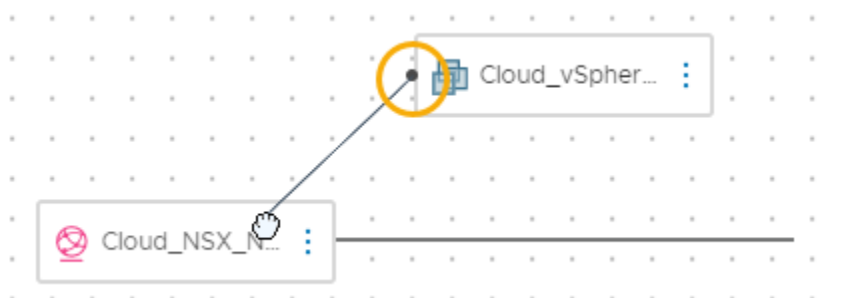
a デザイン キャンバスでテンプレートを開きます。

b [バージョン] をクリックし、**シンプルな展開可能マシン**のような [説明] を入力して、[作成] をクリックします。

c [リソース タイプ] ペインで、[NSX ネットワーク] リソース タイプをキャンバスにドラッグします。

d マシンをネットワークに接続します。

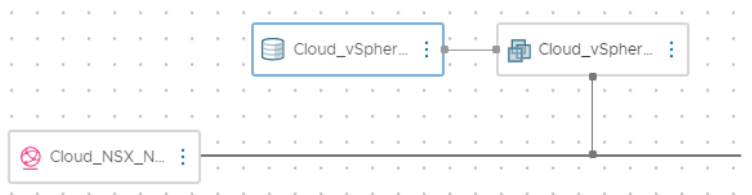
マシン コンポーネント上の小さな円をクリックし、接続をネットワークまでドラッグします。



YAML が次の例のようになっていることに注意してください。

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: centos
      flavor: small
      networks:
        - network: '${resource.Cloud_NSX_Network_1.id}'
      attachedDisks: []
  Cloud_NSX_Network_1:
    type: Cloud.NSX.Network
    properties:
      networkType: existing
```

- e [テスト] をクリックして、テンプレートを検証します。
 - f [[展開]] をクリックします。
 - g 名前に **DevTemplate - machine - network** と入力して、[展開] をクリックします。
 - h 進行状況を追跡し、展開が成功したことを確認します。
- 6 テンプレートのバージョンを作成し、データ ディスクを追加します。
- a デザイン キャンバスでテンプレートを開きます。
 - b テンプレートのバージョンを作成します。
- 説明として **既存ネットワークを使用するマシン** と入力します。
- c [リソース タイプ] ペインで、[vSphere ディスク] リソース タイプをキャンバスにドラッグします。
 - d ディスクをマシンに接続します。



YAML が次の例のようになっていることに注意してください。

```
formatVersion: 1
inputs: {}
resources:
  Cloud_vSphere_Disk_1:
    type: Cloud.vSphere.Disk
    properties:
      capacityGb: 1
  Cloud_vSphere_Machine_1:
```

```

type: Cloud.vSphere.Machine
properties:
  image: centos
  flavor: small
  networks:
    - network: '${resource.Cloud_NSX_Network_1.id}'
  attachedDisks:
    - source: '${resource.Cloud_vSphere_Disk_1.id}'
Cloud_NSX_Network_1:
  type: Cloud.NSX.Network
  properties:
    networkType: existing

```

- e テンプレートをテストします。
- f **DevTemplate-machine-network-storage** という名前を使用して、テンプレートを展開します。
- g 進行状況を追跡し、展開が成功したことを確認します。
- h テンプレートのバージョンを作成します。

説明として **既存ネットワークおよびストレージ ディスクを使用するマシン** と入力します。

この最終バージョンで、作業テンプレートをサービス カタログに追加できるようになります。

チュートリアルの結果

これで、作業システムとして Cloud Assembly を構成するワークフローが完了しました。次の概念について学習しました。

- クラウド アカウントとは、クラウド ベンダーのエンドポイントに vRealize Automation Cloud Assembly を接続するための認証情報です。
- クラウド ゾーンとは、プロジェクトのニーズやコストの管理目標に基づいてさまざまなプロジェクトに割り当て、アカウント/リージョン内の特定のコンピューティング リソースです。
- インフラストラクチャ リソースとは、クラウド テンプレートで使用されるアカウント/リージョンに関連付けられたリソースの定義です。
- プロジェクトとは、プロジェクトのアプリケーション開発目標に基づいて、ユーザーにクラウド ゾーンへのアクセス権を付与する方法です。
- クラウド テンプレートとは、繰り返し開発および展開されるアプリケーション ワークロードの定義です。

このチュートリアルは vRealize Automation Cloud Assembly 開発環境の基盤となります。このプロセスを使用してインフラストラクチャを構築し、クラウド テンプレート開発スキルを向上させることができます。

チュートリアル：本番ワークロードをプロビジョニングするための vRealize Automation Cloud Assembly の構成

クラウド テンプレートの設計者がテンプレートを作成して展開するときに、vRealize Automation Cloud Assembly が作業を代行するように、クラウド管理者としてプロジェクトの展開プロセスを自動化することができます。

す。たとえば、特定のカスタム マシン命名パターンを使用してワークロードを展開したり、マシンを特定の Active Directory 組織単位に追加したり、特定の DNS および IP アドレス範囲を使用したりする場合などです。

プロジェクト展開のプロセスを自動化することで、さまざまなデータセンターやクラウド環境で複数のプロジェクトを簡単に管理できるようになります。

すべてのタスクを完了する必要はありません。管理目標に応じて、これらのタスクを併用および適用することをお勧めします。次は、使用できるタスクの一覧です。

- [マシン名のカスタマイズ](#)
- [Active Directory マシン レコードの作成](#)
- [ネットワーク DNS と内部 IP アドレス範囲の設定](#)

開始する前に

このチュートリアルでは、インフラストラクチャを構成し、マシンとネットワークを含むクラウド テンプレートを正常に展開しておく必要があります。システムで次の構成がすでに行われていることを確認します。

- インフラストラクチャ チュートリアルで指定されたすべての手順が正常に実行されている。[チュートリアル：vRealize Automation Cloud Assembly での vSphere インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。
- Cloud Assembly 管理者ロールが割り当てられている。[vRealize Automation の組織およびサービスのユーザー ロール](#)を参照してください。

マシン名のカスタマイズ

このタスクの目的は、開発プロジェクト用に展開されたマシンに、プロジェクトのコストセンター、展開時に選択されたリソース タイプ、および一意性を確保するためにインクリメントされた値に基づいて名前を付けることです。たとえば、DevProject-centos-021 のような名前を付けます。

この例は、命名の要件に合わせて調整できます。

プロジェクトの詳細については、[5 章 vRealize Automation Cloud Assembly プロジェクトの追加と管理](#)を参照してください。



このカスタム命名の例を示すビデオについては、[展開用のカスタム命名テンプレートの作成方法](#)を参照してください。

- 1 [インフラストラクチャ] - [プロジェクト] の順に選択します。
- 2 既存のプロジェクトを選択するか、新しいプロジェクトを作成します。
このチュートリアルでは、プロジェクト名は Development Project です。
- 3 [作成] をクリックします。
- 4 [プロジェクト] 画面でタイトルのプロジェクト名をクリックし、プロジェクトを構成できるようにします。
- 5 [ユーザー] タブをクリックして、このプロジェクトのメンバーであるユーザーを追加します。

6 [プロビジョニング] タブをクリックします。

- a [ゾーン] セクションで、[ゾーンの追加] をクリックし、このプロジェクトのワークロードが展開されている、使用可能なクラウド ゾーンを追加します。
- b [カスタム プロパティ] セクションで、名前が **costCenter**、値が **DevProject** のカスタム プロパティを追加します。カスタム プロパティとカスタム命名のサンプル値が表示されているプロジェクトの [プロビジョニング] タブ

Custom Properties

Specify the custom properties that should be added to all requests in this project. ⓘ

Define custom properties	Name	Value
	costCenter	DevProject

Custom Naming

Specify the naming template to be used for machines, networks, security groups and disks provisioned in this project.

Template

`$(resource.costCenter)-$(resource.osType)-${###}` ⓘ

Hint: Avoid conflicting names by generating digits in names. `$(#####)`

- c [カスタム命名] セクションで、次の命名テンプレートを追加します。

```
$(resource.costCenter)-$(resource.osType)-${###}
```

`$(resource.osType)` は、クラウド テンプレートを展開するときに選択されたオペレーティング システムに基づきます。

7 [保存] をクリックします。

8 オペレーティング システム タイプの入力値を使用して、クラウド テンプレートを更新します。

入力値を使用すると、ユーザーの展開申請フォームのカスタマイズや、開発プロセスの簡素化を直接行うことができます。入力値を作成することで、単一のクラウド テンプレートを使用して、さまざまな構成のワークロードを展開できるようになります。例としては、サイズやオペレーティング システムなどがあります。

この例では、前のチュートリアルの開発テンプレートを使用します。[手順 5: 基本的なクラウド テンプレートを設計して展開する](#)を参照してください。

- a [デザイン] を選択し、開発テンプレートを開きます。
- b [コード] ペインで、次の変更を加えて YAML を更新します。
 - Inputs セクションで **osType** を追加します。

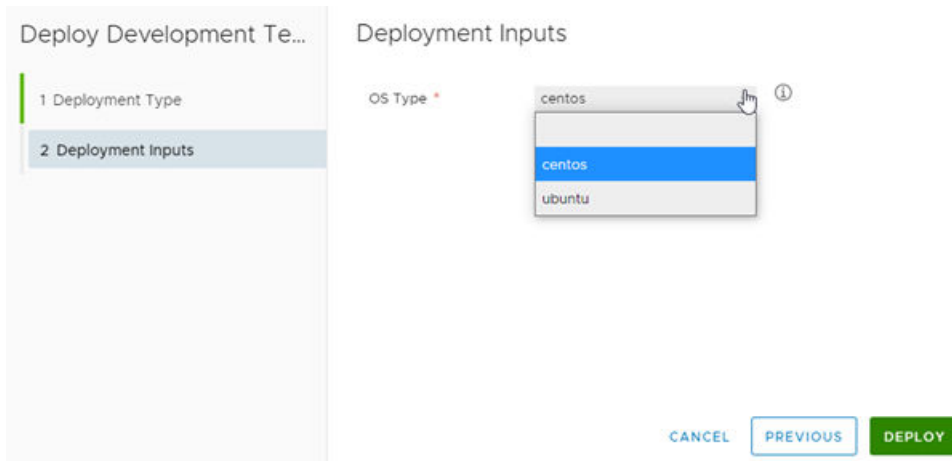
次の手順では、osType の入力を使用してイメージを指定することもできます。enum セクションに文字列を追加する場合、値（この例では centos および ubuntu）は、[インフラストラクチャ] - [構成] - [イメージ マッピング] で定義したイメージ名と一致する必要があります。たとえば、イメージ マッピング名が centos ではなく CentOS の場合、入力セクションでは CentOS を使用する必要があります。

```
inputs:
  osType:
    type: string
    title: OS Type
    description: Select the operating system.
    enum:
      - centos
      - ubuntu
```

- Cloud_vSphere_Machine_1 セクションで、image を osType の入力パラメータ (\${input.osType}) に更新し、同じ入力パラメータを持つ osType カスタム プロパティを追加します。

```
resources:
  Cloud_vSphere_Disk_1:
    type: Cloud.vSphere.Disk
    properties:
      capacityGb: 1
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: ${input.osType}
      osType: ${input.osType}
      flavor: small
      networks:
        - network: '${resource.Cloud_NSX_Network_1.id}'
      attachedDisks:
        - source: '${resource.Cloud_vSphere_Disk_1.id}'
  Cloud_NSX_Network_1:
    type: Cloud.NSX.Network
    properties:
      networkType: existing
```

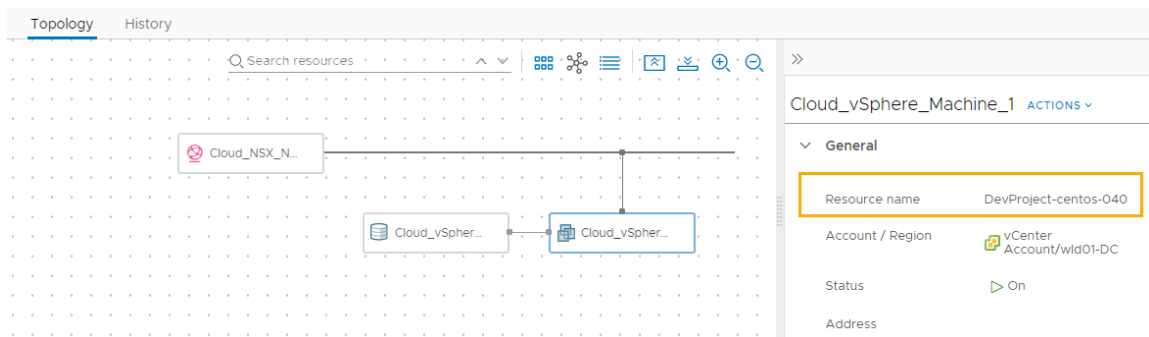
- [展開] をクリックし、名前として「**Custom name deployment test**」を入力します。
- [次へ] をクリックします。
- ドロップダウン メニューで [centos] オペレーティング システムを選択します。



f [[展開]] をクリックします。

- 9 進行状況を追跡し、展開が成功したことを確認します。

この例のマシン名は DevProject-centos-026 です。この例は、このタスクの最初に参照されているチュートリアルに基づいています。



Active Directory マシン レコードの作成

ワークロードをプロビジョニングするときに、Active Directory にマシン レコードを作成できます。このタスクを自動実行してプロジェクトを展開するように vRealize Automation Cloud Assembly を構成することで、クラウド管理者としてのワークロードを軽減することができます。

- 1 Active Directory 統合を追加します。

- a [インフラストラクチャ] - [接続] - [統合] の順に選択します。

これらの手順では、この Active Directory マシン レコードのチュートリアルに関連する基本的な Active Directory 構成について説明します。Active Directory 統合の詳細については、[vRealize Automation Cloud Assembly](#) で [Active Directory 統合を作成する方法](#)を参照してください。

- b [統合の追加] をクリックして、[Active Directory] をクリックします。

c この統合に使用する名前を入力します。

d [LDAP ホスト/IP アドレス] と関連する認証情報を入力します。

e [ベース DN] を入力します。

このチュートリアルでは、**ou=AppDev,dc=cmbu,dc=local** のように入力します。AppDev は、プロジェクトに追加するコンピュータの OU の親 OU です。

f [追加] をクリックします。

2 統合にプロジェクトを追加します。

3 Active Directory 統合で、[プロジェクト] タブをクリックし、[プロジェクトの追加] をクリックします。

Add Projects

Select a project and the OU it will be mapped to by adding its relative DN. The effective DN is created by appending the RDN to the integration base DN (**ou=AppDev,dc=cmbu,dc=local**).

a アプリケーション開発プロジェクトを選択します。

b 相対的な DN を入力します。たとえば、**OU=AppDev-Computers** のように入力します。

c [追加] をクリックします。

4 統合に変更内容を保存するには、[保存] をクリックします。

- 5 プロジェクトのクラウド テンプレートを展開し、マシンが正しい Active Directory OU に追加されていることを確認します。

ネットワーク DNS と内部 IP アドレス範囲の設定

ネットワーク プロファイルを追加または更新して、DNS サーバと内部 IP アドレス範囲を含めます。

vSphere、NSX-V、または NSX-T 用のクラウド アカウントがすでに作成されている必要があります。[チュートリアル : vRealize Automation Cloud Assembly での vSphere インフラストラクチャおよび展開のセットアップとテスト](#)または [vRealize Automation Cloud Assembly へのクラウド アカウントの追加](#)を参照してください。

- 1 [インフラストラクチャ] - [構成] - [ネットワーク プロファイル] を選択します。

- 2 既存のプロファイルを選択するか、プロファイルを作成します。

- 3 [サマリ] タブで [アカウント/リージョン] を選択し、名前を入力します。

このチュートリアルでは、ネットワーク プロファイル名は Network Profile です。

- 4 ネットワークを追加します。

- a [ネットワーク] タブをクリックします。
- b [ネットワークの追加] をクリックします。
- c 1つ以上の NSX または vSphere ネットワークを追加します。
- d [追加] をクリックします。

- 5 DNS サーバを構成します。

- a [ネットワーク] タブのネットワーク リストで、ネットワーク名をクリックします。

Summary

Networks

Network Policies

Load Balancers

Security

Networks listed here are used when provisioning to existing, on-demand, or p

+ ADD NETWORK

TAGS

MANAGE IP RANGES

REMOVE

<input type="checkbox"/>	Name ↑	Account / Region	Zone	Network Domain	CIDR
<input type="checkbox"/>	DevProject --004	NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	192.168.1.64/27

- b このネットワークで使用する DNS サーバの IP アドレスを入力します。

DevProject--004

DNS servers

192.168.1.22
192.168.1.23

DNS search domains

company.local

DNS Servers

Use a comma separated list or new lines.

c [保存] をクリックします。

6 ネットワークの IP アドレス範囲を指定します。

a ネットワーク リストで、ネットワーク名の横にあるチェック ボックスを選択します。

Network Profile **DELETE**

Summary **Networks** Network Policies Load Balancers Security Groups

Networks listed here are used when provisioning to existing, on-demand, or public networks. ⓘ

+ ADD NETWORK TAGS **MANAGE IP RANGES** REMOVE

<input type="checkbox"/>	Name	Account / Region	Zone	Network Domain	CIDR	Subnet
<input type="checkbox"/>	External-mcm1343745-148168716643	NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	172.16.12.64/28	
<input type="checkbox"/>	NSX-mcm1376447-151082888186	NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	192.168.100.32/28	
<input checked="" type="checkbox"/>	NSX-mcm39835-146434698964	NSX-T Account		overlay-tz-sc2vc05-vip-nsx-mgmt.cmbu.local	192.168.1.0/27	

1

b [IP アドレス範囲の管理] をクリックします。

c [IP アドレス範囲の管理] ダイアログ ボックスで、[新規 IP アドレス範囲] をクリックします。

New IP Range

Network *	NSX-mcm1376447-151082888186
Source	<input checked="" type="radio"/> Internal <input type="radio"/> External
Name *	DevProject Range
Description	<div></div>
CIDR	192.168.100.32/28
Start IP address *	192.168.100.34
End IP address *	192.168.100.46

d 名前を入力します。

たとえば、「**DevProject Range**」のように入力します。

e 範囲を定義するには、[開始 IP アドレス] と [終了 IP アドレス] を入力します。

f [追加] をクリックします。

g 範囲を追加するか、[閉じる] をクリックします。

- 構成した関連するネットワーク アカウント/リージョンを含むクラウド ゾーンを、開発プロジェクトに追加します。
- プロジェクトのクラウド テンプレートを展開し、マシンが指定した IP アドレス範囲内でプロビジョニングされていることを確認します。

チュートリアル: vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト

vRealize Automation Cloud Assembly に関するこのエンドツーエンドのチュートリアルでは、マルチクラウド 設定での展開方法について説明します。同じクラウド テンプレートを複数のプロバイダ、この場合は AWS と Microsoft Azure に展開します。

この例では、アプリケーションは WordPress サイトです。一連のセットアップを順番に確認して、設計全体を完成させるプロセスを理解します。

ここに示す名前と値は、単なる例であることに注意してください。使用環境で、この例をそのまま使用することはできません。

それぞれのクラウド インフラストラクチャと展開のニーズに合わせるために、このサンプルの値のどこを独自に置き換えるかを検討してください。

パート 1：サンプル vRealize Automation Cloud Assembly インフラストラクチャの構成

最初に、この後で vRealize Automation Cloud Assembly エンジニアリング部門のユーザーがアプリケーションを開発、テストし、本番環境に移行できるように、リソースを構成します。

インフラストラクチャには、クラウド ターゲットと、WordPress サイトが必要とする使用可能なマシン、ネットワーク、およびストレージの定義が含まれます。

1. クラウド アカウントの追加

この手順では、クラウド管理者が 2 つのクラウド アカウントを追加します。このサンプル プロジェクトでは、AWS での開発およびテスト作業と、Azure での本番環境への移動を想定しています。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に移動します。
- 2 [クラウド アカウントの追加] をクリックし、Amazon Web Services を選択して値を入力します。

設定	サンプルの値
アクセス キーの ID	R5SDR3PXVV2ZW8B7YNSM
プライベート アクセス キー	SZXAINXU4UHNQAQ1E156S
名前	OurCo-AWS
説明	WordPress

すべての値は単なる例であることに注意してください。実際のアカウントの詳細は異なります。

- 3 認証情報を確認するには、[検証] をクリックします。
- 4 [追加] をクリックします。
- 5 新しく追加されたアカウントの [構成] を編集し、us-east-1 および us-west-2 へのプロビジョニングを許可します。
- 6 [クラウド アカウントの追加] をクリックし、Microsoft Azure を選択して値を入力します。

設定	サンプルの値
サブスクリプション ID	ef2avpf-dfdv-zxlugui1i-g4h0-i8ep2jwp4c9arbf
テナント ID	dso9wv3-4zgc-5nrcy5h3m-4skf-nnovp40wfxsro22r
クライアント アプリケーションの ID	bg224oq-3ptp-mbhi6aa05-q511-uf1yjr2sttyik6bs
クライアント アプリケーションのプライベート キー	7uqxi57-0wtn-kymgf9wcj-t2l7-e52e4nu5fig4pmd

設定	サンプルの値
名前	OurCo-Azure
説明	WordPress

- 7 認証情報を確認するには、[検証] をクリックします。
- 8 [追加] をクリックします。
- 9 新しく追加されたアカウントの [設定] を編集し、米国東部リージョンへのプロビジョニングを許可します。

2. クラウド ゾーンの追加

このサンプルの手順では、クラウド管理者は、開発、テスト、および本番用の 3 つのクラウド ゾーンを追加します。クラウド ゾーンは、プロジェクトが WordPress サイトをサポートするために、マシン、ネットワーク、およびストレージを展開するリソースです。

手順

- 1 [インフラストラクチャ] - [構成] - [クラウド ゾーン] の順に移動します。
- 2 [新しいクラウド ゾーン] をクリックして、開発環境の値を入力します。

クラウド ゾーンの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-east-1
名前	OurCo-AWS-US-East
説明	WordPress
配置ポリシー	デフォルト
機能タグ	env:dev

すべての値は単なる例であることに注意してください。ゾーンの詳細は異なります。

- 3 [コンピュート] をクリックし、必要なゾーンがあることを確認します。
- 4 [作成] をクリックします。
- 5 このプロセスを 2 回繰り返し、テスト環境と本番環境の値を指定します。

クラウド ゾーンの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-west-2
名前	OurCo-AWS-US-West
説明	WordPress
配置ポリシー	デフォルト
機能タグ	env:test

クラウド ゾーンの設定	サンプルの値
アカウント/リージョン	OurCo-Azure/East US
名前	OurCo-Azure-East-US
説明	WordPress
配置ポリシー	デフォルト
機能タグ	env:prod

3. フレーバー マッピングの追加

このサンプルの手順で、クラウド管理者は、展開に応じて変化する可能性のあるキャパシティのニーズを考慮して、フレーバー マッピングを追加します。

フレーバー マッピングは、さまざまなサイズのマシン展開に対応しており、非公式には T シャツ サイジングと呼ばれています。

手順

- 1 [インフラストラクチャ] - [設定] - [フレーバー マッピング] の順に移動します。各クラウド ゾーンは、小規模、中規模、および大規模のフレーバーに対応する必要があります。
- 2 [新しいフレーバー マッピング] をクリックし、開発クラウド ゾーンの値を入力します。

設定	サンプルの値
フレーバー名	small
アカウント/リージョン 値	OurCo-AWS/us-east-1 t2.micro
アカウント/リージョン 値	OurCo-AWS/us-west-2 t2.micro
アカウント/リージョン 値	OurCo-Azure/East US Standard_A0

すべての値は単なる例であることに注意してください。フレーバーは異なります。

- 3 [作成] をクリックします。
- 4 このプロセスを 2 回繰り返し、中規模および大規模のフレーバーの値を指定します。

設定	サンプルの値
フレーバー名	medium
アカウント/リージョン 値	OurCo-AWS/us-east-1 t2.medium

設定	サンプルの値
アカウント/リージョン 値	OurCo-AWS/us-west-2 t2.medium
アカウント/リージョン 値	OurCo-Azure/East US Standard_A3

設定	サンプルの値
フレーバー名	large
アカウント/リージョン 値	OurCo-AWS/us-east-1 t2.large
アカウント/リージョン 値	OurCo-AWS/us-west-2 t2.large
アカウント/リージョン 値	OurCo-Azure/East US Standard_A7

4. イメージ マッピングの追加

このサンプルの手順では、クラウド管理者が Ubuntu 用のイメージ マッピング、WordPress サーバのホスト、およびその MySQL データベース サーバを追加します。

イメージ マッピングを追加して、オペレーティング システムを計画します。各クラウド ゾーンには Ubuntu イメージ マッピングが必要です。

手順

- 1 [インフラストラクチャ] - [設定] - [イメージ マッピング] の順に移動します。
- 2 [新しいイメージ マッピング] をクリックし、Ubuntu サーバの値を入力します。

設定	サンプルの値
イメージ名	Ubuntu
アカウント/リージョン 値	OurCo-AWS/us-east-1 ubuntu-16.04-server-cloudimg-amd64
アカウント/リージョン 値	OurCo-AWS/us-west-2 ubuntu-16.04-server-cloudimg-amd64
アカウント/リージョン 値	OurCo-Azure/East US azul-zulu-ubuntu-1604-923eng

すべての値は単なる例であることに注意してください。実際のイメージとは異なります。

- 3 [作成] をクリックします。

5. ネットワーク プロファイルの追加

このサンプルの手順では、クラウド管理者が各クラウド ゾーンにネットワーク プロファイルを追加します。

各プロファイルで、管理者は WordPress マシンのネットワークと、最終的なロード バランサのもう一方の側に配置される 2 つ目のネットワークを追加します。2 つ目のネットワークは、ユーザーが最終的に接続するネットワークになります。

手順

- 1 [インフラストラクチャ] - [構成] - [ネットワーク プロファイル] の順に移動します。
- 2 [新規ネットワーク プロファイル] をクリックして、開発クラウド ゾーンのプロファイルを作成します。

ネットワーク プロファイルの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-east-1
名前	devnets
説明	WordPress

- 3 [ネットワーク] をクリックし、[ネットワークの追加] をクリックします。

- 4 wpnet、appnet-public を選択して [追加] をクリックします。

すべての値は単なる例であることに注意してください。実際のネットワーク名は異なります。

- 5 [作成] をクリックします。

この WordPress の例では、ネットワーク ポリシーまたはネットワーク セキュリティの設定を指定する必要はありません。

- 6 このプロセスを 2 回繰り返して、WordPress のサンプル テストと本番環境のクラウド ゾーンネットワーク プロファイルを作成します。いずれの場合も、wpnet および appnet-public ネットワークを追加します。

ネットワーク プロファイルの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-west-2
名前	testnets
説明	WordPress

ネットワーク プロファイルの設定	値
アカウント/リージョン	OurCo-Azure/East US
名前	prodnets
説明	WordPress

6. ストレージ プロファイルの追加

このサンプルの手順では、クラウド管理者が各クラウド ゾーンにストレージ プロファイルを追加します。

管理者は、本番環境ゾーンに高速ストレージを、開発およびテストには一般的なストレージを配置します。

手順

- 1 [インフラストラクチャ] - [構成] - [ストレージ プロファイル] の順に移動します。
- 2 [新規ストレージ プロファイル] をクリックして、開発クラウド ゾーンのプロファイルを作成します。

アカウントまたはリージョンを選択すると、追加のフィールドが表示されます。

ストレージ プロファイルの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-east-1
名前	OurCo-AWS-US-East-Disk
説明	WordPress
デバイス タイプ	EBS
ボリューム タイプ	汎用 SSD
機能タグ	storage:general

すべての値は単なる例であることに注意してください。

- 3 [作成] をクリックします。
- 4 このプロセスを繰り返して、テスト クラウド ゾーンにプロファイルを作成します。

ストレージ プロファイルの設定	サンプルの値
アカウント/リージョン	OurCo-AWS/us-west-2
名前	OurCo-AWS-US-West-Disk
説明	WordPress
デバイス タイプ	EBS
ボリューム タイプ	汎用 SSD
機能タグ	storage:general

- 5 このプロセスを繰り返して、本番環境のクラウド ゾーンにプロファイルを作成します。これは Azure ゾーンであるため、設定が異なります。

ストレージ プロファイルの設定	サンプルの値
アカウント/リージョン	OurCo-Azure/East US
名前	OurCo-Azure-East-US-Disk
説明	WordPress
ストレージ タイプ	管理対象ディスク
ディスク タイプ	プレミアム LRS

ストレージ プロファイルの設定	サンプルの値
OS ディスク キャッシュ	読み取り専用
データのディスク キャッシュ	読み取り専用
機能タグ	storage:fast

次のステップ

プロジェクトを作成してユーザーを特定し、プロビジョニング設定を定義します。[パート 2：サンプル vRealize Automation Cloud Assembly プロジェクトの作成](#)を参照してください。

パート 2：サンプル vRealize Automation Cloud Assembly プロジェクトの作成

サンプルの vRealize Automation Cloud Assembly プロジェクトによって、プロビジョニング可能なユーザーが有効になり、実行可能なプロビジョニングの程度が設定されます。

プロジェクトで、ユーザーとプロビジョニングの設定を定義します。

- ユーザーと権限のロール レベル
- クラウド ゾーンにプロビジョニングされている展開の優先順位
- クラウド ゾーンあたりの展開インスタンスの最大数

手順

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] の順に移動します。
- 2 [新規プロジェクト] をクリックし、WordPress という名前を入力します。
- 3 [ユーザー] をクリックし、[ユーザーの追加] をクリックします。
- 4 ユーザーのメール アドレスとロールを追加します。

ユーザーを正常に追加するには、VMware Cloud Services 管理者がそのユーザーに対して vRealize Automation Cloud Assembly へのアクセスを有効にしている必要があります。

ここに示されているアドレスは単なる例であることに注意してください。

- chris.ladd@ourco.com、メンバー
 - kerry.mott@ourco.com、メンバー
 - pat.tubb@ourco.com、管理者
- 5 [プロビジョニング] をクリックし、[クラウド ゾーンの追加] をクリックします。

6 ユーザーが展開できるクラウド ゾーンを追加します。

プロジェクトのクラウド ゾーンの設定	サンプルの値
クラウド ゾーン	OurCo-AWS-US-East
プロビジョニングの優先順位	1
インスタンスの制限	5
クラウド ゾーン	OurCo-AWS-US-West
プロビジョニングの優先順位	1
インスタンスの制限	5
クラウド ゾーン	OurCo-Azure-East-US
プロビジョニングの優先順位	0
インスタンスの制限	1

7 [作成] をクリックします。

8 [インフラストラクチャ] - [構成] - [クラウド ゾーン] の順に移動し、作成済みのゾーンを開きます。

9 [プロジェクト] をクリックし、WordPress がゾーンへのプロビジョニングが許可されたプロジェクトであることを確認します。

10 作成済みの他のゾーンを確認します。

次のステップ

基本的なクラウド テンプレートを作成します。

パート 3 : サンプル vRealize Automation Cloud Assembly テンプレートの設計と展開

次に、サンプル アプリケーション（WordPress サイト）を汎用的なクラウド テンプレートの形式で定義します。このテンプレートは、設計を変更することなく別のクラウド ベンダーに展開できます。

この例は、WordPress アプリケーション サーバ、MySQL データベース サーバ、およびサポート リソースで構成されています。テンプレートは数個のリソースから開始し、それらの変更や追加に伴って拡張されます。

クラウド管理者が設定したインフラストラクチャである[パート 1 : サンプル vRealize Automation Cloud Assembly インフラストラクチャの構成](#)で使用されている値は、次のとおりです。

- 2 つのクラウド アカウント、AWS および Azure。
- 3 つのクラウド ゾーン環境 :
 - 開発—OurCo-AWS-US-East
 - テスト—OurCo-AWS-US-West
 - 本番—OurCo-Azure-East-US
- 各ゾーンの小規模、中規模、および大規模のコンピューティング リソースでのフレーバー マッピング。
- 各ゾーンで構成された Ubuntu のイメージ マッピング。
- 各ゾーンの内部および外部サブネットを含むネットワーク プロファイル。

- 展開先のストレージ。開発およびテスト ゾーン用の一般的なストレージ、および本番ゾーン用の高速ストレージ。
- このサンプル プロジェクトには、3 つすべてのクラウド ゾーン環境と、設計を作成できるユーザーが含まれています。

前提条件

実行するためには、使用するインフラストラクチャの値の意味を理解しておく必要があります。このサンプルでは、開発およびテストに AWS を使用し、本番に Azure を使用しています。独自のクラウド テンプレートを作成する場合は、独自の値に置き換えてください。これは通常、クラウド管理者によって設定されています。

手順

1 基本的なクラウド テンプレートの作成

この vRealize Automation Cloud Assembly 設計のサンプルでは、アプリケーション サーバが 1 台だけの場合など、最小限の WordPress リソースのみを含むクラウド テンプレートから開始します。

2 基本的なクラウド テンプレートのテスト

設計中は、通常、基幹部分から開始し、クラウド テンプレートの拡張に伴って展開およびテストすることにより、テンプレートをビルドします。このサンプルでは、vRealize Automation Cloud Assembly に組み込まれた進行中のテストの一部について説明します。

3 クラウド テンプレートの拡張

サンプル アプリケーション用の基本的な vRealize Automation Cloud Assembly テンプレートを作成してテストした後、このテンプレートをマルチティア アプリケーションに拡張して、開発環境とテスト環境、最終的には本番環境に展開できるようにします。

基本的なクラウド テンプレートの作成

この vRealize Automation Cloud Assembly 設計のサンプルでは、アプリケーション サーバが 1 台だけの場合など、最小限の WordPress リソースのみを含むクラウド テンプレートから開始します。

vRealize Automation Cloud Assembly は、infrastructure-as-code ツールです。開始するには、リソースをデザイン キャンバスにドラッグします。次に、キャンバスの右側にあるコード エディタを使用して詳細を設定します。

コード エディタを使用すると、コードの入力、切り取り、貼り付けを直接行うことができます。コードの編集に慣れていない場合は、キャンバスでリソースを選択し、コード エディタの [プロパティ] タブをクリックして値を入力します。入力した値は、コードに直接入力した場合と同じように表示されます。

手順

- 1 [デザイン] - [クラウド テンプレート] の順に選択し、[新規作成元] - [空白のキャンバス] に移動します。
- 2 クラウド テンプレートに **Wordpress-BP** という名前を付けます。
- 3 [WordPress] プロジェクトを選択し、[作成] をクリックします。

- 4 クラウド テンプレート デザイン画面の左側にあるリソースから、クラウドに依存しない 2 台のマシンをキャンバスにドラッグします。

マシンは、WordPress アプリケーション サーバ (WebTier) および MySQL データベース サーバ (DBTier) として機能します。

- 5 右側でマシンの YAML コードを編集して、名前、イメージ、フレーバー、および制約タグを追加します。

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

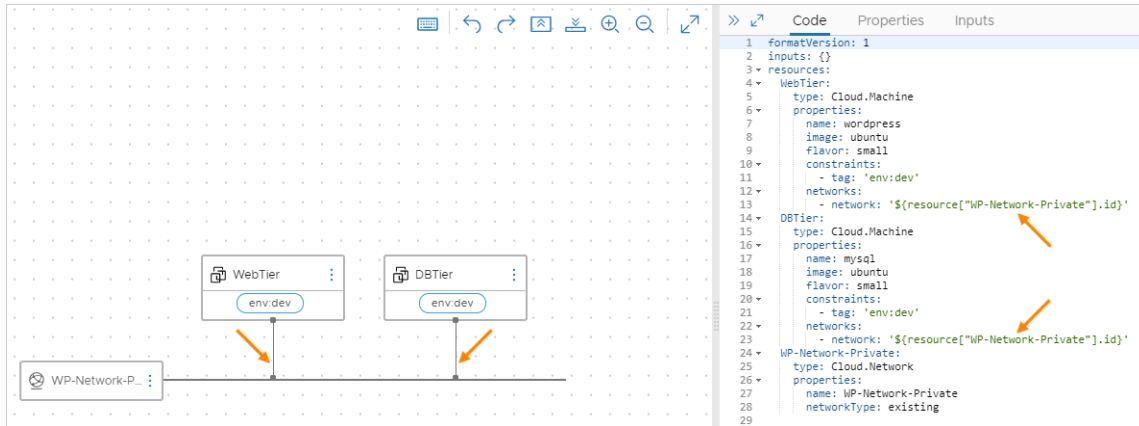
- 6 クラウドに依存しないネットワークをキャンバスにドラッグし、そのコードを編集します。

```
WP-Network-Private:
  type: Cloud.Network
  properties:
    name: WP-Network-Private
    networkType: existing
```

- 7 マシンをネットワークに接続します。

キャンバスで、ネットワーク ブロックの上にカーソルを置き、線がブロックに接触している箇所でバブルをクリックして押したままにし、マシン ブロックにドラッグしてリリースします。

接続線を作成するときは、エディタによってネットワーク コードがマシンに自動的に追加されることに注意します。



8 ユーザー入力プロンプトを追加します。

いくつかの場所では、サンプルのインフラストラクチャが複数のオプションに対して設定されていました。例：

- 開発、テスト、および本番のためのクラウド ゾーン環境
- 小型、中型、および大型マシンのフレーバー マッピング

クラウド テンプレートで特定のオプションを直接設定することもできますが、より優れた方法は、テンプレートの導入時にユーザーがオプションを選択できるようにすることです。ユーザーの入力を求めると、ハードコードされたテンプレートを多用せず、さまざまな方法で導入可能なテンプレートを 1 つ作成できます。

- a ユーザーが導入時にマシン サイズとターゲット環境を選択できるように、コードに `inputs` セクションを作成します。選択可能な値を定義します。

```
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 コードの `resources` セクションで、ユーザーの選択を求めるプロンプトを表示するために、`${input.input-name}` コードを追加します。

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

- 9 最後に、次の例を使用して、WebTier および DBTier コードを強化します。WP-Network-Private コードには追加の変更は必要ありません。

この強化には、データベース サーバへのログイン アクセスと、導入時の cloudConfig 初期化スクリプトが含まれていることに注意してください。

コンポーネント	例
追加の DBTier 入力	<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 </pre>
DBTier リソース	<pre> DBTier: type: Cloud.Machine properties: name: mysql image: ubuntu 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 リソース	<pre> WebTier: type: Cloud.Machine properties: name: wordpress image: ubuntu flavor: '\${input.size}' constraints: - tag: '\${input.env}' networks: - network: '\${resource["WP-Network-Private"].id}' assignPublicIpAddress: true cloudConfig: #cloud-config </pre>

コンポーネント 例

```

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
    - service apache2 reload

```

例：完成した基本的なクラウド テンプレート コードの例

```

formatVersion: 1
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
  resources:
    WebTier:
      type: Cloud.Machine
      properties:
        name: wordpress
        image: ubuntu
        flavor: '${input.size}'
        constraints:
          - tag: '${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 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
        - service apache2 reload
    DBTier:
      type: Cloud.Machine
      properties:

```

```

name: mysql
image: ubuntu
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: []
WP-Network-Private:
  type: Cloud.Network
  properties:
    name: WP-Network-Private
    networkType: existing

```

次のステップ

構文を確認して展開することにより、クラウド テンプレートをテストします。

基本的なクラウド テンプレートのテスト

設計中は、通常、基幹部分から開始し、クラウド テンプレートの拡張に伴って展開およびテストすることにより、テンプレートをビルドします。このサンプルでは、vRealize Automation Cloud Assembly に組み込まれた進行中のテストの一部について説明します。

展開が想定どおりに動作することを確認するために、クラウド テンプレートを数回テストおよび展開することがあります。段階的に、その途中でリソースの追加、再テスト、再展開を行います。

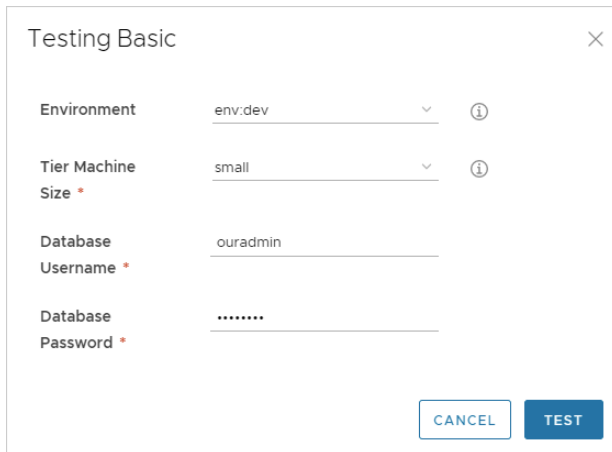
前提条件

基本的なクラウド テンプレートを作成します。[基本的なクラウド テンプレートの作成](#)を参照してください。

手順

- 1 [クラウド テンプレート] をクリックして、WordPress-BP クラウド テンプレートを開きます。
基本的なクラウド テンプレートがデザイン キャンバスとコード エディタに表示されます。
- 2 テンプレートの構文、配置、および基本的な有効性を確認するには、左下の [テスト] をクリックします。

3 入力値を入力して、[テスト] をクリックします。



Testing Basic

Environment: env:dev ⓘ

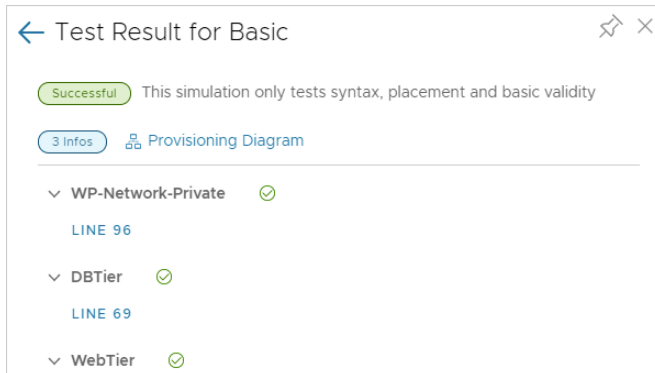
Tier Machine Size: small ⓘ

Database Username: ouradmin

Database Password:

CANCEL TEST

テストはシミュレーションのみで、仮想マシンやその他のリソースを実際に展開するわけではありません。



← Test Result for Basic ☆ ×

Successful This simulation only tests syntax, placement and basic validity

3 Infos Provisioning Diagram

- WP-Network-Private ✓
LINE 96
- DBTier ✓
LINE 69
- WebTier ✓

このテストには [プロビジョニング図] へのリンクが含まれており、シミュレートされた展開フローを調べて、何が起こったかを確認できます。シミュレーションでは、クラウド テンプレートの強い制約に一致するリソース機能が定義されていないなど、潜在的な問題を明らかにします。次のエラーの例では、機能タグ `env:dev` のクラウド ゾーンが定義されたインフラストラクチャにありませんでした。



Request Details EXPORT

NETWORK ALLOCATION → MACHINE ALLOCATION

Request: mysql

Error: No placement exists that satisfies all of the request requirements. See if suitable placements and cloud zones exist for the current project and they have been properly tagged.






Request type	Allocation
Flavor	small
Image	ubuntu
Constraints	env:dev:hard

Annotations: An orange arrow points to the 'env:dev:hard' constraint, and a red arrow points down from the error message to the constraints section.

シミュレーションで正常に実行されても、テンプレートをエラーなしで展開できるとは限りません。

- 4 テンプレートでシミュレーションが成功したら、左下の [展開] をクリックします。
- 5 [新規展開の作成] を選択します。
- 6 展開に **WordPress for OurCo** という名前を付けて、[次へ] をクリックします。
- 7 入力値を入力して、[展開] をクリックします。
- 8 テンプレートが正常に展開されたことを確認するには、[展開] - [展開] で確認します。

展開が失敗した場合は、その名前をクリックして、[履歴] タブをクリックすると、トラブルシューティングに役立つメッセージが表示されます。

Timestamp	Status	Resource type	Resource name
Sep 8, 2020, 1...	CREATE_IN_PROGRESS	 Cloud.Machine	WebTier
Sep 8, 2020, 1...	CREATE_FINISHED	 Cloud.Machine	DBTier
Sep 8, 2020, 1...	CREATE_IN_PROGRESS	 Cloud.Machine	DBTier
Sep 8, 2020, 1...	CREATE_FINISHED	 Cloud.Network	WP-Network-Private
Sep 8, 2020, 1...	CREATE_IN_PROGRESS	 Cloud.Network	WP-Network-Private

一部の履歴エントリには、右端に [プロビジョニング図] リンクがあります。図はシミュレートされたものと似ています。ここでは、プロビジョニング プロセスにおける vRealize Automation Cloud Assembly 決定ポイントのフロー チャートを確認します。

その他のフロー チャートは、[インフラストラクチャ] - [アクティビティ] - [要求] の順に移動して確認できます。

- 9 アプリケーションが動作していることを確認するには、ブラウザで [WordPress の起動] 画面を開きます。

- a WordPress サーバが完全に作成され、初期化されるのを待ちます。

環境によっては、初期化に 30 分以上かかる場合があります。

- b サイトの FQDN または IP アドレスを見つけるには、[展開] - [展開] - [トポロジ] の順に移動します。

- c キャンバスで、WebTier をクリックし、右側のパネルで IP アドレスを見つけます。

- d [WordPress の起動] 画面への完全な URL の一部として IP アドレスを入力します。

このサンプルでは、完全な URL は次のとおりです。

`http://{IP-address}/mywordpresssite`

または

`http://{IP-address}/mywordpresssite/wp-admin/install.php`

- 10 ブラウザで WordPress を確認した後に、アプリケーションの作業がさらに必要になった場合は、テンプレートを変更して、[既存の展開の更新] オプションを使用して再展開します。

- 11 クラウド テンプレートのバージョン管理を行うことを検討してください。変更によって展開が失敗した場合は、動作するバージョンに戻すことができます。
 - a クラウド テンプレート デザイン画面で、[バージョン] をクリックします。
 - b [バージョンの作成] 画面で、**WP-1.0** と入力します。
バージョン名にスペースを含めないでください。
 - c [作成] をクリックします。
バージョンを確認または元に戻すには、デザイン画面で、[バージョン履歴] タブをクリックします。
- 12 現在、基本的な展開が可能であるため、アプリケーション サーバとデータベース サーバの CPU とメモリを増やすことにより、最初の展開時の機能拡張を試行します。
いずれも中規模のノードに更新します。同じテンプレートを使用して、展開時に **medium** を選択し、再展開して、アプリケーションを再検証します。

次のステップ

さらにリソースを追加して、クラウド テンプレートを本番環境に適したアプリケーションに拡張します。

クラウド テンプレートの拡張

サンプル アプリケーション用の基本的な vRealize Automation Cloud Assembly テンプレートを作成してテストした後に、このテンプレートをマルチティア アプリケーションに拡張して、開発環境とテスト環境、最終的には本番環境に展開できるようにします。

クラウド テンプレートを拡張するには、次の機能拡張を追加します。

- アプリケーション サーバをクラスタ化してキャパシティを増やすオプション
- アプリケーション サーバの前にあって一般公開されているネットワークおよびロード バランサ
- アーカイブストレージがあるバックアップ サーバ

前提条件

基本的なクラウド テンプレートを作成してテストします。[基本的なクラウド テンプレートの作成および基本的なクラウド テンプレートのテスト](#)を参照してください。

手順

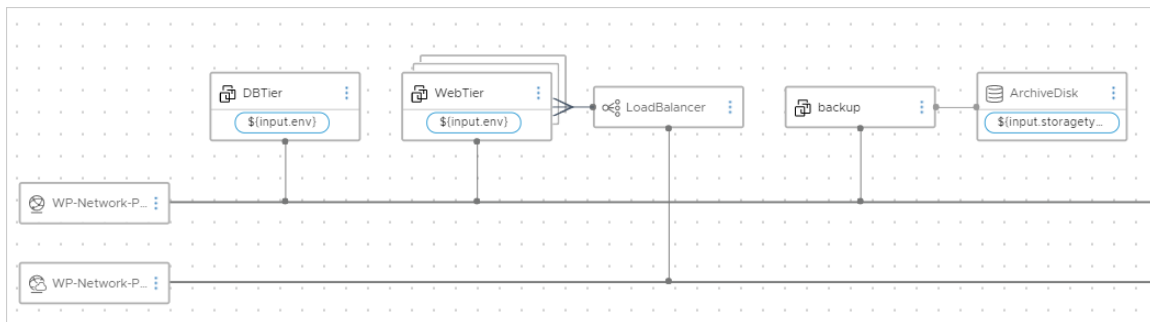
- 1 [クラウド テンプレート] をクリックして、WordPress-BP クラウド テンプレートを開きます。
基本的なテンプレートがデザイン キャンバスとコード エディタに表示されます。
- 2 コード例と図を参考に追加と変更を行います。
GUI を使用して、新しいリソースをロード バランサなどのキャンバスにドラッグし、コード エディタで設定を完了します。
 - a `count` 入力プロンプトを追加して、WordPress アプリケーション サーバをクラスタに含めます。
 - b クラウドに依存しないロード バランサを追加します。
 - c ロード バランサを WordPress アプリケーション サーバ クラスタに接続します。

- d クラウドに依存しないバックアップ マシンを追加します。
- e バックアップ マシンをプライベート/内部ネットワークに接続します。
- f クラウドに依存しないパブリック/外部ネットワークを追加します。
- g ロード バランサをパブリック ネットワークに接続します。
- h クラウドに依存しないストレージ ボリュームをアーカイブ ディスクとして使用できるように追加します。
- i アーカイブ ディスクをバックアップ マシンに接続します。
- j アーカイブ ディスクの速度用に入力プロンプトを追加します。

3 基本的なクラウド テンプレートと同じように展開、テスト、変更を行います。

既存の展開を更新することも、新しいインスタンスを展開して展開を比較することもできます。

その目的は、堅固で繰り返し可能なテンプレートを本番環境への展開に使用できるようにすることです。



例：完成した拡張されたクラウド テンプレート コードの例

```
formatVersion: 1
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
  count:
    type: integer
    default: 2
    maximum: 5
    minimum: 2
    title: WordPress Cluster Size
    description: WordPress Cluster Size (Number of Nodes)
  storagetype:
    type: string
    enum:
      - storage:general
      - storage:fast
    description: Archive Storage Disk Type
    title: Archive Disk Type
resources:
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      image: ubuntu
      flavor: '${input.size}'
      count: '${input.count}'
      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 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
    - service apache2 reload
DBTier:
  type: Cloud.Machine
  properties:
    name: mysql
    image: ubuntu
    flavor: '${input.size}'
    constraints:
      - tag: '${input.env}'
    networks:
      - network: '${resource["WP-Network-Private"].id}'
        assignPublicIpAddress: true
    remoteAccess:
      authentication: usernamePassword
      username: '${input.username}'
      password: '${input.userpassword}'
    cloudConfig: |
      #cloud-config
      repo_update: true
      repo_upgrade: all
      packages:
        - mysql-server
      runcmd:
        - sed -e '/bind-address/ s/^#*#/' -i /etc/mysql/mysql.conf.d/mysqld.cnf
        - service mysql restart
        - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';"
        - mysql -e "FLUSH PRIVILEGES;"
    attachedDisks: []
LoadBalancer:
  type: Cloud.LoadBalancer
  properties:
    name: myapp-lb
    network: '${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
WP-Network-Public:
  type: Cloud.Network
  properties:
    name: WP-Network-Public
    networkType: public
backup:
  type: Cloud.Machine
  properties:
    name: backup
    flavor: '${input.size}'
    image: ubuntu
    networks:
      - network: '${resource["WP-Network-Private"].id}'
    attachedDisks:
      - source: '${resource.ArchiveDisk.id}'
ArchiveDisk:
  type: Cloud.Volume
  properties:
    name: ArchiveDisk
    capacityGb: 5
    constraints:
      - tag: '${input.storageType}'

```

次のステップ

独自のインフラストラクチャを定義し、独自のクラウド テンプレートを作成します。

[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)および [6 章 vRealize Automation Cloud Assembly 展開の設計](#)を参照してください。

チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成

この vRealize Automation Cloud Assembly のチュートリアルでは、VMware Cloud on AWS 環境への展開用にリソース インフラストラクチャとクラウド テンプレートの設定を定義するプロセスを示します。

この手順を実行するには、[VMware Cloud on AWS スタート ガイド](#)の「Software-Defined Data Center の展開と管理」で説明しているように、クラウド管理者が組織の VMware Cloud on AWS SDDC をすでに設定している必要があります。

設定を順番に確認して、VMware Cloud on AWS の環境を設定するプロセスを理解してください。表示される値は、使用事例での例に過ぎないことに注意してください。使用環境独自のクラウド インフラストラクチャや導入のニーズに合わせて、置き換える必要がある値を検討するか、使用事例の値を当てはめます。

類似のワークフローの詳細なビデオが、VMware クラウド管理テクニカル マーケティングにより、[VMware Cloud on AWS を Cloud Assembly 用に設定する方法](#)として公開されています。

手順

1 vRealize Automation での基本的な VMware Cloud on AWS ワークフローの構成

この使用事例では、リソース インフラストラクチャと、VMware Cloud on AWS 環境への展開に対応するクラウド テンプレートを定義するプロセスを示します。

2 vRealize Automation の VMware Cloud on AWS ワークフローで隔離されたネットワークの構成

この手順では、隔離されたネットワークを vRealize Automation の VMware Cloud on AWS 展開に追加します。

vRealize Automation での基本的な VMware Cloud on AWS ワークフローの構成

この使用事例では、リソース インフラストラクチャと、VMware Cloud on AWS 環境への展開に対応するクラウド テンプレートを定義するプロセスを示します。

この手順では、既存の VMware Cloud on AWS 環境内のリソースへのクラウド テンプレートの展開をサポートするインフラストラクチャを構成します。

前提条件

- vRealize Automation Cloud Assembly で VMware Cloud on AWS クラウド アカウントを作成して構成するには、まずユーザーが既存の VMware Cloud on AWS Software-Defined Data Center (SDDC) 環境の組織に所属している必要があります。VMware Cloud on AWS サービスの設定の詳細については、[VMware Cloud on AWS ドキュメント](#)を参照してください。
- vCenter Server の既存の VMware Cloud on AWS ホストの SDDC と vRealize Automation Cloud Assembly の VMware Cloud on AWS クラウド アカウント間で必要な接続の確保を容易にするには、VPN または類似のネットワーク手段を使用して、ネットワーク接続を指定し、ファイアウォール ルールを追加する必要があります。[VMware Cloud on AWS クラウド アカウントに接続するための vRealize Automation での VMware Cloud on AWS SDDC の準備](#)を参照してください。

手順

1 VMware Cloud on AWS クラウド アカウントに接続するための vRealize Automation での VMware Cloud on AWS SDDC の準備

vRealize Automation Cloud Assembly のオンプレミス環境で VMware Cloud on AWS クラウド アカウントを使用する場合は、vCenter Server の SDDC と vRealize Automation のすべての VMware Cloud on AWS クラウド アカウント間の通信をサポートするネットワーク接続を作成する必要があります。

2 サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成

この手順では、vRealize Automation で VMware Cloud on AWS クラウド アカウントを作成します。

3 vRealize Automation での VMware Cloud on AWS 展開用のクラウド ゾーンの設定

この手順では、クラウド ゾーンを作成して、vRealize Automation で VMware Cloud on AWS を使用するとき CloudAdmin ユーザーがアクセスできるコンピューティング リソースを指定します。

4 vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定

この手順では、ネットワーク プロファイルとストレージ プロファイルを設定して、vRealize Automation の VMware Cloud on AWS CloudAdmin ユーザーが使用できるリソースを指定します。

5 vRealize Automation で VMware Cloud on AWS 展開をサポートするためのプロジェクトの作成

この手順では、VMware Cloud on AWS 展開で使用可能なリソースを制御するために使用できる vRealize Automation プロジェクトを定義します。

6 vRealize Automation で VMware Cloud on AWS 展開をサポートするためのクラウド テンプレート デザインでの vCenter マシン リソースの定義

この手順では、デザイン キャンパスに vCenter マシン リソースをドラッグし、vRealize Automation で VMware Cloud on AWS 展開の設定を追加します。

VMware Cloud on AWS クラウド アカウントに接続するための vRealize Automation での VMware Cloud on AWS SDDC の準備

vRealize Automation Cloud Assembly のオンプレミス環境で VMware Cloud on AWS クラウド アカウントを使用する場合は、vCenter Server の SDDC と vRealize Automation のすべての VMware Cloud on AWS クラウド アカウント間の通信をサポートするネットワーク接続を作成する必要があります。

vCenterServer で、既存の VMware Cloud on AWS ホストの SDDC と vRealize Automation の VMware Cloud on AWS クラウド アカウント間で必要な接続の確保を簡素化するには、VPN または類似のネットワーク手段を使用して、2 つの要素間にネットワーク接続を提供する必要があります。

手順

1 パブリック インターネットまたは AWS Direct Connect 経由の VPN 接続を構成します。

VMware Cloud on AWS のドキュメントの [VMware Cloud on AWS のネットワークとセキュリティ](#)を参照してください。

2 vCenter Server の FQDN が管理ネットワーク上のプライベート IP アドレスで解決可能であることを確認します。

[VMware Cloud on AWS のドキュメント](#)の VMware Cloud on AWS のネットワークとセキュリティを参照してください。

3 必要なファイアウォール ルールを構成します。

通信をサポートするには、SDDC の VMware Cloud on AWS コンソールで管理ゲートウェイのファイアウォール ルールを構成する必要があります。ルールは、[管理ゲートウェイ] ファイアウォール ルール セクションに配置する必要があります。SDDC コンソールの [ネットワークとセキュリティ] タブのオプションを使用して、ファイアウォール ルールを作成します。

- HTTPS (TCP 443) サービスの ESXi へのネットワーク トラフィックを、vRealize Automation アプライアンス/サーバの検出された IP アドレスまたは vRealize Automation ロード バランサの仮想 IP アドレスに制限します。
- ICMP (すべての ICMP)、SSO (TCP 7444)、および HTTPS (TCP 443) サービスの vCenter へのネットワーク トラフィックを、vRealize Automation アプライアンス/サーバの検出された IP アドレスまたは vRealize Automation ロード バランサの仮想 IP アドレスに制限します。
- HTTPS (TCP 443) サービスの NSX-T Manager へのネットワーク トラフィックを、vRealize Automation アプライアンス/サーバの検出された IP アドレスまたは vRealize Automation ロード バランサの仮想 IP アドレスに制限します。

次の表に、必要なファイアウォール ルールの概要を示します。

表 2-1. 必要な管理ゲートウェイ ファイアウォール ルールのサマリ

名前	ソース	ターゲット	サービス
vCenter Server	オンプレミス データセンターの CIDR ブロック	vCenter Server	任意 (すべてのトラフィック)
vCenter Server への ping	任意	vCenter Server	ICMP (すべての ICMP)
NSX Manager	オンプレミス データセンターの CIDR ブロック	NSX Manager	任意 (すべてのトラフィック)
オンプレミスから ESXi への ping	オンプレミス データセンターの CIDR ブロック	ESXi の管理のみ	ICMP (すべての ICMP)
オンプレミスから ESXi リモート コンソールおよびプロビジョニング	オンプレミス データセンターの CIDR ブロック	ESXi の管理のみ	TCP 902
オンプレミスから SDDC 仮想マシン	オンプレミス データセンターの CIDR ブロック	SDDC の論理ネットワークの CIDR ブロック	任意 (すべてのトラフィック)
SDDC 仮想マシンからオンプレミス	SDDC の論理ネットワークの CIDR ブロック	オンプレミス データセンターの CIDR ブロック	任意 (すべてのトラフィック)

関連情報については、[VMware Cloud on AWS のドキュメント](#)にある『VMware Cloud on AWS のネットワークとセキュリティ』と『VMware Cloud on AWS Operations Guide』を参照してください。

結果

必要なゲートウェイ アクセスおよびファイアウォール ルールを構成したら、VMware Cloud on AWS クラウド アカウントの作成プロセスを続行できます。

サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成

この手順では、vRealize Automation で VMware Cloud on AWS クラウド アカウントを作成します。

関連情報については、[VMware Cloud on AWS のドキュメント](#)を参照してください。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- この手順では、vCenter 内のターゲット Software-Defined Data Center (SDDC) に対する VMware Cloud on AWS CloudAdmin 認証情報などの必要な管理者認証情報があることと、ポート 443 で HTTPS アクセスを有効にしてあることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- vCenter Server の既存の VMware Cloud on AWS ホストの SDDC と vRealize Automation の VMware Cloud on AWS クラウド アカウント間で必要な接続の確保を簡素化するには、VPN または類似のネットワーク手段を使用して、ネットワーク接続とファイアウォール ルールを提供する必要があります。[VMware Cloud on AWS クラウド アカウントに接続するための vRealize Automation での VMware Cloud on AWS SDDC の準備](#)を参照してください。外部 HTTP インターネット プロキシを使用する場合は、IPv4 用に設定する必要があります。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを構成します。[vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択します。
- 2 [クラウド アカウントの追加] をクリックし、VMware Cloud on AWS を選択して、値を入力します。

次の表に、サンプルの値とサポート情報を示します。

設定	サンプル値と指示	説明
VMC API トークン	<ol style="list-style-type: none"> [VMC API トークン] 行の最後にある i ヘルプ アイコンをクリックし、[ヘルプ] テキストボックスの [API トークン ページ] をクリックして、組織の [マイ アカウント] 画面で [API トークン] タブを開きます。 [トークンを生成] をクリックして、[新しい API トークンの生成] オプションを表示します。 myinitials_mytoken などの新しいトークン名を入力します。 [トークン TTL] を [無期限] に設定します。 <p>有効期限が切れるように設定されたトークンを作成した場合、トークンの有効期限が切れると、vRealize Automation の VMware Cloud on AWS 操作は停止し、新しいトークンを使用してクラウド アカウントを更新するまで停止したままになります。</p> <ol style="list-style-type: none"> [範囲の定義] セクションで、[すべてのロール] を選択します。  <ol style="list-style-type: none"> [生成] をクリックします。 [生成されたトークン] 画面で、[コピー] をクリックしてから [続行] をクリックします。 [新しいクラウド アカウント] 画面に戻り、コピーしたトークンを [VMC API トークン] 行に貼り付け、[API トークンの適用] をクリックします。 	<p>リンクされた [API トークン] 画面で、新しいトークンを作成したり、組織の既存のトークンを使用できます。</p> <p>[範囲の定義] セクションで、API トークンに最低限必要なロールは次のとおりです。</p> <ul style="list-style-type: none"> ■ [組織のロール] <ul style="list-style-type: none"> ■ [組織のメンバー] ■ [組織の所有者] ■ [サービス ロール - VMware Cloud on AWS] <ul style="list-style-type: none"> ■ [管理者] ■ [NSX Cloud 管理者] ■ [NSX Cloud 監査者] <p>注： 生成されたトークンをコピー、ダウンロード、または印刷します。この画面から離れると、生成されたトークンを取得できなくなります。</p> <p>生成されたトークンまたは指定されているトークンを用いて、組織の VMware Cloud on AWS サブスクリプションで使用可能な SDDC 環境に接続し、SDDC 名のリストをポピュレートします。</p> <p>vRealize Automation および VMware Cloud on AWS のサービスが異なる組織にある場合は、VMware Cloud on AWS 組織に切り替えてからトークンを生成する必要があります。</p> <p>API トークンの詳細については、API トークンの生成を参照してください。</p>
SDDC 名	<p>この例では、[Datacenter:Datacenter-abz] を選択します。</p> <p>有効な SDDC 名によって、vCenter および NSX-T の FQDN エントリが自動入力されます。クラウド プロキシがすでに SDDC に展開されている場合は、このクラウド プロキシの値も自動的に入力されます。</p>	<p>VMware Cloud on AWS サブスクリプションで使用可能な SDDC のリストから選択します。SDDC のリストは、VMware Cloud on AWS API トークンに基づいています。</p> <p>NSX-V SDDC は vRealize Automation でサポートされていないため、使用可能な SDDC のリストに表示されません。</p>

設定	サンプル値と指示	説明
vCenter の IP アドレスまたは FQDN	アドレスは、SDDC の選択に基づいて自動入力されます。	指定された SDDC 内の vCenter Server IP アドレスまたは FQDN を入力します。 IP アドレスのデフォルトは、プライベート IP アドレスです。SDDC へのアクセスに使用されているネットワーク接続のタイプによっては、デフォルトのアドレスが、指定した SDDC 内の NSX Manager サーバの IP アドレスと異なる場合があります。
NSX Manager の IP アドレス/FQDN	アドレスは、SDDC の選択に基づいて自動入力されます。	指定された SDDC 内の NSX Manager の IP アドレスまたは FQDN を入力します。 IP アドレスのデフォルトは、プライベート IP アドレスです。SDDC へのアクセスに使用されているネットワーク接続のタイプによっては、デフォルトのアドレスが、指定した SDDC 内の NSX Manager サーバの IP アドレスと異なる場合があります。 VMware Cloud on AWS クラウド アカウントは NSX-T をサポートしています。
vCenter ユーザー名とパスワード	ユーザー名は cloudadmin@vmc.local として自動入力されます。	デフォルトと異なる場合は、指定した SDDC の vCenter Server ユーザー名を入力します。 指定するユーザーには、CloudAdmin 認証情報が必要です。CloudGlobalAdmin 認証情報は必要ありません。 ユーザー パスワードを入力します。
検証	[検証] をクリックします。	検証では、指定された vCenter へのアクセス権と、vCenter が実行されていることを確認します。
名前と説明	クラウド アカウント名に OurCo-VMC を入力します。 クラウド アカウントの説明に、 Sample deployment for VMC を入力します。	
これらのデータセンターへのプロビジョニングを許可	この情報は読み取り専用です。	指定した VMware Cloud on AWS SDDC 環境のデータセンターを一覧表示します。
クラウド ゾーンの作成	チェックボックスを選択解除します。この例では、ワークフローの後半でクラウド ゾーンを作成します。	vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報を参照してください。
機能タグ	空のままにします。このワークフローでは、機能タグは使用しません。	組織のタグ ストラテジで説明するようにタグを使用します。 vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成を参照してください。

3 [追加] をクリックします。

結果

マシンやボリュームなどのリソースは、VMware Cloud on AWS SDDC データセンターから収集され、vRealize Automation[インフラストラクチャ] タブの [リソース] セクションに一覧表示されます。

次のステップ

[vRealize Automation での VMware Cloud on AWS 展開用のクラウド ゾーンの作成。](#)

vRealize Automation での VMware Cloud on AWS 展開用のクラウド ゾーンの作成

この手順では、クラウド ゾーンを作成して、vRealize Automation で VMware Cloud on AWS を使用するとき、CloudAdmin ユーザーがアクセスできるコンピューティング リソースを指定します。

VMware Cloud on AWS には、CloudGlobalAdmin と CloudAdmin という 2 つの主要な管理者認証情報があります。vRealize Automation Cloud Assembly は、CloudAdmin ユーザーをサポートするように設計されています。VMware Cloud on AWS CloudAdmin ユーザーが使用できるリソースに展開します。VMware Cloud on AWS CloudGlobalAdmin の認証情報を必要とするリソースには展開しないでください。

プロジェクト クラウド テンプレートでマシン、ネットワーク、およびストレージを展開するときに、その展開先となるコンピューティング リソースがクラウド ゾーンによって識別されます。[vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報を参照してください。](#)

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- [サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成の手順を完了します。](#)
- この手順では、vCenter Server のターゲット Software-Defined Data Center (SDDC) の VMware Cloud on AWS CloudAdmin 認証情報など、必要な管理者認証情報があることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。](#)
- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについてを参照してください。](#)

手順

- 1 [インフラストラクチャ] - [構成] - [クラウド ゾーン] を選択します。
- 2 [新しいクラウド ゾーン] をクリックし、VMware Cloud on AWS 環境の値を入力します。

設定	サンプルの値
アカウント/リージョン	OurCo-VMC / Datacenter:Datacenter-abz これは、前の手順の サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成 で定義したクラウド アカウントおよびそれに関連付けられたリージョンです。
名前	VMC_cloud_zone-1
説明	VMware Cloud on AWS リソースのみ
配置ポリシー	デフォルト
機能タグ	空のままにします。このワークフローでは、機能タグは使用しません。

3 [コンピューティング] タブをクリックします。

4 下のエリア 1 に示すように、CloudAdmin ユーザーが使用できるコンピューティング リソースを検索して選択します。この例では、Cluster 1/ Compute-ResourcePool という名前のリソースを使用します。

Cluster 1/ Compute-ResourcePool は VMware Cloud on AWS のデフォルトのコンピューティング リソースです。

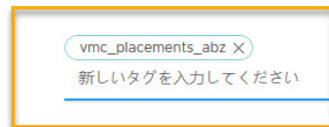


5 上記のエリア 2 に示すように、タグ名 vmc_placements_abz を追加します。

タグ

1個のオブジェクトが選択されました

タグの追加



タグの削除

タグがありません ⓘ

6 [フィルタ タグ] セクションに vmc_placements_abz と入力して、このクラウド ゾーンで使用するコンピューティング リソースをフィルタします。

7 [保存] をクリックします。

名前	アカウント/リージョン	タイプ	タグ
<input type="checkbox"/> ComputeClusterA	LK-TEST 顧客表示もA中CE6監修02506a274 / NSX62-Scale-DC	common title cluster	Cluster ComputeClusterA
<input checked="" type="checkbox"/> ComputeClusterA-New	nsx-v顧客表示もA中CE6監修02506a274 / NSX621-DataCenter	common title cluster	ComputeClusterA
<input type="checkbox"/> ComputeClusterA / Scale	270_VC_account顧客表示もA中CE6監修02506a274 / NSX62-Scale-DC	ResourcePool	ComputeClusterA

この例では、CloudAdmin ユーザーは Cluster 1/ Compute-ResourcePool という名前のコンピューティング リソースのみを使用できます。

次のステップ

vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定。

vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定

この手順では、ネットワーク プロファイルとストレージ プロファイルを設定して、vRealize Automation の VMware Cloud on AWS CloudAdmin ユーザーが使用できるリソースを指定します。

イメージとフレーバーの値も必要ですが、これらに VMware Cloud on AWS ユーザー認証情報特有の点はありません。この例では、クラウド テンプレートを定義するときに、フレーバーの値として `small`、イメージの値として `ubuntu-16` を使用します。

マッピングとプロファイルに関する一般的な情報については、[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- クラウド ゾーンを作成します。[vRealize Automation での VMware Cloud on AWS 展開用のクラウド ゾーンの作成](#)を参照してください。
- この手順では、vCenter Server のターゲット Software-Defined Data Center (SDDC) の VMware Cloud on AWS CloudAdmin 認証情報など、必要な管理者認証情報があることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。

手順

1 VMware Cloud on AWS 展開のネットワーク プロファイルを定義します。


- a [インフラストラクチャ] - [設定] - [ネットワーク プロファイル] の順に選択し、[新規ネットワーク プロファイル] をクリックします。

設定	サンプル値
アカウント/リージョン	OurCo-VMC / Datacenter:Datacenter-abz
名前	vmc-network1
説明	VMware Cloud on AWS CloudAdmin 認証情報を持つクラウド テンプレート管理者がアクセスできるネットワークが含まれます。

注： サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成で作成した VMware Cloud on AWS クラウド アカウントおよびそれに対応する Software-Defined Data Center (SDDC) データセンターを選択します。

- b [ネットワーク] タブをクリックし、[ネットワークの追加] をクリックします。
- c CloudAdmin 認証情報を持つ VMware Cloud on AWS ユーザーが展開先として使用できるネットワークを選択します (例: sddc-cgw-network-1)。

ネットワークの追加



<input type="checkbox"/>	名前	アカウント/リージョン	ゾーン	ネットワーク ドメイン
<input checked="" type="checkbox"/>	ESO_PKS_VC01_VM_PKS	1114VCあア7中表 諸合置ホ CEU8Aaæðe0à U*Use / ESO_PKS_VC01 _DC01		ESO_PKS_VC01_DVS01
<input type="checkbox"/>	ESO_PKS_VC01_Mgmt	1114VCあア7中表 諸合置ホ		ESO_PKS_VC01_DVS01

2 ネットワーク プロファイルを保存します。

3 VMware Cloud on AWS 展開のストレージ プロファイルを定義します。

CloudAdmin ユーザーがアクセスできるデータストアまたはクラスタをターゲットとするストレージ プロファイルを構成します。

- a [インフラストラクチャ] - [設定] - [ストレージ プロファイル] の順に選択し、新しい [新規ストレージ プロファイル] をクリックします。

設定	サンプルの値
アカウント/リージョン	OurCo-VMC / Datacenter:Datacenter-abz サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成で作成した VMware Cloud on AWS クラウド アカウントおよびそれに対応する Software-Defined Data Center (SDDC) データセンターを選択します。
名前	vmc-storage1
説明	VMware Cloud on AWS CloudAdmin 認証情報を持つクラウド テンプレート 管理者が展開先として使用できるデータストア クラスタが含まれます。

- b [データストア/クラスタ] ドロップダウン メニューから、[WorkloadDatastore] データストアを選択します。



vRealize Automation Cloud Assembly の VMware Cloud on AWS の場合、ストレージ ポリシーでは VMware Cloud on AWS 展開をサポートするために [WorkloadDatastore] データストアを使用する必要があります。

4 ストレージ プロファイルを保存します。

次のステップ

vRealize Automation で VMware Cloud on AWS 展開をサポートするためのプロジェクトの作成。

vRealize Automation で VMware Cloud on AWS 展開をサポートするためのプロジェクトの作成

この手順では、VMware Cloud on AWS 展開で使用可能なリソースを制御するために使用できる vRealize Automation プロジェクトを定義します。

プロジェクトについては、[展開時の vRealize Automation Cloud Assembly プロジェクトの動作](#)を参照してください。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定の手順を完了します。

- この手順では、vCenter Server のターゲット Software-Defined Data Center (SDDC) の VMware Cloud on AWS CloudAdmin 認証情報など、必要な管理者認証情報があることを前提としています。
[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。](#)
- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] の順に選択します。
- 2 [新規プロジェクト] をクリックし、プロジェクト名として VMC_proj-1_abz と入力します。
- 3 [ユーザー] をクリックし、[ユーザーの追加] をクリックします。
ユーザーには、組織の VMware Cloud on AWS サブスクリプションに対する CloudAdmin 認証情報が必要です。
 - chris.gray@ourco.com, Administrator
 - kerry.white@ourco.com、メンバー
- 4 [プロビジョニング] をクリックし、[クラウド ゾーンの追加] をクリックします。
- 5 前の手順で設定したクラウド ゾーンを追加します。

設定	サンプルの値
クラウド ゾーン	VMC_cloud_zone-1 このクラウド ゾーンは、前の手順 vRealize Automation での VMware Cloud on AWS 展開用のクラウド ゾーンの作成 で作成しました。
プロビジョニングの優先順位	1
インスタンスの制限	3

- 6 この例では、これら以外のオプションは無視してください。

次のステップ

VMware Cloud on AWS 環境に展開するクラウド テンプレートを作成します。[vRealize Automation で VMware Cloud on AWS 展開をサポートするためのクラウド テンプレート デザインでの vCenter マシン リソースの定義](#)を参照してください。

vRealize Automation で VMware Cloud on AWS 展開をサポートするためのクラウド テンプレート デザインでの vCenter マシン リソースの定義

この手順では、デザイン キャンバスに vCenter マシン リソースをドラッグし、vRealize Automation で VMware Cloud on AWS 展開の設定を追加します。

使用可能な VMware Cloud on AWS リソースに展開できるクラウド テンプレート デザインを作成します。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- この手順は、クラウド テンプレート デザイナの認証情報があることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- この手順では、vCenter 内のターゲット SDDC (Datacenter:Datacenter-abz) に対する VMware Cloud on AWS CloudAdmin 認証情報があることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- 前のセクションの説明に基づき、リソース インフラストラクチャとプロジェクトを構成します。

手順

- 1 [デザイン] タブをクリックして、[新規] をクリックします。

設定	サンプルの値
名前	vmc-bp_abz
説明	1
プロジェクト	VMC_proj-1_abz これは以前に作成したプロジェクトで、以前に作成したクラウド ゾーンをサポートします。現在、このプロジェクトはクラウド ゾーンに関連付けられ、クラウド ゾーンは以前に作成した VMware Cloud on AWS クラウド アカウント/リージョンに関連付けられています。

- 2 vSphere マシン リソースをキャンバス上にスライドします。
- 3 マシン リソースで、次の（太字の）クラウド テンプレート リソース コードを編集します。

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

image には、展開のニーズに適した任意の値を指定できます。

VMware Cloud on AWS 展開をサポートするには、クラウド テンプレート デザイン コードに `folderName: Workloads` ステートメントを追加する必要があります。`folderName: Workloads` 設定は、VMware Cloud on AWS SDDC 環境の CloudAdmin 認証情報をサポートしており、必須です。

注：上記のコード サンプルに表示されている `folderName: Workloads` の設定は必須ですが、上記のようにクラウド テンプレート コードに直接追加することも、関連付けられているクラウド ゾーンまたはプロジェクトに追加することもできます。この設定が、これら 3 つの中の 2 つ以上の場所で指定されている場合、優先順位は次のようになります。

- プロジェクトの設定は、クラウド テンプレートの設定およびクラウド ゾーンの設定をオーバーライドします。
- クラウド テンプレート デザインの設定は、クラウド ゾーンの設定をオーバーライドします。

注：次に示すように、必要に応じて `cpuCount` および `totalMemoryMB` の設定を `flavor` (サイズ変更) エントリに置き換えることができます。

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

クラウド ゾーンのフォルダ値がワークロードに設定されている場合、クラウド ゾーン フォルダの値をオーバーライドする場合を除き、クラウド テンプレートで `folderName` プロパティを設定する必要はありません。

次のステップ

ネットワークの隔離を追加することにより、この基本的な VMware Cloud on AWS ワークフローを拡張します。[vRealize Automation の VMware Cloud on AWS ワークフローで隔離されたネットワークの構成](#)を参照してください。

vRealize Automation の VMware Cloud on AWS ワークフローで隔離されたネットワークの構成

この手順では、隔離されたネットワークを vRealize Automation の VMware Cloud on AWS 展開に追加します。

VMware Cloud on AWS クラウド アカウントを定義すると、VMware Cloud on AWS サービスで設定されている NSX-T 設定が使用可能になります。VMware Cloud on AWS サービスの NSX-T 設定の詳細については、VMware Cloud on AWS の[製品ドキュメント](#)を参照してください。

vRealize Automation では、VMware Cloud on AWS と NSX-T の組み合わせはサポートされます。VMware Cloud on AWS と NSX-V の組み合わせはサポートされません。

vRealize Automation では、VMware Cloud on AWS 展開のネットワーク隔離はサポートされます。VMware Cloud on AWS の他のネットワーク手段はサポートされません。

基本的な VMware Cloud on AWS ワークフローのこの拡張版では、クラウド テンプレートで使用する隔離されたネットワークを作成する以下の方法について説明します。

- オンデマンド ネットワークベースの隔離を構成します。

- オンデマンド セキュリティ グループベースの隔離を構成します。

前提条件

この手順は、基本的な VMware Cloud on AWS ワークフローに対する拡張です。これは、[チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成](#)のワークフローで設定したのと同じクラウド アカウント、リージョン、クラウド ゾーン、プロジェクト、ネットワーク プロファイルを使用します。

手順

1 vRealize Automation での VMware Cloud on AWS 展開向けの隔離されたネットワークの定義

次のいずれかの手順を使用して、VMware Cloud on AWS 展開向けのネットワーク隔離を構成できます。

2 vRealize Automation で VMware Cloud on AWS のネットワークの隔離をサポートするためのクラウド テンプレートでのネットワーク コンポーネントの定義

この手順では、ネットワーク マシン コンポーネントを vRealize Automation のクラウド テンプレート キャンバスにドラッグし、隔離されたネットワーク環境の設定をターゲットの VMware Cloud on AWS 環境に追加します。

vRealize Automation での VMware Cloud on AWS 展開向けの隔離されたネットワークの定義

次のいずれかの手順を使用して、VMware Cloud on AWS 展開向けのネットワーク隔離を構成できます。

- [vRealize Automation でのオンデマンド ネットワークベースの隔離の設定](#)
- [vRealize Automation でのオンデマンド セキュリティ グループベースの隔離の設定](#)

vRealize Automation でのオンデマンド ネットワークベースの隔離の設定

ネットワーク プロファイルでオンデマンド ネットワーク設定を指定して使用することにより、VMware Cloud on AWS 展開のニーズに合わせてネットワークの隔離を設定できます。

隔離されたネットワークを指定するには、セキュリティ グループを使用するか、オンデマンド ネットワーク設定を使用できます。この例では、ネットワーク プロファイルでオンデマンド ネットワーク設定を指定して、ネットワークの隔離を構成します。その後、クラウド テンプレートでネットワークにアクセスし、そのクラウド テンプレートを VMware Cloud on AWS 展開で使用します。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- [vRealize Automation での基本的な VMware Cloud on AWS ワークフローの構成](#)のワークフローを完了します。
- [vRealize Automation の VMware Cloud on AWS ワークフローで隔離されたネットワークの構成](#)を確認します。
- この手順では、vCenter Server のターゲット Software-Defined Data Center (SDDC) の VMware Cloud on AWS CloudAdmin 認証情報など、必要な管理者認証情報があることを前提としています。
[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。](#)

- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。

手順

- 1 基本的な VMware Cloud on AWS ワークフローで使用するネットワーク プロファイル (vmc-network1 など) を開きます。[vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定](#)を参照してください。
- 2 [ネットワーク] タブでは、何も選択を行う必要はありません。
- 3 [ネットワーク ポリシー] タブをクリックします。
- 4 [オンデマンド ネットワークを作成する] オプションを選択し、デフォルトの cgw ネットワーク ドメインを選択します。適切な CIDR とサブネット サイズを指定します。
- 5 [保存] をクリックします。

このネットワーク プロファイルを使用すると、マシンはデフォルトのネットワーク ドメイン内のネットワーク に展開されます。ネットワークは、プライベート ネットワーク アクセスまたは送信ネットワーク アクセスを使用して、他のネットワークから隔離されます。

次のステップ

クラウド テンプレートでネットワーク コンポーネントを構成します。[vRealize Automation で VMware Cloud on AWS のネットワークの隔離をサポートするためのクラウド テンプレートでのネットワーク コンポーネントの定義](#)を参照してください。

vRealize Automation でのオンデマンド セキュリティ グループベースの隔離の設定

ネットワーク プロファイルでオンデマンド セキュリティ グループを指定して使用することにより、VMware Cloud on AWS 展開のニーズに合わせてネットワークの隔離を設定できます。

隔離されたネットワークを指定するには、セキュリティ グループを使用するか、オンデマンド ネットワーク設定を使用できます。この例では、ネットワーク プロファイルでオンデマンド セキュリティ グループを指定することによってネットワークの隔離を構成します。その後、クラウド テンプレートでネットワークを指定し、そのクラウド テンプレートを VMware Cloud on AWS 展開で使用します。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- [vRealize Automation での基本的な VMware Cloud on AWS ワークフローの構成](#)のワークフローを完了します。
- [vRealize Automation の VMware Cloud on AWS ワークフローで隔離されたネットワークの構成](#)を確認します。
- この手順では、vCenter Server のターゲット Software-Defined Data Center (SDDC) の VMware Cloud on AWS CloudAdmin 認証情報など、必要な管理者認証情報があることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- この手順では、クラウド管理者ユーザー ロールが割り当てられていることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。

手順

- 1 基本的な VMware Cloud on AWS ワークフローで使用するネットワーク プロファイル (vmc-network1 など) を開きます。vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定を参照してください。
- 2 基本的な VMware Cloud on AWS ワークフローでを使用した既存のネットワーク (sddc-cgw-network-1 など) を選択します。vRealize Automation での VMware Cloud on AWS 展開のネットワークおよびストレージのプロファイル設定を参照してください。
- 3 [ネットワーク ポリシー] タブをクリックします。
- 4 [オンデマンド セキュリティ グループを作成する] オプションを選択します。



- 5 [保存] をクリックします。

このネットワーク プロファイルを使用すると、マシンは選択したネットワークに展開され、新しいセキュリティ グループ ポリシーによって隔離されます。新しいセキュリティ ポリシーでは、プライベート ネットワーク アクセスまたは送信ネットワーク アクセスが許可されます。

次のステップ

クラウド テンプレートでネットワーク コンポーネントを構成します。vRealize Automation で VMware Cloud on AWS のネットワークの隔離をサポートするためのクラウド テンプレートでのネットワーク コンポーネントの定義を参照してください。

vRealize Automation で VMware Cloud on AWS のネットワークの隔離をサポートするためのクラウド テンプレートでのネットワーク コンポーネントの定義

この手順では、ネットワーク マシン コンポーネントを vRealize Automation のクラウド テンプレート キャンバスにドラッグし、隔離されたネットワーク環境の設定をターゲットの VMware Cloud on AWS 環境に追加します。

以前に作成したクラウド テンプレートにネットワーク隔離を追加します。クラウド テンプレートは、VMware Cloud on AWS 環境への展開をサポートするプロジェクトとクラウド ゾーンのほか、隔離のために設定したネットワーク プロファイルとネットワークにもすでに関連付けられています。

特に指定のない限り、この手順で入力する値は、この例のワークフローでのみ使用します。

前提条件

- [vRealize Automation](#) でのオンデマンド セキュリティ グループベースの隔離の設定または [vRealize Automation](#) でのオンデマンド ネットワークベースの隔離の設定の手順を完了します。
- この手順は、クラウド テンプレート デザイナーの認証情報があることを前提としています。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- この手順では、vCenter 内のターゲット SDDC に対する VMware Cloud on AWS CloudAdmin 認証情報があることを前提としています。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。

手順

- 1 以前のワークフローで作成したクラウド テンプレートを開きます。[vRealize Automation で VMware Cloud on AWS 展開をサポートするためのクラウド テンプレート デザインでの vCenter マシン リソースの定義](#)を参照してください。
- 2 クラウド テンプレート デザイン画面の左側にあるコンポーネントからキャンバスにネットワーク コンポーネントをドラッグします。
- 3 ネットワーク コンポーネントの YAML コードを編集して、太字で示されているように、`private` と `outbound` のどちらかのネットワーク タイプを指定します。

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

または

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

次のステップ

クラウド テンプレートを展開する、または閉じる準備ができました。

チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成

外部 IP アドレス管理プロバイダを使用して、クラウド テンプレート展開の IP アドレス割り当てを管理できます。このチュートリアルでは、外部 IP アドレス管理プロバイダとして Infoblox を使用して vRealize Automation で外部 IP アドレス管理統合を構成する方法について説明します。

この手順では、既存の IP アドレス管理プロバイダ パッケージ（この場合は Infoblox パッケージ）と既存の実行環境を使用して、プロバイダ固有の IP アドレス管理統合ポイントをビルドします。外部 IP アドレス管理プロバイダからの IP アドレス割り当てをサポートするように、既存のネットワークを構成し、ネットワーク プロファイルを作成することができます。最後に、ネットワークおよびネットワーク プロファイルに一致するクラウド テンプレートを作成し、外部 IP アドレス管理プロバイダから取得した IP アドレス値を使用してネットワーク マシンを展開します。

IP アドレス管理プロバイダ パッケージを取得して設定する方法と、クラウド拡張プロキシにアクセスする実行環境が IP アドレス管理プロバイダとの統合をサポートするように設定する方法については、リファレンスに記載されています。

表示される値は、値の例であることに注意してください。使用環境で、使用事例の一つ一つをそのまま使用することはできません。組織のニーズに合わせるために、どの部分を独自の値に置き換えるか、また、例の値に基づいて値を決定するかを検討します。



ビデオ形式の Infoblox IP アドレス管理統合ワークフローを示す同様の vRealize Automation シナリオを参照するには、[Infoblox IP アドレス管理プラグインと vRealize Automation 8.1/vRealize Automation Cloud との 1 対 1 の統合](#)を参照してください。

手順

1 vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加

Infoblox プロバイダ パッケージ (infoblox .zip) を Infoblox Web サイトまたは VMware Marketplace のいずれかからダウンロードおよび展開して vRealize Automation との統合を実行するには、Infoblox の必要な拡張属性を追加する必要があります。

2 vRealize Automation で使用するための外部 IP アドレス管理プロバイダ パッケージのダウンロードと展開

vRealize Automation で外部 IP アドレス管理統合ポイントを定義するには、構成済みの IP アドレス管理プロバイダ パッケージが必要です。

3 vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成

vRealize Automation に外部 IP アドレス管理統合ポイントを定義する前に、IP アドレス管理プロバイダと vRealize Automation の仲介として機能する実行環境を作成するか、既存の実行環境にアクセスする必要があります。実行環境は、通常、Amazon Web Services または Microsoft Azure クラウド アカウントか、クラウド拡張性プロキシに関連付けられたオンプレミスのアクションベース拡張統合ポイントです。

4 vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加

vRealize Automation では、外部 IP アドレス管理プロバイダとの統合がサポートされています。この例では、外部 IP アドレス管理プロバイダとして Infoblox を使用します。

5 vRealize Automation で既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定

IP アドレス値を vRealize Automation から内部的に取得するのではなく、外部の IP アドレス管理プロバイダから取得し、それによって管理されるように、既存のネットワークを定義できます。

6 vRealize Automation での外部 IP アドレス管理プロバイダ範囲の割り当てを使用するクラウド テンプレートの定義と展開

クラウド テンプレートを定義して、外部 IP アドレス管理プロバイダから IP アドレス割り当てを取得および管理することができます。この例では、外部 IP アドレス管理プロバイダとして Infoblox を使用します。

7 vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用

Infoblox 向けに外部 IP アドレス管理統合を含む vRealize Automation プロジェクトには、Infoblox 固有のプロパティを使用できます。

vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加

Infoblox プロバイダ パッケージ (infoblox .zip) を Infoblox Web サイトまたは VMware Marketplace のいずれかからダウンロードおよび展開して vRealize Automation との統合を実行するには、Infoblox の必要な拡張属性を追加する必要があります。

この手順は、vRealize Automation Cloud Assembly との Infoblox 統合用に外部 IP アドレス管理統合ポイントを作成している場合に適用可能です。

infoblox.zip のダウンロードを使用するには、組織アカウント管理者の認証情報を使用して Infoblox アカウントにログインし、次の Infoblox 拡張属性を事前に作成する必要があります。

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

前提条件

- Infoblox のアカウントがあり、組織の Infoblox アカウントへの適切なアクセス認証情報を持っていることを確認します。
- Infoblox WAPI のバージョンがサポートされていることを確認します。Infoblox との IP アドレス管理の統合は、Infoblox WAPI バージョン v2.7 に依存します。WAPI v2.7 をサポートするすべての Infoblox アプリケーションがサポートされます。
- vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用を確認します。

手順

- 1 管理者の認証情報を使用して、Infoblox アカウントにログインします。

これらは、[インフラストラクチャ] - [接続] - [統合] - [] のメニュー シーケンスを使用して vRealize Automation Cloud Assembly で外部 IP アドレス管理統合ポイントを作成した際に指定した管理者ユーザー名とパスワード認証情報と同じものです。

2 Infoblox のドキュメントに記載されている手順を使用して、Infoblox アプリケーションに次の必要な拡張属性を作成します。

- VMware NIC index - 整数型
- VMware resource ID - 文字列型
- Tenant ID - 文字列型
- CMP Type - 文字列型
- VM ID - 文字列型
- VM Name - 文字列型

この手順は、Infoblox ドキュメントのトピック [About Extensible Attributes](#) の「Adding Extensible Attributes」セクションで説明されています。また、[Managing Extensible Attributes](#) も参照してください。

次のステップ

必要な属性を追加した後、[vRealize Automation](#) で使用するための外部 IP アドレス管理プロバイダパッケージのダウンロードと展開で説明されている手順に基づいて Infoblox パッケージのダウンロードと展開のプロセスを再開できます。

vRealize Automation で使用するための外部 IP アドレス管理プロバイダパッケージのダウンロードと展開

vRealize Automation で外部 IP アドレス管理統合ポイントを定義するには、構成済みの IP アドレス管理プロバイダパッケージが必要です。

プロバイダ固有の統合パッケージは、IP アドレス管理プロバイダの Web サイト、[VMware Solution Exchange](#) のマーケットプレイス、または利用できる場合には vRealize Automation の [Marketplace] タブからダウンロードできます。

注： この例では、VMware が提供している Infoblox パッケージ `Infoblox.zip` を使用します。このパッケージは、次の手順に沿って [VMware Marketplace](#) からダウンロードできます。

- [vRA Cloud Infoblox Plug-in バージョン 1.2](#) - vRealize Automation 8.1.x および 8.2.x と互換性があります。
- [vRA Cloud Infoblox Plug-in バージョン 1.1](#) - vRealize Automation 8.1.x と互換性があります。
- [vRA Cloud Infoblox Plug-in バージョン 1.0](#) - グローバル ネットワークにインターネット接続されているかどうかに関係なく、vRealize Automation 8.0.1 と互換性があります。
- [vRA Cloud Infoblox Plug-in バージョン 0.4](#) - グローバル ネットワークにインターネット接続されている場合は、vRealize Automation 8.0.0.x および 8.0.1.x と互換性があります。

Infoblox との IP アドレス管理の統合は、Infoblox WAPI バージョン v2.7 に依存します。WAPI v2.7 をサポートするすべての Infoblox アプライアンスがサポートされます。

他の IP アドレス管理プロバイダ向けに IP アドレス管理統合パッケージを作成する方法については、マーケットプレイ스에 既存のものが ない場合は、[IP アドレス管理 SDK](#) を使用して vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合パッケージを作成する方法を参照してください。

IP アドレス管理プロバイダ パッケージには、メタデータなどの構成ファイルとともにパッケージ化されたスクリプトが含まれています。スクリプトには、vRealize Automation が外部 IP アドレス管理プロバイダと連携して実行する操作で使用するソース コードが含まれています。そのような操作には、Allocate an IP address for a virtual machine、Fetch a list of IP ranges from the provider、Update the MAC address of a host record in the provider などがあります。

前提条件

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- Infoblox や Bluecat などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。
- 外部 IP アドレス管理プロバイダとして Infoblox を使用している場合は、続行する前に、必要な拡張可能属性が Infoblox アカウントに追加されていることを確認します。[vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加](#)を参照してください。

注: Infoblox プラグインの Python 要素による SSL ハンドシェイクの処理方法を原因とする証明書チェーンの問題が発生しています。問題点と必要なアクションの詳細については、ナレッジベースの記事 [vRA Cloud Infoblox Plugin throws a certificate chain error during authentication process \(KB88057\)](#) を参照してください。

手順

- 1 [VMware Marketplace](#) の [vRA Cloud Infoblox Plug-in バージョン 1.1](#) パッケージの画面に移動します。
- 2 ログインして、プラグイン パッケージをダウンロードします。
- 3 必要な拡張属性を Infoblox に追加していない場合は、追加します。[vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加](#)を参照してください。

結果

このパッケージを使用して展開できるようになりました。[vRealize Automation](#) での [Infoblox 用外部 IP アドレス管理統合の追加](#) で説明されているように、[統合] - [統合の追加] - [IP アドレス管理] - [プロバイダの管理] - [パッケージのインポート] の順に選択します。

vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成

vRealize Automation に外部 IP アドレス管理統合ポイントを定義する前に、IP アドレス管理プロバイダと vRealize Automation の仲介として機能する実行環境を作成するか、既存の実行環境にアクセスする必要があります。実行環境は、通常、Amazon Web Services または Microsoft Azure クラウド アカウントか、クラウド拡張性プロキシに関連付けられたオンプレミスのアクションベース拡張統合ポイントです。

外部 IP アドレス管理の統合には、実行環境が必要です。IP アドレス管理統合ポイントを定義する場合は、使用可能な実行環境を指定して、vRealize Automation Cloud Assembly と IP アドレス管理プロバイダの間の接続を作成します。

IP アドレス管理の統合では、ダウンロードされたプロバイダ固有のスクリプトまたはプラグインのセットを実行環境で使用します。これにより、環境は Amazon Web Services Lambda、Microsoft Azure 機能などの FaaS (Feature-as-a-Services) プロバイダ、またはアクションベースの拡張性 (ABX) のオンプレミスの組み込み統合ポイントによって簡素化されます。実行環境を使用して、外部 IP アドレス管理プロバイダ (Infoblox など) に接続します。

注： Infoblox IP アドレス管理統合ポイントには、アクションベースの拡張性 (ABX) のオンプレミスの組み込み統合ポイントが必要です。

各タイプのランタイム環境には、次のような長所と短所があります。

- アクションベースの拡張性 (ABX) の統合ポイント
 - 無償、追加のベンダー使用コストなし
 - パブリックにアクセスできない NAT/ファイアウォールの背後にあるオンプレミス データセンター上の IP アドレス管理ベンダー アプライアンス (例：Infoblox) に接続できる
 - 市販のクラウド ベンダーよりも低速で、パフォーマンスの信頼性がやや低い
- Amazon Web Services
 - ベンダー FaaS 接続/使用量コストが関連付けられている
 - パブリックにアクセスできない NAT/ファイアウォールの背後にあるオンプレミス データセンター上の IP アドレス管理ベンダー アプライアンスに接続できない
 - 高速で信頼性の高いパフォーマンスを持つ
- Microsoft Azure
 - ベンダー FaaS 接続/使用量コストが関連付けられている
 - パブリックにアクセスできない NAT/ファイアウォールの背後にあるオンプレミス データセンター上の IP アドレス管理ベンダー アプライアンスに接続できない
 - 高速で信頼性の高いパフォーマンスを持つ

前提条件

- クラウド管理者権限が付与されていることを確認します。vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。vRealize Automation のユーザー ロールについてを参照してください。
- Infoblox や Bluecat などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。

- Infoblox や BlueCat などの IP アドレス管理プロバイダの導入済みの統合パッケージにアクセスできることを確認します。展開したパッケージは、IP アドレス管理プロバイダの Web サイトまたは vRealize Automation Cloud Assembly の Marketplace から zip 形式のダウンロード ファイルとして取得し、vRealize Automation Cloud Assembly に展開されます。

プロバイダパッケージの .zip ファイルを展開し、IP アドレス管理の統合ページで [プロバイダ] の値として使用できるようにする方法については、[vRealize Automation で使用するための外部 IP アドレス管理プロバイダパッケージのダウンロードと展開](#)を参照してください。

手順

- 1 オンプレミスの FaaS ベースの拡張性アクションを作成して、IP アドレス管理統合の実行環境を使用するには、[拡張性] - [ライブラリ] - [アクション] を選択します。
- 2 [新しいアクション] をクリックし、アクション名と説明を入力して、プロジェクトを指定します。
- 3 [FaaS プロバイダ] ドロップダウン メニューで、[オンプレミス] を選択します。
- 4 機能性アクションを定義するには、フォームに入力します。



実行中の環境の関連情報については、ブログ内のこのビデオ [Infoblox IPAM Plug-in 1.1 Integration](#) の 24 分付近をご覧ください。

vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加

vRealize Automation では、外部 IP アドレス管理プロバイダとの統合がサポートされています。この例では、外部 IP アドレス管理プロバイダとして Infoblox を使用します。

プロバイダ固有の IP アドレス管理統合ポイントを使用して、クラウド テンプレート展開の IP アドレスおよび関連するネットワーク特性を取得および管理できます。

この例では、外部 IP アドレス管理プロバイダを使用して組織のアカウントへのアクセスをサポートする外部 IP アドレス管理統合ポイントを作成します。このワークフローの例では、IP アドレス管理プロバイダは Infoblox で、プロバイダ固有の統合パッケージはすでに存在しています。これらの手順は Infoblox との連携に固有のものです。別の外部 IP アドレス管理プロバイダ向けに IP アドレス管理統合を作成する際にはリファレンスとして使用できます。

プロバイダ固有の統合パッケージは、IP アドレス管理プロバイダの Web サイト、[VMware Solution Exchange のマーケットプレイス](#)、または利用できる場合には vRealize Automation Cloud Assembly の [Marketplace] タブから入手することができます。

この例では、VMware が提供している Infoblox パッケージ `Infoblox.zip` を使用します。このパッケージは、VMware Solution Exchange のマーケットプレイスから次のバージョンをダウンロードできます。

- [vRA Cloud Infoblox Plug-in バージョン 1.1](#) - vRealize Automation 8.1.1 以降をサポート
- [vRA Cloud Infoblox Plug-in バージョン 1.0](#) - vRealize Automation 8.0.1 をサポート
- [vRA Cloud Infoblox Plug-in バージョン 0.1](#) - vRealize Automation 8.0 をサポート

前提条件

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。

- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- 外部 IP アドレス管理プロバイダにアカウントがあること、その IP アドレス管理プロバイダで組織のアカウントへの適切なアクセス認証情報があることを確認します。
- IP アドレス管理プロバイダの導入済みの統合パッケージにアクセスできることを確認します。展開したパッケージは、IP アドレス管理プロバイダの Web サイトまたは VMware Solution Exchange のマーケットプレイスから zip 形式のダウンロード ファイルとして取得し、vRealize Automation に展開されます。

プロバイダ パッケージの .zip ファイルをダウンロードして展開し、IP アドレス管理の統合ページで [プロバイダ] の値として使用できるようにする方法については、[vRealize Automation で使用するための外部 IP アドレス管理プロバイダ パッケージのダウンロードと展開](#)を参照してください。

- IP アドレス管理プロバイダに対して構成されて実行環境に対するアクセス権があることを確認します。実行環境は、通常、アクションベースの拡張性 (ABX) のオンプレミスの組み込み統合ポイントです。

実行環境特性の詳細については、[vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成](#)を参照してください。

- Infoblox アプリケーションに必要な拡張属性を有効にします。[vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加](#)を参照してください。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを設定できます。[vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。
- Infoblox の IP アドレス管理製品へのアクセスと使用に必要なユーザー認証情報があることを確認します。たとえば、Infoblox アプライアンスの [管理] タブを開き、管理者、グループ、およびロールのエントリをカスタマイズします。管理者権限またはスーパーユーザー権限を持つグループ、または DHCP、DNS、IP アドレス管理、およびグリッドの権限を持つカスタム グループのメンバーである必要があります。これらの設定を使用すると、Infoblox プラグインで使用可能なすべての機能にアクセスできます。これにより、ユーザーは Infoblox の IP アドレス管理統合を作成できるようになり、設計者はクラウド テンプレートと展開内で IP アドレス管理統合を使用できるようになります。ユーザー権限の詳細については、Infoblox の製品ドキュメントを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [IP アドレス管理] をクリックします。
- 3 [プロバイダ] ドロップダウン リストで、構成済みの IP アドレス管理プロバイダ パッケージ（たとえば、*Infoblox_hrg* など）を選択します。

リストが空の場合は、[プロバイダ パッケージのインポート] をクリックし、既存のプロバイダパッケージの .zip ファイルに移動して選択します。プロバイダの .zip ファイルがない場合は、IP アドレス管理プロバイダの Web サイトまたは vRealize Automation Cloud Assembly の [マーケットプレイス] タブから取得することができます。

プロバイダ パッケージの .zip ファイルを vCenter に展開し、[統合] ページで [プロバイダ] の値として使用できるようにする方法については、[vRealize Automation で使用するための外部 IP アドレス管理プロバイダ パッケージのダウンロードと展開](#)を参照してください。

既存の IP アドレス管理統合をアップグレードして、ベンダーが提供する新しいバージョンの IP アドレス管理統合パッケージを使用する方法については、[vRealize Automation で新しい外部 IP アドレス管理統合パッケージにアップグレードする方法](#) を参照してください。

- 4 外部 IP アドレス管理プロバイダのアカウントの管理者ユーザー名とパスワードの認証情報を入力します。また、プロバイダのホスト名などのすべての必須フィールドも入力します。

この例では、次の手順を使用して Infoblox IP アドレス管理プロバイダのホスト名を取得します。

- a 別のブラウザ タブで、Infoblox 管理者権限を使用して IP アドレス管理プロバイダ アカウントにログインします。
- b ホスト名の URL をコピーします。
- c IP アドレス管理の統合ページの [ホスト名] フィールドに、ホスト名の URL を貼り付けます。

- 5 [実行環境] ドロップダウン リストで、既存のオンプレミスのアクションベースの拡張性統合ポイント（たとえば *Infoblox_abx_intg*）を選択します。

実行環境で、vRealize Automation と外部 IP アドレス管理プロバイダ間の通信がサポートされます。

注： Amazon Web Services または Microsoft Azure クラウド アカウントを統合の実行環境として使用している場合は、IP アドレス管理プロバイダ アプライアンスがインターネットからアクセス可能で、NAT またはファイアウォールの背後にないこと、パブリックに解決可能な DNS 名があることを確認してください。IP アドレス管理プロバイダにアクセスできない場合、Amazon Web Services Lambda または Microsoft Azure 機能が接続できないため、統合は失敗します。関連情報については、[vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成](#) を参照してください。

IP アドレス管理フレームワークは、アクションベースの拡張性 (ABX) のオンプレミスの組み込み実行環境のみをサポートします。

注： Infoblox IP アドレス管理統合ポイントには、アクションベースの拡張性 (ABX) のオンプレミスの組み込み統合ポイントが必要です。

構成したクラウド アカウントまたは統合ポイントは、関連付けられているクラウド拡張性プロキシを介して、vRealize Automation と IP アドレス管理プロバイダの間の通信を可能にします。すでに作成されているプロバイダを選択するか、新規作成することができます。

実行環境の作成方法の詳細については、[vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成](#) を参照してください。

- 6 [検証] をクリックします。

この例では、実行環境に対してオンプレミスのアクションベースの拡張統合を使用しているため、検証アクションを表示できます。

- a [拡張性] タブをクリックします。
- b [アクティビティ] - [アクションの実行] の順にクリックし、フィルタから [すべての実行] または [統合の実行] を選択して、エンドポイント検証アクションが開始されて実行中であることを確認します。

- 7 IP アドレス管理プロバイダからの自己署名証明書を信頼するように要求するプロンプトが表示されたら、[受け入れる] をクリックします。

自己署名証明書を受け入れると、検証アクションを続行することができます。

- 8 この IP アドレス管理統合ポイントの [名前] (*Infoblox_Integration* など) と [説明] (*Infoblox IPAM with ABX integration for team HRG*) を入力します。

- 9 [追加] をクリックして、新しい外部 IP アドレス管理統合ポイントを保存します。

データ収集アクションは模倣されます。ネットワークおよび IP アドレス範囲が IP アドレス管理プロバイダからデータとして収集されます。データ収集アクションは、次のようにして表示できます。

a [拡張性] タブをクリックします。

b [アクティビティ] - [アクションの実行] の順にクリックして、データ収集アクションが開始され実行されていることを確認します。アクションの実行コンテンツを開いて内容を表示できます。

結果

これで、ネットワークおよびネットワーク プロファイルで、プロバイダ固有の外部の IP アドレス管理統合を使用できるようになりました。

vRealize Automation で既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定

IP アドレス値を vRealize Automation から内部的に取得するのではなく、外部の IP アドレス管理プロバイダから取得し、それによって管理されるように、既存のネットワークを定義できます。

組織の外部 IP アドレス管理プロバイダ アカウントで定義した既存の IP アドレス設定にアクセスするネットワークを定義できます。この手順では、前の手順で作成した Infoblox プロバイダ統合についてさらに詳細に説明します。

この例では、vCenter Server からデータを収集した既存のネットワークを使用してネットワーク プロファイルを構成します。次に、これらのネットワークを構成し、外部 IP アドレス管理プロバイダ（この例では Infoblox）から IP アドレス情報が取得されるようにします。このネットワーク プロファイルと一致する vRealize Automation からプロビジョニングする仮想マシンは、IP アドレスなどの TCP/IP 関連の設定を外部 IP アドレス管理プロバイダから取得します。

ネットワークの詳細については、[vRealize Automation のネットワーク リソース](#)を参照してください。ネットワーク プロファイルの詳細については、[vRealize Automation でネットワーク プロファイルを追加する方法](#) および [vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください。

関連情報については、[vRealize Automation で外部 IP アドレス管理統合用のオンデマンド ネットワークをサポートするようにネットワーク プロファイルを設定する方法](#)を参照してください。

前提条件

この一連の手順は、IP アドレス管理プロバイダの統合ワークフローのコンテキストの一部として表示されます。[チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成](#)を参照してください。

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。

- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- [Infoblox](#) や [Bluecat](#) などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。このワークフローの例では、IP アドレス管理プロバイダは Infoblox です。
- IP アドレス管理プロバイダの IP アドレス管理統合ポイントがあることを確認します。[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。

手順

- 1 ネットワークを構成するには、[インフラストラクチャ] - [リソース] - [ネットワーク] の順にクリックします。
- 2 [ネットワーク] タブで、IP アドレス管理プロバイダの統合ポイントで使用する既存のネットワークを選択します。この例では、ネットワーク名は *net.23.117-only-IPAM* です。

リストされたネットワークは、組織内の vCenter Server から vRealize Automation によってデータ収集されています。
- 3 外部 IP アドレス管理プロバイダから値を取得するには、[アカウント/リージョン]、[名前]、[ネットワーク ドメイン] を除く以下のすべてのネットワーク設定が空であることを確認します。
 - ドメイン（手順 8 の注を参照）
 - CIDR
 - デフォルト ゲートウェイ
 - DNS サーバ
 - DNS 検索ドメイン
- 4 [IP アドレス範囲] タブをクリックして、[IP アドレス管理の IP アドレス範囲の追加] をクリックします。
- 5 [ネットワーク] メニューから、構成したネットワーク（例： *net.23.117-only-IPAM*）を選択します。
- 6 [プロバイダ] メニューから、ワークフローの前の手順で作成した IP アドレス管理統合ポイント *Infoblox_Integration* を選択します。
- 7 現在表示されている [アドレス空間] ドロップダウン メニューから、リストされているネットワーク ビューのいずれかを選択します。

Infoblox のアドレス空間は、ネットワーク ビューと呼ばれます。

ネットワーク ビューは、IP アドレス管理プロバイダ アカウントから取得されます。この例では、構成したネットワーク サブネット *net.23.117-only-IPAM*、ワークフローの前の手順で作成した統合ポイント *Infoblox_Integration*、*default* という名前のアドレス空間を使用しています。

リストされたアドレス空間の値は、外部 IP アドレス管理プロバイダから取得されます。

- 8 選択したアドレス空間で使用可能なネットワークのリストから、1つ以上のネットワークを選択します。たとえば、10.23.117.0/24 を選択します。

この例では、選択したネットワークの [ドメイン] と [DNS サーバ] 列の値に Infoblox からの値が含まれています。

注： 手順 3 で vRealize Automation で [ドメイン] が指定されているネットワークを選択し、[ドメイン] の値を含む外部 IP アドレス管理プロバイダのアドレス空間からネットワークを選択すると、外部 IP アドレス管理プロバイダ ネットワークの [ドメイン] の値が vRealize Automation で指定された [ドメイン] よりも優先されます。IP アドレス管理プロバイダの IP アドレス範囲設定に、前述のように Cloud Assembly または外部 IP アドレス管理プロバイダのいずれにも [ドメイン] の値が設定されていない場合、プロビジョニングは失敗します。

Infoblox の場合は、ブループリントのプロパティ `Infoblox.IPAM.Network.dnsSuffix` をマシンレベルで使用して、ドメインの値を上書きできます。関連情報については、[vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用](#)を参照してください。

- 9 [追加] をクリックして、ネットワークの IP アドレス管理の IP アドレス範囲を保存します。

この範囲は、[IP アドレス範囲] テーブルに表示されます。

- 10 [IP アドレス] タブをクリックします。

外部 IP アドレス管理プロバイダからの新しいアドレス範囲を使用してマシンをプロビジョニングすると、新しいレコードが [IP アドレス] テーブルに表示されます。

- 11 ネットワーク プロファイルを構成してネットワークが使用されるようにするには、[インフラストラクチャ] - [構成] - [ネットワーク プロファイル] の順にクリックします。

- 12 *Infoblox-NP* のようにネットワーク プロファイルに名前を指定し、次のサンプル設定を追加します。

- サマリ タブ

- vSphere クラウドのアカウント/リージョンを指定します。
- ネットワーク プロファイルの機能タグを *infoblox_abx* のような名前を指定して追加します。

機能タグはクラウド テンプレートのプロビジョニングの関連付けの際にクラウド テンプレートの制約タグとして使用するため、書き留めておきます。

- ネットワーク タブ

- 前の手順で作成したネットワークを追加します。例： *net.23.117-only-IPAM*。

- 13 [保存] をクリックし、これらの設定でネットワーク プロファイルを保存します。

結果

これで、クラウド テンプレート デザインで Infoblox IP アドレス管理統合に使用される既存のネットワーク タイプに対して、ネットワークとネットワーク プロファイルが設定されました。

vRealize Automation での外部 IP アドレス管理プロバイダ範囲の割り当てを使用するクラウド テンプレートの定義と展開

クラウド テンプレートを定義して、外部 IP アドレス管理プロバイダから IP アドレス割り当てを取得および管理することができます。この例では、外部 IP アドレス管理プロバイダとして Infoblox を使用します。

外部 IP アドレス管理統合ワークフローの最後の手順では、クラウド テンプレートを定義して展開することで、以前に定義したネットワークおよびネットワーク プロファイルを組織の Infoblox アカウントに接続し、展開した仮想マシンの IP アドレス割り当てを vRealize Automation Cloud Assembly からではなく、外部 IP アドレス管理プロバイダから取得および管理します。

このワークフローでは、外部 IP アドレス管理プロバイダとして Infoblox を使用しており、一部の手順では例の値が Infoblox に固有のものになっていますが、この手順は他の外部 IP アドレス管理の統合にも適用できます。



VMware vRealize Automation および Infoblox DDI を使用した IP アドレス管理および DNS の自動化
関連情報は Infoblox のブログに記載されています。

クラウド テンプレートを展開して仮想マシンを起動すると、展開内の各仮想マシンに使用される IP アドレスは、[リソース] - [ネットワーク] 画面にネットワーク エントリとして表示されます。また、IP アドレス管理プロバイダのアカウントの IP アドレス管理プロバイダ ネットワーク、およびホスト vCenter Server に展開された各仮想マシンの vSphere Web Client レコードでは、新しいホスト レコードとして表示されます。

前提条件

この手順は、外部 IP アドレス管理プロバイダの統合ワークフローのコンテキストで説明しています。[チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成](#) を参照してください。

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- Infoblox や BlueCat などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。
- ホスト アカウントへの管理者アクセス権があること、vSphere Web Client のレコードで、ホスト vCenter Server に展開された仮想マシンのステータス レコードを表示するために必要なロール要件があることを確認します。
- 外部 IP アドレス管理プロバイダ用の IP アドレス管理統合ポイントがあることを確認します。[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。
- 目的の IP アドレス管理統合ポイントで、外部 IP アドレス管理の統合をサポートする vRealize Automation Cloud Assembly ネットワークおよびネットワーク プロファイルを構成していることを確認します。
[vRealize Automation で既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定](#) を参照してください。
- プロジェクトとクラウド ゾーンで、IP アドレス管理統合ポイントのタグとネットワークまたはネットワーク プロファイルが一致するようにタグ付けされていることを確認します。必要に応じて、カスタム リソースの命名をサポートするようにプロジェクトを構成します。

プロジェクトとクラウド ゾーンのロール、クラウド テンプレート内の他のインフラストラクチャ要素のロールの詳細については、[チュートリアル: vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。タグ付けの詳細については、[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。

プロジェクトの設定を使用した仮想マシンのカスタムでの名前付けの詳細については、[vRealize Automation Cloud Assembly を使用して展開されたリソースの名前をカスタマイズする方法](#)を参照してください。

手順

- 1 [クラウド テンプレート] - [新規作成] の順にクリックし、[新規クラウド テンプレート] 画面に次の情報を入力して [作成] をクリックします。

- [名前] = ipam-bpa
- [説明] = Infoblox IP アドレス管理統合を使用するクラウド テンプレート
- [プロジェクト] = 123VC

- 2 この例では、クラウドに依存しないマシン コンポーネントと、クラウドに依存しないネットワーク コンポーネントをクラウド テンプレート キャンバスに追加し、2 つのコンポーネントを接続します。
- 3 クラウド テンプレート コードを編集して、ネットワーク プロファイルに追加した機能タグに一致する制約タグをネットワーク コンポーネントに追加します。この例では、このタグ値は *infoblox_abx* です。
- 4 クラウド テンプレート コードを編集し、ネットワーク割り当てのタイプを *static* に指定します。

外部 IP アドレス管理プロバイダを使用している場合は、`assignment: static` 設定が必要です。

この例では、指定された IP アドレス 10.23.117.4 は、関連付けられたネットワーク プロファイルでネットワークに選択した外部 IP アドレス管理のアドレス空間で現在使用可能であることが確認されています。

`assignment: static` 設定は必須ですが、`address: value` 設定は不要です。外部 IP アドレスの選択範囲を特定のアドレス値で始めることができますが、これは必須ではありません。`address: value` 設定を指定しない場合、外部 IP アドレス管理プロバイダは外部 IP アドレス管理ネットワークで使用可能な次のアドレスを選択します。

- 5 クラウド テンプレート コードを次の例を参照して確認します。

```
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}'
```

クラウド テンプレートで DNS および DHCP の設定を指定する際に使用できる Infoblox プロパティの例については、[vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用](#) を参照してください。

- 6 クラウド テンプレート画面で [展開] をクリックし、展開に *Infoblox-1* という名前を指定し、[展開の種類] 画面の [展開] をクリックします。
- 7 クラウド テンプレートの展開中に、[拡張性] タブをクリックし、[アクティビティ] - [アクションの実行] の順に選択して、*Infoblox_AllocateIP_n* 拡張性アクションが実行されていることを確認します。

拡張性アクションが完了し、マシンがプロビジョニングされると、*Infoblox_Update_n* アクションによって、MAC アドレスが Infoblox に伝達されます。
- 8 Infoblox アカウントにログインして開くと、IP アドレス管理アドレスの新しいホスト レコードが関連付けられた 10.23.117.0/24 ネットワークに表示されます。Infoblox で DNS タブを開き、新しい DNS ホスト レコードを表示することもできます。
- 9 仮想マシンがプロビジョニングされていることを確認するには、ホスト vCenter Server および vSphere Web Client にログインして、プロビジョニングされたマシンを特定し、DNS 名と IP アドレスを確認します。

プロビジョニングされた仮想マシンが起動すると、*Infoblox_AllocateIP* の拡張性アクションによって、MAC アドレスが Infoblox に伝達されます。
- 10 vRealize Automation Cloud Assembly の新しいネットワーク レコードを表示するには、[インフラストラクチャ] - [リソース] - [ネットワーク] の順に選択し、[IP アドレス] タブをクリックして開きます。
- 11 展開を削除すると、展開内の仮想マシンの IP アドレス管理アドレスが解放され、外部 IP アドレス管理プロバイダで再度 IP アドレスを他の割り当てに使用できるようになります。vRealize Automation Cloud Assembly でのこのイベントの拡張性アクションは *Infoblox_Deallocate* です。

vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用

Infoblox 向けに外部 IP アドレス管理統合を含む vRealize Automation プロジェクトには、Infoblox 固有のプロパティを使用できます。

次の Infoblox プロパティは、クラウド テンプレートのデザインおよび展開で Infoblox ip アドレス管理統合と共に使用できます。これらを vRealize Automation で使用することにより、クラウド テンプレートの展開時に IP アドレスの割り当ての制御を強化することができます。これらのプロパティの使用はオプションです。

■ Infoblox.IPAM.createFixedAddress

このプロパティを使用すると、Infoblox 内に固定アドレス レコードを作成できます。利用可能な値は True と False です。デフォルトでは、ホスト レコードが作成されます。デフォルト値は False です。

■ Infoblox.IPAM.Network.dnsView

このプロパティを使用すると、Infoblox 内にホスト レコードを作成するときに DNS ビューを使用できます。

- Infoblox.IPAM.Network.enableDns

Infoblox で IP アドレスを割り当てるときに、このプロパティによって DNS レコードも作成できます。利用可能な値は True と False です。デフォルト値は True です。

- Infoblox.IPAM.Network.enableDhcp

ホスト アドレスの DHCP 構成を有効にするには、このオプションを True に設定します。

- Infoblox.IPAM.Network.dnsSuffix

このプロパティを使用すると、Infoblox ネットワークの *domain* DHCP オプションを新しい値で上書きできます。この機能は、Infoblox ネットワークに *domain* DHCP オプションが設定されていない場合、または *domain* DHCP オプションを上書きする必要がある場合に便利です。デフォルト値は null（空の文字列）です。

Infoblox.IPAM.Network.dnsSuffix は、Infoblox.IPAM.Network.enableDns が True に設定されている場合にのみ適用されます。

Infoblox のプロパティを指定するには、vRealize Automation Cloud Assembly で次のいずれかの方法を使用します。

- プロジェクトでプロパティを指定するには、[インフラストラクチャ] - [管理] - [プロジェクト] 画面の [カスタム プロパティ] セクションを使用します。この方法を使用すると、このプロジェクトの範囲でプロビジョニングされるすべてのマシンに対して、指定されたプロパティが適用されます。

- クラウド テンプレートで各マシン コンポーネントにプロパティを指定できます。Infoblox.IPAM.Network.dnsView プロパティの使用法を示すサンプル クラウド テンプレートコードを以下に示します。

```
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
```

- 拡張性サブスクリプションを使用して、プロパティを指定できます。

このユースケースに関連する Infoblox 拡張可能属性の関連情報については、[vRealize Automation との統合のために Infoblox アプリケーションの必要な拡張属性を追加](#)を参照してください。

クラウド テンプレートのさまざまなマシン NIC での Infoblox プロパティの使用

次の Infoblox プロパティには、クラウド テンプレートのマシン NIC ごとに異なる値を設定できます。

- Infoblox.IPAM.Network.enableDhcp
- Infoblox.IPAM.Network.dnsView
- Infoblox.IPAM.Network.enableDns

たとえば、NIC ごとに異なる Infoblox.IPAM.Network.dnsView 値を使用するには、NIC ごとに Infoblox.IPAM.Network< *nicIndex*>.dnsView エントリを使用します。次の例では、2 つの NIC の Infoblox.IPAM.Network.dnsView 値が異なります。

```
formatVersion: 1
inputs: {}
resources:
  Cloud_Machine_1:
    type: Cloud.Machine
    properties:
      Infoblox.IPAM.Network0.dnsView: default
      Infoblox.IPAM.Network1.dnsView: my-net
      image: ubuntu
      flavor: small
      networks:
        - network: '${resource.Cloud_Network_1.id}'
          deviceIndex: 0
        - network: '${resource.Cloud_Network_2.id}'
          deviceIndex: 1
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      networkType: existing
  Cloud_Network_2:
    type: Cloud.Network
    properties:
      networkType: existing
```

デフォルトでは、Infoblox の統合によって、Infoblox の *default* DNS ビューに DNS ホスト レコードが作成されます。Infoblox 管理者が *custom* DNS ビューを作成した場合は、デフォルトの統合動作を上書きして、マシン コンポーネントの Infoblox.IPAM.Network.dnsView プロパティを使用して名前付きビューを指定できます。たとえば、次のプロパティを Cloud_Machine_1 コンポーネントに追加すると、Infoblox に名前付き DNS ビューを指定できます。

```
Cloud_Machine_1:
  type: Cloud.Machine
  properties:
    image: ubuntu
    flavor: small
    Infoblox.IPAM.Network.dnsView:<dns-view-name>
```

DNS ビューの構成および使用方法の詳細については、Infoblox 製品ドキュメントの [DNS ビュー](#) を参照してください。Infoblox 統合ワークフローの例については、[vRealize Automation での外部 IP アドレス管理プロバイダ範囲の割り当てを使用するクラウド テンプレートの定義と展開](#) を参照してください。

組織での vRealize Automation Cloud Assembly の設定

3

Cloud Assembly 管理者は、ユーザー ロールを理解し、クラウド アカウントのベンダーおよび統合アプリケーションとの接続を設定する必要があります。

クラウド アカウントおよび統合を設定するときは、Cloud Assembly とそれらのターゲット システムの間の通信を設定します。

この章には、次のトピックが含まれています。

- vRealize Automation のユーザー ロールについて
- vRealize Automation Cloud Assembly へのクラウド アカウントの追加
- 他のアプリケーションとの vRealize Automation の統合
- vRealize Automation Cloud Assembly でのオンボーディング プランについて
- vRealize Automation Cloud Assembly 環境の詳細設定

vRealize Automation のユーザー ロールについて

vRealize Automation には、いくつかのレベルのユーザー ロールがあります。これらのレベルにより、組織へのアクセス、サービスへのアクセス、クラウド テンプレート、カタログ アイテム、パイプラインを生成または使用するプロジェクトへのアクセス、およびユーザー インターフェイスの各部分の使用または表示の可否が制御されます。クラウド管理者はこれらのレベルにより、運用上のニーズに応じてさまざまなレベルの粒度を適用できます。

ロール全般の説明

ユーザー ロールは、さまざまなレベルで定義されます。サービス レベルのロールは、サービスごとに定義されます。

サービス ロールの詳細については、次の表を参照してください。

ロール	全般的な権限	ロールが定義される場所
組織の所有者	<p>コンソールにアクセスして、ユーザーを組織に追加できます。</p> <p>組織の所有者は、サービス ロールを持っていないサービスにアクセスすることはできません。</p> <p>組織ユーザー ロールの詳細</p>	組織コンソール
組織のメンバー	<p>コンソールにアクセスできます。</p> <p>組織のメンバーは、サービス ロールを持っていないサービスにアクセスすることはできません。</p> <p>組織ユーザー ロールの詳細</p>	組織コンソール
サービス管理者	<p>コンソールにアクセスでき、サービスの表示、更新、および削除の全般的な権限を持ちます。</p> <ul style="list-style-type: none"> ■ Cloud Assembly サービス ロール ■ Service Broker サービス ロール ■ Code Stream サービス ロール 	組織コンソール
サービス ユーザー	<p>制限付きの権限でコンソールとサービスにアクセスできます。</p> <p>サービス メンバーのユーザー インターフェイスには制限があります。表示または実行できる内容は、プロジェクトのメンバーシップによって異なります。</p> <ul style="list-style-type: none"> ■ Cloud Assembly サービス ロール ■ Service Broker サービス ロール ■ Code Stream サービス ロール 	組織コンソール
サービス閲覧者	<p>コンソールとサービスに表示のみモードでアクセスできます。</p> <ul style="list-style-type: none"> ■ Cloud Assembly サービス ロール ■ Service Broker サービス ロール ■ Code Stream サービス ロール 	組織コンソール
実行者（vRealize Automation Code Stream のみ）	<p>コンソールにアクセスして、パイプラインの実行を管理できます。</p> <p>Code Stream サービス ロール</p>	組織コンソール
vRA Migration Assistant 管理者	<p>コンソールにアクセスでき、vRA Migration Assistant および Cloud Assembly で表示、更新、削除の全般的な権限を持ちます。</p> <p>このロールには、少なくとも Cloud Assembly 閲覧者ロールが必要です。</p>	組織コンソール
vRA Migration Assistant 閲覧者	<p>コンソール、vRA Migration Assistant、および Cloud Assembly に表示専用モードでアクセスできます。</p> <p>このロールには、少なくとも Cloud Assembly 閲覧者ロールが必要です。</p>	組織コンソール
Orchestrator 管理者	<p>特定のグループによって作成されたコンテンツも含めて、vRealize Orchestrator クライアントのすべての機能およびコンテンツにアクセスできます。</p>	組織コンソール、および vRealize Orchestrator クライアント内

ロール	全般的な権限	ロールが定義される場所
Orchestrator ワークフロー設計者	自分の vRealize Orchestrator クライアント コンテンツを作成、実行、編集、および削除できます。自分に割り当てられたグループに自分のコンテンツを追加できます。vRealize Orchestrator クライアントの管理およびトラブルシューティング機能にはアクセスできません。	組織コンソール、および vRealize Orchestrator クライアント内
プロジェクト ロール	プロジェクト ロールに応じて、プロジェクトのリソースを表示および管理できます。 プロジェクト ロールには、管理者、メンバー、閲覧者が含まれます。 vRealize Automation の組織およびサービスのユーザー ロール	vRealize Automation Cloud Assembly、vRealize Automation Service Broker、および vRealize Automation Code Stream
カスタム ロール	すべてのサービスについて vRealize Automation Cloud Assembly によって権限が定義されます。 ユーザーがサービスにアクセスできるようになるには、該当するサービスのサービス閲覧者ロールが少なくとも 1 つ必要です。カスタムロールはサービス ロールよりも優先されます。 vRealize Automation のカスタム ユーザーロール	vRealize Automation Cloud Assembly および vRealize Automation Service Broker

vRealize Automation の組織およびサービスのユーザー ロール

vRealize Automation Cloud Assembly、vRealize Automation Service Broker、および vRealize Automation Code Stream の各サービスに対して定義されている組織およびサービスのユーザー ロールにより、ユーザーが各サービスで表示および実行できることが決まります。

組織ユーザー ロール

ユーザー ロールは、組織の所有者が vRealize Automation コンソールを使用して、組織に対して定義します。組織ロールとサービス ロールという 2 つのタイプのロールがあります。

組織ロールはグローバルで、組織内のすべてのサービスに適用されます。組織レベルのロールは、組織の所有者または組織のメンバーのロールです。

組織のロールの詳細については、[vRealize Automation の管理](#)を参照してください。

サービス固有の権限である vRealize Automation Cloud Assembly サービス ロールは、コンソールの組織レベルでも割り当てられます。

サービス ロール

これらのサービス ロールは、組織の所有者によって割り当てられます。

この記事では、3 つすべてのサービスに関する情報を扱っています。

- [Cloud Assembly サービス ロール](#)
- [Service Broker サービス ロール](#)

■ Code Stream サービス ロール

Cloud Assembly サービス ロール

vRealize Automation Cloud Assembly サービス ロールによって、vRealize Automation Cloud Assembly で表示および実行できる内容が決まります。これらのサービス ロールは、組織の所有者がコンソールで定義します。

表 3-1. vRealize Automation Cloud Assembly サービス ロールの説明

ロール	説明
Cloud Assembly 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権を持っているユーザー。これは、クラウド アカウントの追加、新しいプロジェクトの作成、プロジェクト管理者の割り当てなど、すべてを表示および操作できる唯一のユーザー ロールです。
Cloud Assembly ユーザー	Cloud Assembly 管理者ロールを持たないユーザー。 vRealize Automation Cloud Assembly プロジェクトでは、管理者がユーザーをプロジェクト メンバー、管理者、または閲覧者としてプロジェクトに追加します。管理者は、プロジェクト管理者を追加することもできます。
Cloud Assembly 閲覧者	情報を表示するための読み取りアクセス権は持っているが、作成、更新、削除はできないユーザー。これは、すべてのプロジェクトに対する読み取り専用のロールです。 閲覧者ロールを持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクト メンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバー ロールとは異なり、権限が拡張されることはありません。

サービス ロールに加えて、vRealize Automation Cloud Assembly にはプロジェクト ロールがあります。どのプロジェクトもすべてのサービスで使用できます。

プロジェクト ロールは vRealize Automation Cloud Assembly で定義され、プロジェクトごとに変えることができます。

次の表に、さまざまなサービス ロールおよびプロジェクト ロールで何を表示および実行できるかを示します。サービス管理者にはユーザー インターフェイスのすべての領域に対する完全な権限が付与されていることに注意してください。

プロジェクト ロールに関する説明を参考にして、ユーザーに付与する権限を決定できます。

- プロジェクト管理者は、サービス管理者が作成したインフラストラクチャを活用して、プロジェクト メンバーが開発作業に必要なリソースを確実に使用できるようにします。
- プロジェクト メンバーは、クラウド テンプレートを設計および展開するためにプロジェクト内で作業します。
- プロジェクト閲覧者は、読み取り専用アクセスに制限されていますが、クラウド テンプレートのダウンロードなどの非破壊的な操作を実行できる場合もあります。

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー		
				ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
[Cloud Assembly へのアクセス]						
コンソール	vRA コンソールで Cloud Assembly を表示して開くことができます	はい	はい	はい	はい	はい
[インフラストラクチャ]						
	[インフラストラクチャ] タブを表示して開く	はい	はい	はい	はい	はい
構成 - プロジェクト	プロジェクトの作成	はい				
	プロジェクトのサマリ、プロビジョニング、Kubernetes、統合、およびテストプロジェクトの構成から値を更新または削除します。	はい				
	プロジェクトでユーザーおよびグループを追加し、ロールを割り当てます。	はい		はい。自分のプロジェクト。		
	プロジェクトの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
構成 - クラウド ゾーン	クラウド ゾーンの作成、更新、または削除	はい				
	クラウド ゾーンの表示	はい	はい			
構成 - Kubernetes ゾーン	Kubernetes ゾーンの作成、更新、または削除	はい				
	Kubernetes ゾーンの表示	はい	はい			
構成 - フレーバー	フレーバーの作成、更新、または削除	はい				
	フレーバーの表示	はい	はい			
構成 - イメージ マッピング	イメージ マッピングの作成、更新、または削除	はい				
	イメージ マッピングの表示	はい	はい			
構成 - ネットワーク プロファイル	ネットワーク プロファイルの作成、更新、または削除	はい				

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール (続き)

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
	イメージ ネットワーク プロファイルの表示	はい	はい			
構成 - ストレージ プロファイル	ストレージ プロファイルの作成、更新、または削除	はい				
	イメージ ストレージ プロファイルの表示	はい	はい			
構成 - 価格設定カード	価格設定カードの作成、更新、または削除	はい				
	価格設定カードの表示	はい	はい			
構成 - タグ	タグの作成、更新、または削除	はい				
	タグの表示	はい	はい			
リソース - コンピューティング	検出されたコンピューティング リソースへのタグの追加	はい				
	検出されたコンピューティング リソースの表示	はい	はい			
リソース - ネットワーク	ネットワーク タグ、IP アドレス範囲、IP アドレスの変更	はい				
	検出されたネットワーク リソースの表示	はい	はい			
リソース - セキュリティ	検出されたセキュリティ グループへのタグの追加	はい				
	検出されたセキュリティ グループの表示	はい	はい			
リソース - ストレージ	検出されたストレージへのタグの追加	はい				
	ストレージの表示	はい	はい			
リソース - マシン	マシンの追加と削除	はい				
	マシンの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
リソース - ボリューム	検出されたストレージ ボリュームの削除	はい				
	検出されたストレージ ボリュームの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト。

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール (続き)

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
リソース - Kubernetes	Kubernetes クラスターの展開または追加、および名前空間の作成または追加	はい				
	Kubernetes クラスターと名前空間の表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
アクティビティ - 申請	展開申請レコードの削除	はい				
	展開申請レコードの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
アクティビティ - イベント ログ	イベント ログの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
接続 - クラウド アカウント	クラウド アカウントの作成、更新、または削除	はい				
	クラウド アカウントの表示	はい	はい			
接続 - 統合	統合の作成、更新、または削除	はい				
	統合の表示	はい	はい			
オンボーディング	オンボーディング プランの作成、更新、または削除	はい				
	オンボーディング プランの表示	はい	はい			はい。自分のプロジェクト
[マーケットプレイス]						
	[マーケットプレイス] タブを表示して開く	はい	はい			
	ダウンロードしたクラウド テンプレートを [デザイン] タブで使用する	はい		はい。プロジェクトに関連付けられている場合。	はい。プロジェクトに関連付けられている場合。	
マーケットプレイス: クラウド テンプレート	クラウド テンプレートのダウンロード	はい				
	クラウド テンプレートの表示	はい	はい			

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール (続き)

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
マーケットプレイス - イメージ	イメージのダウンロード	はい				
	イメージの表示	はい	はい			
マーケットプレイス - ダウンロード	ダウンロードしたすべてのアイテムのログの表示	はい	はい			
[拡張性]						
	[拡張性] タブを表示して開く	はい	はい			はい
イベント	拡張性イベントの表示	はい	はい			
サブスクリプション	拡張性サブスクリプションの作成、更新、または削除	はい				
	サブスクリプションの無効化	はい				
	サブスクリプションの表示	はい	はい			
ライブラリ - イベント トピック	イベント トピックの表示	はい	はい			
ライブラリ - アクション	拡張性アクションの作成、更新、または削除	はい				
	拡張性アクションの表示	はい	はい			
ライブラリ - ワークフロー	拡張性ワークフローの表示	はい	はい			
アクティビティ - アクションの実行	拡張性アクションの実行のキャンセルまたは削除	はい				
	拡張性アクションの実行の表示	はい	はい			はい。自分のプロジェクト
アクティビティ - ワークフローの実行	拡張性ワークフローの実行の表示	はい	はい			
[デザイン]						
デザイン	[デザイン] タブを開き、クラウド テンプレートのリストを表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール (続き)

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
クラウド テンプレート	クラウド テンプレートの作成、更新、削除	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
	クラウド テンプレートの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
	クラウド テンプレートのダウンロード	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
	クラウド テンプレートのアップロード	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
	クラウド テンプレートの展開	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
	クラウド テンプレートのバージョン管理とリストア	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
	クラウド テンプレートのカタログへのリリース	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
カスタム リソース	カスタム リソースの作成、更新、または削除	はい				
	カスタム リソースの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
カスタム アクション	カスタム アクションの作成、更新、または削除	はい				
	カスタム アクションの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
[展開]						
	[展開] タブを表示して開く	はい	はい	はい	はい	はい

表 3-2. vRealize Automation Cloud Assembly サービス ロールとプロジェクト ロール （続き）

ユーザー インターフェイスのコンテキスト	タスク	Cloud Assembly 管理者	Cloud Assembly 閲覧者	Cloud Assembly ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者またはプロジェクトメンバーである必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
	展開の詳細、展開履歴、トラブルシューティング情報などを含め、展開を表示する。	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
	ポリシーに基づいて展開に対して Day 2 アクションを実行します。	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	

Service Broker サービス ロール

vRealize Automation Service Broker サービス ロールによって、vRealize Automation Service Broker で表示および実行できる内容が決まります。これらのサービス ロールは、組織の所有者がコンソールで定義します。

表 3-3. Service Broker サービス ロールの説明

ロール	説明
Service Broker 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権が必要です。これは、新しいプロジェクトの作成やプロジェクト管理者の割り当てなど、すべてのタスクを実行できる唯一のユーザー ロールです。
Service Broker ユーザー	vRealize Automation Service Broker 管理者ロールを持たないすべてのユーザー。 vRealize Automation Service Broker プロジェクトでは、管理者がユーザーをプロジェクト メンバー、管理者、または閲覧者としてプロジェクトに追加します。管理者は、プロジェクト管理者を追加することもできます。
Service Broker 閲覧者	情報を表示するための読み取りアクセス権は持っているが、作成、更新、削除はできないユーザー。 閲覧者ロールを持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクト メンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバー ロールとは異なり、権限が拡張されることはありません。

サービス ロールに加えて、vRealize Automation Service Broker にはプロジェクト ロールがあります。どのプロジェクトもすべてのサービスで使用できます。

プロジェクト ロールは vRealize Automation Service Broker で定義され、プロジェクトごとに変えることができます。

次の表に、さまざまなサービス ロールおよびプロジェクト ロールで何を表示および実行できるかを示します。サービス管理者にはユーザー インターフェイスのすべての領域に対する完全な権限が付与されていることに注意してください。

プロジェクト ロールに関する次の説明を利用して、ユーザーに付与する権限を決定します。

- プロジェクト管理者は、サービス管理者が作成したインフラストラクチャを活用して、プロジェクト メンバーが開発作業に必要なリソースを確実に使用できるようにします。
- プロジェクト メンバーは、クラウド テンプレートを設計および展開するためにプロジェクト内で作業します。
- プロジェクト閲覧者は、読み取り専用アクセスに制限されています。

表 3-4. Service Broker サービス ロールとプロジェクト ロール

ユーザー インターフェイスのコンテキスト	タスク	Service Broker 管理者	Service Broker 閲覧者	Service Broker ユーザー		
				ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者である必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
[Service Broker へのアクセス]						
コンソール	コンソールで Service Broker を表示して開くことができます	はい	はい	はい	はい	はい
[インフラストラクチャ]						
	[インフラストラクチャ] タブを表示して開く	はい	はい			
構成 - プロジェクト	プロジェクトの作成	はい				
	プロジェクトのサマリ、プロビジョニング、Kubernetes、および統合の値の更新または削除	はい				
	プロジェクトでユーザーおよびグループを追加し、ロールを割り当てます。	はい		はい。自分のプロジェクト。		
	プロジェクトの表示	はい	はい			
構成 - クラウド ゾーン	クラウド ゾーンの作成、更新、または削除	はい				
	クラウド ゾーンの表示	はい	はい			
構成 - Kubernetes ゾーン	Kubernetes ゾーンの作成、更新、または削除	はい				
	Kubernetes ゾーンの表示	はい	はい			
接続 - クラウド アカウント	クラウド アカウントの作成、更新、または削除	はい				
	クラウド アカウントの表示	はい	はい			

表 3-4. Service Broker サービス ロールとプロジェクト ロール (続き)

ユーザー インターフェイスのコンテキスト	タスク	Service Broker 管理者	Service Broker 閲覧者	Service Broker ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者である必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
接続 - 統合	統合の作成、更新、または削除	はい				
	統合の表示	はい	はい			
アクティビティ - 申請	展開申請レコードの削除	はい				
	展開申請レコードの表示	はい				
アクティビティ - イベント ログ	イベント ログの表示	はい				
[コンテンツとポリシー]						
	[コンテンツとポリシー] タブを表示して開く	はい	はい			
コンテンツ ソース	コンテンツ ソースの作成、更新、または削除	はい				
	コンテンツ ソースの表示	はい	はい			
コンテンツの共有	共有コンテンツの追加または削除	はい				
	共有コンテンツの表示	はい	はい			
コンテンツ	フォームのカスタマイズとアイテムの構成	はい				
	コンテンツの表示	はい	はい			
ポリシー - 定義	ポリシー定義の作成、更新、または削除	はい				
	ポリシー定義の表示	はい	はい			
ポリシー - 適用	適用ログの表示	はい	はい			
通知 - メール サーバ	メール サーバの設定	はい				
[カタログ]						
	[カタログ] タブを表示して開く	はい	はい	はい	はい	はい
	使用可能なカタログ アイテムの表示	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
	カタログ アイテムの要求	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
[展開]						

表 3-4. Service Broker サービス ロールとプロジェクト ロール（続き）

ユーザー インターフェイスのコンテキスト	タスク	Service Broker 管理者	Service Broker 閲覧者	Service Broker ユーザー ユーザーがプロジェクト関連のタスクを表示および実行するには、プロジェクト管理者である必要があります。		
				プロジェクト管理者	プロジェクトメンバー	プロジェクト閲覧者
	[展開] タブを表示して開く	はい	はい	はい。	はい	はい
	展開の詳細、展開履歴、トラブルシューティング情報などを含め、展開を表示する。	はい	はい	はい。自分のプロジェクト	はい。自分のプロジェクト	はい。自分のプロジェクト
	ポリシーに基づいて展開に対して Day 2 アクションを実行	はい		はい。自分のプロジェクト	はい。自分のプロジェクト	
[承認]						
	[承認] タブを表示して開く	はい	はい	はい	はい	はい
	承認申請への応答	はい		Service Broker ユーザー ロールのみ	Service Broker ユーザー ロールのみ	Service Broker ユーザー ロールのみ

Code Stream サービス ロール

vRealize Automation Code Stream サービス ロールによって、vRealize Automation Code Stream で表示および実行できる内容が決まります。これらのロールは、組織の所有者がコンソールで定義します。どのプロジェクトもすべてのサービスで使用できます。

表 3-5. Code Stream サービス ロールの説明

ロール	説明
Code Stream 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権を持っているユーザー。これは、すべてを表示および実行できる唯一のユーザー ロールであり、具体的には、プロジェクトの作成、エンドポイントの統合、トリガの追加、パイプラインとカスタム ダッシュボードの作成、エンドポイントおよび変数に対する制限付きリソースとしてのマーキング、制限付きリソースを使用するパイプラインの実行、パイプラインを vRealize Automation Service Broker で公開する申請が可能です。
Code Stream 開発者	パイプラインを使用できるが制限付きのエンドポイントまたは変数は使用できないユーザー。パイプラインに制限付きのエンドポイントまたは変数が含まれている場合、このユーザーは、制限付きのエンドポイントまたは変数を使用するパイプライン タスクに関する承認を得る必要があります。
Code Stream 実行者	パイプラインを実行でき、ユーザー操作タスクの承認または拒否を行うことができるユーザー。このユーザーは、パイプラインの実行を再開、一時停止、およびキャンセルできますが、パイプラインを変更することはできません。

表 3-5. Code Stream サービス ロールの説明（続き）

ロール	説明
Code Stream ユーザー	vRealize Automation Code Stream にアクセスできるが、vRealize Automation Code Stream 内のそれ以外の権限を持たないユーザー。
Code Stream ビューア	パイプライン、エンドポイント、パイプラインの実行、ダッシュボードを表示するための読み取りアクセス権を持ち、作成、更新、削除することはできないユーザー。サービス閲覧者ロールも持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクトメンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバーロールとは異なり、権限が拡張されることはありません。

サービス ロールに加えて、vRealize Automation Code Stream にはプロジェクト ロールがあります。どのプロジェクトもすべてのサービスで使用できます。

プロジェクト ロールは vRealize Automation Code Stream で定義され、プロジェクトごとに変えることができます。

次の表に、さまざまなサービス ロールおよびプロジェクト ロールで何を表示および実行できるかを示します。サービス管理者にはユーザー インターフェイスのすべての領域に対する完全な権限が付与されていることに注意してください。

プロジェクト ロールに関する次の説明を利用して、ユーザーに付与する権限を決定します。

- プロジェクト管理者は、サービス管理者が作成したインフラストラクチャを活用して、プロジェクト メンバーが開発作業に必要なリソースを確実に使用できるようにします。プロジェクト管理者はメンバーを追加できます。
- サービス ロールを持つプロジェクト メンバーがサービスを使用できます。
- プロジェクト閲覧者はプロジェクトを表示できますが、作成、更新、削除することはできません。

制限付きを除くすべてのアクションは、このロールに、制限付きの変数およびエンドポイントを除くエンティティに対して作成、読み取り、更新、および削除アクションを実行する権限があることを意味します。

表 3-6. vRealize Automation Code Stream サービス ロールの機能

ユーザー インターフェイスのコンテキスト	機能	Code Stream サービス ロール	Code Stream 開発者ロール	Code Stream 実行者ロール	Code Stream 閲覧者ロール	Code Stream ユーザー ロール
[パイプライン]						
	パイプラインの表示	はい	はい	はい	はい	
	パイプラインの作成	はい	はい			
	パイプラインの実行	はい	はい	はい		
	エンドポイントや変数に制限があるパイプラインの実行	はい				
	パイプラインの更新	はい	はい			
	パイプラインの削除	はい	はい			

表 3-6. vRealize Automation Code Stream サービス ロールの機能（続き）

ユーザー インターフェ イスのコン テキスト	機能	Code Stream サービス ロール	Code Stream 開発 者ロール	Code Stream 実行 者ロール	Code Stream 閲 覧者ロール	Code Stream ユ ーザー ロール
[パイプラインの実行]						
	パイプラインの実行状況の表示	はい	はい	はい	はい	
	パイプラインの実行の再開、一時停止、およびキャンセル	はい	はい	はい		
	制限があるリソースの承認のために停止しているパイプラインの再開	はい				
[カスタム統合]						
	カスタム統合の作成	はい	はい			
	カスタム統合の読み取り	はい	はい			
	カスタム統合の更新	はい	はい			
[エンドポイント]						
	実行の表示	はい	はい	はい	はい	
	実行の作成	はい	はい			
	実行の更新	はい	はい			
	実行の削除	はい	はい			
[リソースへの制限付きのマーク付け]						
	エンドポイントまたは変数への制限付きのマーク付け	はい				
[ダッシュボード]						
	ダッシュボードの表示	はい	はい	はい	はい	
	ダッシュボードの作成	はい	はい			
	ダッシュボードの更新	はい	はい			
	ダッシュボードの削除	はい	はい			

vRealize Automation のカスタム ユーザー ロール

vRealize Automation Cloud Assembly 管理者は、vRealize Automation でユーザーが表示および実行できる内容を定義するカスタム ロールを作成できます。その後、これらのロールにユーザーを割り当てることができます。

カスタム ユーザー ロールの権限

vRealize Automation Cloud Assembly を使用して、より詳細なユーザー ロールを定義し、これらのロールにユーザーを割り当てることができます。カスタム ロールには、表示と管理の 2 つのカテゴリがあります。

- **表示**：この権限を持つロールに割り当てられたユーザーは、ユーザー インターフェイスの選択したセクションですべてのプロジェクトのすべてのアイテムを参照できます。このロールは、アカウント、設定、または割り当てられた値を確認する必要があるユーザーに便利です。
- **管理**：この権限を持つロールに割り当てられたユーザーは、ユーザー インターフェイスの選択したセクションですべてのプロジェクトのすべてのアイテムを参照することができ、追加、編集、および削除に関する完全な権限が付与されます。

これらの権限によって、他のロールから付与された権限が拡張されます。これらの権限はプロジェクトのメンバーシップによる制限を受けません。たとえば、インフラストラクチャの一部を管理できるようにプロジェクト管理者の権限を拡張することや、サービス閲覧者が承認を確認して応答できるようにすることができます。

ユーザー ロールを定義してユーザーを割り当てするには、vRealize Automation Cloud Assembly または vRealize Automation Service Broker をサービス管理者として開いて、[インフラストラクチャ] - [管理] - [カスタム ロール] の順に選択します。vRealize Automation Code Stream でカスタム ロールを構成することはできません。ただし、ロールはすべてのサービスに適用されます。

表 3-7. カスタム ロール

ユーザー インターフェイス	権限	説明
[インフラストラクチャ]		
	クラウド アカウントの表示	クラウド アカウントを表示します。
	クラウド アカウントの管理	クラウド アカウントを作成、更新、または削除します。
	イメージ マッピングの表示	イメージ マッピングを表示します。
	イメージマッピングの管理	イメージ マッピングを作成、更新、または削除します。
	フレーバー マッピングの表示	フレーバー マッピングを表示します。
	フレーバーマッピングの管理	フレーバー マッピングを作成、更新、または削除します。
	クラウド ゾーンの表示	クラウド ゾーンを表示します。
	クラウド ゾーンの管理	クラウド ゾーンを作成、更新、または削除します。
	マシンの表示	マシンを表示します。

表 3-7. カスタム ロール （続き）

ユーザー インターフェイス	権限	説明
	申請の表示	アクティビティ申請を表示します。
	申請の管理	リストから申請を削除します。
	統合の表示	統合を表示します。
	統合の管理	統合を作成、更新、または削除します。
	プロジェクトの表示	プロジェクトを表示します。
	プロジェクトの管理	プロジェクトを作成します。プロジェクトでユーザーを追加し、ロールを割り当てます。プロジェクトのサマリ、ユーザー、プロビジョニング、Kubernetes、統合、およびテストプロジェクトの構成から値を更新または削除します。
	オンボーディング プランの表示	オンボーディング プランの表示
	オンボーディング プランの管理	オンボーディング プランを作成、更新、または削除します。
[カタログ]		
	コンテンツの表示	
	コンテンツの管理	コンテンツ ソースを追加、更新、削除します。 コンテンツを共有します。 カタログ アイコンや申請フォームなどのコンテンツをカスタマイズします。
[ポリシー]		
	ポリシーの表示	ポリシーの定義を表示します。
	ポリシーの管理	ポリシー定義を作成、更新、または削除します。
[展開]		
	展開の表示	展開の詳細、展開履歴、トラブルシューティング情報などを含め、すべての展開を表示する。
	展開の管理	すべての展開を表示し、Day 2 ポリシーによって管理者が展開および展開コンポーネントに対して実行することが許可されているすべての Day 2 アクションを実行する。
[クラウド テンプレート]		
	クラウド テンプレートの表示	クラウド テンプレートを表示します。
	クラウド テンプレートの管理	クラウド テンプレートの作成、更新、テスト、削除、バージョン管理、共有、およびクラウドテンプレートのバージョンのリリース/リリース解除を行います。

表 3-7. カスタム ロール （続き）

ユーザー インターフェイス	権限	説明
	クラウド テンプレートの編集	クラウド テンプレートの作成、更新、テスト、バージョン管理、共有、およびクラウド テンプレートのバージョンのリリース/リリース解除を行います。このロールには、クラウド テンプレートを削除する権限はありません。
	クラウド テンプレートの展開	任意のプロジェクトで任意のクラウド テンプレートをテストおよび展開します。
	インライン クラウド テンプレート コンテンツの展開	担当者が関連付けられているプロジェクトで任意のクラウド テンプレートを展開します。このプロジェクト ロールには、管理者、メンバー、または閲覧者を指定できます。
[XaaS]		
	カスタム リソースの表示	カスタムリソースを表示します。
	カスタムリソースの管理	カスタム リソースの作成、更新、または削除
	リソース アクションの編集	カスタム アクションを表示します。
	リソース アクションの管理	カスタム アクションの作成、更新、または削除
[拡張性]		
	拡張性リソースの表示	イベント、サブスクリプション、イベント トピック、アクション、ワークフロー、アクションの実行、およびワークフローの実行を表示します。
	拡張性リソースの管理	拡張性サブスクリプションを作成、更新、削除、および無効にします。 拡張性アクションを作成、更新、または削除します。拡張性アクションの実行をキャンセルまたは削除します。
[パイプライン]		
	パイプラインの管理	パイプライン、エンドポイント、変数、トリガの設定を作成、編集、および削除します。 制限付きモデルは除外されます。
	制限付きパイプラインの管理	パイプライン、エンドポイント、変数、トリガの設定を作成、編集、および削除します。 制限付きモデルは含まれます。
	カスタム統合の管理	カスタム統合を追加、編集、および削除します。
	パイプラインの実行	パイプライン モデルの実行とトリガを実行したり、これらを一時停止、キャンセル、再開、または再実行したりします。

表 3-7. カスタム ロール （続き）

ユーザー インターフェイス	権限	説明
	制限付きパイプラインの実行	パイプライン モデルの実行とトリガを実行したり、これらを一時停止、キャンセル、再開、または再実行したりします。 制限付きエンドポイントと変数を解決します。
	実行の管理	パイプライン モデルの実行とトリガを実行したり、これらを一時停止、キャンセル、再開、または再実行したりします。 制限付きエンドポイントと変数を解決します。 実行を削除します。
[承認]		
	承認の管理	承認申請を承認または却下できる [承認] タブを表示します。 このロールを持つ承認者は、ポリシーで承認者になっていない限り、承認要求に関する E メール通知を受信しません。

使用事例：ユーザー ロールによって vRealize Automation でのアクセスを制御する方法

クラウド管理者として、ユーザーが vRealize Automation で実行できるタスクを制御するとします。管理目標とアプリケーション開発チームの責任に応じて、その目標に適したユーザー ロールを設定する方法は異なります。

次の vRealize Automation Cloud Assembly と vRealize Automation Service Broker の例は、3 つの使用事例に基づいています。これらの例は、ユーザー ロールの利用方法を示すために十分な指示のみを提供しています。

これらの使用事例の対象者は、クラウド管理者（クラウド管理者とも見なされる）とサービス管理者です。

使用事例は、互いに関連付けて利用するように組み立てられています。使用事例 3 に直接進む準備ができている場合は、指定された方法でロールを設定する理由について十分に理解するために使用事例 1 と 2 を確認することをお勧めします。

使用事例の目的はユーザー ロールのデモであり、インフラストラクチャの設定、プロジェクトの管理、クラウド テンプレートの作成、展開の操作に関して詳細な情報を提供することではありません。

開始する前に、vRealize Automation コンソールでクラウド管理者が設定したユーザー ロールのレベルを理解しておく必要があります。

■ 組織ロール

組織ロールは、コンソールにアクセスできるユーザーを制御します。

組織の所有者は、すべてのサービスのすべてのユーザーに、少なくとも組織メンバー ロールを確実に割り当てる必要があります。

ロール	説明
組織の所有者	管理者は、ユーザーの追加、ユーザーのロールの変更、組織からのユーザーの削除を実行できます。所有者は、どのユーザーがサービスにアクセスできるかを管理します。
組織のメンバー	一般的なユーザーは、組織コンソールにログインできます。サービスにアクセスするには、組織の所有者がユーザーにサービス ロールを割り当てる必要があります。

■ サービス ロール

サービス ロールは、割り当てられたサービスにアクセスできるユーザーを制御します。

組織の所有者は、サービスにアクセスする必要があるユーザーに適切なロールが確実に割り当てられるようにする必要があります。ロールを使用して、各サービスでユーザーがどのレベルの操作を実行できるか制御します。

表 3-8. vRealize Automation Cloud Assembly サービス ロールの説明

ロール	説明
Cloud Assembly 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権を持っているユーザー。これは、クラウドアカウントの追加、新しいプロジェクトの作成、プロジェクト管理者の割り当てなど、すべてを表示および操作できる唯一のユーザー ロールです。
Cloud Assembly ユーザー	Cloud Assembly 管理者ロールを持たないユーザー。 vRealize Automation Cloud Assembly プロジェクトでは、管理者がユーザーをプロジェクト メンバー、管理者、または閲覧者としてプロジェクトに追加します。管理者は、プロジェクト管理者を追加することもできます。
Cloud Assembly 閲覧者	情報を表示するための読み取りアクセス権は持っているが、作成、更新、削除はできないユーザー。これは、すべてのプロジェクトに対する読み取り専用のロールです。 閲覧者ロールを持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクト メンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバー ロールとは異なり、権限が拡張されることはありません。

表 3-9. Service Broker サービス ロールの説明

ロール	説明
Service Broker 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権が必要です。これは、新しいプロジェクトの作成やプロジェクト管理者の割り当てなど、すべてのタスクを実行できる唯一のユーザー ロールです。
Service Broker ユーザー	vRealize Automation Service Broker 管理者ロールを持たないすべてのユーザー。 vRealize Automation Service Broker プロジェクトでは、管理者がユーザーをプロジェクト メンバー、管理者、または閲覧者としてプロジェクトに追加します。管理者は、プロジェクト管理者を追加することもできます。
Service Broker 閲覧者	情報を表示するための読み取りアクセス権は持っているが、作成、更新、削除はできないユーザー。 閲覧者ロールを持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクト メンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバー ロールとは異なり、権限が拡張されることはありません。

表 3-10. Code Stream サービス ロールの説明

ロール	説明
Code Stream 管理者	ユーザー インターフェイスと API リソース全体に対する読み取りおよび書き込みアクセス権を持っているユーザー。これは、すべてを表示および実行できる唯一のユーザー ロールであり、具体的には、プロジェクトの作成、エンドポイントの統合、トリガの追加、パイプラインとカスタム ダッシュボードの作成、エンドポイントおよび変数に対する制限付きリソースとしてのマーキング、制限付きリソースを使用するパイプラインの実行、パイプラインを vRealize Automation Service Broker で公開する申請が可能です。
Code Stream 開発者	パイプラインを使用できるが制限付きのエンドポイントまたは変数は使用できないユーザー。パイプラインに制限付きのエンドポイントまたは変数が含まれている場合、このユーザーは、制限付きのエンドポイントまたは変数を使用するパイプライン タスクに関する承認を得る必要があります。
Code Stream 実行者	パイプラインを実行でき、ユーザー操作タスクの承認または拒否を行うことができるユーザー。このユーザーは、パイプラインの実行を再開、一時停止、およびキャンセルできますが、パイプラインを変更することはできません。
Code Stream ユーザー	vRealize Automation Code Stream にアクセスできるが、vRealize Automation Code Stream 内のそれ以外の権限を持たないユーザー。
Code Stream ビューア	パイプライン、エンドポイント、パイプラインの実行、ダッシュボードを表示するための読み取りアクセス権を持ち、作成、更新、削除することはできないユーザー。サービス閲覧者ロールも持つユーザーは、管理者が使用できるすべての情報を表示できます。これらのユーザーは、プロジェクト管理者またはプロジェクト メンバーにされない限り、アクションを実行することはできません。プロジェクトに関連しているユーザーは、そのロールに関連する権限を持ちます。プロジェクト閲覧者は、管理者ロールまたはメンバー ロールとは異なり、権限が拡張されることはありません。

■ プロジェクト メンバーシップのロール

プロジェクト メンバーシップによって、使用可能なインフラストラクチャ リソースとクラウド テンプレートが決まります。

プロジェクト メンバーシップは、サービス管理者ロールを持つユーザーによってサービス内で定義されます。サービス管理者は、1 つまたは複数のプロジェクトへのアクセスを必要とするユーザーに、各プロジェクトで適切なプロジェクト ロールが確実に割り当てられるようにする必要があります。

表 3-11. プロジェクト ロール

ロール	説明
プロジェクト管理者	プロジェクト管理者は、自分のプロジェクトの管理、プロジェクトに関連付けられているクラウド テンプレートの作成と展開、およびすべてのプロジェクト メンバーのプロジェクト展開の管理を実行できます。
プロジェクト メンバー	プロジェクト メンバーは、プロジェクトに関連付けられているクラウド テンプレートの作成と展開、自分の展開の管理、およびすべての共有展開の管理を実行できます。
プロジェクト閲覧者	プロジェクト閲覧者は、プロジェクトのリソース、クラウド テンプレート、および展開に対する読み取り専用アクセス権を持つプロジェクト メンバーです。

■ カスタム ロール

vRealize Automation Cloud Assembly によってカスタム ロールを作成すると、メンバーおよび閲覧者のロールを細かく調整することができます。

これらの使用事例で示されている手順は、ユーザー ロールに着目することを目的としています。vRealize Automation を設定するための詳細で厳密な手順ではありません。

ロールを構成する際は、API 操作を実行しているユーザーにもここで割り当てるロールが適用されることに注意してください。

前提条件

- 組織の所有者ロールが割り当てられていることを確認します。コンソールにログインすると、[ID とアクセスの管理] タブが表示されます。表示されない場合は、組織の所有者にお問い合わせください。
-
- ユーザーが vRealize Automation に追加されていることを確認します。
vRealize Automation をインストールすると、Active Directory ユーザーがプロセスの一部として追加されます。
- さまざまなロールの詳細なタスクとロール リストについては、[vRealize Automation の組織およびサービスのユーザー ロール](#)を参照してください。

手順

1 ユーザー ロールの使用事例 1：小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定

vRealize Automation クラウド管理者は、インフラストラクチャ リソースへのアクセスと予算を管理する責任があります。自分自身、および 2 人のユーザーを管理者として追加します。この小規模なチームでは、インフラストラクチャを作成し、クラウドテンプレートを使用するチームのビジネス目標に一致するクラウドテンプレートを作成できます。自分と小規模な管理者チームによって、管理者以外の利用者のためにクラウドテンプレートを展開します。管理者以外に vRealize Automation へのアクセスを許可しません。

2 ユーザー ロールの使用事例 2：大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定

vRealize Automation 組織の所有者は、インフラストラクチャ リソースへのアクセスと予算を管理する責任があります。利用者に提供できるようになるまで、各プロジェクトにテンプレートを作成および展開する、クラウドテンプレート開発者のチームが必要になります。次に、展開可能なリソースをカタログ内の利用者に提供します。

3 ユーザー ロールの使用事例 3：システム ロールを調整するための vRealize Automation カスタム ユーザー ロールの設定

vRealize Automation 組織の所有者またはサービス管理者は、組織およびサービスのシステム ロールを使用してユーザー アクセスを管理します。ただし、システム ロールでは許可されていないタスクの実行やコンテンツの表示が可能になるカスタム ロールを、一部のユーザーに対して作成することが必要な場合もあります。

ユーザー ロールの使用事例 1：小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定

vRealize Automation クラウド管理者は、インフラストラクチャ リソースへのアクセスと予算を管理する責任があります。自分自身、および 2 人のユーザーを管理者として追加します。この小規模なチームでは、インフラストラクチャを作成し、クラウド テンプレートを使用するチームのビジネス目標に一致するクラウド テンプレートを作成できます。自分と小規模な管理者チームによって、管理者以外の利用者のためにクラウド テンプレートを展開します。管理者以外に vRealize Automation へのアクセスを許可しません。

この使用事例では、組織の所有者として、全員がサービス管理者ロールを持っている小規模なチームを所有しているとします。

次の手順では、1 人のユーザーについてのプロセス全体を追っていきます。各ステップは、複数のユーザーに対して実行できます。

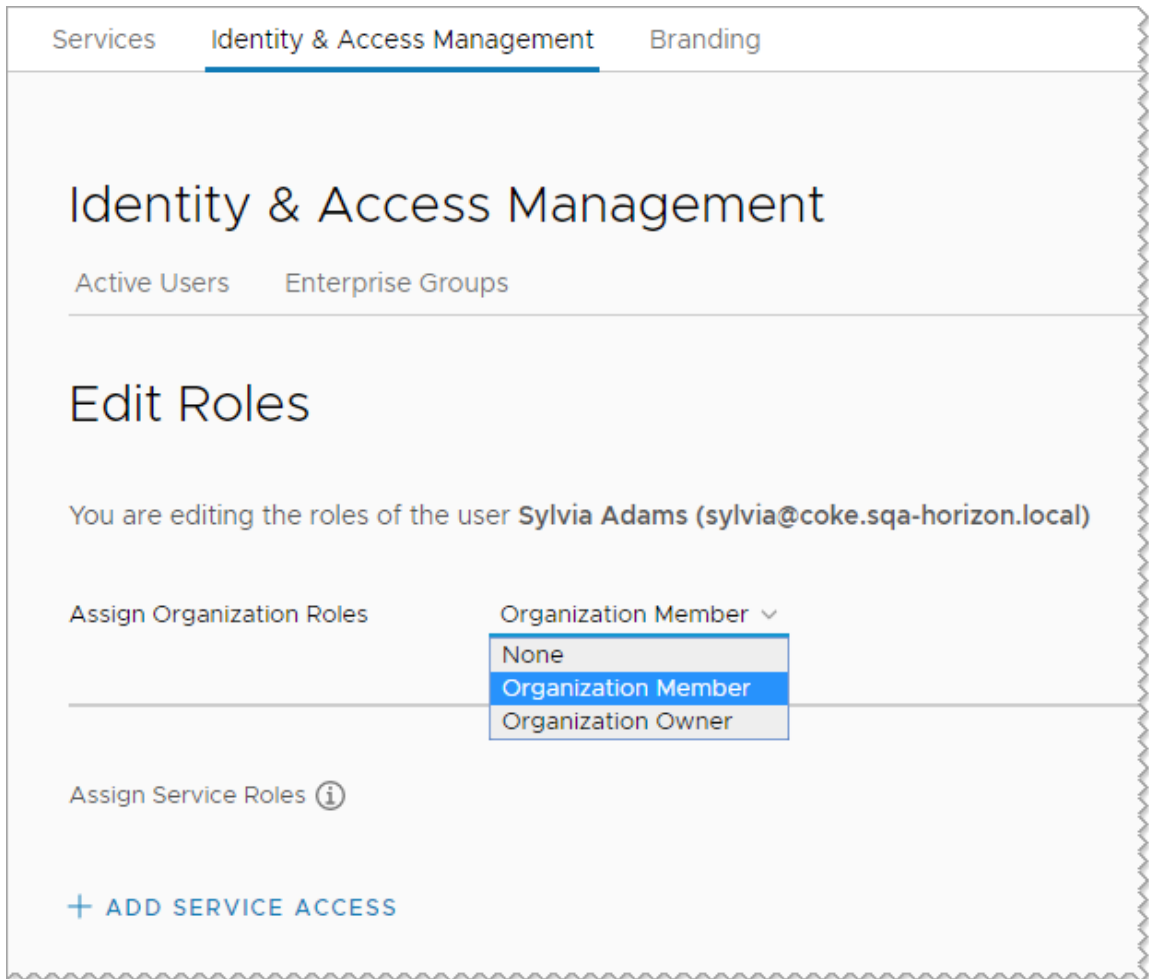
前提条件

- 使用事例の概要で規定されているすべての前提条件を満たしていることを確認します。[使用事例：ユーザー ロールによって vRealize Automation でのアクセスを制御する方法](#)を参照してください。

手順

- 1 組織ロールを割り当てます。[ID およびアクセス権の管理] をクリックします。
 - a vRealize Automation コンソールにログインします。
 - b [ID およびアクセス権の管理] をクリックします。

- c ユーザー名を選択し、[ロールの編集] をクリックします。
- d [組織ロールの割り当て] ドロップダウン メニューで、[組織メンバー] を選択します。



組織のメンバー ロールにより、ユーザーはコンソールおよびコンソールに追加したサービスにアクセスできるようになります。組織のユーザーを管理することはできません。

このユーザーの [ロールの編集] 画面を開いたまま、次の手順に進みます。

- 2 このシナリオでは、自分と他の 1 人または 2 人の管理者に Cloud Assembly 管理者ロールを割り当てます。

サービス管理者ロールには、インフラストラクチャ、プロジェクト、クラウド テンプレート、展開を追加、編集、削除できる完全な権限があります。1 人のユーザーに管理者ロールを、別のユーザーにユーザー ロールを定義する方法については、シナリオ 2 で説明します。この例では、Sylvia を使用します。

- a [サービスへのアクセス権の追加] をクリックします。
- b 次の値を使用してユーザーを構成します。

サービス	ロール
vRealize Automation Cloud Assembly	vRealize Automation Cloud Assembly 管理者

[Services](#)
[Identity & Access Management](#)
[Branding](#)

Identity & Access Management

[Active Users](#)
[Enterprise Groups](#)

Edit Roles

You are editing the roles of the user **Sylvia Adams** (sylvia@coke.sqa-horizon.local)

Assign Organization Roles Organization Member

Assign Service Roles ⓘ

Cloud Assembly
with roles
Cloud Assembly Administrator
×

[+ ADD SERVICE ACCESS](#)

SAVE
CANCEL

- 3 Cloud Assembly にプロジェクトを作成して、リソースのグループ化や、別のビジネス グループに対するリソースの請求の管理に使用します。

- a コンソールで、[サービス] タブをクリックし、[Cloud Assembly] をクリックします。
- b [インフラストラクチャ] - [プロジェクト] - [新しいプロジェクト] の順に選択します。

このユーザー ロールの使用事例は、すべてが定義されたシステムを作成することではなく、ユーザー ロールを実装する方法の例を示すことに重点を置いています。

インフラストラクチャの設定の詳細については、[リソース インフラストラクチャの構築](#)を参照してください。プロジェクトの詳細については、[プロジェクトの追加および管理](#)を参照してください。

- c プロジェクト名として **WebAppTeam** と入力します。
- d [ユーザー] をクリックし、[ユーザーの追加] をクリックします。

- e インフラストラクチャとクラウド テンプレートの構築と管理に協力する個人のメール アドレスを入力します。

たとえば、「tony@mycompany.com,syliva@mycompany.com」のように入力します。

- f [ロールの割り当て] ドロップダウン メニューで、[管理者] を選択します。

vRealize Automation Cloud Assembly 管理者であるこの 2 人のユーザーには、クラウド アカウント、インフラストラクチャ、およびすべてのプロジェクトへの管理者アクセス権がすでに付与されています。この手順は、以降のシナリオで使用されるロールの理解に役立ちます。以降のシナリオでは、さまざまな権限を持つプロジェクト管理者とプロジェクト メンバーのロールを定義します。

- g [プロビジョニング] タブをクリックし、1 つ以上のクラウド ゾーンを追加します。

次の点にも注意してください。この使用事例では、ユーザー ロールについて説明しています。

4 WebAppTeam プロジェクトをテストできるように、シンプルなクラウド テンプレートを作成します。

このクラウド テンプレート セクションは、一部のみを示しています。ここで重視しているのはプロジェクトで定義されるユーザーとユーザー ロールであり、クラウド テンプレートの作成方法ではありません。

- a [クラウド テンプレート] - [新規] を選択します。
- b 新しいクラウド テンプレートの名前として **WebApp** と入力します。
- c [プロジェクト] では、WebAppTeam を選択します。

New Cloud Template

Name * WebApp

Description

Project * WebAppTeam

Cloud template sharing in Service Broker

☒ Share only with this project

☐ Allow an administrator to share with any project in this organization

CANCEL CREATE

- d [プロジェクトとのみ共有] を選択します。

この設定により、クラウド テンプレートはプロジェクト メンバーのみが使用できるようになります。クラウド テンプレートを他のチームに提供する準備ができたなら、[管理者にこの組織の任意のプロジェクトとの共有を許可する] を選択できます。クラウド テンプレートを他のプロジェクトと共有すると、同じ基本テンプレートのインスタンスを重複して維持する必要がなくなります。開発プロジェクトから本番環境のプロジェクトにクラウド テンプレートを移動すると、カタログ利用者は本番環境のインフラストラクチャ リソースに展開できるようになります。

- e [作成] をクリックします。

- f クラウド テンプレート デザイナで、[クラウドに依存しない] - [マシン] コンポーネントをキャンバスにドラッグします。

クラウド テンプレートの設定の詳細については、[展開の設計](#)を参照してください。

- g [[展開]] をクリックします。
- h 利用者に提供する準備が整うまで、クラウド テンプレートに対する作業を繰り返します。
- i [バージョン] をクリックし、クラウド テンプレートをリリースしてバージョンを決定します。

5 通常使用している方法で、ログイン情報をユーザーに送信します。

結果

この使用事例では、2 人の同僚を組織のメンバーにしました。次に、Sylvia を vRealize Automation Cloud Assembly 管理者にしました。Tony を WebApp プロジェクト管理者にしました。このユーザー ロール設定は、展開するアプリケーションをコンシューマにセルフサービス アクセスやカタログで提供するのではなく、小規模なチームが自身で提供する場合にのみ機能します。

ユーザー ロールの使用事例 2: 大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定

vRealize Automation 組織の所有者は、インフラストラクチャ リソースへのアクセスと予算を管理する責任があります。利用者に提供できるようになるまで、各プロジェクトにテンプレートを作成および展開する、クラウド テンプレート開発者のチームが必要になります。次に、展開可能なリソースをカタログ内の利用者に提供します。

この使用事例では、使用事例 1 が管理者のみを対象にした使用事例であることを理解しておく必要があります。さらに多くのチームをサポートし、より大きな目標を達成できるように、システムを拡張する必要があります。

- 開発時に、開発者が独自のアプリケーション クラウド テンプレートを作成して展開できるようにします。自分自身を管理者として追加し、サービス ユーザーとサービス閲覧者の両方のロールを持つユーザーを追加します。次に、ユーザーをプロジェクト メンバーとして追加します。このプロジェクト メンバーは、独自のクラウド テンプレートを開発して展開できます。
- クラウド テンプレートをカタログに公開し、開発者以外がクラウド テンプレートを展開できるようにします。ここで、Service Broker 用のユーザー ロールを割り当てます。Service Broker は、クラウド テンプレートの利用者向けのカタログを提供します。この機能を使用してリースや資格を含むポリシーを作成することもできますが、この機能は、ユーザー ロールに関するこの使用事例には含まれません。

前提条件

- 最初の使用事例を確認します。[ユーザー ロールの使用事例 1: 小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- 付与する権限に応じて、次のユーザーを特定します。
 - vRealize Automation Cloud Assembly ユーザーおよび閲覧者になるクラウド テンプレート開発者
 - vRealize Automation Service Broker 管理者
 - vRealize Automation Service Broker ユーザーとしてカタログ利用者になる開発者以外のユーザー

手順

- 1 組織のメンバーのロールをクラウド テンプレート開発者ユーザーに割り当てます。

指示が必要な場合は、[ユーザー ロールの使用事例 1: 小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。

2 vRealize Automation Cloud Assembly サービス メンバー ロールをクラウド テンプレート開発者に割り当てます。

- a [サービスへのアクセス権の追加] をクリックします。

- b 次の値を使用してユーザーを構成します。

サービス	ロール
vRealize Automation Cloud Assembly	vRealize Automation Cloud Assembly ユーザー
vRealize Automation Cloud Assembly	vRealize Automation Cloud Assembly 閲覧者

この使用事例では、展開可能なクラウド テンプレートを確実に作成できるように、開発者がインフラストラクチャを確認する必要があります。次のステップでプロジェクト管理者として、およびプロジェクト メンバーとして割り当てるユーザーは、インフラストラクチャを表示できません。サービス閲覧者は、インフラストラクチャがどのように設定されているかを確認できますが、変更を加えることはできません。クラウド管理者は、自分で管理を続けつつ、開発者がクラウド テンプレートの開発に必要とする情報にアクセスできるようにします。

3 リソース ユーザーをグループ化するために使用するプロジェクトを、vRealize Automation Cloud Assembly 内に作成します。

この使用事例では、2 つのプロジェクトを作成します。1 つ目のプロジェクトは PersonnelAppDev で、2 つ目は PayrollAppDev です。

- コンソールで、[サービス] タブをクリックし、[Cloud Assembly] をクリックします。
- [インフラストラクチャ] - [プロジェクト] - [新しいプロジェクト] の順に選択します。
- 名前として **PersonnelAppDev** と入力します。
- [ユーザー] をクリックし、[ユーザーの追加] をクリックします。

- e プロジェクト メンバーを追加し、プロジェクト管理者を割り当てます。

プロジェクト ロール	説明
プロジェクト ユーザー	プロジェクト メンバーは、プロジェクトの主要な開発者ユーザーロールです。クラウド テンプレートを展開することによって開発作業をテストする準備ができたときに使用可能なクラウド リソースは、プロジェクトによって決定されます。
プロジェクト管理者	プロジェクト管理者は、プロジェクトのユーザーを追加および削除することで、開発者をサポートします。プロジェクトを削除することもできます。プロジェクトを作成するには、サービス管理者権限が必要です。

- f プロジェクト メンバーとして追加するユーザーについては、各ユーザーのメール アドレスをコンマで区切って入力し、[ロールの割り当て] ドロップダウン メニューで [ユーザー] を選択します。

たとえば、「tony@mycompany.com,sylvia@mycompany.com」のように入力します。

PersonnelAppDev DELETE

Summary Users Provisioning Kubernetes Provisioning Integrations

Deployment sharing ☒ Deployments are shared between all users in the project

User roles Specify the users and groups related to this project.

+ ADD USERS + ADD GROUPS X REMOVE

Q Search users or groups

<input type="checkbox"/>	Name	Account	Role
<input type="checkbox"/>	Sylvia Adams	sylvia	Administrator
<input type="checkbox"/>	Gloria Martinez	gloria	Member
<input type="checkbox"/>	Tony Anteater	tony	Member

1 - 3 of 3 users

SAVE CANCEL

- g 指定された管理者について、[ロールの割り当て] ドロップダウン メニューで [管理者] を選択し、必要なメール アドレスを入力します。
- h [プロビジョニング] タブをクリックし、1つ以上のクラウド ゾーンを追加します。
- このプロジェクトに参加しているクラウド テンプレート開発者がテンプレートを展開すると、クラウド ゾーンで使用可能なリソースにテンプレートが展開されます。クラウド ゾーンのリソースが、プロジェクト開発チームのテンプレートのニーズと一致していることを確認する必要があります。
- i このプロセスを繰り返して、必要なユーザーと管理者と共に PayrollAppDev プロジェクトを追加します。
- 4 必要なログイン情報をサービス ユーザーに提供し、各プロジェクトのメンバーが以下のタスクを実行できることを確認します。
- a vRealize Automation Cloud Assembly を開きます。
- b すべてのプロジェクトのインフラストラクチャを確認します。
- c ユーザーがメンバーになっているプロジェクトのクラウド テンプレートを作成します。

- d プロジェクトで定義されているクラウド ゾーンのリソースにクラウド テンプレートを展開します。
- e 展開を管理します。

5 組織のメンバーのロールをクラウド テンプレート開発者ユーザーに割り当てます。

指示が必要な場合は、[ユーザー ロールの使用事例 1：小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。

6 カタログ管理者、カタログ利用者、およびクラウド テンプレート開発者に、それぞれの職務に基づいてロールを割り当てます。

- a [サービスへのアクセス権の追加] をクリックします。
- b 次の値を使用して、カタログ管理者を構成します。

これは、自分自身（管理者）、クラウド管理者、またはアプリケーション開発チームのユーザーでも実行可能です。

サービス	ロール
vRealize Automation Service Broker	vRealize Automation Service Broker 管理者

- c 次の値を使用して、クラウド テンプレートの利用者を構成します。

サービス	ロール
vRealize Automation Service Broker	vRealize Automation Service Broker ユーザー

Identity & Access Management

Active Users Enterprise Groups

Edit Roles

You are editing the roles of the user **Gloria Martinez** (gloria@coke.sqa-horizon.local)

Assign Organization Roles Organization Member ▼

Assign Service Roles ⓘ

Service Broker ▼ with roles Service Broker User ▼ ×

[+ ADD SERVICE ACCESS](#)

- d 次の値を使用して、クラウド テンプレート開発者を構成します。

サービス	ロール
Cloud Assembly vRealize Automation Cloud Assembly	vRealize Automation Cloud Assembly ユーザー

- 7 リソースとユーザーをグループ化するために使用するプロジェクトを、vRealize Automation Cloud Assembly 内に作成します。

この使用事例では、2 つのプロジェクトを作成します。1 つ目のプロジェクトは PersonnelAppDev で、2 つ目は PayrollAppDev です。

指示が必要な場合は、[ユーザー ロールの使用事例 2：大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。

- 8 プロジェクト チームごとにクラウド テンプレートを作成してリリースします。

指示が必要な場合は、[ユーザー ロールの使用事例 1：小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。

- 9 vRealize Automation Cloud Assembly クラウド テンプレートを vRealize Automation Service Broker にインポートします。

vRealize Automation Service Broker 管理者ロールを持つユーザーでログインする必要があります。

- vRealize Automation Service Broker 管理者ロールを持つユーザーでログインします。
- コンソールで、vRealize Automation Service Broker をクリックします。
- [コンテンツとポリシー] - [コンテンツ ソース] の順に選択し、[新規] をクリックします。

- [Cloud Assembly クラウド テンプレート] を選択します。
- 名前に **PersonnelAppImport** と入力します。
- [ソース プロジェクト] のドロップダウン メニューで、PersonnelAppDev を選択し、[検証] をクリックします。
- ソースが検証されたら、[作成してインポート] をクリックします。
- PayrollAppDev には、コンテンツ ソース名に PayrollAppImport を使用して繰り返します。

10 インポートしたクラウド テンプレートをプロジェクトと共有します。

クラウド テンプレートはすでにプロジェクトに関連付けられていますが、カタログで使用できるようにするために vRealize Automation Service Broker で共有します。

- a vRealize Automation Service Broker 管理者ロールを持つユーザーとして続行します。
- b vRealize Automation Service Broker で、[コンテンツとポリシー] - [コンテンツ共有] の順に選択します。
- c カタログからクラウド テンプレートを展開できる必要があるユーザーを含んでいる、**PersonnelAppDev** プロジェクトを選択します。
- d [アイテムの追加] をクリックしてから、プロジェクト メンバーと共有する PersonnelApp クラウド テンプレートを選択します。

Share Items with PersonnelAppDev
×

Select the templates to share with the project members. ⓘ

CONTENT SOURCES ▾

Filter... ↻

<input checked="" type="checkbox"/>	Items Shared with this Project	Description
<input checked="" type="checkbox"/>	<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">▾</div> <div>PersonnelAppImport</div> </div> <div style="margin-left: 20px;">WebApp for Personnel</div>	

☒ 1

1 item(s)

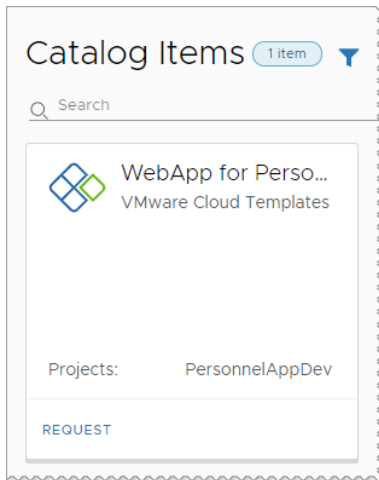
CANCEL

SAVE

- e [保存] をクリックします。

- 11 プロジェクト メンバーが vRealize Automation Service Broker カタログからクラウド テンプレートを使用できることを確認します。

- a プロジェクト メンバーに、ログインして [カタログ] タブをクリックするように要求します。

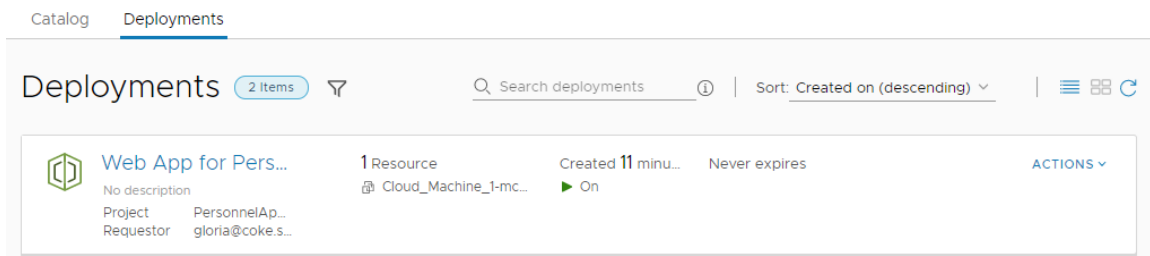


- b PersonnelApp クラウド テンプレート カードの [申請] をクリックします。

- c フォームに入力し、[送信] をクリックします。

- 12 プロジェクト メンバーが展開プロセスを監視できることを確認します。

- a プロジェクト メンバーに、[展開] タブをクリックして、プロビジョニング要求を特定するように要求します。



- b クラウド テンプレートが展開されたら、要求しているユーザーがアプリケーションにアクセスできることを確認します。

- 13 他のプロジェクトについても、このプロセスを繰り返します。

結果

この使用事例では、クラウド テンプレートの開発を開発者に委任する必要があることを踏まえて、さらに組織のメンバーを追加します。このメンバーは、vRealize Automation Cloud Assembly ユーザーにしました。次に、関連するプロジェクトのメンバーとして設定して、クラウド テンプレートを作成して展開できるようにしました。プロジェクト メンバーは管理対象のインフラストラクチャの表示や変更はできませんが、彼らにサービス閲覧者の完全な権限を付与して、設計するインフラストラクチャの制約を理解できるようにしました。

この使用事例では、vRealize Automation Service Broker 管理者やユーザーなど、さまざまなロールを持つユーザーを構成します。その後、開発者以外のユーザーに vRealize Automation Service Broker カタログを提供します。

次のステップ

カスタム ロールを定義してユーザーに割り当てる方法については、[ユーザー ロールの使用事例 3：システム ロールを調整するための vRealize Automation カスタム ユーザー ロールの設定](#)を参照してください。

ユーザー ロールの使用事例 3：システム ロールを調整するための vRealize Automation カスタム ユーザー ロールの設定

vRealize Automation 組織の所有者またはサービス管理者は、組織およびサービスのシステム ロールを使用してユーザー アクセスを管理します。ただし、システム ロールでは許可されていないタスクの実行やコンテンツの表示が可能になるカスタム ロールを、一部のユーザーに対して作成することが必要な場合もあります。

このシナリオでは、サービス ユーザーと閲覧者、および使用事例 2 で定義されているプロジェクト メンバーと閲覧者のロールについて理解していることが前提となります。これらのロールは、使用事例 1 で使用されているサービスおよびプロジェクト管理者のロールよりも制約が多いことがわかります。これで、ローカルな使用事例をいくつか確認しました。これらの使用事例では、一部のユーザーに対して、ある機能については完全な管理権限を付与し、別の機能については表示権限を付与し、それ以外の機能セットについては表示を禁止するように設定します。これらの権限を定義するには、カスタム ロールを使用します。

この使用事例は、使用可能 3 つのローカルな使用事例に基づいています。この手順では、次のカスタム ロールの権限の作成方法について説明します。

- 制限付きインフラストラクチャ管理者。サービス管理者以外の一部のサービス ユーザーに、より広範なインフラストラクチャ権限を付与する必要があります。管理者として、これらのユーザーが、クラウド ゾーン、イメージ、およびフレーバーの設定をサポートできるようにする必要があります。また、これらのユーザーが、検出されたリソースのオンボーディングや管理を実行できるようにする必要があります。これらのユーザーは、クラウド アカウントまたは統合を追加できないことに注意してください。これらのエンドポイントのインフラストラクチャの定義のみが実行可能です。
- 拡張性開発者。一部のサービス ユーザーに、プロジェクト チームやその他のプロジェクトのクラウド テンプレート開発の一環として、拡張性アクションとサブスクリプションを使用するための完全な権限を付与する必要があります。これらのユーザーは、複数のプロジェクト用のカスタム リソース タイプとカスタム アクションも作成します。
- XaaS 開発者。一部のサービス ユーザーには、複数のプロジェクトでカスタム リソース タイプとカスタム アクションの作成ができるように、フル権限を付与する必要があります。
- 展開のトラブルシューティング担当者。プロジェクト管理者に、失敗した展開に対してトラブルシューティングと根本原因分析を実行するために必要な権限を付与する必要があります。イメージやフレーバーのマッピングなど、損害につながりにくいカテゴリや、重要性の低いカテゴリに関する管理権限を付与します。また、失敗した展開のトラブルシューティング担当者ロールの一部として、承認および Day 2 ポリシーを設定する権限も付与する必要があります。

前提条件

- [vRealize Automation ユーザー ロール](#)について示されている、vRealize Automation Cloud Assembly と vRealize Automation Service Broker のサービス ロールおよびプロジェクト ロールに関する表を確認してください。各サービス ユーザー ロールがこれらのサービス内で表示できる内容と実行できる作業について理解しておく必要があります。
- [vRealize Automationのカスタム ユーザー ロール](#)の説明を参照して、ユーザーの権限を変更する方法について確認します。
- 最初の使用事例を確認して、組織のロールとサービス管理者のロールについて確認します。[ユーザー ロールの使用事例 1: 小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- サービス ユーザーおよびプロジェクト メンバーのロールを理解するために 2 番目の使用事例を確認します。[ユーザー ロールの使用事例 2 : 大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- vRealize Automation Service Broker に慣れます。[カタログへのコンテンツの追加](#)を参照してください。

手順

- 1 組織のメンバーのロールをクラウド テンプレート開発者ユーザーに割り当てます。
指示が必要な場合は、[ユーザー ロールの使用事例 1 : 小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- 2 クラウド テンプレート開発者およびカタログ利用者に vRealize Automation Cloud Assembly と vRealize Automation Service Broker のサービス ロールを割り当てます。
指示が必要な場合は、[ユーザー ロールの使用事例 2 : 大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- 3 リソースとユーザーをグループ化するために使用するプロジェクトを、vRealize Automation Cloud Assembly 内に作成します。
カスタム ロールに関する次の手順には、プロジェクト ロールも含まれています。
プロジェクトの作成についての指示が必要な場合は、[ユーザー ロールの使用事例 2 : 大規模な展開チームとカタログをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- 4 プロジェクト チームごとにクラウド テンプレートを作成してリリースします。
指示が必要な場合は、[ユーザー ロールの使用事例 1 : 小規模なアプリケーション開発チームをサポートするための vRealize Automation ユーザー ロールの設定](#)を参照してください。
- 5 サービス管理者として vRealize Automation Cloud Assembly にログインし、[インフラストラクチャ] - [管理] - [カスタム ロール] の順に選択します。

6 制限付きインフラストラクチャ管理者ロールを作成します。

この例では、さまざまなプロジェクトのインフラストラクチャの設定に熟練したユーザー Tony がいますが、Tony には完全なサービス権限を付与しません。Tony はすべてのプロジェクトの作業をサポートするコア インフラストラクチャを構築します。Tony には制限付きのインフラストラクチャ管理権限を付与します。Tony、または外部請負業者は、検出されたマシンをオンボーディングしたり、これらを vRealize Automation の管理対象に組み入れたりするために、同様な権限を持つことがあります。

- a Tony をサービス ユーザーおよび閲覧者として vRealize Automation Cloud Assembly に追加します。

閲覧者の権限を持っている Tony は、作業のトラブルシューティングでの必要性に応じて、基盤となるクラウド アカウントと統合を確認することができますが、変更を加えることはできません。

- b プロジェクトを作成し、Tony をプロジェクト メンバーとして追加します。
- c カスタム ロールを作成するには、[インフラストラクチャ] - [管理] - [カスタム ロール] の順に選択し、[新しいカスタム ロール] をクリックします。
- d 名前として「**制限付きインフラストラクチャ管理者**」を入力し、次の権限を選択します。

選択する権限	ユーザーが実行できる操作
インフラストラクチャ > クラウド ゾーンの管理	クラウド ゾーンを作成、更新、削除します。
インフラストラクチャ > フレーバー マッピングの管理	フレーバー マッピングを作成、更新、削除します。
インフラストラクチャ > イメージ マッピングの管理	イメージ マッピングを作成、更新、削除します。

- e [作成] をクリックします。
- f [カスタム ロール] 画面で、制限付きインフラストラクチャ管理者ロールを選択し、[割り当て] をクリックします。
- g Tony の E メール アカウントを入力し、[追加] をクリックします。
たとえば、Tony@yourcompany.com のように入力します。
定義済みの Active Directory ユーザー グループを入力することもできます。
- h Tony に、ログインするときに、カスタム ロールで定義されている領域内で値を追加、編集、削除できるか確認するように指示します。

7 拡張性開発者ロールを作成します。

この例では、拡張性アクションとサブスクリプションを使用して毎日の開発タスクを管理する方法に精通している複数のクラウド テンプレート開発者 (Sylvia と Igor) がいます。これらのユーザーは vRealize Orchestrator の経験もあるため、さまざまなプロジェクトにカスタム リソースおよびアクションを提供する作業を担当してもらいます。また、カスタム リソースとアクションを管理し、拡張性アクションとサブスクリプションを管理することによって拡張性を管理する追加の権限を付与します。

- a Sylvia と Igor を vRealize Automation Cloud Assembly ユーザーとして追加します。
- b これらのユーザーを、拡張性スキルで貢献しているプロジェクトのメンバーとして追加します。

- c カスタム ユーザー ロールを作成し、「**拡張性開発者**」という名前を付けて、次の権限を選択します。

選択する権限	ユーザーが実行できる操作
XaaS > カスタムリソースの管理	カスタム リソースを作成、更新、または削除します。
XaaS > リソース アクションの管理	カスタム アクションを作成、更新、または削除します。
拡張性 > 拡張性リソースの管理	拡張性アクションとサブスクリプションを作成、更新、または削除します。サブスクリプションを無効にします。アクションの実行をキャンセルして、削除します。

- d [作成] をクリックします。
- e Sylvia および Igor を拡張性開発者ロールに割り当てます。
- f Sylvia と Igor がカスタム リソースおよびアクションを管理できること、および [拡張性] タブのさまざまなオプションを管理できることを確認します。

8 展開のトラブルシューティング担当者ロールを作成します。

この例では、プロジェクト管理者がチームの展開失敗を修復できるように、プロジェクト管理者に付与する管理権限を増やします。

- a プロジェクト管理者、Shauna、Pratap、および Wei を vRealize Automation Cloud Assembly および vRealize Automation Service Broker サービス ユーザーとして追加します。
- b プロジェクト内で、これらのユーザーをプロジェクト管理者として追加します。
- c カスタム ユーザー ロールを作成し、「**開発のトラブルシューティング担当者**」という名前を付けて、次の権限を選択します。

選択する権限	ユーザーが実行できる操作
インフラストラクチャ > フレーバー マッピングの管理	フレーバー マッピングを作成、更新、削除します。
インフラストラクチャ > イメージ マッピングの管理	イメージ マッピングを作成、更新、削除します。
展開 > 展開の管理	複数のプロジェクトにまたがる展開をすべて表示し、展開と展開コンポーネントに対してすべての Day 2 アクションを実行します。
ポリシー > ポリシーの管理	ポリシー定義を作成、更新、または削除します。

- d [作成] をクリックします。
- e Shauna、Pratap、および Wei を展開のトラブルシューティング担当者ロールに割り当てます。
- f vRealize Automation Service Broker でフレーバー マッピング、イメージ マッピング、ポリシーを管理できることを確認します。

結果

この使用事例では、サービスとプロジェクト ロールを展開するカスタム ロールを含む、さまざまなロールを持つ複数のユーザーを構成します。

次のステップ

ローカルな使用事例に対処するカスタム ロールを作成します。

vRealize Automation Cloud Assembly へのクラウド アカウントの追加

クラウド アカウントは、vRealize Automation Cloud Assembly が領域またはデータセンターからデータを収集し、その領域にクラウド テンプレートを展開する際に使用する設定済みの権限です。

収集されるデータには、後でクラウド ゾーンに関連付ける領域が含まれます。

後でクラウド ゾーン、マッピング、およびプロファイルを設定するときに、関連付けられているクラウド アカウントを選択します。

クラウド管理者は、チーム メンバーが作業するプロジェクトのクラウド アカウントを作成します。ネットワークとセキュリティ、コンピューティング、ストレージ、およびタグのコンテンツなどのリソース情報は、クラウド アカウントで収集されます。

注： クラウド アカウントに、領域内にすでに展開されているマシンが関連付けられている場合は、オンボーディング プランを使用してそのマシンを vRealize Automation Cloud Assembly の管理対象にできます。[vRealize Automation Cloud Assembly でのオンボーディング プランについて](#)を参照してください。

展開で使用されているクラウド アカウントを削除すると、その展開に含まれるリソースは管理対象外になります。

vRealize Automation でクラウド アカウントを使用するために必要な認証情報

vRealize Automation でクラウド アカウントを設定して操作するには、次の認証情報があることを確認します。

必要なクラウド アカウントの認証情報

目的	必要なもの
vRealize Automation Cloud Assembly に登録してログインする	<p>VMware ID。</p> <ul style="list-style-type: none"> ■ 会社のメール アドレスを使用して、My VMware アカウントを設定します。
vRealize Automation サービスに接続する	<p>ファイアウォールを介して以下にアクセスする送信トラフィックに対して開いている HTTPS ポート 443。</p> <ul style="list-style-type: none"> ■ *.vmwareidentity.com ■ gaz.csp-vidm-prod.com ■ *.vmware.com <p>ポートとプロトコルの詳細については、VMware Ports and Protocols を参照してください。</p> <p>必要なポートおよびプロトコルの関連情報については、ポートの要件を参照してください。</p>

目的	必要なもの
Amazon Web Services (AWS) クラウド アカウントの追加	<p>読み取りおよび書き込み権限を持つパワー ユーザー アカウントを指定します。ユーザー アカウントは、AWS IAM (ID およびアクセス権の管理) システムのパワー アクセス ポリシー (PowerUserAccess) のメンバーであることが必要です。</p> <ul style="list-style-type: none"> ■ 20 桁のアクセス キー ID および対応するプライベート アクセス キー <p>外部 HTTP インターネット プロキシを使用する場合は、IPv4 用に設定する必要があります。</p> <p>vRealize Automation アクション ベースの拡張性 (ABX) と外部 IP アドレス管理の統合には、追加の権限が必要になることがあります。</p> <p>自動スケーリング機能を使用するには、次の AWS 権限が推奨されます。</p> <ul style="list-style-type: none"> ■ 自動スケーリングのアクション : <ul style="list-style-type: none"> ■ autoscaling:DescribeAutoScalingInstances ■ autoscaling:AttachInstances ■ autoscaling>DeleteLaunchConfiguration ■ autoscaling:DescribeAutoScalingGroups ■ autoscaling>CreateAutoScalingGroup ■ autoscaling:UpdateAutoScalingGroup ■ autoscaling>DeleteAutoScalingGroup ■ autoscaling:DescribeLoadBalancers ■ 自動スケーリングのリソース : <ul style="list-style-type: none"> ■ * <p>自動スケーリングのすべてのリソース権限を指定します。</p> <p>AWS の ID とアクセス権に関する一時的な制限付き権限の認証情報を AWS Security Token Service (AWS STS) の機能がサポートできるようにするには、次の権限が必要です。</p> <ul style="list-style-type: none"> ■ AWS STS リソース : <ul style="list-style-type: none"> ■ * <p>すべての STS リソース権限を付与します。</p> <p>EC2 機能を許可するには、次の AWS 権限が必要です。</p> <ul style="list-style-type: none"> ■ EC2 のアクション : <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

目的	必要なもの
	<ul style="list-style-type: none"> ■ ec2:CreateSnapshot ■ ec2:ModifyInstanceAttribute ■ ec2:DescribeRouteTables ■ ec2:DescribeInstanceType ■ ec2:DescribeInstanceTypeOfferings ■ 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>注： vRealize Automation アクション ベースの拡張性 (ABX) または外部 IP アドレス管理の統合には、SpotFleet 要求権限は必要ありません。</p>
	<ul style="list-style-type: none"> ■ EC2 のリソース： <ul style="list-style-type: none"> ■ * <p>すべての EC2 リソース権限を付与します。</p> <p>弾性ロード バランシング機能を使用するには、次の AWS 権限が必要です。</p> ■ ロード バランサのアクション： <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

目的	必要なもの
	<ul style="list-style-type: none"> ■ elasticloadbalancing:RegisterInstancesWithLoadBalancer ■ elasticloadbalancing:CreateLoadBalancerListeners ■ ロード バランサ リソース : ■ * <p>ロード バランサ リソースのすべての権限を付与します。</p> <p>次の AWS の ID およびアクセス権の管理 (IAM) 権限を有効にすることはできますが、必須ではありません。</p> <ul style="list-style-type: none"> ■ iam:SimulateCustomPolicy ■ iam:GetUser ■ iam:ListUserPolicies ■ iam:GetUserPolicy ■ iam:ListAttachedUserPolicies ■ iam:GetPolicyVersion ■ iam:ListGroupsForUser ■ iam:ListGroupPolicies ■ iam:GetGroupPolicy ■ iam:ListAttachedGroupPolicies ■ iam:ListPolicyVersions

目的	必要なもの
Microsoft Azure クラウド アカウントを追加する	<p>Microsoft Azure インスタンスを設定し、有効な Microsoft Azure サブスクリプションを取得します（サブスクリプション ID が必要となります）。</p> <p>方法：リソースにアクセスできる Azure AD アプリケーションとサービス プリンシパル をポータルで作成するの説明に従って、Active Directory アプリケーションを作成します。</p> <p>外部 HTTP インターネット プロキシを使用する場合は、IPv4 用に設定する必要があります。</p> <p>次の情報をメモしておきます。</p> <ul style="list-style-type: none"> ■ サブスクリプション ID <p>Microsoft Azure サブスクリプションへのアクセスを許可します。</p> ■ テナント ID <p>Microsoft Azure アカウントで作成した Active Directory アプリケーションの認証エンドポイント。</p> ■ クライアント アプリケーションの ID <p>Microsoft Azure 個人アカウントの Microsoft Active Directory へのアクセスを提供します。</p> ■ クライアント アプリケーションのプライベート キー <p>クライアント アプリケーション ID とペアリングするために生成された一意のプライベート キー。</p> <p>Microsoft Azure クラウド アカウントを作成および検証するには、次の権限が必要です。</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 ■ Microsoft.Network/networkSecurityGroups/write

目的	必要なもの
	<ul style="list-style-type: none"> ■ 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>Microsoft Azure をアクションベースの拡張機能とともに使用している場合は、最小限の権限に加えて、次の権限が必要です。</p> <ul style="list-style-type: none"> ■ Microsoft.Web/sites/read ■ Microsoft.Web/sites/write ■ Microsoft.Web/sites/delete ■ Microsoft.Web/sites/*/action ■ 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

目的	必要なもの
	<ul style="list-style-type: none">■ Microsoft.Web/sites/hostruntime/host/read■ Microsoft.Web/sites/functions/masterkey/read■ Microsoft.Web/apimanagementaccounts/apis/read■ Microsoft.Authorization/roleAssignments/read■ Microsoft.Authorization/roleAssignments/write■ Microsoft.Authorization/roleAssignments/delete
	拡張機能を使用して、アクションベースの拡張性を持つ Microsoft Azure を使用している場合は、次の権限も必要です。
	<ul style="list-style-type: none">■ Microsoft.Compute/virtualMachines/extensions/write■ Microsoft.Compute/virtualMachines/extensions/read■ Microsoft.Compute/virtualMachines/extensions/delete

目的	必要なもの
Google Cloud Platform (GCP) クラウド アカウントの追加	<p>Google Cloud Platform クラウド アカウントは、Google Cloud Platform コンピューティング エンジンと連携して動作します。</p> <p>Google Cloud Platform クラウド アカウントを作成および検証するには、プロジェクト管理者および所有者の認証情報が必要です。</p> <p>外部 HTTP インターネット プロキシを使用する場合は、IPv4 用に設定する必要があります。</p> <p>コンピューティング エンジン サービスを有効にする必要があります。vRealize Automation でクラウド アカウントを作成する場合は、コンピューティング エンジンが初期化されたときに作成されたサービス アカウントを使用します。</p> <p>ユーザーが実行できるアクションに応じて、次のコンピューティング エンジンの権限も必要です。</p> <ul style="list-style-type: none"> ■ roles/compute.admin <p>すべてのコンピューティング エンジン リソースに対するフル コントロールを提供します。</p> ■ roles/iam.serviceAccountUser <p>サービス アカウントとして実行するように設定された仮想マシン インスタンスを管理するユーザーに、アクセスを提供します。次のリソースおよびサービスへのアクセスを許可します。</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>イメージに対する他の権限がなくてもイメージの一覧表示と読み取りができる権限を提供します。プロジェクト レベルで compute.imageUser ロールが付与されたユーザーは、そのプロジェクト内のすべてのイメージを一覧表示できます。また、プロジェクト内のイメージに基づいてインスタンスやパーシステント ディスクなどのリソースを作成できます。</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>仮想マシン インスタンスを作成、変更、削除する権限を提供します。これには、ディスクの作成、変更、削除をする権限、およびシールドされた VMBETA 設定を設定する権限が含まれます。</p> <p>仮想マシン インスタンス(サービス アカウントとして実行されるネットワークまたはセキュリティの設定とインスタンスは除く)を管理するユーザーの場合、インスタンスを含む組織、フォルダ、またはプロジェクトに、または個々のインスタンスにこのロールを付与します。</p> <p>サービス アカウントとして実行するように設定された仮想マシン インスタンスを管理するユーザーにも、roles/iam.serviceAccountUser ロールが必要です。</p> <ul style="list-style-type: none"> ■ compute.acceleratorTypes ■ compute.addresses.get ■ compute.addresses.list ■ compute.addresses.use

目的	必要なもの
	<ul style="list-style-type: none"> ■ compute.autoscalers ■ 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.useExternallp ■ 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.useExternallp ■ compute.targetPools.get ■ compute.targetPools.list

目的	必要なもの
	<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>コンピューティング エンジン インスタンス、インスタンス グループ、ディスク、スナップショット、イメージに対するフルコントロールを提供します。また、すべてのコンピューティング エンジン ネットワーク リソースへの読み取りアクセスも提供します。</p> <hr/> <p>注： このロールをインスタンス レベルで付与されたユーザーは、新しいインスタンスを作成できなくなります。</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

目的	必要なもの
	<ul style="list-style-type: none"> ■ 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 ■ compute.targetInstances.list

目的	必要なもの
	<ul style="list-style-type: none"> ■ 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 クラウド アカウントを追加する	<p>以下の読み取りおよび書き込み権限を持つアカウントを指定します。</p> <ul style="list-style-type: none"> ■ NSX-T エンタープライズ管理者のロールとアクセス認証情報 ■ NSX-T の IP アドレスまたは FQDN <p>管理者は、このページの「vCenter ベースのクラウド アカウントでの vSphere エージェントの要件」セクションに説明されているように、vCenter Server へのアクセス権も必要です。</p>
NSX-V クラウド アカウントを追加する	<p>以下の読み取りおよび書き込み権限を持つアカウントを指定します。</p> <ul style="list-style-type: none"> ■ NSX-V エンタープライズ管理者のロールとアクセス認証情報 ■ NSX-V の IP アドレスまたは FQDN <p>管理者は、このページの「vCenter ベースのクラウド アカウントでの vSphere エージェントの要件」セクションに説明されているように、vCenter Server へのアクセス権も必要です。</p>

目的	必要なもの
vCenter クラウド アカウントを追加する	<p>以下の読み取りおよび書き込み権限を持つアカウントを指定します。</p> <ul style="list-style-type: none"> ■ vCenter の IP アドレスまたは FQDN <p>管理者は、このページの「vCenter ベースのクラウド アカウントでの vSphere エージェントの要件」セクションに説明されているように、vCenter Server へのアクセス権も必要です。</p>
VMware Cloud on AWS (VMC) クラウド アカウントを追加する	<p>以下の読み取りおよび書き込み権限を持つアカウントを指定します。</p> <ul style="list-style-type: none"> ■ cloudadmin@vmc.local アカウントまたは CloudAdmin グループ内の任意のユーザー アカウント ■ NSX エンタープライズ管理者のロールとアクセス認証情報 ■ 組織の VMware Cloud on AWS SDDC 環境に対する NSX クラウド管理者アクセス ■ 組織の VMware Cloud on AWS SDDC 環境に対する管理者アクセス ■ 組織の VMware Cloud on AWS サービスにある VMware Cloud on AWS 環境の VMware Cloud on AWS API トークン ■ vCenter の IP アドレスまたは FQDN <p>管理者は、このページの以下の「vSphere ベースのクラウド アカウントでの vCenter エージェントの要件」セクションにリストされているすべての権限を持つ、ターゲットの VMware Cloud on AWS SDDC が使用している vCenter に対するアクセス権も必要です。</p> <p>VMware Cloud on AWS クラウド アカウントを作成して使用するために必要な権限の詳細については、VMware Cloud on AWS 製品ドキュメントの『VMware Cloud on AWS データセンターの管理』を参照してください。</p>

vCenter ベースのクラウド アカウントでの vSphere エージェントの要件

次の表に、VMware Cloud on AWS および vCenter クラウド アカウントを管理するために必要な権限を示します。これらの権限は、エンドポイントをホストするクラスタだけでなく、vCenter Server 内のすべてのクラスタで有効になっている必要があります。

NSX-V、NSX-T、vCenter、VMware Cloud on AWS など、vCenter Server ベースのすべてのクラウド アカウントについて、管理者は vSphere エンドポイント認証情報、またはエージェント サービスが vCenter で実行されるときに認証情報を持っている必要があります。これにより、ホスト vCenter Server に管理者アクセスが提供されます。

vSphere エージェントの要件の詳細については、[VMware vSphere 製品のドキュメント](#)を参照してください。

表 3-12. vCenter Server インスタンスを管理するために vSphere エージェントに必要な権限

属性値	権限
データストア	<ul style="list-style-type: none"> ■ 容量の割り当て ■ データストアの参照 ■ 低レベルのファイル操作
データストア クラスタ	データストア クラスタの設定
フォルダ	<ul style="list-style-type: none"> ■ フォルダの作成 ■ フォルダの削除
グローバル	<ul style="list-style-type: none"> ■ カスタム属性の管理 ■ カスタム属性の設定
ネットワーク	ネットワークの割り当て
権限	権限の変更

表 3-12. vCenter Server インスタンスを管理するために vSphere エージェントに必要な権限（続き）

属性値	権限
リソース	<ul style="list-style-type: none"> ■ 仮想マシンのリソース プールへの割り当て ■ パワーオフ状態の仮想マシンの移行 ■ パワーオン状態の仮想マシンの移行
コンテンツ ライブラリ	<p>コンテンツ ライブラリに権限を割り当てるには、管理者が権限をグローバル権限としてユーザーに付与する必要があります。関連情報については、VMware vSphere のドキュメントで、『vSphere 仮想マシン管理』のコンテンツ ライブラリの権限の階層的な継承を参照してください。</p> <ul style="list-style-type: none"> ■ ライブラリ アイテムの追加 ■ ローカル ライブラリの作成 ■ サブスクライブ済みライブラリの作成 ■ ライブラリ アイテムの削除 ■ ローカル ライブラリの削除 ■ サブスクライブ済みライブラリの削除 ■ ファイルのダウンロード ■ ライブラリ アイテムの消去 ■ サブスクライブ済みライブラリの消去 ■ サブスクリプション情報の検出 ■ ストレージの読み取り ■ ライブラリ アイテムの同期 ■ サブスクライブ済みライブラリの同期 ■ タイプのイントロスペクション ■ 設定の更新 ■ ファイルの更新 ■ ライブラリの更新 ■ ライブラリ アイテムの更新 ■ ローカル ライブラリの更新 ■ サブスクライブ済みライブラリの更新 ■ 設定の表示
タグ	<ul style="list-style-type: none"> ■ vSphere タグの割り当てまたは割り当て解除 ■ vSphere タグの作成 ■ vSphere タグ カテゴリの作成 ■ vSphere タグの削除 ■ vSphere タグ カテゴリの削除 ■ vSphere タグの編集 ■ vSphere タグ カテゴリの編集 ■ カテゴリの UsedBy フィールドの変更 ■ タグの UsedBy フィールドの変更

表 3-12. vCenter Server インスタンスを管理するために vSphere エージェントに必要な権限（続き）

属性値	権限
vApp	<ul style="list-style-type: none"> ■ インポート ■ vApp アプリケーションの設定 <p>vApp.Import アプリケーションの設定が必要になるのは、OVF テンプレートの場合、および仮想マシンをコンテンツ ライブラリからプロビジョニングする場合です。</p> <p>vApp.vApp アプリケーションの設定が必要になるのは、クラウド設定スクリプトで cloud-init を使用する場合です。この設定では、製品情報やプロパティなど、vApp の内部構造を変更できます。</p>
仮想マシン - インベントリ	<ul style="list-style-type: none"> ■ 既存から作成 ■ 新規作成 ■ 移動 ■ 削除
仮想マシン - 相互作用	<ul style="list-style-type: none"> ■ CD メディアの設定 ■ コンソール操作 ■ デバイス接続 ■ パワーオフ ■ パワーオン ■ リセット ■ 中断 ■ ツールのインストール
仮想マシン - 構成	<ul style="list-style-type: none"> ■ 既存ディスクの追加 ■ 新規ディスクの追加 ■ ディスクの削除 ■ 詳細 ■ CPU 数の変更 ■ リソースの変更 ■ 仮想ディスクの拡張 ■ ディスク変更の追跡 ■ メモリ ■ デバイス設定の変更 ■ 名前の変更 ■ 注釈の設定 ■ 設定 ■ スワップファイルの配置
仮想マシン - プロビジョニング	<ul style="list-style-type: none"> ■ カスタマイズ ■ テンプレートのクローン作成 ■ 仮想マシンのクローン作成 ■ テンプレートの展開 ■ カスタム仕様の読み取り
仮想マシンの状態	<ul style="list-style-type: none"> ■ スナップショットの作成 ■ スナップショットの削除 ■ スナップショットに戻す

vRealize Automation Cloud Assembly で使用するための Microsoft Azure の構成

vRealize Automation Cloud Assembly で Microsoft Azure クラウド アカウントを作成するには、いくつかの情報を収集し、いくつかの構成を実行する必要があります。

手順

- 1 Microsoft Azure サブスクリプションおよびテナント ID を特定して記録します。
 - サブスクリプション ID - Azure ポータルの左側のツールバーにある [サブスクリプション] アイコンをクリックして、サブスクリプション ID を表示します。
 - テナント ID - [ヘルプ] アイコンをクリックし、Azure ポータルで診断を表示します。テナントを検索し、見つけたら ID を記録します。
- 2 新しいストレージ アカウントとリソース グループを作成して開始することができます。または、これらを後からブループリントで作成できます。
 - ストレージ アカウント - 次の手順でアカウントを構成します。
 - 1 Azure ポータルで、サイドバーの [ストレージ アカウント] アイコンを見つけます。正しいサブスクリプションが選択されていることを確認し、[追加] をクリックします。また、Azure 検索フィールドでストレージ アカウントを検索することもできます。
 - 2 ストレージ アカウントに必要な情報を入力します。サブスクリプション ID が必要になります。
 - 3 既存のリソース グループを使用するか、新しいリソース グループを作成するかを選択します。後で必要になるため、リソース グループの名前をメモしておきます。

注： 後で必要になるため、ストレージ アカウントの場所を保存します。

- 3 仮想ネットワークを作成します。または、適切な既存のネットワークがある場合は、そのネットワークを選択できます。

ネットワークを作成している場合は、[既存のリソース グループの使用] を選択し、前の手順で作成したグループを指定する必要があります。また、以前に指定したのと同じ場所を選択します。Microsoft Azure は、その場所がオブジェクトが使用する利用可能なすべてのコンポーネント間で一致しない場合、仮想マシンまたはその他のオブジェクトを展開しません。

 - a 左側のパネルで仮想ネットワーク アイコンを見つけてクリックするか、仮想ネットワークを検索します。正しいサブスクリプションを選択し、[追加] をクリックします。
 - b 新しい仮想ネットワークの一意の名前を入力し、後で使用するために記録します。
 - c [アドレス空間] フィールドに、仮想ネットワークの適切な IP アドレスを入力します。
 - d 正しいサブスクリプションが選択されていることを確認し、[追加] をクリックします。
 - e 残りの基本構成情報を入力します。
 - f 必要に応じて他のオプションを変更できますが、ほとんどの構成では、デフォルトのままにしておくことができます。
 - g [作成] をクリックします。

- 4 vRA が認証できるように、Azure Active Directory アプリケーションを設定します。
 - a Azure の左側のメニューで Active Directory アイコンを見つけ、クリックします。
 - b [アプリの登録] をクリックし、[追加] を選択します。
 - c Azure 名の検証に準拠するアプリケーションの名前を入力します。
 - d アプリケーション タイプを Web アプリ/API のままにします。
 - e サインオン URL には、使用状況に適した任意のものを指定できます。
 - f [作成] をクリックします。
- 5 Cloud Assembly でアプリケーションを認証するためのプライベート キーを作成します。
 - a Azure でアプリケーションの名前をクリックします。
後で使用するためにアプリケーション ID をメモしておきます。
 - b 次のペインで [すべての設定] をクリックし、設定リストから [キー] を選択します。
 - c 新しいキーの説明を入力し、期間を選択します。
 - d [保存] をクリックします。キー値は後で取得できないため、安全な場所にコピーしておきます。
 - e 左側のメニューで、アプリケーションの [API のアクセス許可] を選択し、[権限の追加] をクリックして新しい権限を作成します。
 - f [API の選択] ページで、[Azure サービス管理] を選択します。
 - g [権限の委任] をクリックします。
 - h [権限の選択] で user_impersonation を選択し、[権限の追加] をクリックします。
- 6 仮想マシンを展開および管理できるように、Active Directory アプリケーションを認証して Azure サブスクリプションに接続します。
 - a 左側のメニューで [サブスクリプション] アイコンをクリックし、新しいサブスクリプションを選択します。
パネルをスライドさせるには、名前のテキストをクリックする必要がある場合があります。
 - b アクセス コントロール (IAM) オプションを選択すると、サブスクリプションに対する権限が表示されます。
 - c [ロールの割り当ての追加] 見出しの下にある [追加] をクリックします。
 - d [ロール] ドロップダウンから [共同作成者] を選択します。
 - e [アクセスの割り当て] ドロップダウンで、デフォルトの選択をそのままにします。
 - f [選択] ボックスにアプリケーションの名前を入力します。
 - g [保存] をクリックします。
 - h ロールを追加して、新しいアプリケーションに所有者、共同作成者、およびリーダーのロールが割り当てられるようにします。
 - i [保存] をクリックします。

次のステップ

Microsoft Azure コマンド ライン インターフェイス ツールをインストールする必要があります。これらのツールは、Windows および Mac オペレーティング システムの両方で無償で利用できます。これらのツールのダウンロードとインストールの詳細については、Microsoft のドキュメントを参照してください。

コマンド ライン インターフェイスがインストールされている場合は、新しいサブスクリプションに認証する必要があります。

- 1 ターミナル ウィンドウを開き、Microsoft Azure のログイン情報を入力します。認証を行うための URL と短いコードが送付されます。
- 2 ブラウザで、デバイス上のアプリケーションから受信したコードを入力します。
- 3 認証コードを入力し、[続行] をクリックします。
- 4 Azure アカウントを選択してログインします。

複数のサブスクリプションがある場合は、`azure account set <subscription-name>` コマンドを使用して正しいものが選択されていることを確認します。

- 5 続行する前に、`azure provider register microsoft.compute` コマンドを使用して新しい Azure サブスクリプションに Microsoft.Compute プロバイダを登録する必要があります。

コマンドがタイムアウトになり、初回の実行でエラーが発生した場合は、コマンドを再度実行します。

構成が完了したら、`azure vm image list` コマンドを使用して、使用可能な仮想マシン イメージ名を取得できます。必要なイメージを選択して、そのイメージに指定された URN を記録し、後でブループリントで使用できます。

vRealize Automation に Microsoft Azure クラウド アカウントを作成します

クラウド管理者は、チームが vRealize Automation クラウド テンプレートを展開するアカウント リージョンの Microsoft Azure クラウド アカウントを作成できます。

vRealize Automation での Microsoft Azure クラウド アカウントの使用事例を確認するには、「[チュートリアル: vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)」を参照してください。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- 必要なユーザー ロールがあることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- vRealize Automation で使用する Microsoft Azure アカウントを構成します。[vRealize Automation Cloud Assembly で使用するための Microsoft Azure の構成](#)を参照してください。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを構成します。[vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 Microsoft Azure アカウント タイプを選択し、認証情報とその他の値を入力します。
- 3 [検証] をクリックします。
アカウントに関連付けられているアカウント リージョンが収集されます。
- 4 このリソースをプロビジョニングするリージョンを選択します。
- 5 効率を高めるために、[選択したリージョンのクラウド ゾーンの作成] をクリックします。
- 6 タグ付け方法をサポートするためにタグを追加する必要がある場合は、機能タグを入力します。vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 7 [保存] をクリックします。

結果

アカウントが vRealize Automation に追加され、選択したリージョンが指定したクラウド ゾーンで使用可能になります。

次のステップ

このクラウド アカウントのインフラストラクチャ リソースを作成します。

vRealize Automation での Amazon Web Services クラウド アカウントの作成

クラウド管理者は、チームが vRealize Automation クラウド テンプレートを展開するアカウント リージョンの Amazon Web Services (AWS) クラウド アカウントを作成できます。

許可されたユーザーの場合は、AWS クラウド アカウントで AWS GovCloud 構成へのアクセスがサポートされます。この構成では、プロジェクトの構成、タグ、およびインフラストラクチャに関する標準的な vRealize Automation クラウド アカウント機能の大半がサポートされます。Cloud Assembly クラウド テンプレートでは、AWS Platform as a Service (PaaS) プロパティの使用がサポートされます。

次の手順では、AWS クラウド アカウントの構成方法について説明します。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。

- 必要なユーザー ロールがあることを確認します。vRealize Automation のユーザー ロールについてを参照してください。
- 必要な AWS 管理者認証情報を持っていることを確認します。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを構成します。vRealize Automation のインターネット プロキシ サーバの構成方法を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 AWS のアカウント タイプを選択し、認証情報とその他の値を入力します。
- 3 [検証] をクリックします。
アカウントに関連付けられているアカウント リージョンが収集されます。
- 4 このリソースをプロビジョニングするリージョンを選択します。
- 5 効率を高めるために、[選択したリージョンのクラウド ゾーンの作成] をクリックします。
- 6 タグ付け方法をサポートするためにタグを追加する必要がある場合は、機能タグを入力します。vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 7 [追加] をクリックします。

結果

アカウントが vRealize Automation に追加され、選択したリージョンが指定したクラウド ゾーンで使用可能になります。

次のステップ

このクラウド アカウントのインフラストラクチャ リソースを構成します。

vRealize Automation に Google Cloud Platform クラウド アカウントを作成します

クラウド管理者は、チームが vRealize Automation クラウド テンプレートを展開するアカウント リージョンの Google Cloud Platform (GCP) クラウド アカウントを作成できます。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。

- 必要なユーザー ロールがあることを確認します。 [vRealize Automation のユーザー ロールについて](#)を参照してください。
- Google Cloud Platform JSON セキュリティ キーへのアクセス権があることを確認します。
- Google Cloud Platform インスタンスに必要なセキュリティ情報があることを確認します。この情報の多くは、使用中のインスタンスまたは Google のドキュメントから取得できます。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを構成します。 [vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 Google Cloud Platform のアカウント タイプを選択し、適切な認証情報と関連情報を入力します。ソース GCP アカウントのコンピューティング エンジンが初期化されたときに作成されたサービス アカウントを使用します。

上記の「[前提条件]」セクションに示されているように、認証情報の要件は [vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)に記載されています。vRealize Automation でクラウド アカウントを正常に作成するには、ソース GCP アカウントでコンピューティング エンジン サービスを有効にしておく必要があります。

vRealize Automation では、プロジェクト ID は Google Cloud Platform エンドポイントの一部です。これは、クラウド アカウントの作成時に指定します。プロジェクト固有のプライベート イメージのデータ収集中に、vRealize Automation GCP アダプタは Google Cloud Platform API に対してクエリを実行します。

- 3 [検証] をクリックします。
アカウントに関連付けられているアカウント リージョンが収集されます。
- 4 このリソースをプロビジョニングするリージョンを選択します。
- 5 効率を高めるために、[選択したリージョンのクラウド ゾーンの作成] をクリックします。
- 6 タグ付けストラテジをサポートするタグが必要な場合は、機能タグを入力します。 [vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 7 [追加] をクリックします。

結果

アカウントが vRealize Automation に追加され、選択したリージョンが指定したクラウド ゾーンで使用可能になります。

次のステップ

このクラウド アカウントのインフラストラクチャ リソースを作成します。

vRealize Automation に vCenter クラウド アカウントを作成します

vRealize Automation クラウド テンプレートを展開するアカウント リージョンに、vCenter Server クラウド アカウントを追加できます。

ネットワークとセキュリティの目的から、vCenter Server のクラウド アカウントを NSX-T または NSX-V のクラウド アカウントに関連付けることができます。

NSX-T クラウド アカウントは、1 つ以上の vCenter Server クラウド アカウントに関連付けることができます。ただし、NSX-V クラウド アカウントは、1 つの vCenter Server クラウド アカウントにのみ関連付けることができます。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください](#)。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- クラウド アカウントをサポートするようにポートとプロトコルを適切に構成してあることを確認します。[vRealize Automation の製品ドキュメント](#)にある『vRealize Easy Installer を使用した vRealize Automation のインストール』の「vRealize Automation のポートとプロトコル」トピック、および『vRealize Automation リファレンス アーキテクチャ ガイド』の「ポートの要件」トピックを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 vCenter アカウント タイプを選択し、vCenter Server ホスト IP アドレスを入力します。
- 3 vCenter Server の管理者認証情報を入力し、[検証] をクリックします。
アカウントに関連付けられているすべてのデータセンターがデータ収集されます。次の要素は、次の要素のすべての vSphere タグと同様にデータ収集されます。
 - マシン
 - クラスタおよびホスト
 - ポート グループ
 - データ ストア
- 4 このクラウド アカウントにプロビジョニングを許可するために、指定された vCenter Server で使用可能なデータセンターを少なくとも 1 つ選択します。

- 5 効率を高めるために、選択したデータセンターにプロビジョニングするためのクラウド ゾーンを作成します。

組織のクラウド戦略によっては、別の手順でクラウド ゾーンを作成することもできます。

クラウド ゾーンについては、[vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報](#)を参照してください。

- 6 既存の NSX クラウド アカウントを選択します。

この段階で、または、後でクラウド アカウントを編集するときに、NSX アカウントを選択できます。

NSX-V クラウド アカウントについては、[vRealize Automation での NSX-V クラウド アカウントの作成](#)を参照してください。

NSX-T クラウド アカウントについては、[vRealize Automation での NSX-T クラウド アカウントの作成](#)を参照してください。

クラウド テンプレートを展開した後での関連付けの変更については、[vRealize Automation で NSX クラウド アカウントの関連付けを削除した場合の動作](#)を参照してください。

- 7 タグ付け方法をサポートするためにタグを追加する場合は、機能タグを入力します。

この段階で、または、後でクラウド アカウントを編集するときに、タグを追加できます。タグ付けの詳細については、[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 8 [保存] をクリックします。

結果

クラウド アカウントが追加され、選択したデータセンターが指定したクラウド ゾーンで使用できるようになります。マシン、ネットワーク、ストレージ、ボリュームなどの収集されたデータは、[インフラストラクチャ] タブの [リソース] セクションに表示されます。

次のステップ

このクラウド アカウントの残りのインフラストラクチャ リソースを構成します。[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。

vRealize Automation での NSX-V クラウド アカウントの作成

ネットワークとセキュリティのために、NSX-V クラウド アカウントを作成して、vCenter クラウド アカウントに関連付けることができます。

NSX-V クラウド アカウントは、1 つの vCenter Server クラウド アカウントにのみ関連付けることができます。

NSX-V と vCenter Server クラウド アカウントの関連付けは、vRealize Automation の外部、具体的には NSX アプリケーション内で構成する必要があります。vRealize Automation では、NSX と vCenter Server の間の関連付けは作成されません。vRealize Automation では、NSX 内にすでにある関連付けを指定します。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください](#)。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- この NSX クラウド アカウントで使用する vCenter クラウド アカウントを持っていることを確認します。[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。
- クラウド アカウントをサポートするようにポートとプロトコルを適切に構成してあることを確認します。[vRealize Automation の製品ドキュメント](#)にある『vRealize Easy Installer を使用した vRealize Automation のインストール』の「vRealize Automation のポートとプロトコル」トピック、および vRealize Automation リファレンス アーキテクチャ ガイドの「ポートの要件」トピックを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 NSX-V アカウント タイプを選択し、NSX-V ホスト IP アドレスを入力します。
- 3 NSX の管理者認証情報を入力し、[検証] をクリックします。
アカウントに関連付けられているアセットが収集されます。
NSX ホストの IP アドレスが使用できない場合、検証は失敗します。
- 4 使用可能な場合は、この NSX-V アカウントに関連付ける vCenter クラウド アカウントを表す vCenter エンドポイントを選択します。
現在、NSX-T または NSX-V クラウド アカウントに関連付けられていない vCenter Server クラウド アカウントのみを選択できます。
クラウド テンプレートを展開した後での関連付けの変更については、[vRealize Automation で NSX クラウド アカウントの関連付けを削除した場合の動作](#)を参照してください。
- 5 タグ付け方法をサポートするためにタグを追加する場合は、機能タグを入力します。
機能タグは、後で追加または削除できます。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。



機能タグと制約タグによって展開の配置を制御する方法については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 6 [保存] をクリックします。

次のステップ

この NSX クラウド アカウントと関連付ける vCenter クラウド アカウントを作成または編集できます。[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。

このクラウド アカウントで使用するデータセンターで使用される 1 つ以上のクラウド ゾーンを作成して構成します。 [vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報](#)を参照してください。

このクラウド アカウントのインフラストラクチャ リソースを構成します。 [4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。

vRealize Automation での NSX-T クラウド アカウントの作成

ネットワークとセキュリティのために、NSX-T クラウド アカウントを作成して 1 つ以上の vCenter Server クラウド アカウントに関連付けることができます。

NSX-T クラウド アカウントは、1 つ以上の vCenter Server クラウド アカウントに関連付けることができます。ただし、NSX-V クラウド アカウントは、1 つの vCenter Server クラウド アカウントにのみ関連付けることができます。

NSX-T と 1 つ以上の vCenter Server クラウド アカウントの関連付けは、vRealize Automation の外部、具体的には NSX アプリケーション内で構成する必要があります。vRealize Automation では、NSX と vCenter Server の間の関連付けは作成されません。vRealize Automation では、NSX 内にすでにある 1 つ以上の構成の関連付けを指定します。

NSX-T クラウド アカウントは、NSX-T Manager API メソッドまたは NSX-T ポリシー API メソッドのいずれかをサポートするように定義できます。この 2 つのメソッドの詳細については、[NSX-T Data Center の製品ドキュメント](#)の『NSX-T Data Center 管理ガイド』にある「NSX Manager の概要」などのトピックを参照してください。この後の手順でも情報が提供されています。

NSX-T クラウド アカウントの作成後、そのアカウントを別の API メソッドに変換することはできません。代わりに、クラウド アカウントを削除し、他の API モードを使用して再作成する必要があります。

展開のフォルト トレランスと高可用性を簡素化するため、各 NSX-T データセンター エンドポイントは、3 つの NSX Manager を含むクラスタとなっています。

- vRealize Automation は、いずれか 1 つの NSX Manager を参照することができます。このオプションを使用して、1 つの NSX Manager が vRealize Automation からの API 呼び出しを受信します。
- vRealize Automation は、クラスタの仮想 IP アドレスを参照できます。このオプションを使用して、1 つの NSX Manager が仮想 IP アドレスの制御を引き継ぎます。NSX Manager は vRealize Automation からの API 呼び出しを受信します。障害が発生した場合は、クラスタ内の別のノードが仮想 IP アドレスの制御を引き継いで、vRealize Automation からの API 呼び出しを受信します。

NSX の仮想 IP アドレス設定の詳細については、[VMware NSX-T Data Center のドキュメント](#)の『NSX-T Data Center インストール ガイド』にある「クラスタの仮想 IP (VIP) アドレスの設定」を参照してください。

- vRealize Automation は、3 つの NSX Manager への呼び出しをロード バランシングするために、ロード バランサの仮想 IP アドレスを参照できます。このオプションを使用すると、3 つのすべての NSX Manager が vRealize Automation からの API 呼び出しを受信します。

サードパーティのロード バランサの仮想 IP アドレスを設定することも、NSX-T ロード バランサの仮想 IP アドレスを設定することもできます。

環境の規模が大きい場合は、このオプションを使用して、vRealize Automation の API 呼び出しを 3 つの NSX Manager で分割することを検討してください。

前提条件

- 必要な管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしてあることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください](#)。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- この NSX クラウド アカウントで使用する vCenter クラウド アカウントを持っていることを確認します。[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。
- クラウド アカウントをサポートするようにポートとプロトコルを適切に構成してあることを確認します。[vRealize Automation の製品ドキュメント](#)にある『vRealize Easy Installer を使用した vRealize Automation のインストール』の「vRealize Automation のポートとプロトコル」トピック、および vRealize Automation リファレンス アーキテクチャ ガイドの「ポートの要件」トピックを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 NSX-T アカウント タイプを選択し、NSX-T エンドポイント マネージャ インスタンスのホスト IP アドレス、または仮想 IP アドレスを入力します（NSX Manager および仮想 IP アドレスについて予期される動作については上記参照）。
- 3 NSX のユーザー名とパスワード管理者の認証情報を入力し、[検証] をクリックします。
アカウントに関連付けられているアセットが収集されます。
NSX ホストの IP アドレスが使用できない場合、検証は失敗します。
- 4 [NSX-T API メソッド] で、[Manager メソッド] または [ポリシー メソッド] のいずれかを選択します。
 - Manager API メソッド
vRealize Automation の以前のバージョンからオンボーディングまたは移行された既存の NSX-T エンドポイントまたはクラウド アカウントは、Manager API メソッドの NSX-T クラウド アカウントとして扱われます。
Manager API メソッドは、NSX-T 2.4、NSX-T 3.0、NSX-T 3.1 以降でサポートされます。
現在 NSX-T Manager API メソッドを使用している場合は、vRealize Automation に Manager API からポリシー API への移行パスが導入されるまで、Manager API メソッドを使用し続けることをお勧めします。
NSX-T の vRealize Automation オプションの一部では、クラウド テンプレート内の仮想マシンの NIC コンポーネントに対するタグの追加などに関して、NSX-T 3.0 以降が必要です。
 - ポリシー API メソッド（デフォルト）
ポリシー API メソッドは、NSX-T 3.0 および NSX-T 3.1 以降で使用可能です。このオプションを使用すると、NSX-T ポリシー API で使用可能な追加機能を vRealize Automation で使用できるようになります。

vRealize Automation 8.2 にポリシー API メソッドが導入される前に作成した既存の NSX-T クラウド アカウントがある場合、それらは Manager API メソッドを使用します。Manager API からポリシー API への移行ツールを vRealize Automation で使用できるようになるまで待つことをお勧めします。待たない場合は、既存の NSX-T クラウド アカウントを、ポリシー API メソッドを指定する新しい NSX-T クラウド アカウントに置き換える必要があります。

- 5 [関連付け] で、この NSX-T クラウド アカウントに関連付ける 1 つ以上の vCenter Server クラウド アカウントを追加します。既存の vCenter Server クラウド アカウントの関連付けを削除することもできます。

現在、vRealize Automation で NSX-T または NSX-V クラウド アカウントに関連付けられていない vCenter Server クラウド アカウントのみを選択できます。

[vRealize Automation で NSX-T と複数の vCenter Server のマッピングで可能になる事例](#)を参照してください。

クラウド テンプレートを展開した後に関連付けの変更またはクラウド アカウントの削除を行う方法については、[vRealize Automation で NSX クラウド アカウントの関連付けを削除した場合の動作](#)を参照してください。

- 6 タグ付け方法をサポートするためにタグを追加する場合は、機能タグを入力します。

機能タグは、後で追加または削除できます。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 7 [保存] をクリックします。

次のステップ

この NSX クラウド アカウントと関連付ける vCenter クラウド アカウントを作成または編集できます。[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。

このクラウド アカウントで使用するデータセンターで使用する 1 つ以上のクラウド ゾーンを作成して構成します。[vRealize Automation Cloud Assembly クラウド ザーンの詳細情報](#)を参照してください。

このクラウド アカウントのインフラストラクチャ リソースを構成します。[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。

vRealize Automation に VMware Cloud on AWS クラウド アカウントを作成します

クラウド管理者は、チームが vRealize Automation クラウド テンプレートを展開するアカウント リージョンの VMware Cloud on AWS クラウド アカウントを作成できます。

VMware Cloud on AWS では、vRealize Automation にいくつかの独自の設定手順が必要です。クラウド アカウントの API トークン値の設定や、クラウド プロキシのゲートウェイ ファイアウォール ルールの設定など、VMware Cloud on AWS の vRealize Automation の適切な設定については、[チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成](#)のワークフローを参照してください。

前提条件

- vCenter 内のターゲット SDDC の VMware Cloud on AWS CloudAdmin 認証情報を含む、必要な VMware Cloud on AWS 管理者認証情報を持っていること、およびポート 443 での HTTPS アクセスを有効にしていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください。](#)
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- 外部インターネットにアクセスできない場合は、インターネット サーバ プロキシを構成します。[vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。
- SDDC で必要なアクセス権およびファイアウォール ルールが構成されていることを確認します。[VMware Cloud on AWS クラウド アカウントに接続するための vRealize Automation での VMware Cloud on AWS SDDC の準備](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント]の順に選択し、[クラウド アカウントの追加] をクリックして、VMware Cloud on AWS アカウントのタイプを選択します。
- 2 使用可能な SDDC に組織がアクセスするための [VMC API トークン]を追加します。
リンクされた [API トークン] 画面で、新しいトークンを作成したり、組織の既存のトークンを使用できます。詳細については、[サンプル ワークフローの vRealize Automation での VMware Cloud on AWS クラウド アカウントの作成](#)を参照してください。
- 3 展開で使用できる SDDC を選択します。
NSX-V SDDC はサポートされていないため、リストに表示されません。
vCenter および NSX-T Manager の IP アドレス/FQDN 値は、SDDC に基づいて自動的に入力されます。
- 4 指定した SDDC の vCenter のユーザー名とパスワードが cloudadmin@vmc.local のデフォルト値以外である場合は、それを入力します。
- 5 [検証] をクリックして、指定された vCenter へのアクセス権があることと、vCenter が実行されていることを確認します。
アカウントに関連付けられているデータセンターが収集されます。
- 6 効率を高めるために、選択した SDDC にプロビジョニングするためのクラウド ゾーンを作成します。
組織のクラウド戦略によっては、別の手順でクラウド ゾーンを作成することもできます。
- 7 タグ付け方法をサポートするためにタグを追加する場合は、機能タグを入力します。
機能タグは、後で追加または削除できます。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。



機能タグと制約タグによって展開の配置を制御する方法の詳細については、ビデオ チュートリアル [Constraint Tags and Placement](#) を参照してください。

- 8 [保存] をクリックします。

結果

クラウド アカウントが追加され、選択した SDDC が指定したクラウド ゾーンで使用できるようになります。

次のステップ

VMware Cloud on AWS の vRealize Automation を正しく設定するには、[チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成](#)を参照してください。

vRealize Automation 以外の VMware Cloud on AWS に関する関連情報については、[VMware Cloud on AWS ドキュメント](#)を参照してください。

VMware Cloud Foundation クラウド アカウントの作成

ワークロード ドメインを使用するには、vRealize Automation Cloud Assembly 内で VMware Cloud Foundation (VCF) をクラウド アカウントとして構成します。

VCF クラウド アカウントを使用すると、VCF ワークロードを Cloud Assembly に組み込んで、ハイブリッド クラウド管理ソリューションを包括的に効率化することができます。Cloud Assembly には、VCF クラウド アカウントの構成画面を有効にするためのエントリ ポイントがいくつか用意されています。SDDC 統合の [ワークロード ドメイン] タブの [クラウド アカウントの追加] ボタンを使用してこの画面にアクセスすると、vCenter Server と NSX Manager の基本情報であるワークロードが事前に選択されています。

前提条件

VMware SDDC Manager 4.1 以降のインスタンスをこのクラウド アカウントで使用するには、vRealize Automation Cloud Assembly 統合として構成する必要があります。詳細については、[VMware SDDC Manager 統合の構成](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、[クラウド アカウントの追加] をクリックします。
- 2 VCF クラウド アカウントのタイプを選択して、[名前] と [説明] を入力します。
- 3 このクラウド アカウントで使用する SDDC Manager インスタンスの FQDN と認証情報を入力します。
このアカウントで使用する SDDC Manager インスタンスを構成済みの場合は、この手順をスキップできます。
- 4 この VCF クラウド アカウントで使用するワークロード ドメインを 1 つ以上選択します。
- 5 vCenter Server と NSX に Cloud Foundation マネージド サービスの認証情報を使用するように Cloud Assembly を設定する場合は、[サービス認証情報の自動作成] を選択します。あとで、これらの認証情報を変更する場合は、VCF のメカニズムを使用してパスワードを管理する必要があります。
このオプションを選択した場合は、手順 7 と 8 をスキップできます。
- 6 このクラウド アカウントに関連付けられている vCenter Server へのアクセスに必要な認証情報を入力します。
- 7 VCF クラウド アカウントの認証情報を手動で入力する場合は、NSX Manager の見出しの下で、NSX 認証情報を入力します。Cloud Assembly で自動的に NSX 認証情報の作成と検証を行う場合は、[サービス認証情報の作成と検証] をクリックします。

- 8 このクラウド アカウントに関連付けられている NSX-T ネットワークへのアクセスに必要な認証情報を入力します。
- 9 必要に応じて、NSX モードを選択します。
- 10 [検証] をクリックして、SDDC Manager への接続を確認します。
- 11 必要に応じて、[構成] 見出しの下で、プロビジョニング先のデータセンターを選択します。選択したデータセンターのクラウド ゾーンを作成する場合は、チェックボックスをクリックします。
- 12 タグ付けストラテジをサポートするタグを使用する場合は、機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)および[タグ付けストラテジの作成](#)を参照してください。
- 13 [保存] をクリックします。

結果

このクラウド アカウントでは、指定した SDDC Manager に関連付けられている特定のワークロード ドメインを vRealize Automation Cloud Assembly で使用できます。

vRealize Automation を使用して追加のワークロード ドメインを管理する場合は、ドメインごとにこの手順を繰り返します。

次のステップ

VCF クラウド アカウントを構成したら、メインのクラウド アカウント画面でアカウントを選択し、[クラウドのセットアップ] をクリックして、クラウドを構成する VMware Cloud Foundation クイックスタート ウィザードを開始します。

クイックスタート ウィザードの詳細については、[../Getting-Started-Cloud-Assembly/GUID-BDC673B9-D2AD-47BC-93C5-8C500074F931.html](#) を参照してください。

他のアプリケーションとの vRealize Automation の統合

統合を使用すると、外部システムを vRealize Automation に追加できます。

統合の対象には、vRealize Orchestrator、構成管理のほか、GitHub、Ansible、Puppet などの外部システム、および Infoblox などの外部 IP アドレス管理プロバイダが含まれます。

注： 外部インターネットにアクセスできないが、統合で外部インターネットへのアクセスが必要になる場合は、インターネット サーバ プロキシを設定します。[vRealize Automation のインターネット プロキシ サーバの構成方法](#)を参照してください。

vRealize Automation Cloud Assembly での Git 統合の使用方法

vRealize Automation Cloud Assembly は GitLab、GitHub、および BitBucket リポジトリとの統合をサポートしているため、ソース管理で VMware Cloud Templates およびアクション スクリプトを管理できます。この機能により、展開に関連するプロセスを容易に監査し、責任の所在を明確にできます。

vRealize Automation Cloud Assembly には、GitLab、GitHub、BitBucket の 3 種類の Git 統合が用意されています。これらのオプションはそれぞれ別の統合です。

vRealize Automation Cloud Assembly との Git 統合をセットアップするには、指定したすべてのユーザーがアクセスできるように適切なローカル Git リポジトリを設定する必要があります。また、Git でクラウド テンプレートを検出するために、特定の構造でクラウド テンプレートを保存する必要があります。GitLab または GitHub との統合を作成するには、vRealize Automation Cloud Assembly で [インフラストラクチャ] - [接続] - [統合] の順に選択して、該当する項目を選択します。ターゲット リポジトリの url とトークンが必要になります。

既存のリポジトリとの Git 統合を設定すると、資格のあるユーザーであれば、選択したプロジェクトに関連付けられたすべてのクラウド テンプレートを使用できるようになります。これらのテンプレートは、既存の展開で使用することも、新しい展開の基盤として使用することもできます。プロジェクトを追加するときは、そのプロジェクトが Git のどこにどのように保存されているかに関するプロパティをいくつか選択する必要があります。

vRealize Automation Cloud Assembly から直接 Git リポジトリにアクションを保存できます。アクション スクリプトは、Git で直接バージョン管理することも、vRealize Automation Cloud Assembly に複数のバージョンを作成することもできます。vRealize Automation Cloud Assembly 内にアクションのバージョンを作成すると、自動的に 1 つのバージョンとして Git に保存されます。クラウド テンプレートの場合は、vRealize Automation Cloud Assembly から直接 Git 統合に追加できないため、これよりも複雑です。Git インスタンスに直接保存する必要があり、取得する際には、vRealize Automation Cloud Assembly のクラウド テンプレート管理ページで Git から取得します。

はじめに

GitLab または GitHub によってクラウド テンプレートが検出されるようにするには、クラウド テンプレートを特定の構造で作成して保存する必要があります。

- GitLab に正しく統合されるようにクラウド テンプレートを設定して保存します。有効なテンプレートのみが GitLab にインポートされます。
 - クラウド テンプレート用として指定するフォルダを 1 つ以上作成します。
 - すべてのクラウド テンプレートを `blueprint.yaml` ファイル内に格納する必要があります。
 - テンプレートの最上部に、`name:` および `version:` プロパティがあることを確認します。
- 該当するリポジトリの API キーを抽出します。Git アカウントで、右上隅のログインを選択し、[設定] メニューに移動します。[アクセス トークン] を選択し、トークンの名前を入力して、有効期限を設定します。次に、API を選択してトークンを作成します。生成された値をコピーして保存します。

Git 統合で使用するすべてのクラウド テンプレートについて、次のガイドラインを遵守する必要があります。

- 各クラウド テンプレートを個別のフォルダに配置する必要があります。
- すべてのクラウド テンプレートに `blueprint.yaml` という名前を付ける必要があります。
- すべてのクラウド テンプレート YAML ファイルで `name` フィールドと `version` フィールドを使用する必要があります。
- インポートされるのは、有効なクラウド テンプレートのみです。

- Git からインポートされたドラフトのクラウド テンプレートを更新し、そのコンテンツが上位バージョンのものとは異なる場合、そのドラフトは今後の同期で更新されず、新しいバージョンが作成されます。テンプレートを更新し、今後 Git と同期できるようにする場合は、最終変更後に新しいバージョンを作成する必要があります。
- **vRealize Automation Cloud Assembly での GitLab クラウド テンプレート統合の設定**
この手順では、vRealize Automation Cloud Assembly で GitLab 統合を設定することで、リポジトリのクラウド テンプレートを操作できるようにし、さらに指定したプロジェクトに関連付けられている保存済みのテンプレートを自動的にダウンロードできるようにします。GitLab でクラウド テンプレートを使用するには、適切な GitLab インスタンスへの接続を作成し、そのインスタンスに目的のテンプレートを保存する必要があります。
- **vRealize Automation Cloud Assembly で GitHub との統合を設定する**
vRealize Automation Cloud Assembly で、GitHub クラウドベースのリポジトリ ホスティング サービスを統合できます。
- **vRealize Automation Cloud Assembly での Bitbucket 統合の設定**
vRealize Automation Cloud Assembly では、Bitbucket と統合して ABX アクション スクリプトと VMware Cloud Templates の Git ベースのリポジトリとして使用できます。

vRealize Automation Cloud Assembly での GitLab クラウド テンプレート統合の設定

この手順では、vRealize Automation Cloud Assembly で GitLab 統合を設定することで、リポジトリのクラウド テンプレートを操作できるようにし、さらに指定したプロジェクトに関連付けられている保存済みのテンプレートを自動的にダウンロードできるようにします。GitLab でクラウド テンプレートを使用するには、適切な GitLab インスタンスへの接続を作成し、そのインスタンスに目的のテンプレートを保存する必要があります。

既存のリポジトリとの GitLab 統合を設定すると、資格のあるユーザーであれば、選択したプロジェクトに関連付けられたすべてのクラウド テンプレートを使用できるようになります。これらのテンプレートは、既存の展開で使用することも、新しい展開の基盤として使用することもできます。プロジェクトを追加するときは、そのプロジェクトが GitLab のどこにどのように保存されているかに関するプロパティをいくつか選択する必要があります。

注： 新規または更新されたクラウド テンプレートを vRealize Automation Cloud Assembly から Git リポジトリにプッシュすることはできません。また、新しいテンプレートを vRealize Automation Cloud Assembly からリポジトリにプッシュすることはできません。リポジトリにクラウド テンプレートを追加するには、開発者が Git インターフェイスを使用する必要があります。

Git からインポートされたドラフトのクラウド テンプレートを更新し、そのコンテンツが上位バージョンのものとは異なる場合、そのドラフトは今後の同期で更新されず、新しいバージョンが作成されます。クラウド テンプレートを更新し、今後 Git と同期できるようにする場合は、最終変更後に新しいバージョンを作成する必要があります。

GitLab で使用するようクラウド テンプレートを設定し、必要な情報を収集したら、GitLab インスタンスとの統合を設定する必要があります。その後、指定したクラウド テンプレートを GitLab にインポートできます。この手順のビデオ デモは、<https://www.youtube.com/watch?v=h0vqo63Sdgg> で確認できます。

前提条件

- 該当するリポジトリの API キーを抽出します。GitLab アカウントで、右上隅のログインを選択し、[設定] メニューに移動します。[アクセス トークン] を選択し、トークンの名前を入力して、有効期限を設定します。次に、API を選択してトークンを作成します。生成された値をコピーして保存します。

vRealize Automation Cloud Assembly との Git 統合をセットアップするには、指定したすべてのユーザーがアクセスできるように適切なローカル Git リポジトリを設定している必要があります。また、GitLab によってクラウド テンプレートが検出されるようにするには、クラウド テンプレートを特定の構造で作成して保存する必要があります。

- GitLab に正しく統合されるようにクラウド テンプレートを設定して保存します。有効なテンプレートのみが GitLab にインポートされます。[vRealize Automation Cloud Assembly での Git 統合の使用方法を参照してください](#)。

手順

- 1 vRealize Automation Cloud Assembly で、GitLab 環境との統合を設定します。
 - a [インフラストラクチャ] - [統合] - [新規追加] の順に選択し、GitLab を選択します。
 - b GitLab インスタンスの [URL] を入力します。Software as a Service (SaaS) である GitLab インスタンスでは、ほとんどの場合、これは gitlab.com です。
 - c 指定した GitLab インスタンスの [トークン] (API キー) を入力します。GitLab インスタンスからトークンを抽出する方法については、上記の前提条件を参照してください。
 - d 適切な名前と説明を追加します。
 - e [検証] をクリックして、接続を確認します。
 - f 必要に応じて機能タグを追加します。詳細については、[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。
 - g [追加] をクリックします。
- 2 適切なリポジトリでクラウド テンプレートを受け入れるように GitLab 接続を構成します。
 - a [インフラストラクチャ] - [統合] の順に選択し、適切な GitLab 統合を選択します。
 - b [プロジェクト] を選択します。
 - c [新規プロジェクト] を選択し、プロジェクトの名前を作成します。
 - d GitLab 内の [リポジトリ] パスを入力します。通常、これはリポジトリ名に結び付けられている主アカウントのユーザー名です。
 - e 使用する適切な GitLab [ブランチ] を入力します。
 - f 必要に応じて、[フォルダ] の名前を入力します。空白のままにすると、すべてのフォルダが使用可能になります。
 - g 適切な [タイプ] を入力します。必要に応じて、フォルダの名前を入力します。空白のままにすると、すべてのフォルダが使用可能になります。
 - h [次へ] をクリックして、リポジトリの追加を完了します。

[次へ] をクリックすると、クラウド テンプレートをプラットフォームにインポートする自動同期タスクが開始されます。

同期タスクが完了すると、クラウド テンプレートがインポートされたことを示すメッセージが表示されます。

結果

これで、GitLab からクラウド テンプレートを取得できるようになりました。

vRealize Automation Cloud Assembly で GitHub との統合を設定する

vRealize Automation Cloud Assembly で、GitHub クラウドベースのリポジトリ ホスティング サービスを統合できます。

vRealize Automation Cloud Assembly で GitHub との統合を設定するには、有効な GitHub トークンが必要です。トークンの作成および配置の詳細については、GitHub のドキュメントを参照してください。

前提条件

- GitHub にアクセスできる必要があります。
- GitHub に正しく統合されるようにクラウド テンプレートを設定して保存します。有効なクラウド テンプレートのみが GitHub にインポートされます。[vRealize Automation Cloud Assembly での Git 統合の使用方法](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [GitHub] を選択します。
- 3 GitHub 設定画面で必要な情報を入力します。
- 4 [検証] をクリックして、統合を検証します。
- 5 タグ付け方法をサポートするためにタグを追加する必要がある場合は、機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。
- 6 [追加] をクリックします。
- 7 適切なリポジトリでクラウド テンプレートを受け入れるように GitLab 接続を構成します。
 - a [インフラストラクチャ] - [統合] の順に選択し、適切な GitHub 統合を選択します。
 - b [プロジェクト] を選択します。
 - c [新規プロジェクト] を選択し、プロジェクトの名前を作成します。
 - d GitHub 内の [リポジトリ] パスを入力します。通常、これはリポジトリ名に結び付けられている主アカウントのユーザー名です。
 - e 使用する適切な GitHub [ブランチ] を入力します。
 - f 必要に応じて、[フォルダ] の名前を入力します。空白のままにすると、すべてのフォルダが使用可能になります。

- g 適切な [タイプ] を入力します。
- h [次へ] をクリックして、リポジトリの追加を完了します。

クラウド テンプレートをプラットフォームにインポートする自動同期タスクが開始されます。

同期タスクが完了すると、クラウド テンプレートがインポートされたことを示すメッセージが表示されます。

結果

GitHub を vRealize Automation Cloud Assembly のブループリントで使用できます。

次のステップ

これで、GitHub からクラウド テンプレートを取得できるようになりました。

vRealize Automation Cloud Assembly での Bitbucket 統合の設定

vRealize Automation Cloud Assembly では、Bitbucket と統合して ABX アクション スクリプトと VMware Cloud Templates の Git ベースのリポジトリとして使用できます。

vRealize Automation Cloud Assembly では、Bitbucket 統合を使用して VMware Cloud Templates または ABX アクション スクリプトの 2 種類のリポジトリ アイテムを操作できます。Bitbucket 統合を使用する前に、作業対象のプロジェクトを同期する必要があります。ABX アクションは Bitbucket リポジトリへの書き戻しをサポートしていますが、クラウド テンプレートを統合から書き戻すことはできません。クラウド テンプレート ファイルの新しいバージョンを作成する場合は、手動で行う必要があります。

前提条件

- 展開で使用する 1 つ以上の ABX またはクラウド テンプレートベースのプロジェクトを使用して、オンプレミスの Bitbucket Server 環境を設定します。Bitbucket クラウドは現在、サポートされていません。
- Bitbucket 統合を関連付ける vRealize Automation Cloud Assembly プロジェクトを作成または指定します。
- Bitbucket 統合に同期するクラウド テンプレート ファイルの名前は、`blueprint.yaml` にする必要があります。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 Bitbucket を選択します。
- 3 Bitbucket の新しい統合の [サマリ] ページで、サマリ情報と Bitbucket 認証情報を入力します。
- 4 統合を確認するために、[検証] をクリックします。
- 5 タグ付けストラテジをサポートするタグを追加する場合は、機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。
- 6 [追加] をクリックします。

- 7 この Bitbucket 統合にプロジェクトを関連付けるために、Bitbucket 統合のメイン ページで [プロジェクト] タブを選択します。
 - 8 この Bitbucket 統合に関連付けるプロジェクトを選択します。
 - 9 [次へ] をクリックして Bitbucket プロジェクトにリポジトリを追加し、追加するリポジトリのタイプを指定して、[リポジトリ] 名、[ブランチ]、および [フォルダ] を指定します。
 - 10 [追加] をクリックします。
- 1 つまたは複数のリポジトリをプロジェクトに追加するには、[リポジトリの追加] をクリックします。

結果

指定したリポジトリ構成を使用して Bitbucket 統合が構成され、構成されたリポジトリに含まれる ABX アクションとクラウド テンプレートを表示および操作できるようになります。Bitbucket 統合にプロジェクトを追加すると同期操作が実行されて、指定したリポジトリから ABX アクション スクリプトとクラウド テンプレート ファイルの最新バージョンが取得されます。Bitbucket 統合ページの [履歴] タブには、統合のために実行されたすべての同期操作の履歴が表示されます。デフォルトでは、ファイルは 15 分ごとに自動的に同期されますが、ファイルを選択して [同期] をクリックすることにより、いつでも手動でファイルを同期できます。

次のステップ

ABX アクションは vRealize Automation Cloud Assembly 拡張性画面で、クラウド テンプレートはデザイン画面で操作できます。vRealize Automation Cloud Assembly の [拡張性] 領域で ABX アクションの変更後のバージョンを保存すると、新しいバージョンのスクリプトが作成され、リポジトリに書き戻されます。

vRealize Automation での外部 IP アドレス管理統合の構成方法

プロバイダ固有の外部 IP アドレス管理統合ポイントを作成して、クラウド テンプレート展開環境で使用される IP アドレスを管理できます。外部 IP アドレス管理統合ポイントを使用すると、IP アドレスは、vRealize Automation からではなく、指定した IP アドレス管理プロバイダから取得および管理されます。

プロバイダ固有の IP アドレス管理統合ポイントを作成し、vRealize Automation のクラウド テンプレート展開および仮想マシンの IP アドレスと DNS 設定を管理できます。

これらの前提条件の構成方法、およびサンプル ワークフローのコンテキスト内でプロバイダ固有の外部 IP アドレス管理統合ポイントを作成する方法の例については、[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。このワークフローは Infoblox IP アドレス管理統合用ですが、外部 IP アドレス管理ベンダーのリファレンスとしても使用できます。

外部 IP アドレス管理パートナーおよびベンダーが IP アドレス管理ソリューションを vRealize Automation と統合できるようにするために必要な資産を作成する方法については、[IP アドレス管理 SDK を使用して vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合パッケージを作成する方法](#) を参照してください。

前提条件

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。

- [Infoblox](#) や [Bluecat](#) などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。
- Infoblox や BlueCat などの IP アドレス管理プロバイダで展開した統合パッケージにアクセスできることを確認します。展開したパッケージは、最初は IP アドレス管理プロバイダまたは vRealize Automation の Marketplace から zip 形式のダウンロード ファイルとして取得され、vRealize Automation に展開されます。
- IP アドレス管理プロバイダに対して構成されて実行環境に対するアクセス権があることを確認します。
- アクションベースの拡張性 (ABX) のオンプレミスの組み込み実行環境を使用している場合は、HTTP プロキシサーバが [gcr.io](#) や [storage.googleapis.com](#) などの外部サイトに送信トラフィックを渡すことができる vRealize Automation ネットワークにあることを確認します。詳細については、ナレッジベースの記事「[Pulling Docker images behind proxy in vRealize Automation 8.x \(75180\)](#)」を参照してください。
- IP アドレス管理ベンダーの製品へのアクセスと使用に必要なユーザー認証情報があることを確認します。必要なユーザー権限については、統合ベンダーの製品ドキュメントを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [IP アドレス管理] をクリックします。
- 3 [プロバイダ] ドロップダウンで、リストから構成された IP アドレス管理プロバイダ パッケージを選択します。
リストが空の場合は、[プロバイダ パッケージのインポート] をクリックし、既存のプロバイダパッケージの .zip ファイルに移動して選択します。.zip ファイルがない場合は、プロバイダの Web サイトまたは vRealize Automation の [Marketplace] タブから取得することができます。
- 4 外部 IP アドレス管理プロバイダのアカウントの管理者ユーザー名とパスワードの認証情報を入力します。また、プロバイダのホスト名などのすべての必須フィールドも入力します。
- 5 [実行環境] ドロップダウン リストで、オンプレミスのアクションベースの拡張統合ポイントなど、既存の実行環境を選択します。

実行環境で、vRealize Automation と IP アドレス管理プロバイダ間の通信がサポートされます。

IP アドレス管理フレームワークは、アクションベースの拡張性 (ABX) のオンプレミスの組み込み実行環境のみをサポートします。

注： Amazon Web Services または Microsoft Azure クラウド アカウントを統合の実行環境として使用している場合は、IP アドレス管理プロバイダ アプライアンスがインターネットからアクセス可能で、NAT またはファイアウォールの背後にないこと、パブリックに解決可能な DNS 名があることを確認してください。IP アドレス管理プロバイダにアクセスできない場合、Amazon Web Services Lambda または Microsoft Azure 機能が接続できないため、統合は失敗します。

- 6 [検証] をクリックします。
- 7 外部 IP アドレス管理プロバイダからの自己署名証明書を信頼するように要求するプロンプトが表示されたら、[受け入れる] をクリックします。

自己署名証明書を受け入れると、検証アクションを続行することができます。

- 8 この IP アドレス管理統合ポイントの名前を入力し、[追加] をクリックして、新しい IP アドレス管理統合ポイントを保存します。

データ収集アクションは模倣されます。ネットワークおよび IP アドレスは、外部 IP アドレス管理プロバイダからデータ収集されます。

vRealize Automation で新しい外部 IP アドレス管理統合パッケージにアップグレードする方法

既存の外部 IP アドレス管理統合ポイントをアップグレードして、より新しいバージョンのベンダー固有の IP アドレス管理統合パッケージを取得できます。

外部 IP アドレス管理プロバイダまたは VMware は、特定のベンダー用のソース IP アドレス管理統合パッケージをアップグレードする場合があります。たとえば、Infoblox 用の外部 IP アドレス管理統合パッケージは、何回かアップグレードされています。名前付き IP アドレス管理統合ポイントを使用する既存の vRealize Automation インフラストラクチャの設定を保持するには、新しい IP アドレス管理統合ポイントを作成せずに、IP アドレス管理統合ポイントを編集して、更新された IP アドレス管理統合パッケージを提供します。

前提条件

この手順では、外部 IP アドレス管理統合ポイントがすでに作成されていること、およびその統合ポイントをアップグレードして、より新しいバージョンのベンダーの IP アドレス管理統合パッケージを使用することを想定しています。

IP アドレス管理統合ポイントの作成方法の詳細については、[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報を参照してください](#)。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- 外部 IP アドレス管理プロバイダにアカウントがあること、その IP アドレス管理プロバイダで組織のアカウントへの適切なアクセス認証情報があることを確認します。
- IP アドレス管理プロバイダの導入済みの統合パッケージにアクセスできることを確認します。展開したパッケージは、IP アドレス管理プロバイダの Web サイトまたは vRealize Automation の Marketplace から zip 形式のダウンロード ファイルとして取得し、vRealize Automation に展開されます。

プロバイダパッケージの .zip ファイルをダウンロードして展開し、IP アドレス管理の統合ページで [プロバイダ] の値として使用できるようにする方法については、[vRealize Automation で使用するための外部 IP アドレス管理プロバイダパッケージのダウンロードと展開](#)を参照してください。

- IP アドレス管理プロバイダに対して構成されて実行環境に対するアクセス権があることを確認します。実行環境は、通常、アクションベースの拡張性 (ABX) のオンプレミスの組み込み統合ポイントです。

実行環境特性の詳細については、[vRealize Automation での IP アドレス管理統合ポイント用の実行環境の作成](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合][IP アドレス管理] の順に選択し、既存の IP アドレス管理統合ポイントを開きます。
- 2 [プロバイダの管理] をクリックします。
- 3 更新された IP アドレス管理統合パッケージに移動して、このパッケージをインポートします。
- 4 [検証] をクリックし、[保存] をクリックします。

vRealize Automation Cloud Assembly での My VMware 統合の設定

My VMware と vRealize Automation Cloud Assembly を統合して、VMware 関連のアクションおよび機能（クラウド テンプレート用の VMware Marketplace へのアクセスなど）をサポートできます。

各組織で作成できる My VMware 統合は 1 つのみです。

前提条件

My VMware に適切な権限を持つユーザー アカウントが必要です。

- ユーザーを My VMware アカウントに招待する方法については、[ナレッジベースの記事 KB2070555](#) を参照してください。
- My VMware アカウントにユーザー権限を割り当てる方法については、[KB2006977](#) を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [My VMware] を選択します。
- 3 My VMware の設定画面で、必要な情報を入力します。
- 4 タグ付けストラテジをサポートするタグが必要な場合は、機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。
- 5 [追加] をクリックします。

結果

My VMware はクラウド テンプレートで使用できます。

次のステップ

目的のクラウド テンプレートに My VMware コンポーネントを追加します。

Cloud Assembly で vRealize Orchestrator との統合を設定する

1 つ以上の vRealize Orchestrator 統合を設定して、ワークフローを拡張性の一部として使用することができます。

vRealize Automation には、拡張性サブスクリプションに使用できる事前構成済みの vRealize Orchestrator インスタンスが含まれています。また、vRealize Automation Cloud Services コンソールから、組み込みの vRealize Orchestrator のクライアントにアクセスすることもできます。

vRealize Automation Cloud Assembly への vRealize Orchestrator の統合では、外部 vRealize Orchestrator インスタンスを追加して、拡張性サブスクリプションに含まれているワークフロー ライブラリを使用できます。詳細については、[拡張性ワークフロー サブスクリプション](#)を参照してください。

前提条件

- クラウド管理者権限が付与されていることを確認します。詳細については、[vRealize Automation のユーザーロールについて](#)を参照してください。
- vRealize Orchestrator 8.1 にアップグレードまたは移行します。『VMware vRealize Orchestrator のアップグレードとインストール』を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択します。
- 2 [統合の追加] をクリックします。
- 3 vRealize Orchestrator を選択します。
- 4 vRealize Automation Cloud Assembly に、vRealize Orchestrator インスタンスの URL を入力します。
- 5 統合を確認するには、[検証] をクリックします。
- 6 vRealize Orchestrator 統合の名前を入力します。
- 7 (オプション) vRealize Orchestrator 統合の説明を入力します。
- 8 (オプション) 機能タグを追加します。機能タグの詳細については、[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。

注： 機能タグを使用すると、複数の vRealize Orchestrator 統合を管理できます。[プロジェクトの制約を使用した複数の vRealize Orchestrator 統合の管理](#)を参照してください。

- 9 [追加] をクリックします。

vRealize Orchestrator 統合が保存されます。

次のステップ

統合が設定され、ワークフローが追加されていることを確認するには、[拡張性] - [ライブラリ] - [ワークフロー] の順に選択します。

プロジェクトの制約を使用した複数の vRealize Orchestrator 統合の管理

プロジェクトの制約を使用して、ワークフロー サブスクリプションで使用される vRealize Orchestrator 統合を管理できます。

vRealize Automation Cloud Assembly では、ワークフロー サブスクリプションで利用できる複数の vRealize Orchestrator サーバの統合がサポートされます。ソフト制約またはハード制約により、プロジェクトによってプロビジョニングされるクラウド テンプレートで使用される vRealize Orchestrator 統合を管理できます。プロジェクトの制約の詳細については、[vRealize Automation Cloud Assembly のプロジェクト タグとカスタム プロパティの使用](#)を参照してください。

前提条件

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- vRealize Automation Cloud Assembly で、複数の vRealize Orchestrator 統合を構成します。[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。
- 機能タグを vRealize Orchestrator 統合に追加します。[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] の順に移動し、プロジェクトを選択します。
- 2 [プロビジョニング] タブを選択します。
- 3 [拡張性の制約] テキスト ボックスに vRealize Orchestrator 統合の機能タグを入力し、それをプロジェクトのソフト制約またはハード制約として設定します。
- 4 [保存] をクリックします。

結果

クラウド テンプレートを展開するときに、vRealize Automation Cloud Assembly はプロジェクトの制約を使用して、ワークフロー サブスクリプションで使用する vRealize Orchestrator 統合を管理します。

次のステップ

または、機能タグを使用して、複数の vRealize Orchestrator 統合をクラウド アカウント レベルで管理できます。詳細については、[クラウド アカウント機能タグによる複数の vRealize Orchestrator 統合の管理](#)を参照してください。

クラウド アカウント機能タグによる複数の vRealize Orchestrator 統合の管理

機能タグを使用して、ワークフロー サブスクリプションで使用する vRealize Orchestrator 統合を管理できます。

vRealize Automation Cloud Assembly では、ワークフロー サブスクリプションで利用できる複数の vRealize Orchestrator サーバの統合がサポートされます。クラウド アカウントに機能タグを追加することにより、ワークフロー サブスクリプションで使用する vRealize Orchestrator 統合を管理できます。

前提条件

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- vRealize Automation Cloud Assembly で、複数の vRealize Orchestrator 統合を構成します。詳細については、[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。
- 機能タグを vRealize Orchestrator 統合に追加します。[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に移動します。
- 2 クラウド アカウントを選択します。
- 3 使用する vRealize Orchestrator 統合の機能タグを入力します。

機能タグは、自動的にソフト制約に変換されます。統合の管理でハード制約を使用するには、プロジェクトの制約を使用する必要があります。詳細については、[プロジェクトの制約を使用した複数の vRealize Orchestrator 統合の管理](#)を参照してください。

- 4 [保存] をクリックします。

結果

クラウド テンプレートを展開するときに、vRealize Automation Cloud Assembly は関連付けられたクラウド アカウント内のタグ付けを使用して、ワークフロー サブスクリプションで使用される vRealize Orchestrator 統合を管理します。

vRealize Automation Cloud Assembly で Kubernetes を使用する方法

vRealize Automation Cloud Assembly には、Kubernetes リソースを管理および展開するためのいくつかのオプションが用意されています。

vRealize Automation Cloud Assembly で Kubernetes リソースを操作するには、主に次の 2 つの方法があります。VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) (旧称は PKS)、または Red Hat OpenShift を vRealize Automation Cloud Assembly と統合して、Kubernetes リソースの構成、管理、展開を行うことができます。2 番目の方法として、vCenter クラウド アカウントを利用してスーパーバイザーの名前空間にアクセスし、vSphere Project Pacific の Kubernetes ベース機能と連携することができます。vRealize Automation Cloud Assembly に外部 Kubernetes リソースを統合することもできます。

VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) または Openshift 統合の使用

TKGI、外部クラスタ、または Openshift 構成の場合、ユーザーは vRealize Automation Cloud Assembly が提供する Kubeconfig を使用することで、適切な Kubernetes クラスタにアクセスできます。

TKGI または OpenShift 統合を作成すると、該当する Kubernetes クラスタが vRealize Automation Cloud Assembly で使用可能になります。また、vRealize Automation Cloud Assembly に対して Kubernetes コンポーネントを追加したり、作成したりすることにより、クラスタおよびコンテナ アプリケーションの管理がサポートされるようになります。これらのアプリケーションは、Service Broker カタログから使用できるセルフサービス展開の基盤となります。

vSphere Project Pacific Kubernetes クラスタの操作

Project Pacific は、Kubernetes を制御プレーンとして使用する vSphere の機能強化です。これを使用することで、ユーザーは仮想マシンとコンテナの両方を 1 つのインターフェイスで管理することができます。また、vRealize Automation Cloud Assembly を使用することで、vSphere に組み込まれた Pacific Kubernetes 機能を利用できます。Pacific 機能にアクセスするには、スーパーバイザー クラスタを含む vSphere の実装を行い、vCenter 環境との統合を作成します。Pacific では、従来の仮想マシンと Kubernetes クラスタの両方を vCenter から管理できます。

Pacific ベースのスーパーバイザー名前空間では、ユーザーはスーパーバイザー名前空間の詳細への指定リンクにログインできるように、適切な vSphere SSO にアクセスする必要があります。これにより、ユーザーは vSphere 認証を使用してカスタマイズされた Kubectl をダウンロードして、スーパーバイザーの名前空間を使用することができます。

この機能を使用するには、スーパーバイザー名前空間が設定されている vSphere クラウド アカウントを持つ vCenter が必要です。ユーザーは、ログインした後、該当する名前空間の使用を開始できます。

- **vRealize Automation Cloud Assembly での PKS 統合の設定**

オンプレミスおよびクラウド内で PKS リソース接続を構成して、vRealize Automation Cloud Assembly で Kubernetes の統合および管理機能をサポートできます。

- **vRealize Automation Cloud Assembly での Red Hat OpenShift 統合の構成**

オンプレミスおよびクラウド内で Red Hat OpenShift リソース接続を構成して、vRealize Automation Cloud Assembly でエンタープライズ レベルの Kubernetes の統合および管理機能をサポートできます。

- **vRealize Automation Cloud Assembly での Kubernetes ゾーンの構成**

Kubernetes ゾーンを使用すると、クラウド管理者は、vRealize Automation Cloud Assembly 展開で使用する Kubernetes のクラスタおよび名前空間とスーパーバイザー名前空間のポリシー ベースの配置を定義できます。管理者はこのページを使用して、Kubernetes 名前空間のプロビジョニングに使用できるクラスタを指定し、どのプロパティがクラスタに対して許容されるかを指定できます。

- **vRealize Automation Cloud Assembly での Pacific スーパーバイザー クラスタおよび名前空間の使用**

管理者は、ユーザーがクラウド テンプレート内の名前空間を展開して Service Broker カタログ内で要求できるように、Pacific が有効になっている既存の vSphere 統合のスーパーバイザー名前空間を使用するように vRealize Automation Cloud Assembly を構成できます。

- **vRealize Automation Cloud Assembly での Kubernetes クラスタと名前空間の操作**

vRealize Automation Cloud Assembly で Kubernetes 展開の基盤となる Kubernetes クラスタと名前空間の構成を、汎用と Pacific ベースの両方で追加、表示、および管理できます。

- **vRealize Automation Cloud Assembly のクラウド テンプレートへの Kubernetes コンポーネントの追加**

Kubernetes コンポーネントを vRealize Automation Cloud Assembly クラウド テンプレートに追加するときに、クラスタを追加するか、ユーザーがさまざまな構成で名前空間を作成できるようにするかを選択できます。通常、この選択は、アクセス コントロールの要件、Kubernetes コンポーネントの構成方法、および展開の要件によって異なります。

■ Kubernetes での vRealize Automation Cloud Assembly 拡張性の使用

vRealize Automation Cloud Assembly には、Kubernetes クラスターの展開に関連する一般的なアクションに対応する一連の標準イベント トピックが用意されています。ユーザーは必要に応じてこれらのトピックを購読でき、購読済みのトピックに関連するイベントが発生したときに通知を受け取ります。また、イベント通知に基づいて実行するように vRO ワークフローを構成することもできます。

vRealize Automation Cloud Assembly での PKS 統合の設定

オンプレミスおよびクラウド内で PKS リソース接続を構成して、vRealize Automation Cloud Assembly で Kubernetes の統合および管理機能をサポートできます。

PKS の統合により、オンプレミスおよびクラウド内の PKS インスタンスと、PKS および外部クラスターでプロビジョニングされた Kubernetes クラスターを管理できます。ポリシーベースのリソース配置をサポートするには、Kubernetes プロファイルを作成してプロジェクトに関連付ける必要があります。

前提条件

- UAA 認証を使用して適切に設定された Pivotal Container Service (PKS) サーバが必要です。
- クラウド管理者権限が付与されていることを確認します。詳細については、[vRealize Automation のユーザーロールについて](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 VMware Enterprise PKS を選択します。
- 3 作成する PKS クラウド アカウントの IP アドレスまたは FQDN、および PKS アドレスを入力します。
 - IP アドレスは、PKS ユーザー認証サーバの FQDN または IP アドレスです。
 - PKS アドレスは、メイン PKS サーバの FQDN または IP アドレスです。
- 4 この PKS サーバがローカルにあるか、パブリック クラウドまたはプライベート クラウドに配置されているかを選択します。
- 5 PKS サーバとその他の関連情報について、適切な [ユーザー名] と [パスワード] を入力してください。
- 6 タグ付けストラテジをサポートするタグを使用する場合は、機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。
- 7 [追加] をクリックします。

結果

Kubernetes ゾーンを作成してプロジェクトに割り当てるか、外部の Kubernetes クラスターを検出してそれらのクラスターをプロジェクトに割り当てることができます。さらに、大規模なグループおよび組織間のクラスターの管理を簡素化する Kubernetes 名前空間を追加または作成できます。

次のステップ

適切な Kubernetes ゾーンを作成または選択してから、1 つ以上のクラスタまたは名前空間を選択してプロジェクトに割り当てます。その後、クラウド テンプレートを作成して公開し、Kubernetes を使用するセルフサービス展開をユーザーが生成できるようにします。

vRealize Automation Cloud Assembly での Red Hat OpenShift 統合の構成

オンプレミスおよびクラウド内で Red Hat OpenShift リソース接続を構成して、vRealize Automation Cloud Assembly でエンタープライズ レベルの Kubernetes の統合および管理機能をサポートできます。

vRealize Automation Cloud Assembly は、OpenShift バージョン 3.x との統合をサポートしています。

前提条件

- Red Hat OpenShift の実装を適切に構成しておく必要があります。
- クラウド管理者権限が付与されていることを確認します。詳細については、[vRealize Automation のユーザーロールについて](#)を参照してください。
- VMware では、クラウド テンプレートを含む OpenShift クラスタの作成に使用可能なリソースを次の場所で提供しています：<https://flings.vmware.com/enterprise-openshift-as-a-service-on-cloud-automation-services>。これらのリソースを使用して作成されたクラスタを Kubernetes ゾーンでグローバル クラスタとして使用して、セルフサービス名前空間を作成できます。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 Red Hat OpenShift を選択します。
- 3 OpenShift サーバの [アドレス] および [場所] を入力します。
- 4 適切な [認証情報のタイプ] を選択し、適切な認証情報を入力します。

OpenShift 統合では、OAuth ユーザー名/パスワード、パブリック キー、またはベアラー トークン認証がサポートされます。

- 5 OpenShift 統合の適切な [名前] および [説明] を入力します。
- 6 タグ付けストラテジをサポートするタグを使用する場合は、適切な機能タグを入力します。[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法およびタグ付けストラテジの作成](#)を参照してください。
- 7 [追加] をクリックします。

結果

統合が作成されると、[Kubernetes] ページの関連するセクションに新しい Kubernetes クラスタが表示されます。Kubernetes ゾーンを作成してプロジェクトに割り当てることができます。さらに、大規模なグループおよび組織間のクラスタの管理を簡素化する Kubernetes 名前空間を構成できます。

次のステップ

適切な Kubernetes ゾーンを作成または選択してから、1 つ以上のクラスタまたは名前空間を選択してプロジェクトに割り当てます。その後、クラウド テンプレートを作成して公開し、Kubernetes を使用するセルフサービス展開をユーザーが生成できるようにします。

vRealize Automation Cloud Assembly での Kubernetes ゾーンの構成

Kubernetes ゾーンを使用すると、クラウド管理者は、vRealize Automation Cloud Assembly 展開で使用される Kubernetes のクラスタおよび名前空間とスーパーバイザー名前空間のポリシー ベースの配置を定義できます。管理者はこのページを使用して、Kubernetes 名前空間のプロビジョニングに使用できるクラスタを指定し、どのプロパティがクラスタに対して許容されるかを指定できます。

クラウド管理者は、Kubernetes ゾーンを、Cloud Assembly 用に構成されている PKS クラウド アカウントと、または、プロジェクトに関連付けられていない外部 Kubernetes クラスタと関連付けることができます。

Kubernetes ゾーンを作成するときに、複数のプロバイダ固有のリソースをゾーンに割り当てることができます。これらのリソースは、ワーカー、マスター、使用可能な CPU、メモリ、およびその他の設定に関して、新しくプロビジョニングされたクラスタに対してどのプロパティを設定できるかを示します。PKS プロバイダの場合、これらは PKS プランに対応しています。管理者は、新たにプロビジョニングされた Kubernetes の名前空間の配置に使用される Kubernetes ゾーンに複数のクラスタを割り当てすることもできます。管理者は、オンボーディングされていないクラスタ、または CMX で管理されていないクラスタ、および事前選択されたクラスタ プロバイダを使用してプロビジョニングされたクラスタのみを割り当てることができます。管理者は、複数の Kubernetes ゾーンを単一のプロジェクトに割り当てることができるため、このプロジェクト内で発生する配置操作ですべてのゾーンを使用できるようになります。

クラウド管理者は、複数のレベルで優先順位を割り当てることができます。

- プロジェクト内の Kubernetes ゾーンの優先順位。
- Kubernetes ゾーン内のリソースの優先順位。
- Kubernetes ゾーン内のクラスタの優先順位。

クラウド管理者は、複数のレベルでタグを割り当てすることもできます。

- Kubernetes ゾーンごとの機能タグ。
- リソース割り当てごとのタグ。
- クラスタ割り当てごとのタグ。

汎用 Kubernetes 名前空間と同じ方法で、vSphere のスーパーバイザー名前空間を使用して Kubernetes ゾーンを作成できます。Kubernetes ゾーンにスーパーバイザー名前空間を追加するには、目的の Pacific 名前空間リソースを含む vSphere 7 エンドポイントにこのゾーンを関連付ける必要があります。

Service Broker には、Service Broker 管理者が既存の Kubernetes ゾーンにアクセスできるバージョンの Kubernetes ゾーン ページが含まれています。それにより、Service Broker 管理者は、カタログからプロビジョニングされる Kubernetes 名前空間およびクラスタの配置ポリシーを作成できます。

前提条件

適切な PKS 展開との統合を構成します。[vRealize Automation Cloud Assembly での PKS 統合の設定](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [構成] - [Kubernetes ゾーン] の順に選択し、[新しい Kubernetes ゾーン] をクリックします。

- 2 このゾーンを適用する PKS 統合 [アカウント] の名前を入力します。

これにより、ゾーンに関連付けられているクラウド アカウントまたはエンドポイントが定義されます。各ゾーンには、1つのエンドポイントのみを割り当てることができます。vSphere のスーパーバイザー名前空間を使用する場合は、スーパーバイザー名前空間を含む vSphere のインスタンスのみを選択できます。

- 3 Kubernetes ゾーンの [名前] と [説明] を追加します。

- 4 該当する場合は機能タグを追加します。詳細については、[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。

- 5 [保存] をクリックします。

- 6 [オンデマンド] タブをクリックし、クラスタ プロビジョニングに使用するゾーンに応じて PKS プランを追加します。

1つ以上のプランを選択し、優先順位を割り当てることができます。数字が小さいほど優先度が高くなります。タグ ベースの選択は優先順位の割り当てよりも優先されます。

- 7 [クラスタ] タブをクリックし、[コンピューティングの追加] ボタンをクリックして、Kubernetes またはスーパーバイザー クラスタをゾーンに追加します。外部クラスタを使用している場合、選択すると、自動的に vRealize Automation Cloud Assembly にオンボーディングされます。

vRealize Automation Cloud Assembly の [Kubernetes クラスタ] 画面で、クラスタに Kubernetes の名前空間を追加できます。

結果

Kubernetes ゾーンは、vRealize Automation Cloud Assembly 展開で使用するよう構成されます。

次のステップ

プロジェクトに Kubernetes ゾーンを割り当てます。

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] を選択してから、Kubernetes ゾーンに関連付けるプロジェクトを選択します。

- 2 [プロジェクト] ページの [Kubernetes プロビジョニング] タブをクリックします。

- 3 [Kubernetes ゾーンの追加] をクリックし、作成したゾーンを追加します。必要に応じて複数のゾーンを使用できます。また、ゾーンの優先順位も設定します。

- 4 [保存] をクリックします。

プロジェクトにゾーンを割り当てると、[デザイン] タブの [クラウド テンプレート] 画面を使用して、Kubernetes ゾーンとプロジェクト構成に基づいて展開をプロビジョニングできます。[クラウド テンプレート] 画面には、K8S クラスタ、K8S 名前空間、およびスーパーバイザー名前空間を追加するためのオプションがあります。目的の Kubernetes リソースに適したオプションを選択します。

vRealize Automation Cloud Assembly での Pacific スーパーバイザー クラスタおよび名前空間の使用

管理者は、ユーザーがクラウド テンプレート内の名前空間を展開して Service Broker カタログ内で要求できるように、Pacific が有効になっている既存の vSphere 統合のスーパーバイザー名前空間を使用するように vRealize Automation Cloud Assembly を構成できます。

このタスクでは、展開で使用するために vRealize Automation Cloud Assembly でスーパーバイザー クラスタを追加する方法と、特定の Kubernetes リソースにアクセスできる vRealize Automation Cloud Assembly プロジェクトおよびユーザーを定義する名前空間を作成または追加する方法について説明します。この機能は、PKS や Openshift などの統合ではなく、適切な vSphere クラウド アカウントに依存します。スーパーバイザー クラスタは、vSphere に関連付けられているカスタマイズされた Kubernetes クラスタです。これらのクラスタは Kubernetes API をエンド ユーザーに公開し、ワーカー ノードのプラットフォームとして Linux ではなく ESXi を使用します。スーパーバイザー名前空間は、Kubernetes リソースへのアクセス コントロールを簡素化します。これは、通常、個々の仮想マシンよりも名前空間にポリシーを適用するほうが容易であるためです。各スーパーバイザー クラスタには、複数の名前空間を作成できます。

Pacific が有効になっている vSphere インスタンスで使用する場合、スーパーバイザー名前空間を使用したプロビジョニングに使用できるスーパーバイザー クラスタは、Kubernetes ゾーンによって定義されます。スーパーバイザー名前空間は、Pacific が有効になっている vSphere インスタンスに固有のものです。Pacific が有効になっている vSphere インスタンスに、汎用 Kubernetes リソースをプロビジョニングすることはできません。

プロジェクト閲覧者として指定された vRealize Automation Cloud Assembly ユーザーには、名前空間に対して表示のみのアクセス権が付与されますが、プロジェクト メンバーは名前空間を編集できます。

名前空間に関連付けられているスーパーバイザー クラスタは、必要に応じて構成できます。

前提条件

- vRealize Automation Cloud Assembly で Pacific 名前空間を使用するには、vSphere 7.x エンドポイントが構成されている必要があります。vSphere は、vCenter Server クラウド アカウントの一部としてインストールされます。[vRealize Automation に vCenter クラウド アカウントを作成します](#)を参照してください。
- Project Pacific を vSphere クラウド アカウントで有効にする必要があります。また、適切なスーパーバイザー名前空間が含まれている必要があります。
- ユーザーを同期するために、vCenter Server と vRealize Automation 展開の両方で同じ Active Directory を使用する必要があります。そのようにしなかった場合でもプロビジョニングは機能しますが、vRealize Automation ユーザーは名前空間への自動アクセスを取得できません。

手順

- 1 vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [構成] - [Kubernetes ゾーン] の順に選択します。

この画面には使用可能な管理対象クラスタが表示され、さらにクラスタを追加できます。クラスタのいずれかをクリックすると、その詳細が表示されます。

- 2 [新しい Kubernetes ゾーン] を選択します。
- 3 ターゲット vSphere クラウド アカウントの [アカウント] 詳細を指定します。

- 4 テキスト ボックスで検索アイコンをクリックして、すべての vSphere アカウントを表示するか、アカウントを名前で検索します。
- 5 新しいゾーンの [名前] と [説明] を入力します。
- 6 該当する場合は機能タグを追加します。詳細については、[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。
- 7 [プロビジョニング] タブをクリックして、名前空間に関連付けるスーパーバイザー クラスタを選択します。
- 8 [コンピューティングの追加] をクリックして、使用可能なスーパーバイザー クラスタを表示して選択します。
- 9 [追加] をクリックします。
- 10 [インフラストラクチャ] - [管理] - [プロジェクト] を選択してから、Kubernetes ゾーンに関連付けるプロジェクトを選択します。
- 11 [プロジェクト] ページの [Kubernetes プロビジョニング] タブをクリックします。
- 12 [Kubernetes ゾーンの追加] をクリックし、作成したゾーンを追加します。必要に応じて複数のゾーンを使用できます。また、ゾーンの優先順位も設定します。
- 13 [保存] をクリックします。

次のステップ

名前空間が構成されると、該当するユーザーには、vRealize Automation Cloud Assembly の [インフラストラクチャ] - [リソース] - [Kubernetes] 画面でその名前空間が表示されます。ユーザーは [サマリ] タブの [アドレス] リンクをクリックして vSphere 向け Kubernetes CLI Tools を開き、名前空間を管理できます。ユーザーがスーパーバイザー名前空間の詳細へのリンクにアクセスするには、クラウド管理者となるか、指定されたプロジェクトの名前空間のメンバーになることが必要です。また、カスタマイズされた Kubectl をダウンロードして、スーパーバイザー名前空間を利用することも可能です。ユーザーは、スーパーバイザー名前空間にログインして、他の名前空間と同じように使用し、クラウド テンプレートを作成してアプリケーションを展開できます。

名前空間をクラウド テンプレートに追加するには、[デザイン] - [クラウド テンプレート] の順に選択し、既存のクラウド テンプレートを選択するか、新しいクラウド テンプレートを作成します。次に、左側のメニューで [スーパーバイザー名前空間] の項目を選択し、キャンバスにドラッグします。

スーパーバイザー名前空間を含むクラウド テンプレートを展開すると、ユーザーも Service Broker カタログからスーパーバイザー名前空間を要求できるようになります。また、Cloud Assembly の [展開] 画面をクリックして、展開に関する情報を表示したり、vSphere 上の名前空間に対して kubectl を実行するコマンドを含むリンクにアクセスしたりできます。

vRealize Automation Cloud Assembly での Kubernetes クラスタと名前空間の操作

vRealize Automation Cloud Assembly で Kubernetes 展開の基盤となる Kubernetes クラスタと名前空間の構成を、汎用と Pacific ベースの両方で追加、表示、および管理できます。

[インフラストラクチャ] - [リソース] - [Kubernetes] ページでアクセスの資格が付与された、Kubernetes クラスタと名前空間を表示、追加、および管理できます。多くの場合、このページでは展開されたクラスタと名前空間の管理が簡素化されます。

- クラスタ：クラスタとは、1 台以上の物理マシンに分散された Kubernetes ノードのグループです。このページには、vRealize Automation Cloud Assembly インスタンスで使用するよう構成されている、プロビジョニング済みおよび未展開のクラスタが表示されます。クラスタをクリックすると、その現在のステータスに関する情報が表示されます。クラスタを展開すると、クラウド管理者のみがアクセスできる Kubconfig ファイルへのリンクが含まれます。このファイルで、名前空間のリストを含む完全な管理者権限がクラスタ全体に対して付与されます。

スーパーバイザー クラスタは vSphere インスタンスに対して固有であり、Linux の代わりに ESXi をワーカーノードとして使用します。

- 名前空間：名前空間は、クラスタ リソースを分離する方法を管理者に提供する仮想クラスタです。ユーザーおよび組織の大規模なグループ間にまたがるリソース管理が簡素化されます。ロールベースのアクセス コントロールのフォームとして、クラウド管理者は、ユーザーが展開を申請したときに、ユーザーがプロジェクトに名前空間を追加して後で Kubernetes のクラスタ ページから名前空間を管理できるようにすることが可能です。名前空間を展開すると、開発者などの適切なユーザーがその名前空間の一部の要素を表示および管理できるようにする Kubeconfig ファイルへのリンクが名前空間に含まれます。

スーパーバイザー名前空間は vSphere インスタンスにのみ存在し、vSphere オブジェクトへの Kubernetes のようなアクセスを提供します。

新規または既存のクラスタを構成している場合は、接続にマスター IP アドレスを使用するか、マスター ホスト名を使用するかを選択する必要があります。

vRealize Automation Cloud Assembly での汎用 Kubernetes クラスタの操作

このページのオプションを使用して、新しいクラスタ、既存のクラスタ、または外部クラスタを vRealize Automation Cloud Assembly に追加できます。

- 1 [インフラストラクチャ] - [リソース] - [Kubernetes] を選択し、[クラスタ] タブがアクティブであることを確認します。

現在 vRealize Automation Cloud Assembly インスタンス向けに構成されているクラスタがある場合は、このページに表示されます。

- 新規または既存のクラスタを追加している場合、またはクラスタを展開している場合は、次の表を参考にして適切なオプションを選択します。

オプション	説明	詳細
展開	vRealize Automation Cloud Assembly への新しいクラスタの追加	このクラスタが展開される TKGI クラウド アカウントと、必要なプランおよびノードの数を指定する必要があります。
既存を追加	プロジェクトと連携するように既存のクラスタを構成します。	TKGI クラウド アカウント、使用するクラスタ、および対象となる開発者に適したプロジェクトを指定する必要があります。また、共有範囲を指定する必要もあります。グローバルに共有する場合は、Kubernetes ゾーンと名前空間を適切に構成する必要があります。
外部を追加	TKGI に関連付けられていない可能性のある基本的な Kubernetes クラスタを vRealize Automation Cloud Assembly に追加します。	クラスタが関連付けられているプロジェクトを指定し、目的のクラスタの IP アドレスを入力して、このクラスタに接続するために必要なクラウド ブロキシと証明書情報を選択する必要があります。

- [追加] をクリックして、vRealize Automation Cloud Assembly 内でクラスタを使用できるようにします。

vRealize Automation Cloud Assembly での Kubernetes 名前空間の操作

クラウド管理者は、名前空間を使用して Kubernetes クラスタ リソースをグループ化および管理できます。ユーザーの場合、名前空間は展開環境の Kubernetes クラスタ内の領域です。管理者およびユーザーは、[インフラストラクチャ] - [リソース] - [Kubernetes] ページにある [名前空間] タブを使用して名前空間にアクセスできます。

Kubernetes の名前空間を vRealize Automation Cloud Assembly でリソースに追加するには、いくつかの方法があります。次の手順では、一般的な方法について説明します。

- [インフラストラクチャ] - [リソース] - [Kubernetes] を選択し、[名前空間] タブをクリックします。
- 新しい名前空間を追加するには、[新しい名前空間] をクリックします。既存の名前空間を追加するには、[名前空間を追加] をクリックします。
- 名前空間の [名前] と [説明] を入力します。
この時点で、Kubernetes リソースで使用する名前空間が追加されましたが、関連付けは行われていません。
- この名前空間に関連付ける [クラスタ] を指定します。
- [作成] をクリックして、名前空間を vRealize Automation Cloud Assembly に追加します。

スーパーバイザー クラスタおよびスーパーバイザー名前空間の操作

vRealize Automation Cloud Assembly の [Kubernetes] 画面で、スーパーバイザー クラスタと名前空間の構成を表示および変更できます。

- vRealize Automation Cloud Assembly で [インフラストラクチャ] - [リソース] - [Kubernetes] の順に選択します。
- [スーパーバイザー クラスタの追加] を選択します。
- ターゲット vSphere クラウド アカウントの詳細を指定します。
- スーパーバイザー クラスタのテキスト ボックスで検索アイコンをクリックして、すべてのスーパーバイザー クラスタを表示するか、クラスタを名前で検索します。

- 5 目的のクラスタを選択し、[追加] をクリックします。
- 6 [スーパーバイザー名前空間] タブを選択し、[新しいスーパーバイザー名前空間] ボタンをクリックして、新しい名前空間を追加します。
- 7 [スーパーバイザー名前空間] タブを選択し、[新しいスーパーバイザー名前空間] ボタンをクリックして、新しい名前空間を追加します。
 - a 新しい名前空間を作成する場合は、[名前] と [説明] に名前と説明を追加します。
 - b 名前空間に関連付ける適切なクラウド アカウントを、[アカウント] で選択します。
 - c [スーパーバイザー クラスタ] で、この名前空間に関連付けるスーパーバイザー クラスタを選択します。
 - d 名前空間に関連付けるプロジェクトを、[プロジェクト] で選択します。
 - e [作成] をクリックします。
- 8 新しい名前空間に関する詳細を確認します。

現在 vSphere の名前空間にアクセスできるユーザーとグループが [ユーザー] タブに表示されます。新しいユーザーまたはグループがプロジェクトに追加された場合は、このタブの [ユーザーの更新] ボタンをクリックして、リストを更新します。リストは自動的に更新されないため、ボタンを使用して更新する必要があります。

注： ユーザーの同期が有効であるのは、vRealize Automation Cloud Assembly と vCenter Server が共通の Active Directory/LDAP サービスを使用するように構成されている場合に限られます。

名前空間が構成されると、該当するユーザーには、vRealize Automation Cloud Assembly の [インフラストラクチャ] - [リソース] - [Kubernetes] 画面でその名前空間が表示されます。ユーザーは [サマリ] タブの [アドレス] リンクをクリックして vSphere 向け Kubernetes CLI Tools を開き、名前空間を管理できます。ユーザーがスーパーバイザー名前空間の詳細へのリンクにアクセスするには、クラウド管理者となるか、指定されたプロジェクトの名前空間のメンバーになることが必要です。また、カスタマイズされた Kubectl をダウンロードして、スーパーバイザー名前空間を利用することも可能です。ユーザーは、スーパーバイザー名前空間にログインして、他の名前空間と同じように使用し、クラウド テンプレートを作成してアプリケーションを展開できます。

vRealize Automation Cloud Assembly のクラウド テンプレートへの Kubernetes コンポーネントの追加

Kubernetes コンポーネントを vRealize Automation Cloud Assembly クラウド テンプレートに追加するときに、クラスタを追加するか、ユーザーがさまざまな構成で名前空間を作成できるようにするかを選択できます。通常、この選択は、アクセス コントロールの要件、Kubernetes コンポーネントの構成方法、および展開の要件によって異なります。

vRealize Automation Cloud Assembly のクラウド テンプレートに Kubernetes コンポーネントを追加するには、[デザイン] - [クラウド テンプレート] の順に選択し、[新規] をクリックし、左側のメニューで Kubernetes オプションを検索して展開します。次に、クラスタまたは KBS 名前空間のいずれかをキャンバスにドラッグすることで選択を行います。

プロジェクトに関連付けられた Kubernetes クラスタをクラウド テンプレートに追加する方法は、有効なユーザーが Kubernetes リソースを使用できるようにするための最も効率的な方法です。他の Cloud Assembly リソースと同様に、クラスタでタグを使用して、展開する場所を制御できます。クラスタ展開の割り当てフェーズでは、タグを使用してゾーンと VMware Tanzu Kubernetes Grid Integrated Edition (TKGI) プランを選択できます。

この方法でクラスタを追加すると、有効なすべてのユーザーが自動的にクラスタを使用できるようになります。

クラウド テンプレートの例

最初のクラウド テンプレートの例は、タグ付けによって制御される単純な Kubernetes 展開のテンプレートを示しています。Kubernetes ゾーンは、[新しい Kubernetes ゾーン] 画面上で構成され、2 つの展開プランを使用して作成されています。この例では、`placement:tag` というタグがゾーンの機能として追加され、クラウド テンプレートの同様な制約との照合に使用されています。複数のゾーンがこのタグを使用して構成されている場合、優先順位の番号が最も小さいゾーンが選択されます。

```
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
```

2 番目のクラウド テンプレートの例では、ユーザーが展開を申請するときに目的のクラスタのホスト名を入力できるように、`$(input.hostname)` という変数を使用してテンプレートを設定する方法を示します。また、タグを使用して、クラスタ展開のリソース割り当てフェーズ中に、ゾーンと TKGI プランを選択することもできます。

```
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
```

名前空間を使用してクラスタ使用量を管理する場合は、クラウド テンプレートに `name: ${input.name}` という変数を設定できます。これは、展開の申請時にユーザーが入力する名前空間名を置き換えるために使用されます。このような展開では、テンプレートを次の例のように作成します。

```
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}

```

ユーザーは、[インフラストラクチャ] - [リソース] - [Kubernetes クラスタ] ページからアクセス可能な kubeconfig ファイルを使用して、展開されたクラスタを管理できます。目的のクラスタのページでカードを見つけて、[Kubeconfig] をクリックします。

VMware Cloud Templates のスーパーバイザー名前空間

vRealize Automation Cloud Assembly クラウド テンプレート内の基本的なスーパーバイザー名前空間のスキームは次のとおりです。

```

{
  "title": "Supervisor namespace schema",
  "description": "Request schema for provisioning of Supervisor namespace resource",
  "type": "object",
  "properties": {
    "name": {
      "title": "Name",
      "description": "Alphabetic (a-z and 0-9) string with maximum length of 63 characters. The character '-' is allowed anywhere except the first or last position of the identifier.",
      "type": "string",
      "pattern": "^[a-z0-9-]{1,63}$",
      "ignoreOnUpdate": true
    },
    "description": {
      "title": "Description",
      "description": "An optional description of this Supervisor namespace.",
      "type": "string",
      "ignoreOnUpdate": true
    },
    "constraints": {
      "title": "Constraints",
      "description": "To target the correct resources, blueprint constraints are matched against infrastructure capability tags. Constraints must include the key name. Options include value, negative [!], and hard or soft requirement.",
      "type": "array",
      "recreateOnUpdate": true,
      "items": {
        "type": "object",
        "properties": {
          "tag": {
            "title": "Tag",
            "description": "Constraint definition in syntax `[!]:[[:hard|soft]]` \nExamples:\n`location:eu:hard`\n`location:us:soft`\n`pci`",
            "type": "string",
            "recreateOnUpdate": true
          }
        }
      }
    },
    "limits": {
      "title": "Limits",

```

```

    "description": "Defines namespace resource limits such as pods, services, etc.",
    "type": "array",
    "recreateOnUpdate": false,
    "items": {
      "type": "object",
      "properties": {
        "stateful_set_count": {
          "title": "stateful_set_count",
          "description": "This represents the new value for 'statefulSetCount' option which
is the maximum number of StatefulSets in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "deployment_count": {
          "title": "deployment_count",
          "description": "This represents the new value for 'deploymentCount' option which
is the maximum number of deployments in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "cpu_limit_default": {
          "title": "cpu_limit_default",
          "description": "This represents the new value for the default CPU limit (in Mhz)
for containers in the pod. If specified, this limit should be at least 10 Mhz.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "config_map_count": {
          "title": "config_map_count",
          "description": "This represents the new value for 'configMapCount' option which
is the maximum number of ConfigMaps in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "pod_count": {
          "title": "pod_count",
          "description": "This represents the new value for 'podCount' option which is the
maximum number of pods in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "job_count": {
          "title": "job_count",
          "description": "This represents the new value for 'jobCount' option which is the
maximum number of jobs in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        },
        "secret_count": {
          "title": "secret_count",
          "description": "This represents the new value for 'secretCount' option which is
the maximum number of secrets in the namespace.",
          "type": "integer",
          "recreateOnUpdate": false
        }
      }
    }
  },

```

```

    "cpu_limit": {
      "title": "cpu_limit",
      "description": "This represents the new value for 'limits.cpu' option which is
equivalent to the maximum CPU limit (in MHz) across all pods in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "cpu_request_default": {
      "title": "cpu_request_default",
      "description": "This represents the new value for the default CPU request (in
Mhz) for containers in the pod. If specified, this field should be at least 10 MHz.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "memory_limit_default": {
      "title": "memory_limit_default",
      "description": "This represents the new value for the default memory limit (in
mebibytes) for containers in the pod.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "memory_limit": {
      "title": "memory_limit",
      "description": "This represents the new value for 'limits.memory' option which is
equivalent to the maximum memory limit (in mebibytes) across all pods in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "memory_request_default": {
      "title": "memory_request_default",
      "description": "This represents the new value for the default memory request (in
mebibytes) for containers in the pod.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "service_count": {
      "title": "service_count",
      "description": "This represents the new value for 'serviceCount' option which is
the maximum number of services in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "replica_set_count": {
      "title": "replica_set_count",
      "description": "This represents the new value for 'replicaSetCount' option which
is the maximum number of ReplicaSets in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "replication_controller_count": {
      "title": "replication_controller_count",
      "description": "This represents the new value for 'replicationControllerCount'
option which is the maximum number of ReplicationControllers in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    }
  }

```

```

    },
    "storage_request_limit": {
      "title": "storage_request_limit",
      "description": "This represents the new value for 'requests.storage' which is the
limit on storage requests (in mebibytes) across all persistent volume claims from pods in the
namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "persistent_volume_claim_count": {
      "title": "persistent_volume_claim_count",
      "description": "This represents the new value for 'persistentVolumeClaimCount'
option which is the maximum number of PersistentVolumeClaims in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    },
    "daemon_set_count": {
      "title": "daemon_set_count",
      "description": "This represents the new value for 'daemonSetCount' option which
is the maximum number of DaemonSets in the namespace.",
      "type": "integer",
      "recreateOnUpdate": false
    }
  },
  "additionalProperties": false
}
}
},
"required": [
  "name"
]
}

```

VMware Cloud Templates では、スーパーバイザー名前空間に関する制限の使用がサポートされています。制限を使用すると、CPU およびメモリのリソース使用量、および展開されるマシンによって名前空間で許可されるポッドの最大数を制御できます。

```

formatVersion: 1
inputs: {}
resources:
  Cloud_SV_Namespace_1:
    type: Cloud.SV.Namespace
    properties:
      name: '${env.deploymentName}'
      limits:
        - cpu_limit: 1000
          cpu_request_default: 800
          memory_limit: 2000
          memory_limit_default: 1500
          pod_count: 200

```

Kubernetes での vRealize Automation Cloud Assembly 拡張性の使用

vRealize Automation Cloud Assembly には、Kubernetes クラスターの展開に関連する一般的なアクションに対応する一連の標準イベント トピックが用意されています。ユーザーは必要に応じてこれらのトピックを購読でき、購読済みのトピックに関連するイベントが発生したときに通知を受け取ります。また、イベント通知に基づいて実行するように vRO ワークフローを構成することもできます。

次のトピックは、vRealize Automation Cloud Assembly の [拡張性] - [ライブラリ] - [イベント トピック] ページから購読できます。これらのトピックを表示するには、イベント トピックの検索テキスト ボックスで Kubernetes を検索します。

- Kubernetes クラスターの割り当て
- Kubernetes クラスターのプロビジョニング後
- Kubernetes クラスターの削除後
- Kubernetes クラスターのプロビジョニング
- Kubernetes クラスターの削除

いずれかのトピックをクリックして、収集および転送されたすべての情報を示すトピックのスキーマを表示します。このスキーマ情報のいずれかを使用して、さまざまな通知および管理タスクとレポート作成タスクを設定できます。

[拡張性] - [ライブラリ] - [アクション] ページで、CMX 関連アクションのアクション スクリプトを設定できます。アクション スクリプトは、さまざまな目的で使用できます。たとえば、Kubernetes クラスター プロビジョニングの DNS レコードを作成します。DNS レコードを作成している場合、アクション スクリプトで REST コマンドを使用して、Kubernetes クラスターのプロビジョニング後のトピックから `masternodeips` フィールドを使用して DNS レコードを作成できます。

[サブスクリプション] ページには、イベント トピックとアクション スクリプトの関係が定義されています。これらのコンポーネントは、vRealize Automation Cloud Assembly の [サブスクリプション] ページで表示および管理できます。

vRealize Automation Cloud Assembly の構成管理について

vRealize Automation Cloud Assembly では、展開の設定とエラーを管理できるよう、Puppet Enterprise、Ansible Open Source、および Ansible Tower との連携がサポートされています。

Puppet との連携

Puppet ベースの構成管理を統合するには、vSphere ワークロードを使用して、パブリック クラウドまたはプライベート クラウドにインストールされている Puppet Enterprise の有効なインスタスが必要です。この外部システムと vRealize Automation Cloud Assembly インスタンス間の接続を確立する必要があります。その後、Puppet 構成管理を、適切なブループリントに追加することによって、vRealize Automation Cloud Assembly で使用できるようになります。

vRealize Automation Cloud Assembly のブループリント サービスの Puppet プロバイダは、展開されたコンピューティング リソースに Puppet エージェントをインストールし、設定して、実行します。Puppet プロバイダでは、次の前提条件を満たした SSH と WinRM の両方の接続がサポートされます。

■ SSH 接続：

- ユーザー名は、スーパー ユーザー、または NOPASSWD でコマンドを実行する sudo 権限を持つユーザーである必要があります。
- 指定したユーザーに対して `requiretty` を無効にします。
- cURL が展開コンピューティング リソースで使用可能になっている必要があります。

■ WinRM 接続：

- PowerShell 2.0 が展開コンピューティング リソースで使用可能になっている必要があります。
- vRealize Orchestrator のドキュメントの説明どおりに、Windows テンプレートを構成します。

Puppet マスターへの接続の管理と、特定の展開への Puppet ロール（設定ルール）の適用は、DevOps 管理者が行います。展開の後、構成管理をサポートするよう設定された仮想マシンを、指定した Puppet マスターに登録します。

仮想マシンが展開されると、ユーザーは外部システムとして Puppet マスターを追加または削除することや、Puppet マスターに割り当てられたプロジェクトを更新することができます。最後に、適切なユーザーは、展開した仮想マシンが使用されなくなったときに、そのマシンを Puppet マスターから登録解除できます。

Ansible Open Source との連携

Ansible 連携を設定する場合は、Ansible Open Source を Ansible のインストール手順に基づいてインストールします。インストールの詳細については Ansible のドキュメントを参照してください。

Ansible ではデフォルトでホスト キーのチェックが有効になります。ホストを `known_hosts` ファイル内の別のキーで再インストールすると、エラー メッセージが表示されます。ホストが `known_hosts` ファイルに含まれていない場合は、起動時にキーを入力する必要があります。`/etc/ansible/ansible.cfg` または `~/.ansible.cfg` ファイルの次の設定を使用して、ホスト キーの確認を無効にすることができます。

```
[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
```

ホスト キー チェック エラーを回避するには、`host_key_checking` および `record_host_keys` を `False` に設定し、`ssh_args` で設定される追加オプション `UserKnownHostsFile=/dev/null` を追加します。また、インベントリが初期状態で空の場合、ホスト リストが空であることを警告するメッセージが Ansible によって表示されます。これにより、プレイブック構文チェックが失敗します。

Ansible Vault により、パスワードやキーなどの機密情報をプレーンテキストの形式ではなく、暗号化されたファイルに保存することが可能になります。Vault はパスワードで暗号化されています。vRealize Automation Cloud Assembly では、Ansible が Vault を使用することでホスト マシンの SSH パスワードなどのデータを暗号化します。これは、Vault パスワードへのパスが設定されていることが前提になります。

ansible.cfg ファイルを変更し、パスワード ファイルの場所を次の形式で指定できます。

```
vault_password_file = //パス/file.txt
```

Ansible がパスワードを自動的に検索するように ANSIBLE_VAULT_PASSWORD_FILE 環境変数を設定することもできます。例：ANSIBLE_VAULT_PASSWORD_FILE=~/.vault_pass.txt

vRealize Automation Cloud Assembly は、Ansible インベントリ ファイルを管理しているため、vRealize Automation Cloud Assembly ユーザーはインベントリ ファイルに rwx アクセスできるようにする必要があります。

```
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 のオープンソース統合で非 root ユーザーを使用する場合、ユーザーには、vRealize Automation Cloud Assembly オープンソース プロバイダが使用するコマンドを実行するための一連の権限が必要です。次のコマンドをユーザーの sudoers ファイルで設定する必要があります。

```
Defaults:myuser !requiretty
```

askpass アプリケーションが指定されていない管理者グループにユーザーが属していない場合は、ユーザーの sudoers ファイルで次のコマンドを実行します。

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

Ansible 統合の設定時にエラーなどの問題が発生した場合は、Ansible コントロール マシンの 'cat~/var/tmp/vmware/provider/user_defined_script/\${ls -t ~/var/tmp/vmware/provider/user_defined_script/ | head -1}/' にある log.txt ファイルを参照してください。

Ansible Tower の統合

サポートされているオペレーティング システムのタイプ

- Red Hat Enterprise Linux 8.0 以降の 64 ビット (x86) 版では、Ansible Tower 3.5 以降のみがサポートされます。
- Red Hat Enterprise Linux 7.4 以降の 64 ビット (x86) 版。
- CentOS 7.4 以降の 64 ビット (x86) 版。

次に、インベントリ ファイルの例を示します。このファイルは、Ansible Tower のインストール時に生成されます。vRealize Automation Cloud Assembly 統合で使用する場合は、変更が必要になることがあります。

```
[root@cava-env8-dev-001359 ansible-tower-setup-bundle-3.5.2-1.el8]# pwd

/root/ansible-tower-install/ansible-tower-setup-bundle-3.5.2-1.el8
```

```
[root@cava-env8-dev-001359 ansible-tower-setup-bundle-3.5.2-1.el8]# cat inventory

[tower]

localhost ansible_connection=local


[database]


[all:vars]

admin_password='VMware1!'


pg_host=''

pg_port=''


pg_database='awx'

pg_username='awx'

pg_password='VMware1!'


rabbitmq_port=5672

rabbitmq_vhost=tower

rabbitmq_username=tower

rabbitmq_password='VMware1!'

rabbitmq_cookie=cookiemonster


# Needs to be true for fqdns and ip addresses

rabbitmq_use_long_name=false
```



```
# Isolated Tower nodes automatically generate an RSA key for authentication;

# To disable this behavior, set this value to false

# isolated_key_generation=true
```

vRealize Automation Cloud Assembly での Puppet Enterprise 統合の設定

vRealize Automation Cloud Assembly では、Puppet Enterprise 構成管理との統合がサポートされます。

Puppet Enterprise を外部システムとして Cloud Assembly に追加すると、デフォルトではすべてのプロジェクトで Puppet Enterprise を使用できます。特定のプロジェクトに限定することもできます。

Puppet Enterprise 統合を追加するには、Puppet のマスター名と、マスターのホスト名または IP アドレスが必要です。

エラーまたは情報に関して Puppet ログを確認する必要がある場合は、次の場所を参照してください。

説明	ログの場所
作成およびインストールに関連するイベントのログ	ログは、展開されたマシンの <code>~/var/tmp/vmware/provider/user_defined_script/\$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ head -1)/`</code> にあります。 完全なログを確認するには、[log.txt] ファイルを参照してください。詳細な Puppet エージェント ログを確認するには、 https://puppet.com/docs/puppet/4.8/services_agent_unix.html#logging を参照してください。
Puppet の削除および実行に関連するタスクのログ	ログは、PE の <code>~/var/tmp/vmware/provider/user_defined_script/\$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ head -1)/`</code> にあります。完全なログを確認するには、[log.txt] ファイルを参照してください。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [Puppet] を選択します。
- 3 Puppet の設定画面で、必要な情報を入力します。
- 4 [検証] をクリックして、統合を検証します。
- 5 [追加] をクリックします。

結果

Puppet はクラウド テンプレートで使用できます。

次のステップ

目的のクラウド テンプレートに Puppet コンポーネントを追加します。

- 1 Cloud Assembly の [クラウド テンプレート] のクラウド テンプレート メニューで、[構成管理] の見出しの下にある [Puppet] を選択し、Puppet コンポーネントをキャンバスにドラッグします。

2 右側のペインで Puppet のプロパティを入力します。

プロパティ	説明
マスター	このクラウド テンプレートで使用する Puppet プライマリ マシンの名前を入力します。
環境	Puppet プライマリ マシンの環境を選択します。
ロール	このクラウド テンプレートで使用する Puppet のロールを選択します。
エージェント実行間隔	このクラウド テンプレートに関連する展開済みの仮想マシンに設定の詳細を適用するために、Puppet エージェントが Puppet プライマリ マシンをポーリングする頻度。

3 右側のペインの [コード] タブをクリックすると、Puppet 設定プロパティの YAML コードが表示されます。

vRealize Automation Cloud Assembly で Ansible オープン ソースとの統合を設定する

vRealize Automation Cloud Assembly は、Ansible オープン ソースの構成管理との統合をサポートします。統合の設定後、新しい展開または既存の展開に Ansible コンポーネントを追加できます。

Ansible オープン ソースを vRealize Automation Cloud Assembly と統合すると、新しいマシンをプロビジョニングして構成管理を自動化するときに、1 つ以上の Ansible プレイブックを指定した順序で実行するように設定できます。クラウド テンプレートに、展開に必要な Playbook を指定します。

Ansible との統合を設定するには、リソース管理に関する情報を定義するインベントリ ファイル パスとともに、Ansible オープン ソース ホスト マシンを指定する必要があります。さらに、Ansible オープン ソース インスタンスにアクセスするための名前とパスワードを指定する必要があります。後で、展開に Ansible コンポーネントを追加するときに、キーベースの認証を使用するように接続を更新できます。

デフォルトでは、Ansible は SSH を使用して物理マシンに接続します。クラウド テンプレートで osType Windows プロパティによって指定されているとおりに Windows マシンを使用している場合、connection_type 変数は自動的に winrm に設定されます。

最初は、Ansible 統合は統合で提供されるユーザー/パスワードまたはユーザー/キー認証情報を使用して、Ansible 制御マシンに接続します。接続が成功すると、クラウド テンプレート内の指定された Playbook が構文に対して検証されます。

検証に成功すると、Ansible 制御マシンの `~/var/tmp/vmware/provider/user_defined_script/` に実行フォルダが作成されます。このフォルダからスクリプトが実行されて、インベントリにホストが追加され、ホストに接続するための認証モードの設定を含むホスト変数ファイルが作成されて、最後に Playbook が実行されます。この時点で、クラウド テンプレートで指定された認証情報を使用して、Ansible 制御マシンからホストに接続します。

Ansible 統合は、IP アドレスを使用しない物理マシンをサポートします。AWS、Azure、GCP などのパブリッククラウドにプロビジョニングされたマシンの場合、作成されたリソースの address プロパティは、マシンがパブリックネットワークに接続されているときにのみマシンのパブリック IP アドレスに設定されます。パブリックネットワークに接続されていないマシンの場合、Ansible 統合はマシンが接続されているネットワークから IP アドレス

を探します。複数のネットワークに接続されている場合は、deviceIndex が最小のネットワークを探します。この値は、マシンに接続されているネットワーク インターフェイス カード (NIC) のインデックス番号です。deviceIndex プロパティがブループリントで指定されていない場合は、最初に接続されたネットワークが使用されます。

vRealize Automation Cloud Assembly での統合のために Ansible Open Source を設定する方法の詳細については、[vRealize Automation Cloud Assembly の構成管理について](#)を参照してください。

前提条件

- Ansible 制御マシンでは、2.6.0 以降の Ansible バージョンを使用する必要があります。
- ユーザーは、使用している Ansible インベントリ ファイルが配置されているディレクトリに対する読み取りまたは書き込みアクセス権を持っている必要があります。また、すでにインベントリ ファイルがある場合は、ここへの読み取りまたは書き込みアクセス権がユーザーに付与されている必要があります。
- 非 root ユーザーで sudo オプションが使用できる場合は、sudoers ファイルに次の行が設定されていることを確認します。

```
Defaults:user_name !requiretty
```

および

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

- /etc/ansible/ansible.cfg または ~/.ansible.cfg に host_key_checking = False が設定され、ホスト キーのチェックが無効になっていることを確認します。
- 次の行を /etc/ansible/ansible.cfg または ~/.ansible.cfg ファイルに追加して、Vault のパスワードが設定されていることを確認します。

```
vault_password_file = /path/to/password_file
```

Vault パスワード ファイルには、プレーン テキスト形式のパスワードが含まれています。Vault パスワード ファイルが使用されるのは、ACM とノード間で使用されるユーザー名とパスワードの組み合わせが、クラウド テンプレートまたは展開で提供される場合のみです。次に例を示します。

```
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
```

- Playbook の実行時にホスト キーの障害の発生を回避するには、/etc/ansible/ansible config に次の設定を含めることをお勧めします。

```
[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.
```

手順

1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。

2 [Ansible] をクリックします。

Ansible の設定画面が表示されます。

3 ホスト名、インベントリ ファイルのパス、および Ansible オープン ソース インスタンスに必要なその他の情報を入力します。

4 [検証] をクリックして、統合を検証します。

5 [追加] をクリックします。

結果

Ansible はクラウド テンプレートで使用できます。

次のステップ

目的のクラウド テンプレートに Ansible コンポーネントを追加します。

1 [クラウド テンプレート キャンバス] 画面のクラウド テンプレート オプション メニューで、[構成管理] の見出しの下にある [Ansible] を選択し、Ansible コンポーネントをキャンバスにドラッグします。

2 右側のパネルで、実行する Playbook の指定など、適切な Ansible プロパティを構成します。

Ansible では、ユーザーは 1 台のホストに変数を割り当て、後からプレイブックで使用できます。Ansible Open Source 統合により、クラウド テンプレートでこれらのホスト変数を指定できます。hostVariables プロパティは、Ansible 制御マシンで想定されている適切な YAML 形式である必要があります。この内容は、次の場所に配置されます。

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

Ansible インベントリ ファイルのデフォルトの場所は、Cloud Assembly の [統合] ページで追加された Ansible アカウントで定義されています。Ansible 統合ではクラウド テンプレートの hostVariable YAML 構文は検証されませんが、フォーマットや構文に誤りがあると、Playbook の実行時に Ansible コントロール マシンでエラーが発生します。

次のクラウド テンプレートの YAML スニペットは、hostVariables プロパティの使用方法的例を示しています。

```
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}
```

Ansible 統合では、次のいずれかの方法で、クラウド テンプレートに認証情報があると想定しています。

- Ansible リソースのユーザー名とパスワード。
- Ansible リソースのユーザー名と privateKeyFile。
- generatedPublicPrivateKey に remoteAccess を指定することによる、Ansible リソースのユーザー名とコンピューティング リソースの privatekey。

クラウド テンプレートで、統合アカウントで指定されているユーザーが Ansible Playbook へのパスにアクセスできることを確認します。絶対パスを使用して Playbook の場所を指定できますが、必須ではありません。ユーザーのホーム フォルダへの絶対パスを指定することをお勧めします。そうすることで、将来的に Ansible 統合の認証情報が変更されても、パスが有効なまま維持されます。

vRealize Automation Cloud Assembly での Ansible Tower 統合の設定

Ansible Tower を vRealize Automation Cloud Assembly に統合すると、展開されたリソースの構成管理をサポートできます。統合の設定後、クラウド テンプレート エディタを使用して、新しい展開または既存の展開に Ansible コンポーネントを追加できます。

vRealize Automation Cloud Assembly は、Ansible Tower バージョン 3.5、3.6、および 3.7 との統合をサポートしています。

前提条件

- 管理者以外のユーザーに、Ansible Tower へのアクセスに適切な権限を付与します。ほとんどの構成で機能する 2 つのオプションがあります。構成に最適なオプションを選択してください。
 - 組織レベルでインベントリ管理者ロールおよびジョブ テンプレート管理者ロールをユーザーに付与します。
 - 特定のインベントリに対する管理者権限と、プロビジョニングに使用するすべてのジョブ テンプレートに対する実行ロールをユーザーに付与します。
- 展開で使用する認証情報とテンプレートを Ansible Tower で適切に設定する必要があります。テンプレートでは、展開で使用するインベントリと Playbook を定義します。ジョブ テンプレートと Playbook は 1:1 でマッピングされます。Playbook では、YAML に似た構文を使用して、テンプレートに関連付けられたタスクを定義します。最も一般的な展開では、認証にマシンの認証情報を使用します。
 - a Ansible Tower にログインして、[ジョブ テンプレート] セクションに移動します。
 - b [新規ジョブ テンプレートの追加] を選択します。
 - すでに作成されている認証情報を選択します。これらは、Ansible Tower で管理されるマシンの認証情報です。それぞれのジョブ テンプレートでは、認証情報オブジェクトを 1 つ使用できます。

- [制限] の選択項目では [起動時にプロンプトを表示] を選択します。これにより、vRealize Automation Cloud Assembly からプロビジョニングまたはプロビジョニング解除されるノードに対して、ジョブ テンプレートが実行されます。このオプションが選択されていない場合は、ジョブ テンプレートを含むブループリントを展開するときに、「制限が設定されていない」という旨のエラーが表示されます。
- vRealize Automation Cloud Assembly から起動されたジョブ テンプレートの実行は、[Ansible Tower ジョブ] タブで確認できます。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 [Ansible Tower] をクリックします。
Ansible の設定画面が表示されます。
- 3 Ansible Tower インスタンスの [ホスト名] (IP アドレスも可) とその他の必須情報を入力します。
- 4 該当する Ansible Tower インスタンスのユーザー インターフェイス ベース認証の [ユーザー名] と [パスワード] を入力します。
- 5 [検証] をクリックして、統合を検証します。
- 6 統合の [名前] と [説明] を適切に入力します。
- 7 [追加] をクリックします。

結果

Ansible Tower はクラウド テンプレートで使用できます。

次のステップ

目的のクラウド テンプレートに Ansible Tower コンポーネントを追加します。統合アカウントで指定されたユーザーに対して、実行権限を含む適切なジョブ テンプレートを指定します。

- 1 [クラウド テンプレート キャンバス] 画面のブループリント オプション メニューで、[構成管理] の見出しの下にある [Ansible] を選択し、Ansible Tower コンポーネントをキャンバスにドラッグします。
- 2 右側のパネルで、ジョブ テンプレートなどの該当する Ansible プロパティを構成します。

vRealize Automation Cloud Assembly で Active Directory 統合を作成する方法

vRealize Automation Cloud Assembly は、Active Directory サーバとの統合をサポートしており、仮想マシンをプロビジョニングする前に、指定された組織単位 (OU) 内のコンピュータ アカウントが Active Directory サーバ内に最初から作成されます。Active Directory では、Active Directory サーバへの LDAP 接続がサポートされています。

プロジェクトに関連付けられている Active Directory ポリシーは、そのプロジェクトの範囲内にプロビジョニングされるすべての仮想マシンに適用されます。ユーザーは 1 つ以上のタグを指定することにより、一致する機能タグを持つクラウド ゾーンにプロビジョニングされる仮想マシンにポリシーを選択的に適用できます。

オンプレミス展開では、Active Directory 統合により、統合とその基盤となる ABX 統合（必要な拡張クラウド プロキシなど）のステータスを示す、健全性チェック機能を設定できます。Active Directory ポリシーを適用する前に、vRealize Automation Cloud Assembly によって、基盤となる統合のステータスがチェックされます。統合が健全な場合、vRealize Automation Cloud Assembly は、指定された Active Directory で展開されるコンピュータ オブジェクトを作成します。統合が健全でない場合、展開操作ではプロビジョニング中に Active Directory フェーズがスキップされます。

前提条件

- Active Directory 統合では、Active Directory サーバへの LDAP 接続が必要です。
- vCenter オンプレミスと Active Directory の統合を構成する場合は、拡張クラウド プロキシとの ABX 統合を構成する必要があります。[拡張性] - [アクティビティ] - [統合] の順に選択し、[オンプレミスの拡張性アクション] を選択します。
- クラウドでの Active Directory との統合を設定する場合は、Microsoft Azure または Amazon Web Services アカウントが必要です。
- 適切なクラウド ゾーンで設定されたプロジェクトと、Active Directory 統合で使用するイメージとフレーバーのマッピングが必要です。
- Active Directory 統合をプロジェクトに関連付ける前に、Active Directory 上に目的の OU を事前に作成しておく必要があります。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[新しい統合] をクリックします。
- 2 [Active Directory] をクリックします。
- 3 [サマリ] タブで、適切な LDAP ホスト名と環境名を入力します。
- 4 LDAP サーバの名前とパスワードを入力します。
- 5 Active Directory 内の目的のユーザーおよびグループの適切なベース識別名 (DN) を入力します。

注： 各 Active Directory 統合ごとに 1 つの識別名 (DN) のみを指定できます。

- 6 [検証] をクリックして、統合が機能することを確認します。
- 7 この統合の名前と説明を入力します。
- 8 [保存] をクリックします。
- 9 [プロジェクト] タブをクリックして、Active Directory 統合にプロジェクトを追加します。
[プロジェクトの追加] ダイアログで、プロジェクト名と、[サマリ] タブで指定したベース識別名 (DN) 内にある相対識別名 (DN) を選択する必要があります。
- 10 [保存] をクリックします。

結果

これで、Active Directory 統合を含むプロジェクトをクラウド テンプレートに関連付けられるようになりました。このクラウド テンプレートを使用してマシンをプロビジョニングすると、マシンは指定された Active Directory および組織単位に事前にステージングされます。

オンプレミスの Active Directory 統合に対するタグベースの健全性チェックを、次のように実装することもできます。

- 1 前の手順で説明したように、Active Directory 統合を作成します。
- 2 [プロジェクト] タブをクリックして、Active Directory 統合にプロジェクトを追加します。
- 3 プロジェクト名を選択し、[プロジェクトの追加] ダイアログで相対 DN を選択します。相対 DN は、指定したベース DN 内に存在している必要があります。
- 4 適切なタグを追加します。これらのタグは、Active Directory ポリシーを適用できるクラウド ゾーンに適用できます。
- 5 [保存] をクリックします。

Active Directory 統合のステータスは、vRealize Automation Cloud Assembly の [インフラストラクチャ] - [接続] - [統合]画面の各統合に表示されます。

Active Directory 統合を含むプロジェクトをクラウド テンプレートに関連付けることができます。このテンプレートを使用してマシンをプロビジョニングすると、マシンは指定された Active Directory および OU に事前にステージングされます。

VMware SDDC Manager 統合の構成

vRealize Automation に VMware SDDC Manager 統合を追加することにより、ワークロード ドメインを vRealize Automation 内の VMware Cloud Foundation (VCF) クラウド アカウントの一部として使用することができます。

前提条件

- vRealize Automation では、VMware SDDC Manager 4.1 以降との統合のみがサポートされます。

手順

- 1 [インフラストラクチャ] - [接続] - [統合] の順に選択して、[統合の追加] をクリックします。
- 2 SDDC Manager を選択します。
SDDC Manager 統合の構成画面が表示されます。
- 3 [サマリ] セクションで、統合の [名前] と [説明] を入力します。
- 4 [SDDC Manager 認証情報] セクションで、SDDC Manager サーバ マシンの [SDDC Manager の IP アドレス/FQDN] を入力します。

- 5 SDDC Manager への最初の接続に使用する管理者アカウントのユーザー名とパスワードを入力します。ベストプラクティスとして、接続に管理者アカウントは使用しないようにします。サービス ロールを作成する場合は、SDDC Manager で管理者権限を持つ別のアカウントを使用します。

SDDC Manager への接続を最初にセットアップする場合は、これらの認証情報が使用されます。その後、VCF クラウド アカウントから接続するときに、サービスの認証情報が作成されて使用されます。

- 6 [検証] をクリックして、SDDC Manager への接続を検証します。

- 7 [追加] をクリックします。

結果

統合が作成されたら、完了した統合画面に表示される [ワークロード ドメイン] タブで、SDDC に関連付けられているワークロードを表示できます。また、統合に関連付けられているワークロードを表示して選択し、[クラウド アカウントの追加] ボタンをクリックして、選択したワークロードを使用する VCF クラウド アカウントの作成画面を開くこともできます。

次のステップ

VCF クラウド アカウントを構成すると、[クラウドのセットアップ] ボタンが画面の上部に表示されます。このボタンをクリックして、VCF クラウド セットアップ ウィザードを開始します。

vRealize Operations Manager との統合

vRealize Automation は vRealize Operations Manager と連携して、高度なワークロード配置の実行、展開の健全性と仮想マシンのメトリックの提供、および価格の表示を行うことができます。

統合の数とタイプ

2 つの製品間の統合は、オンプレミスとクラウドを混在させるのではなく、オンプレミス間で行う必要があります。

1 つの vRealize Automation インスタンスを複数の vRealize Operations Manager インスタンスと統合できますが、vRealize Operations Manager インスタンスは 1 つの vRealize Automation インスタンスにのみ接続できます。

集約された vRealize Operations Manager クラスタを vRealize Automation に接続することはできません。

統合するための基本的な要件

vRealize Operations Manager と統合するには、[インフラストラクチャ] - [接続] - [統合] の順に移動します。統合を追加するには、次のセクションで説明されているログイン アカウントの vRealize Operations Manager URL と認証情報が必要です。さらに、vRealize Automation と vRealize Operations Manager は同じ vSphere エンドポイントを管理する必要があります。

統合のためのログイン アカウント

vRealize Operations Manager では、統合で使用するローカルまたは非ローカルの vRealize Operations Manager ログイン アカウントが必要です。このアカウントには、vSphere エンドポイントの vCenter Server アダプタ インスタンスへの読み取り専用権限が必要です。非ローカルのアカウントは、vRealize Operations Manager へのインポートと読み取り専用のロールの割り当てが必要になる場合があります。統合では、非ローカルアカウント ログインのユーザー名の形式は `username@domain@authenticated-source` (jdoe@company.com@workspaceone など) です。認証されるソースは、vRealize Operations Manager サーバの初期セットアップ時に定義されます。

詳細については、次のセクションを参照してください。価格については、[価格設定カードとは](#)を参照してください。

vRealize Operations Manager を使用した高度なワークロード配置

vRealize Automation および vRealize Operations Manager を連携して、展開ワークロードを最適に配置できます。

ワークロード配置は、vSphere ベースのクラウド ゾーン レベルで有効にします。vRealize Operations Manager を使用した高度な配置の対象となるのは、クラウド ザーンの Distributed Resource Scheduler (DRS) が有効なクラスタのみです。

- **vRealize Automation 配置** : vRealize Automation 配置エンジンは、アプリケーション インテントに基づいています。タグベースの制約、プロジェクトのメンバーシップと関連クラウド ザーン、およびネットワーク、ストレージ、コンピューティングに関連するアフィニティ フィルタが考慮されます。リソースの配置は、これらのすべての要素と、同じ展開内の他の関連するターゲット リソースの存在に依存します。
- **vRealize Operations Manager 配置** : vRealize Operations Manager は、最適な配置のために運用インテントを考慮します。運用インテントは、過去のワークロードと将来の what-if 予測を考慮することができます。

高度なワークロード配置を使用する場合は、vRealize Operations Manager のビジネス インテント オプションを使用するのではなく、vRealize Automation タグ付けを適用することによってビジネス インテントの決定を実装する必要があります。

vRealize Operations Manager と統合する場合、vRealize Automation は引き続きアプリケーション インテント モデルとその関連制約に従って、ターゲットの配置をフィルタリングします。その後、それらの結果から vRealize Operations Manager の推奨事項に従って、より詳細に配置を調整します。

推奨事項がない場合

高度なワークロード配置を有効にし、vRealize Operations Manager 分析で推奨事項が返されない場合は、vRealize Automation を構成して、デフォルトのアプリケーション インテントの配置にフォールバックすることができます。

ワークロード配置の制限事項

vRealize Operations Manager を使用してワークロードを配置する場合、特定の制限が適用されます。

- vRealize Operations Manager は、vCenter Server でリソース プールへのワークロード配置をサポートしていません。

- vRealize Operations Manager がダウンすると、ワークロード配置で vRealize Operations Manager を呼び出すときに使用されるタイムアウトが期限切れになる場合があります。
- 配置は複数のクラウド ゾーンにまたがりません。vRealize Automation は、vRealize Operations Manager に 1 つのクラウド ゾーンを送信し、その単一のクラウド ゾーン内での配置の推奨を行います。

ワークロード配置を有効にする方法

ワークロード配置を有効にするには、vSphere、vRealize Operations Manager、および vRealize Automation で実行する手順があります。

- 1 vRealize Automation Cloud Assembly で、vCenter Server クラウド アカウントに接続します。
オプションは、[インフラストラクチャ] - [接続] - [クラウド アカウント] の下にあります。
- 2 vCenter Server で、DRS が有効なクラスタが存在し、完全自動化に設定されていることを確認します。
- 3 vRealize Operations Manager で、同じ vCenter Server が管理されていることを確認します。
vRealize Operations Manager 8 以降が必要です。
- 4 vRealize Automation Cloud Assembly で、vRealize Operations Manager 統合を追加します。
オプションは、[インフラストラクチャ] - [接続] - [統合] の下にあります。

統合を追加するには、その下にある vRealize Operations Manager プライマリ ノードの URL に加え、ログイン ユーザー名とパスワードを入力する必要があります。

`https://operations-manager-IP-address-or-FQDN/suite-api`

値を入力したら、[検証] をクリックします。
- 5 [同期] をクリックして、vCenter Server との統合を同期します。

また、vRealize Automation Cloud Assembly と vRealize Operations Manager が新しい vCenter Server の管理を開始するときには、必ず同期を実行します。
- 6 vRealize Automation Cloud Assembly で、vCenter Server アカウント用のクラウド ゾーンを作成します。

オプションは、[インフラストラクチャ] - [構成] - [クラウド ゾーン] の下にあります。
- 7 クラウド ゾーンの [サマリ] タブで、配置ポリシーを [詳細] に設定します。
- 8 vRealize Operations Manager が推奨事項を返さない場合は、配置ポリシーで、vRealize Automation をデフォルトの配置にフォールバックするかどうかを選択します。

ワークロード配置のトラブルシューティング

vRealize Operations Manager が想定どおりにワークロードの配置を推奨しない場合は、vRealize Automation Cloud Assembly または vRealize Automation Service Broker の展開申請の詳細を確認してください。

- 1 [インフラストラクチャ] - [アクティビティ] - [申請] に移動して、申請をクリックします。
- 2 [申請の詳細] で、割り当てフェーズを確認します。

正常に識別されたか失敗したターゲットを探します。

- 3 [申請の詳細] の右上で、[開発モード] を有効にします。
- 4 申請パスに沿ってフィルタ ブロックを見つけます。
- 5 フィルタ ブロックをクリックし、次のセクションを確認します。

```
filterName: ComputePlacementPolicyAffinityHostFilter
  v computeLinksBefore
  v computeLinksAfter
  v filteredOutHostsReasons
```

エントリ	説明
computeLinksBefore	vRealize Automation アルゴリズムに基づく潜在的な配置ホストのリスト。
computeLinksAfter	選択した配置ホスト。
filteredOutHostsReasons	ホストが選択または拒否された理由を示すメッセージ。 vRealize Operations Manager がホストを選択すると、次のメッセージが表示されます。 advance policy filter: Filtered hosts based on recommendation from vROPS.

vRealize Operations Manager を使用した継続的な最適化

vRealize Operations Manager で vRealize Automation アダプタを追加すると、vRealize Operations Manager は自動的に vRealize Automation ベースのワークロード用の新しいカスタム データセンター (CDC) を作成します。

継続的な最適化では、ワークロードの再調整および再配置を活用し、初期状態のワークロード配置に縛られることなく、vRealize Automation を vRealize Operations Manager と組み合わせて使用できます。仮想化リソースはさまざまな大きさの負荷で利用されるため、vRealize Automation でプロビジョニングされるワークロードは必要に応じて移動します。

- 継続的な最適化では、vRealize Operations Manager 内に新しい CDC が自動的に作成されます。
vRealize AutomationvSphere クラウド ゾーンごとに 1 つ、新しい CDC があります。
- 新しく作成された CDC では、管理対象の各 vRealize Automation クラスタがクラウド ゾーンに関連付けられています。

注: vRealize Automation と非 vRealize Automation のクラスタが混在する CDC を手動で作成しないでください。

- vRealize Operations Manager を使用して、新しく作成された vRealize Automation ベースの CDC の最適化を継続的に実行します。
- ワークロードを再調整したり、再配置したりできるのは、同じクラウド ゾーン内または CDC 内に限られます。
- 最適化によって vRealize Automation または vRealize Operations Manager の配置違反が新しく発生することはありません。
 - 既存の配置違反がある場合、最適化によって vRealize Operations Manager の運用インテントの問題を修正できます。

- 既存の配置違反がある場合、最適化によって vRealize Operations Manager のビジネス インテントの問題は修正できません。

たとえば、vRealize Operations Manager を使用して、制約がサポートされないクラスタに仮想マシンを手動で移動した場合、vRealize Operations Manager は違反を検出せず、解決も試みません。

- このリリースは、CDC レベルで運用インテントに従います。すべてのメンバー vRealize Automation クラスタが同じ設定で最適化されます。

クラスタごとに異なる運用インテントを設定するには、別の vSphere クラウド ゾーンに関連付けられている別の vRealize Automation CDC 内にクラスタを構成する必要があります。テスト用と本番環境用のクラスタを分けることは、1つの例です。

- vRealize Automation アプリケーション インテント、および vRealize Automation に定義されている制約は、最適化の再調整または再配置の操作時に遵守されます。
- vRealize Operations Manager の配置タグは、vRealize Automation でプロビジョニングされるワークロードには適用できません。

また、複数のマシンに関わる最適化をスケジュール設定によって実行できます。定期的なスケジュールが設定された最適化は、ゼロか全体かというプロセスではありません。状況によってマシンの移動が中断された場合は、再配置が成功したマシンはその再配置が維持され、vRealize Operations Manager の通常の動作として、次の vRealize Operations Manager サイクルで残りの再配置が試みられます。このように、最適化が部分的に完了した場合に vRealize Automation に対して悪影響を及ぼすことはありません。

継続的な最適化を有効にする方法

vRealize Operations Manager で vRealize Automation アダプタを追加すると、vRealize Operations Manager は自動的に vRealize Automation ベースのワークロード専用の新しいデータセンターを作成します。

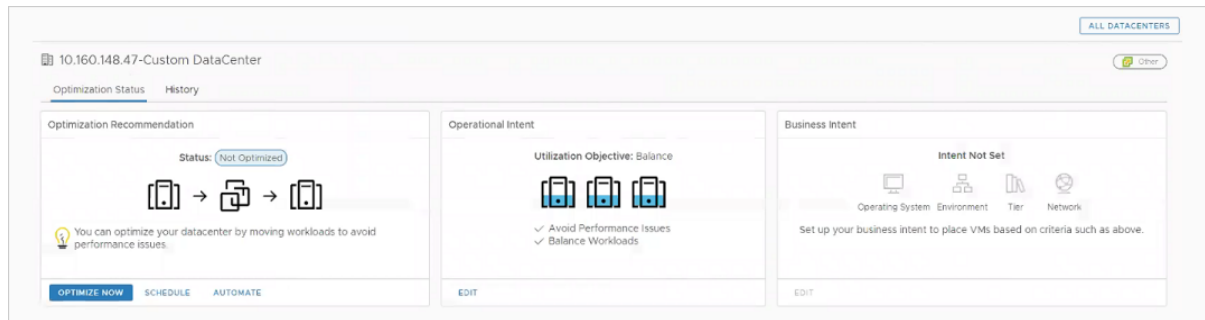
vRealize Automation Cloud Assembly 内で統合を追加する以外には、継続的な最適化のための独立したインストール手順はありません。新しいデータセンター内で、ワークロードの再配置のために vRealize Operations Manager の設定と使用を開始できます。[継続的な最適化の例](#) を参照してください。

継続的な最適化の例

次の例は、vRealize Operations Manager による vRealize Automation の継続的な最適化でリバランスを行うワークフローを示します。

- 1 vRealize Operations Manager のホーム ページで、[ワークロードの最適化] をクリックします。
- 2 自動的に作成された vRealize Automation データセンターを選択します。
- 3 [運用インテント] の下で [編集] をクリックし、[負荷分散] を選択します。

データセンターが vRealize Automation の最適化のためのものである場合、ビジネス インテントは無効なため、選択または編集できません。



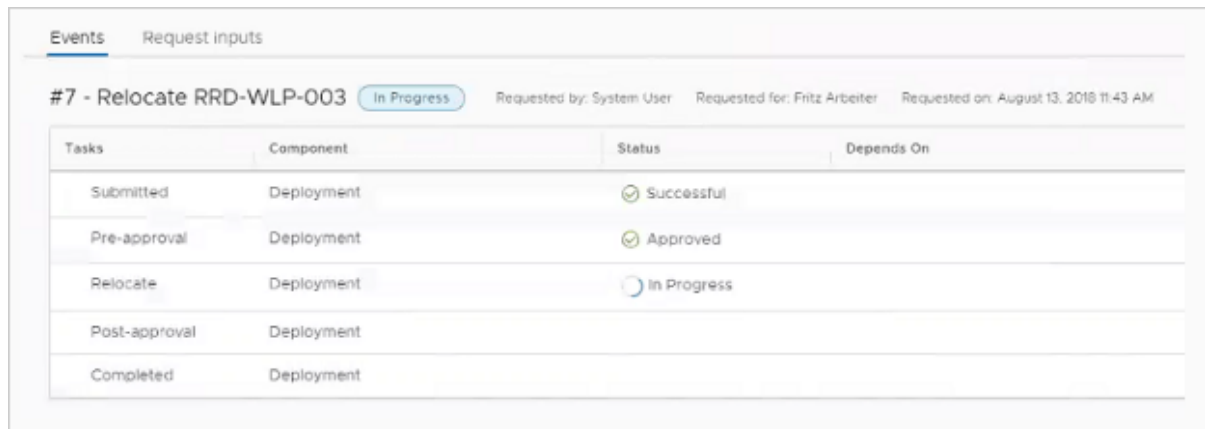
- 4 [最適化の推奨] の下で、[今すぐ最適化] をクリックします。

提案された処理の前と後を示す図が vRealize Operations Manager によって表示されます。

- 5 [次へ] をクリックします。

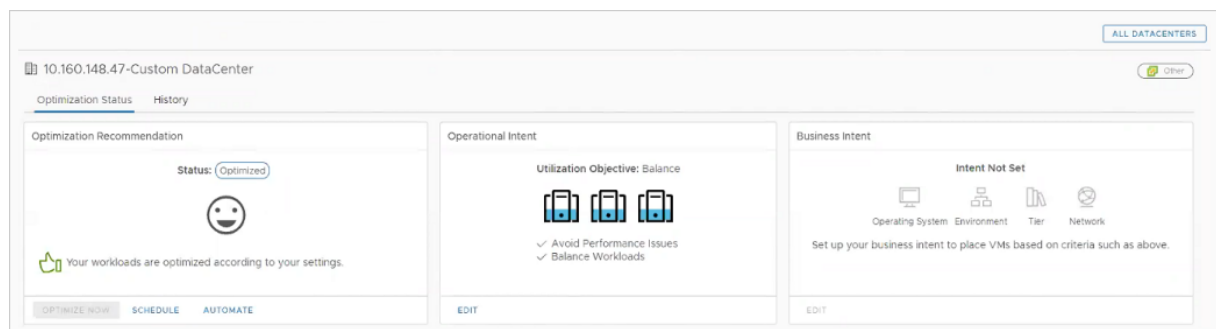
- 6 [アクションの開始] をクリックします。

- 7 vRealize Automation で、[展開] をクリックし、イベントのステータスを観察することにより、進行中の処理を監視します。



リバランスが完了すると、vRealize Automation の表示が更新されます。[コンピューティング リソース] ページに、マシンが移動したことが示されます。

vRealize Operations Manager では、次のデータ収集によって表示が更新され、最適化が完了したことが示されます。



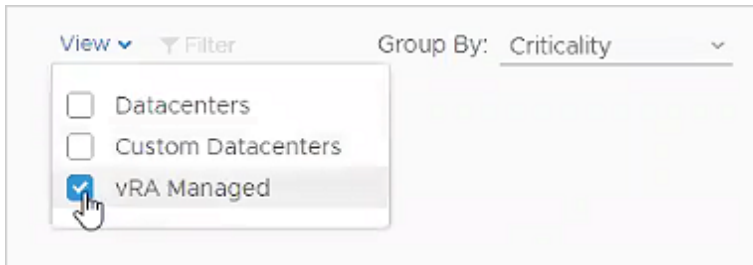
vRealize Operations Manager で処理を確認するには、[管理] - [履歴] - [最近のタスク] の順にクリックします。

管理対象の vRealize Automation データセンターの特定

vRealize Operations Manager を使用して、管理対象の vRealize Automation データセンターだけを表示できます。

手順

- 1 vRealize Operations Manager のホーム ページで、[ワークロードの最適化] をクリックします。
- 2 右側の上部にある [表示] ドロップダウンをクリックします。
- 3 管理対象の vRealize Automation データセンターのみを選択します。



vRealize Operations Manager に基づく展開の監視

vRealize Automation では、展開に関する vRealize Operations Manager データを表示できます。

フィルタされたセットの メトリックを vRealize Automation で直接確認すると、vRealize Operations Manager へのアクセスまたは検索のタスクを実施する必要がありません。その場合、vRealize Operations Manager のコンテキストに沿って起動することはできませんが、必要に応じて vRealize Operations Manager にログインして追加のデータを使用できます。

vRealize Operations Manager データの有効化

vRealize Automation に vRealize Operations Manager データを表示するには、vRealize Operations Manager 統合を追加します。

手順

- 1 vRealize Operations Manager で、[管理] - [ソリューション] の順に移動します。
- 2 [設定済みのアダプタ インスタンス] で、vRealize Automation によってプロビジョニングされる vSphere クラウド ゾーン用の **vCenter Server アダプタ** があることと、それがデータを受信していることを確認します。
- 3 vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [接続] - [統合] に移動します。
- 4 vRealize Operations Manager プライマリ ノードの URL と、vRealize Operations Manager のログイン ユーザー名およびパスワードを入力します。

`https://operations-manager-IP-address-or-FQDN/suite-api`

- 5 [展開] をクリックし、展開を選択して、[監視] タブが表示されることを確認します。

vRealize Operations Manager によって提供される健全性とアラート

監視を有効にすると、vRealize Automation は展開に関する vRealize Operations Manager の健全性および関連アラートを取得します。

監視にアクセスするには、展開をクリックし、[監視] タブを選択します。タブが表示されない場合は、[vRealize Operations Manager データの有効化](#)を参照してください。

アラートを確認するには、左側パネルのコンポーネント ツリーが一番上に表示されている展開名を選択します。

- アラートの重要度とテキストを確認できます。
- 懸念のある領域に注目するには、列のデータをフィルタリングしたりソートしたりします。
- 表示されるのは、健全性バッジと健全性アラートのみです。効率性、リスクなどの他のアラート タイプはサポートされていません。

vRealize Operations Manager によって提供されるメトリック

監視を有効にすると、vRealize Automation は展開に関する vRealize Operations Manager メトリックを取得します。

監視にアクセスするには、展開をクリックし、[監視] タブを選択します。タブが表示されない場合は、[vRealize Operations Manager データの有効化](#)を参照してください。

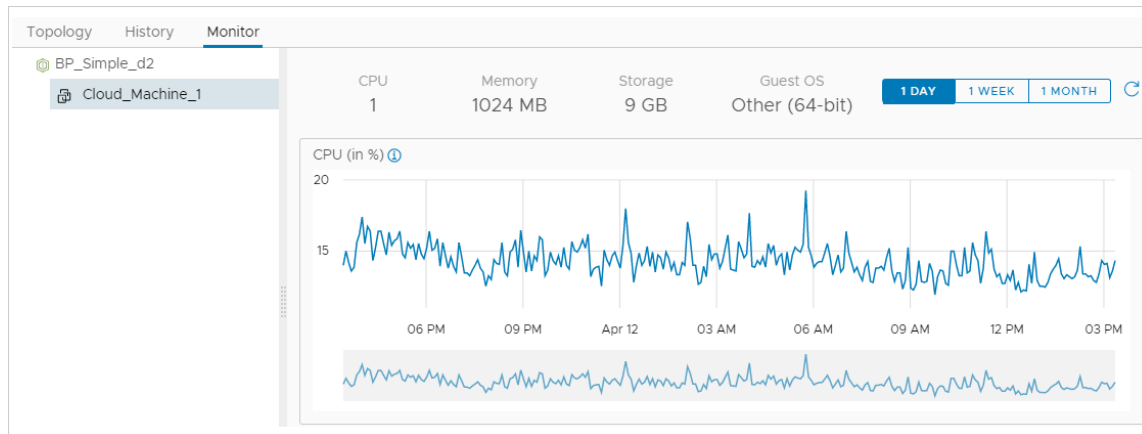
メトリックを表示するには、左側にあるコンポーネント ツリーを展開し、仮想マシンをハイライト表示します。

- メトリックはキャッシュされません。vRealize Operations Manager から直接取得されるため、ロードにしばらくかかる場合があります。
- 仮想マシンのメトリックのみが表示されます。vCloud Director、ソフトウェア、XaaS などの他のコンポーネントのメトリックはサポートされていません。
- vSphere 仮想マシンのメトリックのみが表示されます。AWS や Azure などの他のクラウド プロバイダはサポートされていません。

メトリックは、以下の測定値の最高値と最低値を示すタイムライン グラフとして表示されます。

- CPU
- メモリ
- ストレージの IOPS
- ネットワークの MBPS

特定のメトリックの名前を確認するには、タイムラインの左上隅にある青い情報アイコンをクリックします。

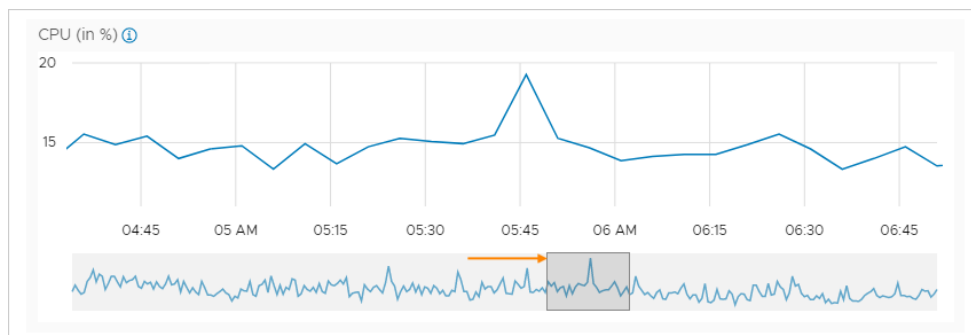


vRealize Operations Manager によって提供されるデータに基づくアクション

vRealize Operations Manager によって提供されるメトリックに問題がある場合は、問題のある領域を vRealize Automation で直接特定できます。

vRealize Operations Manager によって提供されるメトリックを表示するには、展開をクリックし、[監視] タブを選択します。タブが表示されない場合は、[vRealize Operations Manager データの有効化](#)を参照してください。

メトリックは、過去の日、週、または月について提供されます。問題のある領域を拡大表示するには、メトリックのタイムラインの下側にある、淡色表示された小さい領域を選択します。



vRealize Automation Cloud Assembly でのオンボーディング プランについて

ワークロードのオンボーディング プランを使用して、ターゲット リージョンまたはデータセンターのクラウド アカウント タイプからデータ収集されたものの、まだ vRealize Automation Cloud Assembly プロジェクトで管理されていないマシンを特定します。

追加したクラウド アカウントに含まれるマシンが vRealize Automation Cloud Assembly の外部に展開されている場合、そのマシンはオンボーディングされるまで Cloud Assembly によって管理されません。管理対象外のマシンを vRealize Automation Cloud Assembly 管理に移動するには、オンボーディング プランを使用します。プランを作成してマシンに追加します。次に、そのプランを実行してマシンをインポートします。オンボーディング プランを使用すると、クラウド テンプレートを作成でき、1 つまたは複数の展開を作成することもできます。

1つまたは複数の管理対象外のマシンを1つのプランでオンボーディングすることができます。マシンは、手動で、またはフィルタリングルールを使用して選択できます。フィルタリングルールは、オンボーディングするマシンをマシン名、ステータス、IP アドレス、タグなどの基準に基づいて選択します。

- 1つのオンボード プラン内で、管理対象外のマシンを1時間あたり 3,500 台までオンボードすることができます。
- 複数のオンボード プラン内で、管理対象外のマシンを1時間あたり 17,000 台まで同時にオンボードすることができます。

ワークロードのオンボーディングで使用可能なマシンは、特定のクラウド アカウント タイプおよびリージョンに関連する [リソース] - [マシン] 画面にリストされ、[発生元] 列に Discovered というラベルが付きます。データが収集されたマシンのみがリストされます。オンボーディングしたマシンは、[発生元] 列に Deployed と表示されます。

ワークロードのオンボーディング プランを実行するユーザーは、自動的にマシンの所有者として割り当てられます。

オンボーディングの例

オンボーディング技術の例については、[例：選択されたマシンを vRealize Automation Cloud Assembly で単一の展開としてオンボーディング](#)および[例：ルール フィルタを適用したマシンを vRealize Automation Cloud Assembly の個別の展開としてオンボーディング](#)を参照してください。

オンボーディング イベントのサブスクリプション

プランを実行すると、Deployment Onboarded イベントが作成されます。[拡張性] タブのオプションを使用して、これらの展開イベントをサブスクライブし、アクションを実行できます。

例：選択されたマシンを vRealize Automation Cloud Assembly で単一の展開としてオンボーディング

この例では、2 台の管理対象外のマシンを単一の vRealize Automation Cloud Assembly 展開としてオンボーディングし、プラン内のすべてのマシン用に単一のクラウド テンプレートを作成します。

クラウド アカウントを作成すると、アカウントに関連付けられたすべてのマシンでデータが収集され、[インフラストラクチャ] - [リソース] - [マシン] 画面に表示されます。クラウド アカウントに、vRealize Automation Cloud Assembly の外部に展開されたマシンがある場合は、オンボーディング プランを使用してマシンの展開を vRealize Automation Cloud Assembly で管理できます。

注： 展開の名前は、オンボーディングの前にのみ変更できます。オンボーディング後、[名前の変更] オプションは無効になります。

前提条件

- 必要なユーザー ロールがあることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- [vRealize Automation Cloud Assembly](#) でのオンボーディング プランについてを確認します。
- vRealize Automation Cloud Assembly プロジェクトを作成して準備します。

この手順には、基本的な Wordpress 使用事例の手順がいくつか含まれています。[チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。

- プロジェクトを作成して、ユーザーを追加し、ユーザー ロールを割り当てます。[パート 2：サンプル vRealize Automation Cloud Assembly プロジェクトの作成](#)を参照してください。
- プロジェクト用に Amazon Web Services クラウド アカウントを作成します。[1. クラウド アカウントの追加](#)を参照してください。

この手順での Amazon Web Services クラウド アカウントには、そのクラウド アカウントが vRealize Automation Cloud Assembly に追加される前に vRealize Automation Cloud Assembly 以外のアプリケーションによって展開されたマシンが含まれています。

- [マシン] 画面に、オンボーディングするマシンが含まれていることを確認します。[vRealize Automation のマシン リソース](#)を参照してください。

手順

- 1 [インフラストラクチャ] - [オンボーディング] の順に移動します。
- 2 [新しいオンボーディング プラン] をクリックして、サンプル値を入力します。

設定	サンプルの値
プラン名	VC-sqa-deployments
説明	OurCo-AWS クラウド アカウント用の AWS マシンのオンボーディング プランのサンプル
クラウド アカウント	OurCo-AWS
デフォルトのプロジェクト	WordPress

- 3 [作成] をクリックします。
- 4 プランの [展開] タブで [マシンの選択] をクリックし、1 台以上のマシンを選択して、[OK] をクリックします。

マシンの選択



- 5 [すべてのマシンを含む 1 つの展開を作成する] を選択して、[作成] をクリックします。
- 6 新しい展開名の横にあるチェック ボックスをクリックし、[クラウド テンプレート...] をクリックします。
- 7 [Cloud Assembly フォーマットでクラウド テンプレートを作成する] をクリックします。

- 8 クラウド テンプレートの名前を入力して、[保存] をクリックします。

注： オンボーディング プランで vSphere マシンを使用している場合は、オンボーディング プロセスが完了した後にクラウド テンプレートを編集する必要があります。オンボーディング プロセスではソースの vSphere マシンとそのマシン テンプレートをリンクできず、生成されたクラウド テンプレートのクラウド テンプレート コードに `imageRef: "no image available"` エントリが含まれます。imageRef: フィールドに正しいテンプレート名を指定するまで、クラウド テンプレートを展開することはできません。オンボーディング プロセスの完了後にクラウド テンプレートを見つけて更新しやすくするには、展開の [クラウド テンプレート構成] 画面で [クラウド テンプレート名] オプションを使用します。自動生成されたクラウド テンプレート名を記録するか、選択したクラウド テンプレート名を入力して記録します。オンボーディングが完了したら、クラウド テンプレートを見つけて開き、imageRef: フィールドの "no image available" エントリを正しいテンプレート名に置き換えます。

- 9 展開名のチェック ボックスをクリックし、[実行] をクリックします。次に、[プランの実行] 画面で [実行] を再度クリックします。

選択された Amazon Web Services マシンは、付属するクラウド テンプレートと共に単一の展開としてオンボーディングされます。

- 10 [クラウド テンプレート] タブをクリックし、クラウド テンプレート名をクリックすることにより、クラウド テンプレートを開いて確認します。

- 11 [展開] タブをクリックし、展開名をクリックすることにより、展開を開いて確認します。

例：ルール フィルタを適用したマシンを vRealize Automation Cloud Assembly の個別の展開としてオンボーディング

この例では、フィルタリング ルールを使用して、状態が ON で名前が BG で始まるすべてのマシンをオンボーディングします。このプランでは、マシンごとに個別の vRealize Automation Cloud Assembly クラウド テンプレートと展開も作成します。

クラウド アカウントを作成すると、アカウントに関連付けられたすべてのマシンでデータが収集され、[インフラストラクチャ] - [リソース] - [マシン] 画面に表示されます。クラウド アカウントに、vRealize Automation Cloud Assembly の外部に展開されたマシンがある場合は、オンボーディング プランを使用してマシンの展開を vRealize Automation Cloud Assembly で管理できます。

前提条件

- 必要なユーザー ロールがあることを確認します。vRealize Automation のユーザー ロールについてを参照してください。
- vRealize Automation Cloud Assembly でのオンボーディング プランについてを確認します。
- vRealize Automation Cloud Assembly プロジェクトを作成して準備し、1 つ以上のクラウド アカウントを追加します。

これには、ガイダンスありのセットアップ手順の基本的な手順の一部が含まれます。

- プロジェクトを作成して、ユーザーを追加し、ユーザー ロールを割り当てます。パート 2：サンプル vRealize Automation Cloud Assembly プロジェクトの作成を参照してください。

- プロジェクトの指定された領域に 1 つ以上のクラウド アカウントを作成します。
- [マシン] 画面に、オンボーディングするマシンが含まれていることを確認します。vRealize Automation の [マシン リソース](#) を参照してください。

手順

- 1 [インフラストラクチャ] - [オンボーディング] の順に移動します。
- 2 [新しいオンボーディング プラン] をクリックして、値を入力します。

設定	サンプルの値
プラン名	ob_rules_1
説明	rules1 を使用したマシンのオンボーディング
クラウド アカウント	rs-aws
デフォルトのプロジェクト	rs-project

新しいオンボーディング プラン




プラン名

説明

前提条件

クラウド アカウントを追加し、オンボーディングするマシンが配置されているコンピューティング リソースにクラウド ゾーンを作成します。
1 人以上のユーザーを含むプロジェクトを作成し、プロジェクトにクラウド ゾーンへのアクセス権を付与します。

クラウド アカウ
ント  

デフォルトのプロ
ジェクト  

キャンセル

作成

3 [作成] をクリックします。

4 [ルール] タブをクリックしてから、[ルールの追加] をクリックします。

1 つ以上のルールを作成し、特定のマシン特性に基づいてオンボーディングするマシンのグループを選択できます。

5 ルール名（ob_rules_1 など）を入力します。

6 フィルタを追加してルールを作成します。

この例では、[フィルタ] のドロップダウン メニューで [状態] と [名前] のフィルタを使用して、名前に BG* が含まれていて状態が On のすべてのマシンを指定しています。

ルール名 *

ob_rules_1

フィルタリングしています...

プロパティ

任意:

名前:

ステータス:

アドレス:

タグ:

状態	アドレス	作成日時	タグ
オン	10.184.68.223	2020年1月09日	

4 マシン

ルール名 *

ob_rules_1

名前: BG* ステータス: オン

7 [保存] をクリックします。

追加のルールを作成することもできますが、この例では 1 つ使用します。

ob_rules_1

サマリ ルール マシン 展開

このプランにマシンを追加するには、ルールを使用します。 ①

ルールの追加 適用 削除

名前	ステータス	フィルタ
ob_rules_1	OK	Name: BG*

8 [マシン] タブをクリックします。この例では、4 台のマシンが選択されています。そのうち 3 台が「BG」という文字で始まり、1 台に「BG」という文字が含まれています。

ob_rules_1

サマリ ルール マシン 展開

ここにリストされているマシンは、プランの実行時にオンボーディングされます。プランルールは 24 時間ごとに評価されます。プランには新しいマシンを追加できます。

マシンの追加 適用 除外 削除

フィルタリングしています...

名前	ステータス	電源	アドレス	展開	ルール	タグ
tf-machine-mcm332-12460625552	保留中	オン	10.196.157.207	Deployment-a46900d1-f2f1-44d5...	ob_rules_1	US:Ubuntu too bar
Terraform-Provider-003-mcm423-124577617759	保留中	オン	10.196.157.229	Deployment-3468b529-b7b1-4bfc...	ob_rules_1	US:Ubuntu too bar
vm345-mcm593-124545097052	保留中	オン	10.196.157.161	Deployment-56791dfe-cb65-4276...	ob_rules_1	vmtagkey-vm4b
tf-machine-mcm332-12460625555	保留中	オン	10.196.157.213	Deployment-57d8f746-8847-44fa...	ob_rules_1	US:Ubuntu too bar

4 マシン

9 名前が「BG」で始まらないマシンを削除するには、そのチェック ボックスを選択して [除外] をクリックします。

ob_rules_1

サマリ ルール マシン 展開

ここにリストされているマシンは、プランの実行時にオンボーディングされます。プランルールは 24 時間ごとに評価されます。プランには新しいマシンを追加できます。

マシンの追加 適用 除外 削除

フィルタリングしています...

名前	ステータス	電源	アドレス	展開	ルール	タグ
tf-machine-mcm332-12460625552	保留中	オン	10.196.157.207	Deployment-a46900d1-f2f1-44d5...	ob_rules_1	US:Ubuntu too bar
Terraform-Provider-003-mcm423-124577617759	保留中	オン	10.196.157.229	Deployment-3468b529-b7b1-4bfc...	ob_rules_1	US:Ubuntu too bar
vm345-mcm593-124545097052	保留中	オン	10.196.157.161	Deployment-56791dfe-cb65-4276...	ob_rules_1	vmtagkey-vm4b
tf-machine-mcm332-12460625555	保留中	オン	10.196.157.213	Deployment-57d8f746-8847-44fa...	ob_rules_1	US:Ubuntu too bar

4 マシン

10 [展開] タブをクリックします。

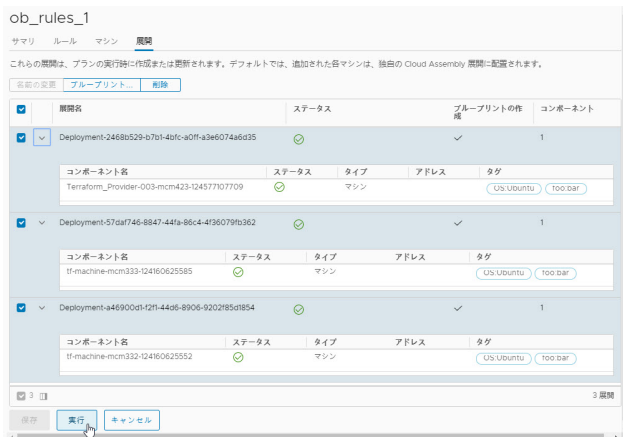
「BG」という文字で始まり、電源状態が On の 3 台のマシンは展開する準備ができています。デフォルトでは、各マシンに対して個別のクラウド テンプレートと展開が作成されます。



- 11 3つの展開名の横にあるチェックボックスをクリックして、[クラウド テンプレート] > [Cloud Assembly フォーマットでブループリントを作成する] > [保存] の順にクリックします。

注： オンボーディング プランで vSphere マシンを使用している場合は、オンボーディング プロセスが完了した後にクラウド テンプレートを編集する必要があります。オンボーディング プロセスではソースの vSphere マシンとそのマシン テンプレートをリンクできず、生成されたクラウド テンプレートのクラウド テンプレートコードに `imageRef: "no image available"` エントリが含まれます。imageRef: フィールドに正しいテンプレート名を指定するまで、クラウド テンプレートを展開することはできません。オンボーディング プロセスの完了後にクラウド テンプレートを見つけて更新しやすくするには、展開の [クラウド テンプレート構成] 画面で [クラウド テンプレート名] オプションを使用します。自動生成されたクラウド テンプレート名を記録するか、選択したクラウド テンプレート名を入力して記録します。オンボーディングが完了したら、クラウド テンプレートを見つけて開き、imageRef: フィールドの "no image available" エントリを正しいテンプレート名に置き換えます。

- 12 [展開] 画面で、3つの展開名の横にあるチェックボックスをクリックして、[実行] をクリックします。



- 13 確認を求めるプロンプトが表示されたら、[実行] をクリックして、マシンをオンボーディングします。

プランの実行

×

プラン名	ob_rules_1
説明	Machine onboarding with rules1
クラウド アカウント	346test_vc_account
デフォルトのプロジェクト	123
展開	3
前回の実行	なし

キャンセル

実行

プランが実行され、マシンが vRealize Automation Cloud Assembly の管理下に入ります。各マシンに対して個別のクラウド テンプレートと展開が作成されます。

vRealize Automation Cloud Assembly 環境の詳細設定

vRealize Automation Cloud Assembly 環境を構成して、プロジェクトの設定、統合、および展開をさらにサポートすることができます。

ユーザーとログの操作、カスタマー エクスペリエンス向上プログラムへの参加または離脱など、管理方法の関連および追加情報については、[vRealize Automation の管理](#)に関するヘルプを参照してください。

vRealize Automation のインターネット プロキシ サーバの構成方法

インターネットへの直接アクセスを持たない隔離されたネットワーク上の vRealize Automation インストールでは、インターネット プロキシ サーバを使用してプロキシ機能によるインターネット接続を許可することができます。インターネット プロキシ サーバでは、HTTP および HTTPS がサポートされています。

Amazon Web Services (AWS)、Microsoft Azure、Google Cloud Platform (GCP) などのパブリック クラウド プロバイダや、IP アドレス管理、Ansible、Puppet などの外部統合ポイントを設定して vRealize Automation と一緒に使用するには、内部の vRealize Automation インターネット プロキシ サーバにアクセスするようにインターネット プロキシ サーバを設定する必要があります。

vRealize Automation には、インターネット プロキシ サーバと通信する内部プロキシ サーバが含まれています。このサーバがプロキシ サーバと通信するのは、プロキシ サーバが `vracli proxy set ...` コマンドで設定されている場合です。組織でインターネット プロキシ サーバが設定されていない場合は、内部の vRealize Automation プロキシ サーバが直接インターネットに接続を試みます。

指定された `vracli` コマンドライン ユーティリティを使用することにより、インターネット プロキシ サーバを使用するように vRealize Automation を設定できます。`vracli` API の使用方法は、`vracli` コマンドラインで `--help` 引数を使用して参照できます (`vracli proxy --help` など)。

インターネット プロキシ サーバへのアクセスでは、vRealize Automation に組み込まれているアクションベースの拡張性 (ABX) のオンプレミスの組み込みコントロールを使用する必要があります。

注： インターネット プロキシ経由での Workspace ONE Access (旧名 : VMware Identify Manager) へのアクセスは、サポートされていません。vracli set vidm コマンドを使用して、インターネット プロキシ サーバ経由で Workspace ONE Access にアクセスすることはできません。

内部プロキシ サーバは、IPv4 がデフォルトの IP アドレス形式である必要があります。TLS (HTTPS) 証明書トラフィックに対して、インターネット プロトコルの制限、認証、または中間者アクションは必要ありません。

前提条件

- インターネット プロキシ サーバとして使用できる既存の HTTP または HTTPS サーバが、外部サイトに送信トラフィックを渡すことができる vRealize Automation ネットワークにあることを確認します。接続は IPv4 用に設定されている必要があります。
- ターゲットのインターネット プロキシ サーバが、デフォルトの IP アドレス形式として IPv6 ではなく IPv4 をサポートするように設定されていることを確認します。
- インターネット プロキシ サーバが TLS を使用していて、クライアントとの HTTPS 接続が必要な場合は、プロキシを構成する前に、次のコマンドのいずれかを使用してサーバ証明書をインポートする必要があります。

```
■ vracli certificate proxy --set path_to_proxy_certificate.pem
```

```
■ vracli certificate proxy --set stdin
```

インタラクティブな入力を行うには、stdin パラメータを使用します。

手順

- 1 Kubernetes によって使用されるポッドまたはコンテナのプロキシ構成を作成します。この例では、HTTP スキームを使用してプロキシ サーバにアクセスします。

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

- 2 プロキシ構成を表示します。

```
vracli proxy show
```

結果は次のようになります。

```
{
  "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# End autogen configuration\n",
  "internal.proxy.config.type": "default"
}

```

注： 組織でインターネット プロキシ サーバが設定されていない場合は、上の例で 'default' ではなく、"internal.proxy.config.type": "non-default" が表示されます。セキュリティのため、パスワードは表示されません。

注： -proxy-exclude パラメータを使用する場合は、デフォルト値を編集する必要があります。たとえば、インターネット プロキシ サーバを使用してアクセスできないドメインとして acme.com を追加する場合は、次の手順を使用します。

- a vracli proxy default-no-proxy と入力して、デフォルトのプロキシ除外設定を取得します。これは、ドメインとネットワークの自動生成されたリストです。
- b 値を編集して .acme.com を追加します。
- c vracli proxy set --proxy-exclude ... と入力して構成の設定を更新します。
- d /opt/scripts/deploy.sh コマンドを実行して、環境を再展開します。

- 3 (オプション) DNS ドメイン、FQDN、および IP アドレスを、インターネット プロキシ サーバからアクセスできないように除外します。

proxy-exclude 変数のデフォルト値を変更するときは、必ず `parameter --proxy-exclude` を使用します。ドメイン `exclude.vmware.com` を追加するには、最初に `vrali proxy show` コマンドを使用してから、proxy-exclude 変数をコピーし、次のように `vracli proxy set ...` コマンドを使用してドメイン値を追加します。

```
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"
```

注： 値を置き換えるのではなく、proxy-exclude に要素を追加します。デフォルト値 proxy-exclude を削除すると、vRealize Automation は適切に機能しません。この問題が発生した場合は、プロキシ構成を削除し、最初からやり直してください。

- 4 `vracli proxy set ...` コマンドを使用してインターネット プロキシ サーバを設定した後は、`vracli proxy apply` コマンドを使用してインターネット プロキシ サーバの設定を更新し、最新のプロキシ設定を有効にすることができます。

- 5 スクリプトの変更をまだ有効にしていない場合は、次のコマンドを実行して有効にします。

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

- 6 (オプション) 必要に応じて、ポート 22 で外部アクセスをサポートするようにプロキシ サーバを構成します。

Puppet や Ansible などの統合をサポートするには、関連するホストにポート 22 でアクセスできるようにプロキシ サーバで許可する必要があります。

例：サンプルの Squid 構成

手順 1 では、Squid プロキシを設定する場合、次のサンプルに合わせて `/etc/squid/squid.conf` で構成を調整することができます。

```
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 Automation で NSX-T と複数の vCenter Server のマッピングで可能になる事例

NSX-T のクラウド アカウントを 1 つ以上の vCenter クラウド アカウントに関連付けて、さまざまな展開の目的に対応できます。

同じ既存の NSX-T ネットワークを異なる vCenter Server のネットワーク プロファイルに関連付け、制約に基づいていずれかの vCenter Server で展開をプロビジョニングすることができます。次に例をいくつか示します。

- 同じネットワーク プロファイルを使用する複数の NIC を備えた 1 台のマシンが含まれ、このネットワーク プロファイルに複数の vCenter Server にまたがる NSX-T ネットワークが含まれているクラウド テンプレート。
- サブネット ベースの分離が指定されたネットワーク プロファイルを使用し、複数の vCenter Server にまたがる既存の NSX-T ネットワークを使用する、プライベート ネットワーク上のマシンが含まれているクラウド テンプレート。
- セキュリティ グループ ベースの分離が指定されたネットワーク プロファイルを使用し、複数の vCenter Server にまたがる NSX-T ネットワークを使用する、プライベート ネットワーク上の単一のマシンが含まれているクラウド テンプレート。
- 複数の vCenter Server にまたがる NSX-T ネットワークが含まれているネットワーク プロファイルを使用する、ルーティング ネットワーク上の単一のマシンが含まれているクラウド テンプレート。
- オンデマンドのロード バランサが含まれ、そのロード バランサがネットワーク上のすべての vCenter Server マシンに対して適用されるようにネットワーク プロファイルで定義されているクラウド テンプレート。
- オンデマンド ネットワークが含まれ、ネットワーク プロファイルを使用するすべての vCenter Server でそのオンデマンド ネットワークが使用されるようにネットワーク プロファイルで定義されているクラウド テンプレート。
- 必要に応じてファイアウォール ルールが含まれ、ネットワーク上のすべての vCenter Server に関連付けられているオンデマンド セキュリティ グループが含まれているクラウド テンプレート。

NSX-T ネットワークに vRealize Automation の内部または外部 IP アドレス管理を設定し、異なる vCenter Server でプロビジョニングされたマシンに対して同じ IP アドレスを共有することができます。

システム内でネットワーク プロファイルが定義されていない場合は、既存の単一の NSX-T ネットワークを共有する異なる vCenter Server で、複数のマシンを含むクラウド テンプレートをプロビジョニングできます。

vRealize Automation で NSX クラウド アカウントの関連付けを削除した場合の動作

NSX クラウド アカウントと vCenter Server クラウド アカウント間の関連付けを削除する場合は、関連するネットワーク プロファイルを更新して、関連付けられている NSX オブジェクトも削除する必要があります。

NSX クラウド アカウントと vCenter Server クラウド アカウント間の関連付けを削除しても、インフラストラクチャ要素が vRealize Automation によって自動的に更新されることはありません。関連付けられている NSX オブジェクトを削除するには、既存のネットワーク プロファイルを更新する必要があります。

ユーザー インターフェイスには、影響を受けるネットワーク プロファイル要素を特定するのに役立つ次のような情報が表示されます。

- ネットワーク プロファイルで NSX の既存のネットワークが選択されている場合：
 - オブジェクトは無効としてマークされ、「一部のネットワーク オブジェクトが見つからないか無効です。」というメッセージが表示されます。
 - これらのオブジェクトは、ネットワーク プロファイルを保存すると削除されます。
- ネットワーク プロファイルでアプリケーションの隔離が設定されている場合は、ネットワーク プロファイルを保存する前に、隔離ポリシーの設定を更新する必要があります。
- ネットワーク プロファイルでセキュリティ グループまたはロード バランサが選択されている場合、ネットワーク プロファイルを保存すると、これらのオブジェクトが削除されます。

既存の展開は既存のコンポーネントに対して設計どおりに機能しますが、スケールアウト操作など、新しいコンポーネントの作成に失敗します。

関連付けを再び確立すると、ネットワーク プロファイルが再設定され、既存の展開が設計どおりに機能します。

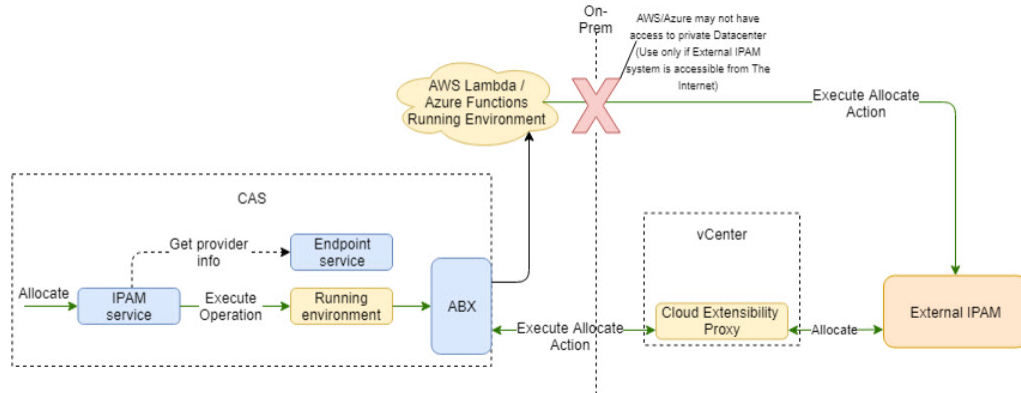
NSX クラウド アカウントを削除した場合、上記の動作は同じですが、ネットワーク オブジェクトは、無効ではなく、不明としてマークされます。

IP アドレス管理 SDK を使用して vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合パッケージを作成する方法

外部 IP アドレス管理ベンダーおよびパートナーは、IP アドレス管理 SDK をダウンロードして使用することで、IP アドレス管理の統合パッケージを作成して、vRealize Automation がプロバイダ固有の IP アドレス管理ソリューションをサポートできるようにすることができます。

提供される IP アドレス管理 SDK を使用して vRealize Automation に対してカスタムの IP アドレス管理統合パッケージをビルドして展開するプロセスは、[VMware Cloud Assembly のプロバイダ固有の IP アドレス管理統合パッケージの作成および展開](#)ドキュメントに記載されています。このドキュメントで説明されているように、[VMware Code](#) サイトから最新の VMware vRealize Automation Third-Party IPAM SDK をダウンロードできます。次の IP アドレス管理 SDK パッケージを入手できます。

- [VMware vRealize Automation Third-Party IPAM SDK 1.1.0](#)
- [VMware vRealize Automation Third-Party IPAM SDK 1.0.0](#)



IP アドレス管理 SDK を使用してベンダー固有の IP アドレス管理統合パッケージを作成する前に、vRealize Automation 用にすでに用意されているかどうかを確認します。プロバイダ固有の IP アドレス管理統合パッケージは、IP アドレス管理プロバイダの Web サイト、[VMware Marketplace](#)、および vRealize Automation の [マーケットプレイス] タブから入手することができます。

チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成 の例はベンダーに固有ですが、リファレンスとして使用できる情報も含まれています。

vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド

4

vRealize Automation Cloud Assembly リソース インフラストラクチャでは、クラウド アカウント リージョンを、クラウド テンプレートとそのワークロードを展開するためのゾーンとして定義します。

さらに、リソース インフラストラクチャでは、イメージとマシン サイズの一般的なマッピングと、複数のクラウド アカウント リージョンまたはデータセンターのネットワークおよびストレージ機能を定義するプロファイルの作成も行われます。

この章には、次のトピックが含まれています。

- vRealize Automation Cloud Assembly のターゲット配置領域またはデータセンターを定義するクラウド ゾーンを追加する方法
- vRealize Automation でフレーバー マッピングを追加して一般的なマシン サイズを指定する方法
- vRealize Automation でイメージ マッピングを追加して一般的なオペレーティング システムにアクセスする方法
- vRealize Automation でネットワーク プロファイルを追加する方法
- 要件の異なる複数の vRealize Automation Cloud Assembly ストレージ プロファイルを追加する方法
- vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法
- vRealize Automation でリソースを操作する方法
- vRealize Automation を使用したマルチプロバイダ テナント リソースの構成

vRealize Automation Cloud Assembly のターゲット配置領域またはデータセンターを定義するクラウド ゾーンを追加する方法

vRealize Automation Cloud Assembly クラウド ゾーンは、AWS や vSphere などのクラウド アカウント タイプ内のリソースのセットです。

特定のアカウント リージョン内のクラウド ゾーンに、クラウド テンプレートからワークロードが展開されます。各クラウド ゾーンは、vRealize Automation Cloud Assembly プロジェクトに関連付けられています。

[インフラストラクチャ] - [設定] - [クラウド ゾーン] の順に選択し、[新規ゾーンの追加] をクリックします。

vRealize Automation Cloud Assembly クラウド ゾーンの詳細情報

vRealize Automation Cloud Assembly クラウド ゾーンは、AWS や vSphere などのクラウド アカウント タイプに固有のコンピューティング リソースのセクションです。

クラウド ゾーンはリージョンに固有であり、プロジェクトに割り当てる必要があります。クラウド ゾーンとプロジェクトの関係は多対多です。vRealize Automation Cloud Assembly では、Azure、AWS、GCP などの主要なパブリック クラウドと vSphere への展開がサポートされています。[vRealize Automation Cloud Assembly へのクラウド アカウントの追加](#)を参照してください。

追加の配置制御には、配置ポリシー オプション、機能タグ、コンピューティング タグが含まれます。

■ 配置ポリシー

配置ポリシーにより、指定したクラウド ゾーンの展開環境内でホストを選択しやすくなります。

- default - コンピューティング リソースをクラスタおよびホスト全体にランダムに分散します。このオプションは、個々のマシン単位で機能します。たとえば、ある特定の展開内のすべてのマシンが、要件を満たす使用可能なクラスタおよびホストにランダムに分散されます。
- binpack - 指定されたコンピューティングを実行するのに使用可能な十分なリソースのある、最も負荷の大きいホストにコンピューティング リソースを配置します。
- spread - 仮想マシンの数が最も少ないクラスタまたはホストに、展開レベルでコンピューティング リソースをプロビジョニングします。vSphere の場合、Distributed Resource Scheduler (DRS) が仮想マシンをホスト間で分散します。たとえば、ある展開環境のすべての要求されたマシンが同じクラスタに配置されても、次の展開では、その時点での負荷に応じて別の vSphere クラスタが選択されることがあります。

たとえば、次の構成を考えます。

- 5 台の仮想マシンを含む DRS クラスタ 1
- 9 台の仮想マシンを含む DRS クラスタ 2
- 6 台の仮想マシンを含む DRS クラスタ 3

3 台の仮想マシンのクラスタを申請し、spread ポリシーを選択すると、これらの仮想マシンはすべてクラスタ 1 に配置されます。更新された負荷は、クラスタ 1 については 8 台の仮想マシンになりますが、クラスタ 2 と 3 の負荷は 9 台と 6 台のままです。

さらに他の 2 台の仮想マシンを追加で申請すると、これらの仮想マシンは DRS クラスタ 3 に配置され、クラスタ 3 の仮想マシンは 8 台になります。クラスタ 1 と 3 の負荷は 8 台と 9 台で変わりません。

2 つのクラウド ゾーンの両方が、プロビジョニングに必要なすべての条件と一致する場合は、配置ロジックによって優先度が高いものが選択されます。

■ 機能タグ

ブループリントには、展開の配置を決定するのに役立つ制約タグが含まれています。展開時に、ブループリントの制約タグがクラウド ゾーンの一一致する機能タグにマッピングされ、コンピューティング リソースの配置に使用できるクラウド ゾーンが決まります。

■ コンピューティング

ワークロードのプロビジョニングに使用可能な AWS のアベイラビリティ ゾーンや vCenter クラスタなどのコンピューティング リソースを、このクラウド ゾーンに表示および管理できます。

vCenter Server コンピューティング クラスタが DRS に対応している場合、クラウド ゾーンには、コンピューティングのリストに含まれるクラスタのみが表示され、子ホストは表示されません。vCenter コンピューティング クラスタが DRS に対応していない場合、クラウド ゾーンにはスタンドアローンの ESXi ホストのみが表示されます（ある場合）。

クラウド ゾーンの必要に応じたコンピューティング リソースを追加します。初期状態で選択されているフィルタでは、すべてのコンピューティング リソースが含まれます。次のリストに示されているのが使用可能なすべてのコンピューティング リソースであり、これらは該当するゾーンに割り当てられます。クラウド ゾーンにコンピューティング リソースを追加する場合には、他に 2 つのオプションを選択できます。

- [コンピューティングの手動選択] - 下のリストからコンピューティング リソースを手動で選択する場合は、このオプションを選択します。選択したら、[コンピューティングの追加] をクリックしてリソースをゾーンに追加します。
- [コンピューティングをタグによって動的に含める] - ゾーンに追加するコンピューティング リソースをタグに基づいて選択する場合は、このオプションを選択します。適切なタグを追加する前の状態では、すべてのコンピューティング リソースが表示されています。このオプションでは、1 つまたは複数のタグを選択または入力できます。

どちらのコンピューティング オプションでも、表示されているコンピューティング リソースの右側のボックスを 1 つまたは複数選択して [削除] をクリックすることで削除できます。

コンピューティング タグを使用すると、配置をさらに制御できます。次の例に示すように、タグを使用して、使用可能なコンピューティング リソースを 1 つ以上のタグに一致するリソースのみにフィルタできます。

- コンピューティングにタグが含まれておらず、フィルタも使用されていない場合



- 2 つのコンピューティングに同じタグが含まれているが、フィルタは使用されていない場合



- 2つのコンピューティングに同じタグが含まれており、タグ フィルタが2つのコンピューティングで使われるタグに一致する場合



■ プロジェクト

このクラウド ゾーンへのワークロード プロビジョニングをサポートするように構成されているプロジェクトを表示できます。

クラウド ゾーンを作成したら、その構成を検証できます。

vRealize Automation でフレーバー マッピングを追加して一般的なマシン サイズを指定する方法

vRealize Automation フレーバー マップでは、自然言語を使用して、特定のクラウド アカウント/リージョンに対するターゲットの展開サイズを定義します。

フレーバー マップは、環境に適した展開サイズを表します。たとえば、指定されたデータセンターの vCenter アカウントおよび指定されたリージョンの Amazon Web Services アカウントの t2.nano 用として、CPU が1個とメモリが2 GB の small および CPU が2個とメモリが8 GB の large が考えられます。

[インフラストラクチャ] - [設定] - [フレーバー マッピング] の順に選択し、[新しいフレーバー マッピング] をクリックします。

vRealize Automation でのフレーバー マッピングの詳細

フレーバー マッピングでは、自然言語による名前付けを使用して、vRealize Automation 内の特定のクラウド アカウントとリージョンに対する一連のターゲット展開のサイズ設定をグループ化します。

フレーバー マッピングを使用すると、お使いのアカウントのリージョンで類似のフレーバー サイズ設定を含む名前付きマッピングを作成できます。たとえば、`standard_small` という名前のフレーバー マップに、プロジェクト内で使用可能なアカウントとリージョンの一部またはすべてに対して同様のフレーバー サイズ設定（1 CPU、2 GB RAM など）を含めることができます。クラウド テンプレートをビルドするときは、ニーズに合った使用可能なフレーバーを選択します。

展開の目的に応じて、プロジェクトのフレーバー マッピングを編成します。

クラウド テンプレートの作成を簡素化するために、新しいクラウド アカウントを追加するときに事前構成オプションを選択できます。事前構成オプションを選択すると、指定されたリージョンに対する組織で最も一般的なフレーバー マッピングとイメージ マッピングが選択されます。

vSphere リソースを含むクラウド テンプレート内のイメージ マッピングに関して、vSphere クラウドゾーンに対してフレーバー マッピングが定義されていない場合、クラウド テンプレートで vSphere 固有の設定を使用してメモリと CPU を無制限に構成できます。vSphere のクラウド ゾーンに対してフレーバー マッピングが定義されている場合、フレーバー マッピングはクラウド テンプレートで vSphere 固有の構成の制限として機能します。

vRealize Automation でイメージ マッピングを追加して一般的なオペレーティング システムにアクセスする方法

vRealize Automation イメージ マップでは、自然言語を使用して、特定のクラウド アカウント/リージョンに対するターゲットの展開オペレーティング システムを定義します。

[インフラストラクチャ] - [設定] - [イメージ マッピング] の順に選択し、[新しいイメージ マッピング] をクリックします。

vRealize Automation でのイメージ マッピングの詳細

イメージ マッピングでは、自然言語による名前付けを使用して、vRealize Automation での特定のクラウド アカウントとリージョンに対する一連の事前定義済みターゲットの OS 仕様をグループ化します。

Microsoft Azure や Amazon Web Services などのクラウド ベンダー アカウントは、イメージを使用して、OS や関連する設定などの一連のターゲットの展開条件をグループ化します。vCenter や VMware Cloud on AWS などの NSX ベースの環境では、同様のグループ分けメカニズムを使用して、一連の OS 展開条件を定義します。クラウド テンプレートをビルドし、最終的に展開して、繰り返し使用する場合は、ニーズに最も適した使用可能なイメージを選択します。

プロジェクトのイメージ マッピングは、同様のオペレーティング システム設定、タグ付け方法、および機能展開の目的に合わせて編成します。

クラウド テンプレートの作成を簡素化するために、新しいクラウド アカウントを追加するときに事前構成オプションを選択できます。事前構成オプションを選択すると、指定されたリージョンに対する組織で最も一般的なフレーバー マッピングとイメージ マッピングが選択されます。

クラウド テンプレートにイメージ情報を追加する場合は、マシン コンポーネントの `properties` セクションの `image` または `imageRef` のどちらかのエントリを使用します。たとえば、スナップショットからクローンを作成する場合は、`imageRef` プロパティを使用します。

クラウド テンプレート コードの image および imageRef エントリの例については、[6 章 vRealize Automation Cloud Assembly 展開の設計](#)を参照してください。

コンテンツ ライブラリに権限を割り当てるには、管理者が権限をグローバル権限としてユーザーに付与する必要があります。関連情報については、[VMware vSphere のドキュメント](#)で、『vSphere 仮想マシン管理』の[コンテンツ ライブラリの権限の階層的な継承](#)を参照してください。

クラウド アカウント/リージョンのイメージの同期

イメージの同期を実行すると、[インフラストラクチャ] - [構成] - [イメージ マッピング] ページの特定のクラウド アカウント/リージョンに対して追加または削除しているイメージが、最新のものであることを確認できます。

- 1 [インフラストラクチャ] - [接続] - [クラウド アカウント] を選択して、関連付けられている [クラウド アカウント/リージョン] を開きます。既存のクラウド アカウント/リージョンを選択します。
- 2 [イメージの同期] ボタンをクリックして、アクションを完了させます。



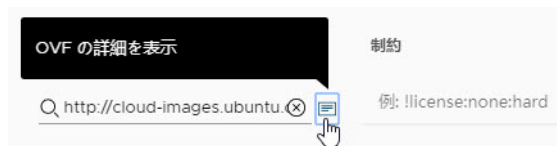
- 3 アクションが完了したら、[インフラストラクチャ] - [構成] - [イメージ マッピング] をクリックします。イメージ マッピングを新しく定義するか、既存のものを編集して、手順 1 のクラウド アカウント/リージョンを選択します。
- 4 [イメージ マッピング] ページのイメージ同期アイコンをクリックします。



- 5 [イメージ マッピング] ページで、指定されたクラウド アカウント/リージョンのイメージ マッピングを構成します。

OVF 詳細の表示

vCenter マシン コンポーネントやイメージ マップなど、OVF 仕様を vRealize Automation Cloud Assembly クラウド テンプレート オブジェクトに含めることができます。イメージに OVF ファイルが含まれている場合は、ファイルを開かずにそのコンテンツを見つけることができます。OVF の上にマウスを移動すると、名前と場所を含む OVF 詳細が表示されます。OVF ファイル フォーマットの詳細については、[vcenter ovf: property](#) を参照してください。



イメージの選択を調整するための制約とタグの使用

クラウド テンプレートのイメージの選択をさらに絞り込むには、1 つまたは複数の制約を追加して、展開可能なイメージ タイプにタグベースの制限を指定します。イメージ マッピングの構成を作成または編集するときに表示される、提供される [制約] の例は、`!license:none:hard` です。この例では、タグベースの制限が示されています。これは、クラウド テンプレートに `license:none` タグが表示されない場合にのみ、イメージを使用できることを示しています。`license:88` や `license:92` などのタグを追加すると、`license:88` および `license:92` タグがクラウド テンプレート内に含まれている場合にのみ、指定されたイメージを使用できます。

制約

例: `!license:none:hard`

クラウド設定スクリプトを使用した展開の制御

イメージ マップとクラウド テンプレートのどちらかまたはその両方でクラウド設定スクリプトを使用して、vRealize Automation Cloud Assembly 展開で使用するカスタムの OS 特性を定義できます。たとえば、クラウド テンプレートをパブリック クラウドまたはプライベート クラウドのどちらかに展開するかに応じて、特定のユーザー権限、OS 権限、またはその他の条件をイメージに適用することができます。クラウド設定スクリプトは、Linux ベースのイメージの場合は `cloud-init`、Windows ベースのイメージの場合は `cloudbase-init` の形式になります。vRealize Automation Cloud Assembly は、Linux システム用の [cloud-init](#) ツールと Windows 用の [cloudbase-init](#) ツールをサポートしています。

Windows マシンの場合、`cloudbase-init` でサポートされている任意のクラウド設定スクリプト形式を使用できます。

次のサンプル クラウド テンプレート コードのマシン リソースは、クラウド設定スクリプトを含むイメージを使用しており、その内容が `image` エントリに表示されます。

```
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}
```

イメージ マッピングとクラウド テンプレートにクラウド設定スクリプトが含まれている場合の動作

クラウド設定スクリプトを含むクラウド テンプレートがクラウド設定スクリプトを含むイメージ マッピングを使用する場合、両方のスクリプトが結合されます。このマージ動作では、スクリプトが `#cloud-config` の形式になっているかどうかを考慮しながら、まずイメージ マッピング スクリプトの内容、次にクラウド テンプレート スクリプトの内容が処理されます。

- `#cloud-config` 形式のスクリプトの場合、マージでは各モジュールの内容（たとえば `runcmd`、`users`、`write_files`）が次のように結合されます。
 - 内容がリストであるモジュールの場合、イメージ マッピングのコマンドのリストとクラウド テンプレートのコマンドのリストがマージされ、両方のリストで同一であるコマンドが除外されます。
 - 内容がディクショナリであるモジュールの場合、コマンドはマージされ、結果は両方のディクショナリの組み合わせになります。両方のディクショナリに同じキーが存在する場合、イメージ マッピング スクリプト ディクショナリのキーは保持され、クラウド テンプレート スクリプト ディクショナリのキーは無視されます。
 - 内容が文字列であるモジュールの場合、イメージ マッピング スクリプトの内容値は保持され、クラウド テンプレート スクリプトの内容値は無視されます。
- 両スクリプトの形式が `#cloud-config` 以外の場合、または片方のスクリプトが `#cloud-config` 形式でもう片方が別の形式の場合、両スクリプトの結合では、イメージ マッピング スクリプトが最初に行われ、イメージ マッピング スクリプトの終了後にクラウド テンプレート スクリプトが実行されます。

関連情報については、「[Merging user-data sections](#)」を参照してください。

クラウド構成スクリプトの設定と使用に関する詳細情報

クラウド構成スクリプトの使用法の詳細については、[vRealize Automation Cloud Assembly テンプレートでマシンを自動的に初期化する方法](#)を参照してください。

また、VMware ブログの記事の [vSphere Customization with Cloud-init While Using vRealize Automation 8 or Cloud](#) と [Customizing Cloud Assembly Deployments with Cloud-Init](#) も参照してください。

vRealize Automation でネットワーク プロファイルを追加する方法

vRealize Automation ネットワーク プロファイルは、展開されるネットワークの動作を示します。

たとえば、ネットワークは内部専用ではなく、インターネットに接続しなければならない場合があります。

ネットワークとそのプロファイルは、クラウドごとに固有です。

[インフラストラクチャ] - [設定] - [ネットワーク プロファイル] の順に選択し、[新規ネットワーク プロファイル] をクリックします。

vRealize Automation でのネットワーク プロファイルの詳細

ネットワーク プロファイルは、特定のリージョンまたはデータセンターのクラウド アカウントで使用可能なネットワークおよびネットワーク設定のグループを vRealize Automation 内で定義したものです。

通常、ネットワーク プロファイルではターゲットの展開環境をサポートします。たとえば、既存のネットワークに送信アクセスのみを設定する小規模なテスト環境や、一連のセキュリティ ポリシーが必要な、ロード バランシングされた大規模な本番環境などです。ワークロード固有のネットワーク特性を収集したものがネットワーク プロファイルであると考えてください。

ネットワーク プロファイルの内容

ネットワーク プロファイルには、次の設定を含む、vRealize Automation の名前付きクラウド アカウント タイプとリージョンに関する特定の情報が含まれています。

- ネットワーク プロファイルの名前付きクラウド アカウント/リージョンおよびオプションの機能タグ。
- 名前付きの既存のネットワークとその設定。
- ネットワーク プロファイルのオンデマンドなどの要素を定義するネットワーク ポリシー。
- オプションでの既存のロード バランサの包含。
- オプションでの既存のセキュリティ グループの包含。

ネットワーク プロファイルに基づいてネットワーク IP アドレス管理機能を決定します。

ネットワークの選択を制御しやすくするために、ネットワーク プロファイルの機能タグはクラウド テンプレートの制約タグと照合されます。さらに、ネットワーク プロファイルによって収集されるネットワークに割り当てられているすべてのタグも、クラウド テンプレートのタグと照合され、クラウド テンプレートが展開されるときにネットワークの選択が制御しやすくなります。

機能タグはオプションです。機能タグはネットワーク プロファイル内のすべてのネットワークに適用されますが、実際に適用されるのはネットワークがネットワーク プロファイルの一部として使用されている場合だけです。ネットワーク プロファイルに機能タグが含まれていない場合は、ネットワーク タグでのみタグの照合が行われます。一致したネットワーク プロファイル内で定義されているネットワーク設定とセキュリティ設定が、クラウド テンプレートの展開時に適用されます。

固定 IP アドレスを使用する場合、アドレス範囲は vRealize Automation によって管理されます。DHCP の場合、IP アドレスの開始アドレスと終了アドレスは、vRealize Automation ではなく独立した DHCP サーバによって管理されます。DHCP または混合型ネットワーク アドレス割り当てを使用する場合、ネットワーク使用率の値はゼロに設定されます。オンデマンド ネットワーク割り当て範囲は、ネットワーク プロファイルで指定された CIDR およびサブネット サイズに基づきます。展開で固定割り当てと動的割り当ての両方をサポートするために、割り当てられる範囲は、静的な割り当て用と動的な割り当て用の 2 つの範囲に分かれています。

ネットワーク

ネットワークは、サブネットとも呼ばれ、IP ネットワークを論理的に細分化したものです。ネットワークでは、クラウド アカウント、IP アドレス (IP アドレス範囲)、およびネットワーク タグをグループ化して、クラウド テンプレートの展開をプロビジョニングする方法と場所を制御します。プロファイルのネットワーク パラメータでは、展開内のマシンが IP レイヤー 3 を介して他のマシンと相互に通信する方法を定義します。ネットワークにはタグを付けることができます。

ネットワーク プロファイルにネットワークを追加したり、ネットワーク プロファイルで使用されるネットワークの要素を編集することができます。また、ネットワーク プロファイルからネットワークを削除することもできます。

注： VMware Cloud Foundation (VCF) クラウド アカウント タイプの場合、ネットワーク プロファイルに追加できるのは NSX ネットワークのみで、vSphere ネットワークは追加できません。NSX ネットワーク セグメントは、NSX-T ネットワーク上にローカルで作成され、グローバル ネットワークとしては作成されません。

■ [ネットワーク ドメイン] または [トランスポート ゾーン]

このネットワーク ドメインまたはトランスポート ゾーンは、vSphere vNetwork 分散ポート グループ (dvPortGroup) 向けの Distributed Switch (dvSwitch) です。トランスポート ゾーンは、*dvSwitch* または *dvPortGroup* のような用語と同様の既存の NSX 概念です。

NSX クラウド アカウントを使用する場合、ページの要素名は [トランスポート ゾーン] です。それ以外の場合は、[ネットワーク ドメイン] です。

標準スイッチの場合、ネットワーク ドメインまたはトランスポート ゾーンはスイッチ自体と同じです。ネットワーク ドメインまたはトランスポート ゾーンは、vCenter Server 内のサブネットの境界を定義します。

トランスポート ゾーンは、NSX 論理スイッチでアクセスできるホストを制御します。トランスポート ゾーンは 1 つ以上の vSphere クラスタにまたがって設定できます。トランスポート ゾーンでは、特定のネットワークを使用できるクラスタと仮想マシンを制御します。同じ NSX トランスポート ゾーンに属するサブネットは、同じマシン ホストに使用できます。

■ [ドメイン]

ターゲット仮想マシン向けの vCenter Single Sign-On ドメインを表します。ドメインは、vSphere の構成中に vCenter 管理者によって構成されます。ドメインによって、vCenter のローカルの認証領域が決定されます。

■ [IPv4 CIDR] および [IPv4] デフォルト ゲートウェイ

vSphere クラウド アカウント、およびクラウド テンプレート内の vSphere マシン コンポーネントは、デュアル IPv6 および IPv4 インターネット プロトコル方式をサポートします。たとえば、192.168.100.14/24 は、IPv4 アドレス 192.168.100.14 とそれに関連付けられているルーティング プリフィックス 192.168.100.0 を表しています。これは、24 個の先頭 1 ビットが含まれるサブネット マスク 255.255.255.0 に相当します。IPv4 ブロック 192.168.100.0/22 は、192.168.100.0 から 192.168.103.255 までの範囲の 1024 IP アドレスを表します。

■ [IPv6 CIDR] および [IPv6] デフォルト ゲートウェイ

vSphere クラウド アカウント、およびクラウド テンプレート内の vSphere マシン コンポーネントは、デュアル IPv6 および IPv4 インターネット プロトコル方式をサポートします。たとえば、2001:db8::/48 は、2001:db8:0:0:0:0:0:0 から 2001:db8:0:ffff:ffff:ffff:ffff:ffff までの範囲の IPv6 アドレス ブロックを表しています。

IPv6 フォーマットは、オンデマンド ネットワークではサポートされていません。

■ [DNS サーバ] および [DNS 検索ドメイン]

■ [パブリック IP アドレスのサポート]

このオプションは、ネットワークがパブリックであるというフラグを付ける場合に選択します。クラウド テンプレート内のネットワーク コンポーネントのうち、`network type: public` プロパティを持つものが、パブリックのフラグが付いているネットワークと照合されます。クラウド テンプレートの展開時に、ネットワークの選択を決定するために、追加の照合が行われます。

■ [ゾーンのデフォルト]

このオプションは、ネットワークがクラウド ゾーンのデフォルトであるというフラグを付ける場合に選択します。クラウド テンプレートの展開時は、デフォルト ネットワークが他のネットワークより優先されます。

■ [起点]

ネットワーク ソースを識別します。

■ [タグ]

ネットワークに割り当てられた 1 つ以上のタグを指定します。タグはオプションです。タグの照合は、クラウド テンプレート環境で使用可能なネットワークに影響を与えます。

ネットワーク タグは、ネットワーク プロファイルに関係なく、ネットワーク アイテム自体にあります。ネットワーク タグは、ネットワーク タグが追加されているすべてのネットワークと、そのネットワークが含まれているすべてのネットワーク プロファイルに適用されます。ネットワークは、任意の数のネットワーク プロファイルにインスタンス化できます。ネットワーク タグは、ネットワーク プロファイルの有無とは関係なく、対応するネットワークに関連付けられます。そのネットワークがどこで使用されていてもかまいません。

クラウド テンプレートを展開すると、クラウド テンプレートのネットワーク コンポーネントの制約タグが、ネットワーク プロファイルの機能タグなどのネットワーク タグと照合されます。ネットワーク プロファイルに機能タグが含まれている場合、その機能タグはそのネットワーク プロファイルで使用できるどのネットワークにも適用されます。一致したネットワーク プロファイル内で定義されているネットワーク設定とセキュリティ設定が、クラウド テンプレートの展開時に適用されます。

ネットワーク ポリシー

ネットワーク プロファイルを使用することで、固定、DHCP、または固定および DHCP IP アドレス設定の組み合わせを含む既存のネットワーク ドメインのサブネットを定義できます。[ネットワーク ポリシー] タブを使用して、サブネットを定義し、IP アドレス設定を指定できます。

NSX-V、NSX-T、または VMware Cloud on AWS を使用している場合、クラウド テンプレートで `networkType: outbound` または `networkType: private` が必要になるか、NSX ネットワークで `networkType: routed` が必要になると、ネットワーク ポリシー設定が使用されます。

関連付けられたクラウド アカウントによっては、ネットワーク ポリシーを使用して、`outbound`、`private` および `routed` のネットワーク タイプと、オンデマンド セキュリティ グループの設定を定義できます。ネットワークに関連付けられているロード バランサがある場合は、ネットワーク ポリシーを使用して `existing` ネットワークを制御することもできます。

送信ネットワークでは、アップストリーム ネットワークへの一方向のアクセスが許可されます。プライベート ネットワークでは、外部からのアクセスは許可されません。ルーティング ネットワークでは、ルーティング ネットワーク間での East/West トラフィックが許可されます。プロファイルの既存のネットワークとパブリック ネットワークが、基盤となるネットワークまたはアップストリーム ネットワークとして使用されます。

次のオンデマンドの選択に対するオプションについては、[ネットワーク プロファイル] の画面上のヘルプと以下の概要を参照してください。

- [オンデマンド ネットワークまたはオンデマンド セキュリティ グループを作成しない]

このオプションは、existing または public のネットワーク タイプを指定するときに使用できます。outbound、private、または routed ネットワークを必要とするクラウド テンプレートは、このプロファイルと一致しません。

- [オンデマンド ネットワークを作成]

このオプションは、outbound、private、または routed のネットワーク タイプを指定するときに使用できます。

Amazon Web Services、Microsoft Azure、NSX、vSphere、および VMware Cloud on AWS は、このオプションをサポートしています。

- [オンデマンド セキュリティ グループを作成]

このオプションは、outbound または private のネットワーク タイプを指定するときに使用できます。

一致したクラウド テンプレートのネットワーク タイプが outbound または private の場合は、新しいセキュリティ グループが作成されます。

Amazon Web Services、Microsoft Azure、NSX、および VMware Cloud on AWS は、このオプションをサポートしています。

ネットワーク ポリシー設定は、クラウド アカウント タイプによって異なる場合があります。これらの設定は、画面上の Signpost のヘルプと以下の概要で説明されています。

- [ネットワーク ドメイン] または [トランスポート ゾーン]

このネットワーク ドメインまたはトランスポート ゾーンは、vSphere vNetwork 分散ポート グループ (dvPortGroup) 向けの Distributed Switch (dvSwitch) です。トランスポート ゾーンは、dvSwitch または dvPortGroup のような用語と同様の既存の NSX 概念です。

NSX クラウド アカウントを使用する場合、ページの要素名は [トランスポート ゾーン] です。それ以外の場合は、[ネットワーク ドメイン] です。

標準スイッチの場合、ネットワーク ドメインまたはトランスポート ゾーンはスイッチ自体と同じです。ネットワーク ドメインまたはトランスポート ゾーンは、vCenter Server 内のサブネットの境界を定義します。

トランスポート ゾーンは、NSX 論理スイッチでアクセスできるホストを制御します。トランスポート ゾーンは 1 つ以上の vSphere クラスタにまたがって設定できます。トランスポート ゾーンでは、特定のネットワークを使用できるクラスタと仮想マシンを制御します。同じ NSX トランスポート ゾーンに属するサブネットは、同じマシン ホストに使用できます。

- [外部のサブネット]

送信アクセスのあるオンデマンド ネットワークには、送信アクセスを持つ外部サブネットが必要です。外部サブネットは、クラウド テンプレートで要求された場合に送信アクセスを提供するために使用され、ネットワークの配置は制御しません。たとえば、外部サブネットはプライベート ネットワークの配置には影響しません。

- [CIDR]

CIDR 表記は、IP アドレスとそれに関連付けられているルーティング プリフィックスを簡潔に表したものです。CIDR 値は、サブネットを作成するためにプロビジョニング中に使用されるネットワーク アドレス範囲を指定します。[ネットワーク ポリシー] タブでのこの CIDR 設定は、/nn で終わり、0 ~ 32 の値を含む IPv4 表記を受け入れます。

■ [サブネット サイズ]

このオプションでは、IPv4 表記を使用して、このネットワーク プロファイルを使用する展開内の隔離された各ネットワークに対してオンデマンド ネットワークのサイズを指定します。サブネット サイズの設定は、内部または外部の IP アドレス管理に使用できます。

IPv6 フォーマットは、オンデマンド ネットワークではサポートされていません。

■ [分散論理ルーター]

オンデマンド ルーティング ネットワークの場合、NSX-V クラウド アカウントを使用するには分散論理ネットワークを指定する必要があります。

分散論理ルーター (DLR) は、NSX-V 上のオンデマンド ルーティング ネットワーク間で、East/West トラフィックをルーティングするために使用されます。このオプションは、ネットワーク プロファイルのアカウント/地域の値が NSX-V クラウドアカウントに関連付けられている場合にのみ表示されます。

■ [IP アドレス範囲の割り当て]

このオプションは、vSphere を含む NSX または VMware Cloud on AWS をサポートするクラウド アカウントで使用できます。

IP アドレス範囲の設定は、外部 IP アドレス管理統合ポイントで既存ネットワークを使用している場合に使用できます。

次の 3 つのオプションのいずれかを選択して、展開ネットワークの IP アドレス範囲割り当てタイプを指定できます。

■ [固定および DHCP]

デフォルトおよび推奨。この混在オプションでは、割り当てられた [CIDR] と [サブネット範囲] の設定を使用して、DHCP サーバ プールが、IP アドレス空間の割り当ての半分で DHCP (動的) の方法を使用し、もう半分で固定の方法を使用するように構成します。このオプションは、オンデマンド ネットワークに接続されている一部のマシンでは割り当てられた固定 IP アドレスが必要で、他のマシンでは動的な IP アドレスが必要な場合に使用します。2 つの IP アドレス範囲が作成されます。

このオプションは、オンデマンド ネットワークに接続されているマシンでの展開 (一部のマシンには固定 IP アドレスが割り当てられ、その他のマシンには NSX DHCP サーバによって IP アドレスが動的に割り当てられる) と、ロード バランサの仮想 IP アドレスが固定である展開に最も効果的です。

■ [DHCP (動的)]

このオプションでは、割り当てられた CIDR を使用して、DHCP サーバ上に IP アドレス プールを構成します。このネットワークのすべての IP アドレスが動的に割り当てられます。割り当てられた CIDR ごとに 1 つの IP アドレス範囲が作成されます。

■ [固定]

このオプションでは、割り当てられた CIDR を使用して、IP アドレスを固定で割り当てます。このオプションは、このネットワークに DHCP サーバを構成する必要がない場合に使用します。割り当てられた CIDR ごとに 1 つの IP アドレス範囲が作成されます。

■ [IP アドレス ブロック]

IP ブロックの設定は、外部 IP アドレス管理統合ポイントでオンデマンド ネットワークを使用している場合に使用できます。

IP アドレス ブロック設定を使用すると、統合された外部 IP アドレス管理プロバイダからネットワーク プロファイルに名前付き IP アドレス ブロックまたは範囲を追加できます。追加された IP アドレス ブロックをネットワーク プロファイルから削除することもできます。IP アドレス管理統合の作成方法の詳細については、[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。

外部 IP アドレス管理は、次のクラウド アカウント/リージョン タイプで使用できます。

- vSphere
- vSphere と NSX-T
- vSphere と NSX-V

■ [ネットワークリソース - 外部ネットワーク]

外部ネットワークは、既存のネットワークとも呼ばれます。これらのネットワークはデータ収集され、選択できるようになります。

■ [ネットワークリソース - Tier-0 論理ルーター]

NSX-T では、Tier-0 論理ルーターを使用して、NSX 環境の外部のネットワークへのゲートウェイを提供します。Tier-0 論理ルーターは、オンデマンド ネットワークの送信アクセスを構成します。

■ [ネットワークリソース - Edge クラスタ]

指定した Edge クラスタは、ルーティング サービスを提供します。Edge クラスタは、オンデマンド ネットワークおよびロード バランサの送信アクセスを構成するために使用されます。Edge クラスタは、Edge アプライアンスが展開される Edge クラスタ（リソース プール）を識別します。

■ [ネットワークリソース - Edge データストア]

指定された Edge データストアを使用して、Edge アプライアンスをプロビジョニングします。この設定は、NSX-V にのみ適用されます。

タグを使用して、クラウド テンプレートで使用可能なネットワークを指定できます。

ロード バランサ

ネットワーク プロファイルにロード バランサを追加できます。ソース クラウド アカウントからデータ収集された情報に基づいて、ロード バランサが一覧表示されます。

ネットワーク プロファイルのいずれかのロード バランサのタグが、クラウド テンプレートのロード バランサ コンポーネント内のタグと一致する場合、このロード バランサは展開時に考慮されます。クラウド テンプレートが展開されると、一致するネットワーク プロファイルのロード バランサが使用されます。

詳細については、[vRealize Automation Cloud Assembly のネットワーク プロファイルでのロード バランサ設定の使用](#)および[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

セキュリティ グループ

クラウド テンプレートが展開されると、ネットワーク プロファイル内のセキュリティ グループが、プロビジョニングされたマシン NIC に適用されます。Amazon Web Services 固有のネットワーク プロファイルの場合、ネットワーク プロファイル内のセキュリティ グループは、[ネットワーク] タブに表示されているネットワークと同じネットワーク ドメイン (VPC) 内で使用できます。ネットワーク プロファイルの [ネットワーク] タブにネットワークが表示されていない場合は、使用可能なすべてのセキュリティ グループが表示されます。

セキュリティ グループを使用すると、オンデマンドの private または outbound ネットワークの隔離設定をさらに定義できます。また、セキュリティ グループは existing ネットワークにも適用されます。

表示されているセキュリティ グループは、ソース クラウド アカウントからデータ収集された情報、またはプロジェクト クラウド テンプレートにオンデマンド セキュリティ グループとして追加された情報に基づいて使用できます。詳細については、[vRealize Automation のセキュリティ リソース](#)を参照してください。

セキュリティ グループは、ネットワーク プロファイルに一致するネットワークに接続されている展開内のすべてのマシンに適用されます。クラウド テンプレートにネットワークが複数あり、それぞれが異なるネットワーク プロファイルに一致することがあります。その場合、ネットワークごとに異なるセキュリティ グループを使用できます。

既存のセキュリティ グループにタグを追加すると、クラウド テンプレート Cloud.SecurityGroup コンポーネントでセキュリティ グループを使用できるようになります。セキュリティ グループには、1 つ以上のタグが必要です。タグがないと、クラウド テンプレートで使用することはできません。詳細については、[vRealize Automation のセキュリティ リソース](#)および[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

ネットワーク プロファイル、ネットワーク、クラウド テンプレート、およびタグに関する詳細情報

ネットワークの詳細については、[vRealize Automation のネットワーク リソース](#)を参照してください。

クラウド テンプレートのサンプル ネットワーク コンポーネントのコードの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

ネットワーク自動化ワークフローのサンプルについては、[Network Automation with Cloud Assembly and NSX](#)を参照してください。

タグおよびタグの使用の詳細については、[vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法](#)を参照してください。

vRealize Automation のネットワーク プロファイルおよびクラウド テンプレートでのネットワーク設定の使用

vRealize Automation でネットワークおよびネットワーク プロファイルを使用して、お使いの展開環境に対するネットワーク プロビジョニングの動作を定義できます。

vRealize Automation では、クラウド固有のネットワーク プロファイルを定義できます。[vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください。

ネットワークおよびネットワーク プロファイル設定を使用して、vRealize Automation クラウド テンプレートおよび展開でのネットワーク IP アドレスの使用方法を制御できます。

vRealize Automation ネットワークでの IPv4 および IPv6 のサポート

vRealize Automation ネットワークは、ピュア IPv4 または IPv4 と IPv6 のデュアル スタックをサポートします。現在、ピュア IPv6 はサポートしていません。

ピュア IPv4 はすべてのクラウド アカウントと統合タイプでサポートされますが、IPv4 と IPv6 のデュアルスタックは、vSphere クラウド アカウントとそのエンドポイントでのみサポートされます。

現在、IPv6 では、ロード バランサ、NSX オンデマンド ネットワーク、外部のサードパーティ製 IP アドレス管理 プロバイダとの使用はサポートされていません。

外部 IP アドレス管理プロバイダのサポート

提供されている内部 IP アドレス管理がサポートされているだけでなく、外部 IP アドレス管理プロバイダを使用してネットワークの IP アドレスを動的または静的に割り当てることができます。クラウド テンプレートのデザインと展開内の既存ネットワークに対しては IP アドレス範囲として割り当てることができ、クラウド テンプレートのデザインと展開内のオンデマンド ネットワークに対しては、IP アドレス ブロックとして割り当てることができます。

Infoblox などの外部 IP アドレス管理プロバイダのサポートは、[インフラストラクチャ] - [接続] - [統合の追加] - [IP アドレス管理] メニュー シーケンスを使用して作成するベンダー固有の IP アドレス管理の統合ポイントで使用できます。

[ネットワーク ポリシー] - [IP アドレス管理の IP アドレス範囲の追加] 画面で [IP アドレス管理の IP アドレス範囲の追加] オプションを使用すると、外部 IP アドレス管理プロバイダのアドレス情報を定義するオプションを使用できます。

IP アドレス管理統合ポイントの作成方法の詳細については、[vRealize Automation での外部 IP アドレス管理統合の構成方法](#) を参照してください。特定の IP アドレス管理ベンダーに対して IP アドレス管理統合ポイントを作成する方法の例については、[チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成](#) を参照してください。

ネットワーク タイプ

クラウド テンプレート内のネットワーク コンポーネントは、以下のいずれかの `networkType` タイプとして定義されます。

ネットワーク タイプ	定義
<code>existing</code>	vCenter、Amazon Web Services、Microsoft Azure など、基盤となるクラウド プロバイダで構成されている既存のネットワークを選択します。outbound のオンデマンド ネットワークでは、既存のネットワークが必要です。 既存のネットワークで固定 IP アドレスの範囲を定義できます。
<code>public</code>	パブリック ネットワーク上のマシンには、インターネットからアクセスできます。IT 管理者は、これらのネットワークを定義します。public ネットワークの定義は、パブリック ネットワークでネットワーク トラフィックの発生が許可されるネットワークの existing ネットワークの定義と同じです。

ネットワーク タイプ	定義
private	<p>オンデマンド ネットワーク タイプ。</p> <p>ネットワーク トラフィックが、展開されたネットワーク上のリソース間でのみ発生するように制限します。これによって受信および送信トラフィックが防止されます。NSX では、オンデマンド NAT の 1 対多に相当します。</p>
outbound	<p>オンデマンド ネットワーク タイプ。</p> <p>ネットワーク トラフィックが展開内のコンピューティング リソース間で発生するように制限しますが、一方向の送信ネットワーク トラフィックも許可します。NSX では、外部 IP アドレスを持つオンデマンド NAT の 1 対多に相当します。</p>
routed	<p>オンデマンド ネットワーク タイプ。</p> <p>ルーティング ネットワークには、一緒にリンクされた使用可能なサブネット全体に分配されるルーティング可能な IP アドレス空間が含まれます。同じルーティング ネットワーク プロファイルを持ち、ルーティング ネットワークを使用してプロビジョニングされる仮想マシンは、互いに通信できるほか、既存のネットワークとも通信できます。</p> <p>ルーティング ネットワークは、NSX-V ネットワークおよび NSX-T ネットワークに使用できるオンデマンド ネットワーク タイプです。Microsoft Azure および Amazon Web Services では、デフォルトでこの接続が提供されます。</p> <p>routed ネットワークは、Cloud.NSX.Network ネットワーク コンポーネントのクラウド テンプレート仕様でのみ使用できます。</p>

ネットワーク コンポーネント データを含む、ポピュレート済みのクラウド テンプレートの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

ネットワーク シナリオ

次のネットワーク プロファイル設定を使用するクラウド テンプレートを展開すると、以下の動作が予想されます。

ネットワークのタイプまたはシナリオ	クラウド ソーンで使用できるネットワーク プロファイルなし	クラウド ソーンで使用可能なネットワーク プロファイルあり
ネットワークなし	<p>クラウド テンプレートにネットワークが指定されていない場合は、コンピューティングと同じプロビジョニング リージョンからランダムにネットワークが選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>使用可能なプロビジョニング リージョンにネットワークがないと、プロビジョニングは失敗します。</p>	<p>ネットワークは、一致するネットワーク プロファイルから選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>どのネットワーク プロファイルも基準を満たしていない場合、プロビジョニングは失敗します。</p>
既存のネットワーク	<p>クラウド テンプレートのネットワーク コンポーネントに制約タグが含まれている場合は、これらの制約を使用して使用可能なネットワークのリストをフィルタします。クラウド テンプレートのネットワーク コンポーネントに含まれる制約タグは、ネットワーク タグ、および利用可能な場合はネットワーク プロファイルの制約タグに一致します。</p> <p>ネットワークのフィルタ済みリストから、単一のネットワークがコンピューティングと同じプロビジョニング リージョンから選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>制約に基づいてフィルタした後で、プロビジョニング リージョンにネットワークがなくなると、プロビジョニングは失敗します。</p>	<p>ネットワークは、一致するネットワーク プロファイルから選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>どのネットワーク プロファイルも基準を満たしていない場合、プロビジョニングは失敗します。</p> <p>ネットワーク制約を使用し、事前割り当て済みのタグに基づいてプロファイル内の既存のネットワークをフィルタできます。</p>
パブリック ネットワーク	<p>ネットワークに制約がある場合は、これらの制約を使用して、supports public IP 属性セットが含まれる使用可能なネットワークのリストをフィルタします。</p> <p>ネットワークのフィルタ済みリストから、ネットワークがコンピューティングと同じプロビジョニング リージョンからランダムに選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>制約に基づいてフィルタした後で、プロビジョニング リージョンにパブリック ネットワークがなくなると、プロビジョニングは失敗します。</p>	<p>一致するネットワーク プロファイルから、supports public IP 属性を持つネットワークが選択されます。</p> <p>デフォルトとしてラベル付けされたネットワークに、環境設定が指定されます。</p> <p>ネットワーク制約を使用し、事前割り当て済みのタグに基づいてプロファイル内の既存のパブリック ネットワークをフィルタできます。</p>
プライベート ネットワーク	<p>プライベート ネットワークにはネットワーク プロファイルからの情報が必要なため、プロビジョニングは失敗します。</p>	<p>新しいネットワークまたは新しいセキュリティ グループが、一致するネットワーク プロファイルの設定に基づいて作成されます。</p> <p>ネットワーク制約タグを使用して、ネットワーク プロファイルおよびネットワークをフィルタできます。</p>

ネットワークのタイプまたはシナリオ	クラウド ソーンで使用できるネットワーク プロファイルなし	クラウド ソーンで使用可能なネットワーク プロファイルあり
送信ネットワーク	送信ネットワークにはネットワーク プロファイルからの情報が必要なため、プロビジョニングは失敗します。	新しいネットワークまたは新しいセキュリティ グループが、一致するネットワーク プロファイルの設定に基づいて作成されます。 ネットワーク制約タグを使用して、ネットワーク プロファイルおよびネットワークをフィルタできます。
オンデマンド ルーティング ネットワーク	ルーティング ネットワークにはネットワーク プロファイルからの情報が必要なため、プロビジョニングは失敗します。	NSX-V の場合は、分散論理ルーター (DLR) を選択する必要があります。 NSX-T および VMware Cloud on AWS の場合、プライベート ネットワークや送信ネットワークと同様のオンデマンド設定が必要です。
既存のネットワークまたはパブリック ネットワークでの WordPress の使用事例サンプル	既存のネットワークまたはパブリック ネットワークでの説明のとおりプロビジョニングが実行されます。	既存のネットワークおよびパブリック ネットワークの動作については、上記の説明を参照してください。 チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト を参照してください。
既存のネットワークまたはパブリック ネットワーク、およびプライベート ネットワークまたは送信ネットワークでの WordPress の使用事例サンプル	ネットワークにはネットワーク プロファイルからの情報が必要であるため、プロビジョニングは失敗します。	プライベート ネットワークおよび送信ネットワークについては、上記の説明を参照してください。 チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト を参照してください。
ロード バランサでの WordPress の使用事例サンプル	ロード バランサはネットワーク プロファイルからの情報を必要とするため、プロビジョニングは失敗します。 プロビジョニングは、既存のロード バランサがある場合に実行される可能性があります。	新しいロード バランサは、ネットワーク プロファイルの設定に基づいて作成されます。 ネットワーク プロファイルで有効になっている既存のロード バランサを指定できます。 既存のロード バランサを申請した場合、プロビジョニングは失敗しますが、ネットワーク プロファイルで制約されることはありません。 チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト を参照してください。

vRealize Automation Cloud Assembly のネットワーク プロファイルおよびクラウド テンプレート デザインでのセキュリティ グループ設定の使用

ネットワーク プロファイルおよびクラウド テンプレート デザインで、セキュリティ グループ設定を定義および変更することができます。

セキュリティ グループ機能は、次のようないくつかの方法で使用できます。

- ネットワーク プロファイルで指定されている既存のセキュリティ グループ

既存のセキュリティ グループをネットワーク プロファイルに追加できます。クラウド テンプレート デザインでそのネットワーク プロファイルを使用する場合、そのマシンはセキュリティ グループのメンバーとしてグループ化されます。この方法では、クラウド テンプレート デザインにセキュリティ グループ リソースを追加する必要はありません。また、この構成でロード バランサを使用することもできます。関連情報については、[vRealize Automation クラウド テンプレートでのロード バランサ リソースの使用](#)を参照してください。

- クラウド テンプレート デザイン内のマシン リソースに関連付けられたセキュリティ グループ コンポーネント
セキュリティ グループ リソースをクラウド テンプレート デザインにドラッグ アンド ドロップし、セキュリティ グループ リソースをマシン NIC にバインドすることができます。その際には、クラウド テンプレート デザイン内の既存のセキュリティ グループ リソースと、データ収集されたリソース内の既存のセキュリティ グループに対して、制約タグを使用します。また、この関連付けは、クラウド テンプレート デザイン キャンバス上のマシンにネットワークを関連付ける方法と同様に、クラウド テンプレート デザイン キャンバスの接続ラインにオブジェクトを接続することによって行うこともできます。

セキュリティ グループのリソースをクラウド テンプレート デザイン キャンバスにドラッグ アンド ドロップすると、existing または new のタイプになることがあります。existing セキュリティ グループ タイプの場合は、プロンプトに従って、タグ制約値を追加する必要があります。new セキュリティ グループ タイプの場合は、ファイアウォール ルールを設定できます。

- タグ制約を使用して割り当てられ、クラウド テンプレートのマシン NIC に関連付けられた既存のセキュリティ グループ

たとえば、2 つのリソース間でタグを一致させることで、クラウド テンプレート デザインのマシン リソースのマシン NIC とセキュリティ グループ リソースを関連付けることができます。

NSX-T の例として、ソース エンドポイントでタグが指定されている場合、NSX-T アプリケーションで指定されている NSX-T タグを使用できます。その後、ネットワーク リソースがクラウド テンプレート デザインのマシン NIC に接続されているときに、クラウド テンプレート デザイン内のネットワーク リソースの制約として指定された NSX-T タグを使用できます。NSX-T タグを使用すると、NSX-T ソース エンドポイントからデータ収集された事前定義済みの NSX-T タグを使用して、マシンを動的にグループ化できます。NSX-T で NSX-T タグを作成するときに、論理ポートを使用します。

- クラウド テンプレート デザインのオンデマンド セキュリティ グループ リソースのファイアウォール ルール
クラウド テンプレート デザインで、オンデマンド セキュリティ グループにファイアウォール ルールを追加できます。

使用可能なファイアウォール ルールについては、[vRealize Automation クラウド テンプレートでのセキュリティ グループ リソースの使用](#)を参照してください。

詳細情報

ネットワーク プロファイルでのセキュリティ グループの定義については、[vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください。

インフラストラクチャ リソース ページでのセキュリティ グループ設定の表示と変更については、[vRealize Automation のセキュリティ リソース](#)を参照してください。

クラウド テンプレート デザインでのセキュリティ グループの定義については、[vRealize Automation クラウド テンプレートでのセキュリティ グループ リソースの使用](#)を参照してください。

クラウド テンプレート デザインにおけるセキュリティ グループ リソースの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

vRealize Automation Cloud Assembly のネットワーク プロファイルでのロード バランサ設定の使用

ネットワーク プロファイル構成で、ロード バランサを設定できます。

[ロード バランサ] タブを使用して、既存のロード バランサをネットワーク プロファイルに追加できます。

ロード バランサをクラウド テンプレート デザインに追加するには、1 つ以上のロード バランサを含むネットワーク プロファイルに関連付けます。クラウド テンプレート デザイン キャンバスまたはコードでロード バランサ リソースを使用して、直接追加することもできます。

ネットワーク プロファイルでセキュリティ グループの使用に基づいてロード バランサ仮想 IP アドレスを含める例

ネットワーク プロファイルで使用できるセキュリティ グループには 2 つのタイプがあります。[セキュリティ グループ] タブから選択する既存のセキュリティ グループと [ネットワーク ポリシー] タブで隔離ポリシーを使用して作成するオンデマンド セキュリティ グループです。

ロード バランサ仮想 IP アドレスが、ネットワーク プロファイル設定に基づくセキュリティ グループに関連付けられている場合、セキュリティ グループ設定はネットワーク プロファイルによって提供されます。

次の表に、サンプル シナリオを示します。

クラウド テンプレート デザイン トポロジ - 関連付けられているリソース	ネットワーク プロファイル構成	セキュリティ グループ メンバーシップ
プライベート ネットワーク上の仮想 IP アドレスを持つ 1 アームのロード バランサと同じプライベート ネットワーク上のマシン。	選択したネットワーク プロファイルは、オンデマンド セキュリティ グループとして定義された隔離ポリシーを使用します。	マシン NIC とロード バランサの仮想 IP アドレスが隔離セキュリティ グループに追加される。
プライベート ネットワーク上の仮想 IP アドレスを持つ 1 アームのロード バランサと同じプライベート ネットワーク上のマシン。	選択したネットワーク プロファイルは、既存のセキュリティ グループを使用し、オンデマンド セキュリティ グループとして定義された隔離ポリシーを使用します。	マシン NIC とロード バランサの仮想 IP アドレスが隔離セキュリティ グループと既存のセキュリティ グループに追加される。
パブリック ネットワーク上の仮想 IP アドレスを持つ 2 アームのロード バランサと、プライベート ネットワーク上のマシン。	選択したネットワーク プロファイルは、既存のセキュリティ グループを使用し、オンデマンド セキュリティ グループとして定義された隔離ポリシーを使用します。	マシン NIC とロード バランサの仮想 IP アドレスが隔離セキュリティ グループと既存のセキュリティ グループに追加される。
パブリック ネットワーク上の仮想 IP アドレスを持つ 2 アームのロード バランサと、プライベート ネットワーク上のマシン。	選択したネットワーク プロファイルは、既存のセキュリティ グループを使用します。	マシン NIC とロード バランサの仮想 IP アドレスが既存のセキュリティ グループに追加される。
2 アームのロード バランサ、仮想 IP アドレスはネットワーク 1 上、マシンはネットワーク 2 上。	2 つのネットワーク プロファイル： <ul style="list-style-type: none"> ■ ネットワーク プロファイル 1: 既存のセキュリティ グループ 1 を使用します。 ■ ネットワーク プロファイル 2: 既存のセキュリティ グループ 2 を使用します。 	ロード バランサはネットワーク プロファイル 1 に、マシンはネットワーク プロファイル 2 に格納される。 ロード バランサの仮想 IP アドレスがセキュリティ グループ 1 に追加され、マシン NIC がセキュリティ グループ 2 に追加される。

詳細情報

クラウド テンプレート デザインにロード バランサ リソースを追加する方法については、[vRealize Automation クラウド テンプレートでのロード バランサ リソースの使用](#)を参照してください。

ロード バランサを含むクラウド テンプレート デザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

vRealize Automation で外部 IP アドレス管理統合用のオンデマンド ネットワークをサポートするようにネットワーク プロファイルを設定する方法

ネットワーク プロファイルが、外部 IP アドレス管理統合を使用する vRealize Automation クラウド テンプレートで使用されている場合、オンデマンド ネットワークの IP アドレスのブロックをサポートするようにこのネットワーク プロファイルを設定できます。

特定の外部 IP アドレス管理プロバイダに既存の統合を使用すると、外部 IP アドレス管理システムに新しいネットワークを作成するようにオンデマンド ネットワークをプロビジョニングすることができます。

このプロセスを使用して、親 CIDR を指定する代わりに、IP アドレスのブロックを構成します (vRealize Automation の内部 IP アドレス管理を使用する場合と同様)。IP アドレス ブロックは、新しいネットワークをセグメント化するためにオンデマンド ネットワークのプロビジョニング中に使用されます。統合でオンデマンド ネットワークがサポートされている場合は、外部 IP アドレス管理プロバイダから IP アドレス ブロックのデータが収集されます。たとえば、Infoblox IP アドレス管理統合を使用している場合、IP アドレス ブロックは Infoblox ネットワーク コンテナを表します。

クラウド テンプレートでオンデマンド ネットワーク プロファイルと外部 IP アドレス管理統合を使用している場合は、クラウド テンプレートの展開時に次のイベントが発生します。

- 外部 IP アドレス管理プロバイダにネットワークが作成されます。
- vRealize Automation にもネットワークが作成され、CIDR やゲートウェイ プロパティなどの設定を含む新しいネットワーク設定が IP アドレス管理プロバイダから反映されます。
- 展開された仮想マシンの IP アドレスは、新しく作成されたネットワークから取得されます。

このオンデマンド ネットワークの例では、クラウド テンプレート展開において外部 IP アドレス管理プロバイダとして Infoblox を使用することにより、vSphere からオンデマンド ネットワークにマシンをプロビジョニングできるように、ネットワーク プロファイルを構成します。

関連情報については、[vRealize Automation で外部 IP アドレス管理統合用の既存ネットワークをサポートするようにネットワーク プロファイルを設定する方法](#)を参照してください。どちらのネットワーク設定の例も、チュートリアル : [vRealize Automation 用 VMware Cloud on AWS の構成](#)に示されている、外部 IP アドレス管理統合用のベンダー固有のワークフロー全体に適合します。

前提条件

次の前提条件は、ネットワーク プロファイルを作成または編集するユーザーに適用されますが、IP アドレス管理統合を含むクラウド テンプレート展開で使用する場合は、ネットワーク プロファイル自体が適用されます。ベンダー固有の IP アドレス管理統合ポイントについては、[vRealize Automation での外部 IP アドレス管理統合の構成方法](#)を参照してください。

この一連の手順は、IP アドレス管理プロバイダの統合ワークフローのコンテキストの一部として表示されます。[チュートリアル：vRealize Automation のプロバイダ固有の外部 IP アドレス管理統合の構成](#) を参照してください。

- クラウド管理者権限が付与されていることを確認します。[vRealize Automation でクラウド アカウントを使用するために必要な認証情報](#)を参照してください。
- クラウド管理者ユーザー ロールが割り当てられていることを確認します。[vRealize Automation のユーザーロールについて](#)を参照してください。
- [Infoblox](#) や [Bluecat](#) などの外部 IP アドレス管理プロバイダにアカウントがあること、IP アドレス管理プロバイダに組織のアカウントへの適切なアクセス認証情報があることを確認します。このワークフローの例では、IP アドレス管理プロバイダは Infoblox です。
- IP アドレス管理プロバイダの IP アドレス管理統合ポイントがあること、および IP アドレス管理統合を作成するために使用される IP アドレス管理パッケージがオンデマンド ネットワークをサポートしていることを確認します。[vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) を参照してください。

Infoblox IP アドレス管理パッケージがオンデマンド ネットワークをサポートしていても、使用しているのが別のプロバイダの外部 IP アドレス管理統合である場合は、その IP アドレス管理統合パッケージがオンデマンド ネットワークをサポートしていることを確認してください。

手順

- 1 ネットワーク プロファイルを設定するには、[インフラストラクチャ] - [構成] - [ネットワーク プロファイル] の順にクリックします。
- 2 [新しいネットワーク プロファイル] をクリックします。
- 3 [サマリ] タブをクリックして、次のサンプル設定を指定します。
 - vSphere クラウド アカウント/リージョンを指定します (**vSphere-IPAM-OnDemandA/Datacenter** など)。

この例では、NSX クラウド アカウントに関連付けられていない vSphere クラウド アカウントを使用することを想定しています。

 - ネットワーク プロファイルに名前を付けます (**Infoblox-OnDemandNP** など)。
 - ネットワーク プロファイルの機能タグ (**infoblox_ondemandA** など) を追加します。

機能タグは、クラウド テンプレートをプロビジョニングするときに使用するネットワーク プロファイルの関連付けを作成するためのクラウド テンプレートの制約タグとしても使用する必要があるため、書き留めておきます。
- 4 [ネットワーク ポリシー] タブをクリックして、次のサンプル設定を指定します。
 - [隔離ポリシー] ドロップダウン メニューで [オンデマンド ネットワーク] を選択します。

このオプションでは、外部 IP アドレス管理の IP アドレス ブロックを使用できます。クラウド アカウントによっては、新しいオプションが表示されることがあります。たとえば、NSX クラウド アカウントに関連付けられている vSphere クラウド アカウントを使用している場合は、次のオプションが表示されます。

 - トランスポート ゾーン
 - Tier-0 論理ルーター

■ Edge クラスタ

この例では、vSphere クラウド アカウントが NSX に関連付けられていないため、[ネットワーク ドメイン] メニュー オプションが表示されます。

- [ネットワーク ドメイン] オプションは空白のまま残します。

- 5 アドレス管理の [ソース] として [外部] をクリックします。
- 6 [IP アドレス ブロックの追加] をクリックして、[IP アドレス管理の IP アドレス ブロックの追加] 画面を開きます。
- 7 [IP アドレス管理の IP アドレス ブロックの追加] 画面の [プロバイダ] メニューで、既存の外部 IP アドレス管理統合を選択します。たとえば、サンプル ワークフローの [vRealize Automation での Infoblox 用外部 IP アドレス管理統合の追加](#) で *Infoblox_Integration* 統合ポイントを選択します。
- 8 [アドレス空間] メニューで、使用可能な一覧表示されている IP ブロックの中から 1 つ ([10.23.118.0/24]) を選択して、追加します。

IP アドレス管理プロバイダがアドレス空間をサポートしている場合は、[アドレス空間] メニューが表示されます。Infoblox 統合の場合、アドレス空間は Infoblox ネットワーク ビューで表されます。
- 9 [サブネット サイズ] ([/29 (-6 IP addresses)]) など) を選択します。
- 10 [作成] をクリックします。

結果

ネットワーク プロファイルが作成され、指定した外部 IP アドレス管理統合を使用してオンデマンド ネットワークをプロビジョニングする際に使用できるようになります。次のサンプル クラウド テンプレートは、この新しいネットワーク プロファイルで定義されているネットワークに展開する単一のマシンを示しています。

```
formatVersion: 1
inputs: {}
resources:
  Cloud_Machine_1:
    type: Cloud.Machine
    properties:
      image: ubuntu
      flavor: small
      networks:
        - network: '${resource.Cloud_Network_1.id}'
          assignment: static
  Cloud_Network_1:
    type: Cloud.Network
    properties:
```



```
networkType: private
constraints: - tag: infoblox_ondemandA
```

注： クラウド テンプレートが展開されている場合は、指定した IP アドレス ブロック内の使用可能な最初のネットワークが取得され、ネットワーク CIDR と見なされます。クラウド テンプレートで NSX ネットワークを使用している場合は、代わりに、次のようにネットワーク プロパティ `networkCidr` を使用して、ネットワークの CIDR を手動で設定できます。これにより、CIDR を手動で設定し、関連付けられたネットワーク プロファイルで指定されている IP アドレス ブロックとサブネット サイズの設定をオーバーライドすることができます。

```
Cloud_Network_1:
  type: Cloud.Network
  properties:
    networkCidr: 10.10.0.0/16
```

vRealize Automation で外部 IP アドレス管理統合用の既存ネットワークをサポートするようにネットワーク プロファイルを設定する方法

ネットワーク プロファイルが、外部 IP アドレス管理統合を使用する vRealize Automation ブループリントで使用されている場合、既存ネットワークの IP アドレス範囲をサポートするようにこのネットワーク プロファイルを設定できます。

vRealize Automation で既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定 に、ベンダー固有のサンプル ワークフローのコンテキストにおける例を示します。外部 IP アドレス管理統合用のベンダー固有のワークフロー全体については、[チュートリアル：vRealize Automation 用 VMware Cloud on AWS の構成](#)を参照してください。

関連情報については、[vRealize Automation で外部 IP アドレス管理統合用のオンデマンド ネットワークをサポートするようにネットワーク プロファイルを設定する方法](#)を参照してください。

要件の異なる複数の vRealize Automation Cloud Assembly ストレージ プロファイルを追加する方法

vRealize Automation Cloud Assembly ストレージ プロファイルは、展開されるストレージの種類を示します。

ストレージは通常、サービス レベルやコスト、パフォーマンス、目的（バックアップなど）などの特性に基づいてプロファイルが決定されます。

[インフラストラクチャ] - [設定] - [ストレージ プロファイル] の順に選択し、[新規ストレージ プロファイル] をクリックします。

vRealize Automation でのストレージ プロファイルの詳細

クラウド アカウント リージョンには、クラウド管理者が vRealize Automation のリージョンにストレージを定義できるストレージ プロファイルが含まれます。

ストレージ プロファイルには、ディスクのカスタマイズ、および機能タグによってストレージの種類を識別する手段が含まれます。その後、タグは、展開時に目的のストレージを作成するために、プロビジョニング サービス要求の制約と照合されます。

ストレージ プロファイルは、クラウド固有のリージョンの下に編成されます。1つのクラウド アカウントに、それぞれに複数のストレージ プロファイルを持つ複数のリージョンがある場合があります。

ベンダーに依存しない配置が可能です。たとえば、3つの異なるベンダー アカウントとそれぞれのリージョンを視覚化します。各リージョンには、fast というタグが付けられた機能を持つストレージ プロファイルが含まれます。プロビジョニング時に、ハードな fast 制約タグを含む要求は、リソースを提供しているベンダー クラウドに関係なく、一致する fast の機能を検索します。次に、一致する内容によって、展開されたストレージ アイテムの作成時に、関連付けられたストレージ プロファイルからの設定が適用されます。

注： クラウド ストレージが異なると、パフォーマンス特性が異なる場合がありますが、タグ付けした管理者によって提供される fast が考慮されます。

ストレージ プロファイルに追加する機能タグは、実際のリソース ターゲットを識別しないようにする必要があります。代わりに、ストレージの種類を記述します。実際のリソースの詳細については、[vRealize Automation のストレージ リソース](#)を参照してください。

ストレージ プロファイル画面の [ディスク タイプ] オプション、または vRealize Automation API を使用して、First Class Disk (FCD) ストレージまたは標準ディスク ストレージをサポートするストレージ プロファイルを作成できます。First Class Disk (FCD) オプションを選択すると、実質的には vSphere ストレージ プロファイルが作成されます。

■ First Class Disk

First Class Disk は、vSphere 仮想マシンとは無関係に作成と管理をすることができます。FCD のライフサイクル管理機能は、仮想マシンから独立して動作します。FCD は vSphere バージョン 6.7 Update 2 以降で使用でき、現在、vRealize Automation には API のみの機能として実装されています。

vRealize Automation API を使用して利用できる機能や API ドキュメント自体へのリンクも含め、First Class Disk (FCD) ストレージについては [vRealize Automation の First Class Disk ストレージで実行できることを参照してください](#)。

■ 標準ディスク

標準ディスク ストレージは、仮想マシンの統合コンポーネントとして作成および管理されます。

標準ディスク ストレージの詳細については、[vRealize Automation の標準ディスク ストレージで実行できる操作](#)と [vRealize Automation のパーシステント ディスク ストレージで実行できる操作](#)を参照してください。

vRealize Automation Cloud Assembly のリソースと展開を管理するためにタグを使用する方法

タグは、機能と制約を照合して展開を配置する vRealize Automation Cloud Assembly の重要なコンポーネントです。vRealize Automation Cloud Assembly を最適に使用するには、タグを理解して効果的に実装する必要があります。

基本的に、タグは vRealize Automation Cloud Assembly アイテムに追加するラベルです。組織と実装に適したタグを任意に作成できます。ただし、タグの機能はラベルをはるかに超えています。タグは、展開可能なサービスをビルドする際に、vRealize Automation Cloud Assembly によるリソースおよびインフラストラクチャの使用方法や使用場所を制御するためです。また、タグは Cloud Assembly 内のガバナンスもサポートしています。

タグの構造

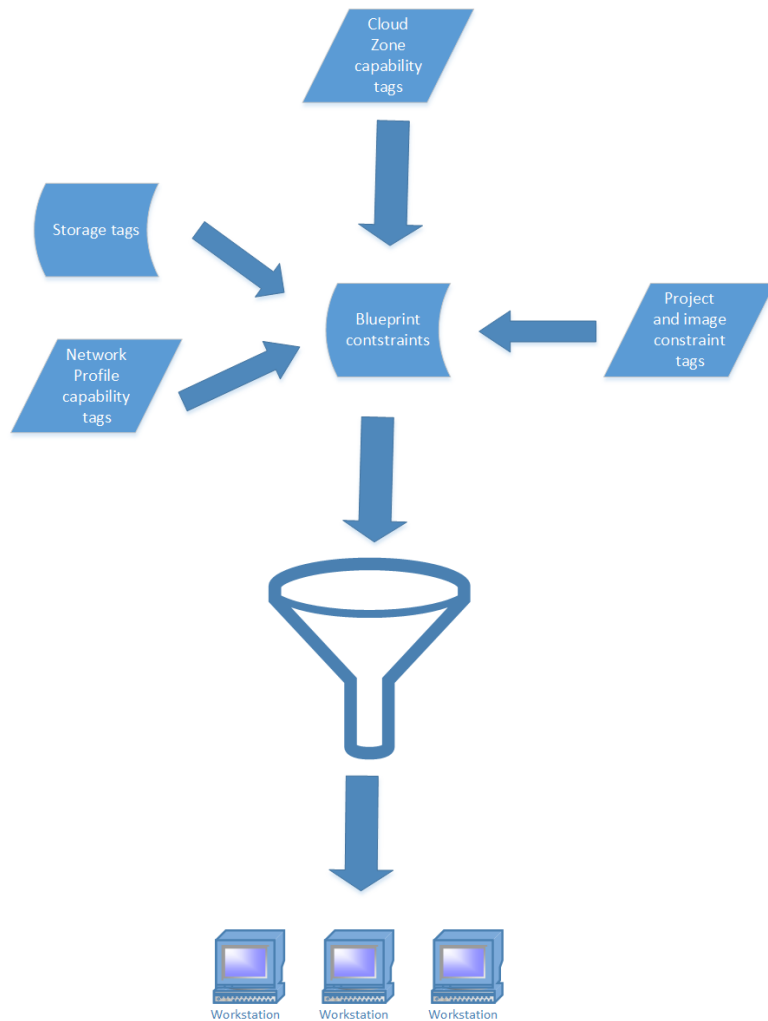
タグは、構造的には `name:value` ペアの規則に準拠する必要がありますが、それを除けばほぼ自由に形成できます。vRealize Automation Cloud Assembly 全体で、いずれのタグも同じように表示され、タグの機能はコンテキストによって決まります。

たとえば、インフラストラクチャ リソースのタグは主に機能タグとして動作します。vRealize Automation Cloud Assembly がこのタグを使用してリソースと展開とを照合するためです。また、リソースも特定します。

タグの機能

タグの主な機能は、vRealize Automation Cloud Assembly が展開の定義に使用する機能と制約を示すことです。タグの機能は、コンテキストによって決定されます。クラウド ゾーン、ネットワーク プロファイル、ストレージ プロファイル、各インフラストラクチャ リソースに配置されたタグは、機能タグとして機能し、展開に使用されるインフラストラクチャに必要な機能を定義します。クラウド テンプレートに配置されたタグは、展開用のリソースを定義する制約として機能します。また、クラウド管理者は、プロジェクトに制約タグを配置することで、これらのプロジェクトに対する制御を実行することもできます。この制約タグは、クラウド テンプレートで表現される他の制約に追加されます。

プロビジョニング時に、vRealize Automation Cloud Assembly はクラウド テンプレートでこのような機能を、制約（タグとも呼ばれる）と照合して、展開構成を定義します。このタグベースの機能と制約機能は、vRealize Automation Cloud Assembly で展開構成の基盤として機能します。たとえば、タグを使用して、特定のリージョンの PCI リソースでのみインフラストラクチャを使用するようにできます。



また、タグをセカンダリ レベルで使用すると、ストレージ アイテムやネットワーク アイテムをはじめとするインフラストラクチャ リソースの検索および特定が容易になります。

たとえば、クラウド ゾーンを設定しており、使用できるコンピューティング リソースが多数あるとします。コンピューティング リソースに適切にタグ付けしている場合は、[クラウド ゾーン] 画面の [計算] タブにある検索機能を使用して、その特定のクラウド ゾーンに関連付けられているリソースをフィルタリングできます。

また、[タグの管理] 画面およびリソース構成画面には、タグ名でアイテムを検索できる検索機能があります。タグ名で検索して特定する機能を使用するには、これらのアイテムに論理的で人間が判読可能なタグを使用することが重要です。

タグの使用の詳細と例については、次の YouTube のビデオを参照してください： <https://youtu.be/4zNQ33RyQio>

外部タグ

vRealize Automation Cloud Assembly には外部タグが含まれる場合もあります。外部タグは、vRealize Automation Cloud Assembly インスタンスに関連付けられたクラウド アカウントから自動的にインポートされます。また、vSphere、AWS や Azure をはじめとする外部ソフトウェア製品からインポートされる場合もあります。インポートされた外部タグは、ユーザーが作成したタグと同じように使用できます。

タグの管理

vRealize Automation Cloud Assembly の [タグの管理] 画面を使用して、タグ ライブラリを監視および管理できます。この画面でタグを作成することもできます。また、[タグの管理] 画面は外部タグを表示および識別できる唯一の画面でもあります。



タグ ストラテジ

混乱を最小限に抑えるため、vRealize Automation Cloud Assembly でタグを作成する前に、適切なタグ ストラテジとタグ付け規則を策定し、タグを作成して使用するすべてのユーザーがタグの意味と使い方を理解できるようにします。[タグ付けストラテジの作成](#)を参照してください。

タグ付けストラテジの作成

Cloud Assembly の機能を最大限に活用し、混乱が生じる可能性を最小限に抑えるためには、組織の IT 構造および目標に基づいて、適切なタグ付けストラテジを綿密に計画して実装する必要があります。

タグ付けは共通性のある複数の目的に利用できますが、タグ付け戦略は展開のニーズ、構造、目標に合わせて調整する必要があります。

タグ付けのベスト プラクティス

効果的なタグ戦略の一般的な特徴は次のとおりです。

- お客様のビジネスの構造と関連性がある一貫したタグ付けストラテジを設計して実装し、該当するすべてのユーザーにこのプランを通達します。ストラテジは、展開ニーズに対応でき、人間が判読できる明確な言葉を使用し、該当するすべてのユーザーが理解できる必要があります。
- タグには、シンプルかつ明確で、わかりやすい名前と値を使用します。たとえば、ストレージとネットワークのアイテムには明確で一貫性のあるタグ名を使用して、ユーザーが何を選択するのかをすぐに理解したり、展開済みのリソースのタグ割り当てを確認したりできるようにします。
- 値のない名前を使用してタグを作成することはできますが、ベスト プラクティスとして、タグの使用方法が他のユーザーに明確に伝わるように、各タグ名に適切な値を作成することをお勧めします。
- 重複するタグや無関係なタグの作成は避けます。たとえば、ストレージの問題に関連したストレージ アイテムのタグのみを作成します。

タグ付けの実装

基本的なタグ付けストラテジについての主要な考慮事項をまとめます。次のリストには、ストラテジを策定する際の一般的な考慮事項を示しています。これらの考慮事項は確定的なものではなく代表的なものであることに注意してください。各自のユースケースとの関連性が高い他の考慮事項がある場合があります。具体的なストラテジは個別のユースケースに適したものである必要があります。

- 展開先となる環境の数。通常は、各環境を表すタグを作成します。
- 展開をサポートするために、どのようにコンピューティング リソースを構造化し、使用するか。
- 展開先となるリージョンや場所の数。通常、これらの各リージョンや各場所を表すタグをプロファイル レベルで作成します。
- 展開に使用可能なストレージ オプションの数と、それらの特徴をどのように説明するか。タグでこれらのオプションを表す必要があります。
- ネットワーク オプションを分類し、該当するすべてのオプションに対応するタグを作成します。
- 一般的な展開変数。たとえば、展開先となる環境の数を指定します。一般的に、多くの組織では、テスト、開発、および本番環境の数を最小限に抑えています。これらの 1 つ以上の環境への展開を簡単に設定できるように、一致する制約タグとクラウド ゾーン機能タグを作成して調整する必要があります。
- ネットワーク リソースとストレージ リソースのタグを調整して、それらのタグが使用されているネットワーク プロファイルとストレージ プロファイルのコンテキストで論理的に理解できるようにします。リソース タグは、リソースの展開をより細かく制御するために使用できます。
- クラウド ゾーンおよびネットワーク プロファイルの機能タグやその他の機能タグと、制約タグとを調整します。一般的に、管理者は最初にクラウド ゾーンとネットワーク プロファイルの機能タグを作成します。その後、他のユーザーはこれらの機能タグに一致する制約を使用して設計ができます。

組織にとって重要な考慮事項を把握すると、その考慮事項に対応する適切なタグ名を論理的方法で計画することができます。その後、ストラテジの概要を策定し、タグを作成または編集する権限を持つすべてのユーザーが利用できるようにします。

有用な実装方法としては、すべてのコンピューティング インフラストラクチャ リソースに個別にタグ付けすることから始めます。説明したように、特定のリソースに関連するタグ名には論理的なカテゴリを使用します。たとえば、ストレージ リソースには tier1、tier2 などのタグを付けることができます。また、Windows や Linux などのオペレーティング システムに基づいてコンピューティング リソースにタグを付けることもできます。

リソースにタグを付けた後は、クラウド ゾーン、ストレージ、およびネットワーク プロファイルのタグを作成するための、ニーズに最適な方法を検討できます。

vRealize Automation Cloud Assembly での機能タグの使用

vRealize Automation Cloud Assembly では、機能タグを使用して、インフラストラクチャ コンポーネントの展開機能を定義できます。この機能は、制約とともに、vRealize Automation の配置ロジックの基礎として機能します。

機能タグは、コンピューティング リソース、クラウド ゾーン、イメージおよびイメージ マップ、ネットワークとネットワーク プロファイルに作成することができます。これらのリソースを作成するためのページには、機能タグを作成するためのオプションが含まれています。または、vRealize Automation Cloud Assembly の [タグの管理] 画面を使用して、機能タグを作成することもできます。クラウド ゾーンおよびネットワーク プロファイルの機能タグは、それらのゾーンまたはプロファイル内のすべてのリソースに影響します。ストレージまたはネットワーク コンポーネントの機能タグは、それらが適用されるコンポーネントにのみ影響します。

通常、機能タグは、コンピューティング リソースの場所、ネットワークのアダプタ タイプ、またはストレージ リソースの階層レベルなどの特性を定義しています。また、環境の場所やタイプなどのビジネス上の考慮事項を定義することもできます。タグ戦略全体と同様に、機能タグはビジネス ニーズに応じて論理的な方法で編成する必要があります。

vRealize Automation Cloud Assembly は、展開時にクラウド ゾーンからの機能タグをクラウド テンプレートの制約と照合します。そのため、機能タグの作成と使用の際には、照合が目的どおりに実行されるように適切なクラウド テンプレートの制約を理解してから作成を計画する必要があります。

たとえば、ドキュメントに付属しているチュートリアル : [vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)の「クラウド ゾーンの追加」トピック セクションには、OurCo-AWS-US-East および OurCo AWS-US-West クラウド ゾーンの開発およびテスト用タグを作成する方法が記載されています。このチュートリアルでは、これらのタグは OurCo-AWS-US-East が開発環境で、OurCo-AWS-US_ West ゾーンがテスト環境であることを示しています。クラウド テンプレートに同様の制約タグを作成すると、これらの機能タグによって目的の環境に直接展開することができます。

vRealize Automation Cloud Assembly での制約タグの使用

プロジェクトおよびクラウド テンプレートに追加されたタグは、インフラストラクチャ リソース、プロファイル、およびクラウド ゾーンの機能タグを照合するために使用される制約タグとして機能します。クラウド テンプレートの場合、vRealize Automation Cloud Assembly は、この照合機能を使用して、展開にリソースを割り当てます。

vRealize Automation Cloud Assembly では、2 つの主要な方法で制約タグを使用できます。1 つ目の方法は、プロジェクトとイメージを構成する場合に使用します。タグを制約として使用することで、リソースをプロジェクトまたはイメージに関連付けることができます。2 つ目の方法は、制約として指定されたタグを使用して展開用のリソースを選択する場合に、クラウド テンプレート内で使用します。いずれの方法でも、適用される制約はクラウド テンプレートでマージされ、展開で使用可能なリソースを定義する一連の展開要件を形成します。

プロジェクトでの制約タグの仕組み

vRealize Automation Cloud Assembly リソースを構成する場合、クラウド管理者はプロジェクトに制約タグを適用できます。このように、管理者はプロジェクト レベルで直接ガバナンスの制約を適用できます。このレベルで追加されたすべての制約は、該当するプロジェクトに対して申請されたすべてのクラウド テンプレートに適用され、これらの制約タグは他のタグよりも優先されます。

プロジェクトの制約タグがクラウド テンプレートの制約タグと競合する場合には、プロジェクトのタグが優先されるため、クラウド管理者はガバナンス ルールを適用できます。たとえば、クラウド管理者がプロジェクトに `location:london` タグを作成し、一方で開発者がクラウド テンプレートに `location:boston` タグを配置した場合、前者が優先されるため、リソースは `location:london` タグを含むインフラストラクチャに展開されます。

プロジェクトには、最大 3 つの制約を適用できます。プロジェクト制約には、強い制約と弱い制約があります。デフォルトでは、強い制約になっています。強い制約を使用すると、展開の制限を厳格に適用できます。満たされない強い制約が 1 つ以上ある場合、展開は失敗します。弱い制約は、環境設定であり、適用可能であれば選択されます。弱い制約が満たされなくても、展開が失敗することはありません。

クラウド テンプレートでの制約タグの仕組み

クラウド テンプレートでは、クラウド管理者がリソース、クラウド ゾーン、ストレージ プロファイル、およびネットワーク プロファイルに作成した適切な機能タグに一致するように、制約タグを YAML コードとしてリソースに追加します。その他にも、制約タグを実装するための複雑なオプションがあります。たとえば、申請に 1 つ以上の タグをポピュレートする変数を使用できます。これにより、申請時に 1 つ以上のタグを指定できます。

制約タグを作成するには、クラウド テンプレート YAML コードの制約見出しの下の `tag` ラベルを使用します。クラウド テンプレートで作成した制約タグに、プロジェクトの制約タグが追加されます。

vRealize Automation Cloud Assembly は、YAML ファイルで制約を簡単に使用できるように、単純な文字列形式をサポートしています。

```
[!tag_key[:tag_value][:hard|soft]
```

デフォルトでは、vRealize Automation Cloud Assembly は強い正の制約を作成します。アプリケーションの他の部分と同様に、タグ値はオプションですが、設定することをお勧めします。

次の「WordPress with MySQL」の例では、コンピューティング リソースの位置情報を表す YAML 制約タグを示しています。

```
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 ...
```

クラウド テンプレートの操作方法の詳細については、[パート 3 : サンプル vRealize Automation Cloud Assembly テンプレートの設計と展開](#)を参照してください。

プロジェクトおよびクラウド テンプレートでの強い制約および弱い制約の仕組み

プロジェクトとクラウド テンプレートのどちらでも、制約を強い制約にも弱い制約にもすることができます。上記のコード スニペットは、強い制約の例でもあり、弱い制約の例でもあります。デフォルトでは、すべての制約が強い制約になります。強い制約を使用すると、展開の制限を厳格に適用できます。満たされない強い制約が 1 つ以上ある場合、展開は失敗します。弱い制約は環境設定であり、使用可能な場合に適用されます。弱い制約が満たされなくても、展開が失敗することはありません。

特定のリソース タイプに対して一連の強い制約および弱い制約がある場合、弱い制約はタイ ブレーカとしても機能します。つまり、複数のリソースが強い制約を満たしている場合は、弱い制約を使用して、展開で使用する実際のリソースを選択します。

たとえば、ネットワーク、ストレージ、および拡張性の各項目を任意に組み合わせて、プロジェクトに対して最大 3 つの制約を指定できます。また、各制約が強いかわかりやすいかを選択できます。たとえば、`location:boston` のタグで強いストレージ制約を作成するとします。プロジェクト内のストレージがこの制約に一致しない場合、関連する展開は失敗します。

標準タグ

vRealize Automation Cloud Assembly は、標準タグを一部の展開に適用して、展開されたリソースの分析、監視、およびグループ化をサポートします。

標準タグは vRealize Automation Cloud Assembly 内で一意です。他のタグとは異なり、展開の設定中にユーザーがこれらのタグを使用することはなく、制約も適用されません。これらのタグは、AWS、Azure、および vSphere の展開でプロビジョニング中に自動的に適用されます。これらのタグは、システムのカスタム プロパティとして保存され、プロビジョニング後に展開に追加されます。

標準タグのリストを以下に示します。

表 4-1. 標準タグ

説明	タグ
組織	<code>org: orgID</code>
プロジェクト	<code>project: projectID</code>
申請者	<code>requester: username</code>
展開	<code>deployment: deploymentID</code>
クラウド テンプレートのリファレンス（該当する場合）	<code>blueprint: blueprintID</code>
ブループリントのコンポーネント名	<code>blueprintResourceName: CloudMachine_1</code>
配置の制約（ブループリント、申請パラメータ、または IT ポリシーを使用して適用）	<code>constraints: key:value:soft</code>
クラウド アカウント	<code>cloudAccount: accountID</code>
ゾーンまたはプロファイル（該当する場合）	<code>zone: zoneID, networkProfile: profileID, storageProfile: profileID</code>

vRealize Automation Cloud Assembly のタグの処理方法

vRealize Automation Cloud Assembly では、タグの表す機能と制約により、プロビジョニング プロセスでプロビジョニングされる展開にリソースを割り当てる方法と場所が決定されます。

vRealize Automation Cloud Assembly は、プロビジョニングされる展開を作成するためにタグを解決するときの操作について、特定の順序と階層を使用します。このプロセスの基本について理解しておくと、タグを効率よく実装して予測可能な展開を作成するのに役立ちます。

次のリストに、Cloud Assembly でタグの解決や展開の定義を行う際に使用される操作とシーケンスの概要を示します。

- クラウド ゾーンは、可用性とプロファイルを含め、いくつかの基準によってフィルタリングされます。このとき、ゾーンが属する地域のプロファイル内のタグが照合されます。
- 残りのクラウド ゾーンは、ゾーンおよびコンピューティング機能タグを使用して強い制約によってフィルタリングされます。
- フィルタリングされたゾーンから、優先順位を使用してクラウド ゾーンが選択されます。同じ優先順位を持つ複数のクラウド ゾーンがある場合は、クラウド ゾーンとコンピューティング機能の組み合わせを使用して、一致する弱い制約によってソートされます。
- クラウド ゾーンが選択された後、クラウド テンプレートに示されている強い制約および弱い制約を含む一連のフィルタを照合することにより、ホストが選択されます。

単純なタグ付け構造を設定する方法

このトピックでは、vRealize Automation Cloud Assembly での論理タグ付けストラテジに関する基本的なアプローチとオプションについて説明します。これらの例を実際の展開の開始点として使用することも、ニーズに合った別のストラテジを考案することもできます。

通常、タグを作成および維持するための主要な役割はクラウド管理者が担います。

このトピックでは、vRealize Automation Cloud Assembly のドキュメントの他の場所で説明されている WordPress のユースケースを参照して、いくつかのキー項目にタグを追加する方法について説明します。また、WordPress のユースケースで示されているタグ付けの例に関連して、その代替方法と応用についても説明します。

WordPress のユースケースの詳細については、[チュートリアル : vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。

WordPress のユースケースでは、クラウド ゾーンやストレージ プロファイルおよびネットワーク プロファイルにタグを配置する方法について説明します。これらのプロファイルは、リソースを体系化したパッケージのようなものです。プロファイルに配置されたタグは、そのプロファイル内のすべての項目に適用されます。タグを作成し、コンピューティング リソースのほか、ストレージ リソースや個別のネットワーク アイテムに配置することもできますが、そのようなタグは、配置された特定のリソースのみに適用されます。タグを設定する場合、通常はコンピューティング リソースへのタグ付けから始め、その後、プロファイルやクラウド ゾーンに追加することをお勧めします。また、これらのタグを使用して、クラウド ゾーン用のコンピューティング リソースのリストをフィルタリングできます。

たとえば、このユースケースに示されているようにストレージ プロファイルにタグを配置することもできますが、個別のストレージ ポリシー、データストア、ストレージ アカウントにタグを配置することもできます。これらのリソースにタグを配置すると、ストレージ リソースの展開方法を細かく制御できます。展開の準備のための処理中に、これらのタグはプロファイル タグの後に、次の処理レベルとして解決されます。

一般的なユーザーを設定するシナリオの例のように、ネットワーク プロファイルに `region: eastern` のタグを配置することもできます。このタグは、そのプロファイル内のすべてのリソースに適用されます。そのため、`networktype:pci` のタグはプロファイル内の pci ネットワーク リソースに配置することも考えられます。`eastern` および `pci` の制約があるクラウド テンプレートからは、この pci ネットワークを `eastern` 地域に使用する展開が作成されます。

手順

1 コンピューティング インフラストラクチャ リソースに対するタグ付けは、論理的で適切な方法で実行します。

特に重要なことは、[クラウド ゾーンの作成] 画面の [コンピューティング] タブで検索機能を使用して見つけることができるように、コンピューティング リソースに対するタグ付けを論理的にすることです。この検索機能を使用することで、クラウド ゾーンに関連付けられているコンピューティング リソースを素早くフィルタリングすることができます。ストレージとネットワークにプロファイル レベルでタグ付けすると、個別のストレージ リソースやネットワーク リソースに対するタグ付けが不要になる可能性があります。

- a vRealize Automation Cloud Assembly インスタンスに対してインポート済みのコンピューティング リソースを表示するには、[リソース] - [コンピューティング] の順に選択します。
- b 適切なコンピューティング リソースを選択し、[タグ] をクリックしてリソースにタグを追加します。必要に応じて、各リソースに複数のタグを追加することができます。
- c 対象となるストレージ リソースとネットワーク リソースについて、前の手順を繰り返します。

2 クラウド ゾーンとネットワーク プロファイルの機能タグを作成します。

各実装に応じて、クラウド ゾーンとネットワーク プロファイルの両方に同じタグを使用することも、それぞれのアイテムに一意のタグを作成することもできます。

ネットワーク プロファイルでは、プロファイル全体とそのサブネットに対してタグを配置できます。プロファイル レベルで適用されたタグは、サブネットなど、そのプロファイル内のすべてのコンポーネントに適用されます。サブネットに対するタグは、配置された特定のサブネットにのみ適用されます。タグの処理時には、プロファイル レベルのタグがサブネット レベルのタグよりも優先されます。

この例では、vRealize Automation Cloud Assembly のクラウド ゾーンおよびネットワーク プロファイル タグのユースケースに関するドキュメント全体で使用されている 3 つのシンプルなタグを作成します。これらのタグは、プロファイル コンポーネントの環境を識別します。

- `zone:test`
- `zone:dev`
- `zone:prod`

3 ストレージ コンポーネントのストレージ プロファイル タグを作成します。

通常、ストレージ タグは、`tier1`、`tier2` などのストレージ項目のパフォーマンス レベルを識別します。または、`pci` などのストレージ項目の特性を識別します。

ストレージ プロファイルに対するタグの追加については、[6. ストレージ プロファイルの追加](#)を参照してください。

- `usage:general`
- `usage:fast`

結果

タグ付けの基本的な構造を作成したら、それに対する作業を開始し、必要に応じてタグを追加または編集してタグ付け機能を強化したり拡張したりできます。

vRealize Automation でリソースを操作する方法

クラウド管理者は、データ収集によって公開されている vRealize Automation リソースを確認できます。

クラウド管理者は、リソースに機能タグをラベル付けして、vRealize Automation クラウド テンプレートが展開される場所を制御できます。

vRealize Automation のコンピューティング リソース

クラウド管理者は、データ収集によって公開されているコンピューティング リソースを確認できます。

クラウド管理者は、リソースに直接タグを適用するように選択し、vRealize Automation のプロビジョニングで照合する目的で機能にラベルを付けることができます。

vRealize Automation のネットワーク リソース

vRealize Automation では、クラウド管理者はプロジェクトにマッピングされているクラウド アカウントおよび統合からデータ収集されたネットワーク リソースを表示および編集できます。

クラウド アカウントを vRealize Automation Cloud Assembly インフラストラクチャに追加した後に、たとえば [インフラストラクチャ] - [接続] - [クラウド アカウント] メニュー シーケンスを使用すると、データ収集によってクラウド アカウントのネットワークとセキュリティの情報が検出されます。この情報は、ネットワーク、ネットワーク プロファイル、およびその他の定義で使用できるようになります。

ネットワークは、使用可能なネットワーク ドメインまたはトランスポート ゾーンの IP アドレス固有のコンポーネントです。Amazon Web Services または Microsoft Azure ユーザーの場合は、ネットワークをサブネットと考えます。

プロジェクト内のネットワークに関する情報を表示するには、[インフラストラクチャ] - [リソース] - [ネットワーク] 画面を使用します。

vRealize Automation Cloud Assembly[ネットワーク] 画面には、次のような情報が含まれています。

- クラウド アカウントのネットワーク ドメイン（たとえば、vCenter、NSX-V、Amazon Web Services）に外部から定義されるネットワークおよびロード バランサ。
- クラウド管理者によって展開されたネットワークとロード バランサ。
- クラウド管理者によって定義または変更された IP アドレス範囲やその他のネットワーク特性。

- プロバイダ固有の外部 IP アドレス管理統合の特定アドレス空間における、外部 IP アドレス管理プロバイダの IP アドレス範囲。

ネットワークの詳細については、次の情報を参照してください。[ネットワーク] ページのさまざまな設定に関する Signpost のヘルプと、[vRealize Automation でのネットワーク プロファイルの詳細](#)。

ネットワーク

ネットワークとその特性を表示および編集できます。たとえば、タグの追加や、パブリック IP アドレスへのアクセスのサポート解除です。DNS、CIDR、ゲートウェイ、タグ値などのネットワーク設定を管理することもできます。また、新しい IP アドレス範囲を定義できるほか、ネットワーク内の既存の IP アドレス範囲を管理することもできます。

既存のネットワークの場合、IP アドレス範囲とタグの設定を変更できます。そのためには、ネットワークのチェックボックスを選択し、[IP アドレス範囲の管理] または [タグ] を選択します。それ以外の場合、ネットワーク自体を選択してその情報を編集できます。

タグを使用すると、該当するネットワークをクラウド テンプレートのネットワーク コンポーネントと照合できるほか、オプションでネットワーク プロファイルも照合できます。ネットワーク タグは、対応するネットワークのあらゆるインスタンスに適用されます。そのネットワークがどのネットワーク プロファイルにあるかは関係ありません。ネットワークは、任意の数のネットワーク プロファイルにインスタンス化できます。ネットワーク タグは、ネットワーク プロファイルの有無とは関係なく、対応するネットワークに関連付けられます。そのネットワークがどこで使用されていてもかまいません。ネットワーク タグがクラウド テンプレートの他のコンポーネントと一致するためには、まずクラウド テンプレートが 1 つ以上のネットワーク プロファイルと一致する必要があります。

IP アドレス範囲

IP アドレス範囲を使用して、組織内の特定のネットワークの開始 IP アドレスと終了 IP アドレスを定義または変更します。リストされたネットワークの IP アドレス範囲を表示および管理できます。ネットワークが外部 IP アドレス管理プロバイダによって管理されている場合は、関連付けられた IP アドレス管理統合ポイントに関連する IP アドレス範囲を管理できます。

ネットワークに IP アドレス範囲を追加するには、[新規 IP アドレス範囲] をクリックします。[内部 IP アドレス範囲] を指定できます。また、有効な IP アドレス管理統合が使用可能な場合は、[外部 IP アドレス範囲] を指定できます。

デフォルト ゲートウェイを IP アドレス範囲に含めることはできません。サブネットの IP アドレス範囲にサブネットゲートウェイの値を含めることはできません。

特定の IP アドレス管理プロバイダに対して外部 IP アドレス管理統合を使用する場合は、[外部 IP アドレス範囲] を使用して、使用可能な外部 IP アドレス管理統合ポイントから IP アドレス範囲を選択することができます。このプロセスは、外部 IP アドレス管理の全体的な統合ワークフローのコンテキストで [vRealize Automation で既存ネットワークに外部 IP アドレス管理を使用するためのネットワークおよびネットワーク プロファイルの設定](#) に記述されています。

IP アドレス

組織で現在使用している IP アドレスを確認し、そのステータス (available や allocated など) を表示できます。表示される IP アドレスは、vRealize Automation によって内部で管理される IP アドレス、または外部 IP アドレス管理プロバイダの統合を含む展開用に指定された IP アドレスのいずれかです。外部 IP アドレス管理プロバイダは、独自の IP アドレス割り当てを管理します。

ネットワークが外部 IP アドレス管理プロバイダではなく vRealize Automation によって内部で管理されている場合は、IP アドレスを解放することもできます。

IP アドレスを使用していたマシンを削除した後などに、内部 IP アドレス管理を使用して IP アドレスを解放すると、IP アドレスが解放されてから再利用できるようになるまでに 30 分間の待機時間が生じます。待機時間中に、DNS キャッシュをクリアできます。その後、IP アドレスを新しいマシンに割り当てることができます。たとえば、以前に削除したマシンと同じ IP アドレスを使用してマシンをプロビジョニングできます。

ロード バランサ

組織内のアカウント/リージョン クラウド アカウントで使用可能なロード バランサに関する情報を管理できます。使用可能な各ロードバランサの構成された設定を開いて表示することができます。ロード バランサのタグを追加および削除することもできます。

ネットワーク ドメイン

ネットワーク ドメインには、重複していない、関連付けられているネットワークが表示されます。

vRealize Automation のセキュリティ リソース

vRealize Automation Cloud Assembly でクラウド アカウントを追加すると、データ収集によってクラウド アカウントのネットワークとセキュリティの情報が検出され、ネットワーク プロファイルやその他のオプションで使用するようになります。

セキュリティ グループとファイアウォール ルールでは、ネットワークの隔離がサポートされます。セキュリティ グループはデータ収集されます。ファイアウォール ルールはデータ収集されません。

セキュリティ グループ

[インフラストラクチャ] - [リソース] - [セキュリティ] メニュー シーケンスを使用して、vRealize Automation Cloud Assembly クラウド テンプレート デザインで作成されたオンデマンド セキュリティ グループおよびソース アプリケーション (NSX-T や Amazon Web Services など) で作成された既存のセキュリティ グループを表示できます。使用可能なセキュリティ グループは、データ収集プロセスによって公開されます。

使用可能なセキュリティ グループを表示し、選択したセキュリティ グループに対してタグを追加または削除できます。クラウド テンプレートの作成者は、マシン NIC に 1 つ以上のセキュリティ グループを割り当てて、展開のセキュリティを制御できます。

クラウド テンプレート デザインでは、セキュリティ グループ リソースの securityGroupType パラメータは、existing (既存のセキュリティ グループの場合) または new (オンデマンド セキュリティ グループの場合) として指定します。

NSX-V、NSX-T、または Amazon Web Services アプリケーションなど、基盤となるクラウド アカウント エンドポイントの既存のセキュリティ グループが使用可能です。組織のクラウド テンプレート デザインで作成されたオンデマンド セキュリティ グループもデータ収集されます。オンデマンド セキュリティ グループは、現在 NSX-V および NSX-T でのみ使用できます。

既存のセキュリティ グループが表示され、[発生源] 列で Discovered と分類されます。vRealize Automation Cloud Assembly でクラウド テンプレートまたはネットワーク プロファイル内に作成したオンデマンド セキュリティ グループは、[発生源] 列で Managed by Cloud Assembly として表示および分類されます。ネットワーク プロファイルの一部として作成したオンデマンド セキュリティ グループは、事前構成済みのファイアウォール ルールを持つ隔離セキュリティ グループとして内部で分類され、セキュリティ グループ リソースとしてクラウド テンプレート デザインに追加されません。クラウド テンプレート デザインで作成し、express ファイアウォール ルールを含めることができるオンデマンド セキュリティ グループは、new に分類されるセキュリティ グループ リソースの一部として追加されます。

vRealize Automation Cloud Assembly ではなく、ソース NSX アプリケーションなどのソース アプリケーションで既存のセキュリティ グループを直接編集すると、vRealize Automation Cloud Assembly 内から関連するクラウド アカウントまたは統合ポイントのデータ収集を実行するまで、更新は vRealize Automation Cloud Assembly に表示されません。データ収集は 10 分ごとに自動で実行されます。

クラウド管理者は、1 つ以上のタグを既存のセキュリティ グループに割り当て、クラウド テンプレートで使用するようにすることができます。クラウド テンプレートの作成者は、クラウド テンプレート デザイン内の Cloud.SecurityGroup リソースを使用して、タグ制約により既存のセキュリティ グループを割り当てることができます。既存のセキュリティ グループには、クラウド テンプレート デザインのセキュリティ リソースで少なくとも 1 つの制約タグを指定する必要があります。

セキュリティ グループでのファイアウォール ルールの使用

NSX-V および NSX-T のオンデマンド セキュリティ グループのファイアウォール ルールは、クラウド テンプレート デザイン コードのセキュリティ グループ リソースで直接作成できます。

[適用先] 列には、NSX 分散ファイアウォール (DFW) によって分類または管理されているセキュリティ グループは含まれていません。アプリケーションに適用されるファイアウォール ルールは、East/West DFW トラフィック用です。

一部のファイアウォール ルールは、ソース アプリケーションでのみ管理でき、vRealize Automation Cloud Assembly では編集できません。たとえば、イーサネット、緊急、インフラストラクチャ、および環境のルールは、NSX-T で管理されます。

詳細情報

ネットワーク プロファイルでのセキュリティ グループの使用に関する詳細については、[vRealize Automation のネットワーク プロファイルの詳細](#)を参照してください。

ファイアウォール ルールの定義については、[vRealize Automation Cloud Assembly のネットワーク プロファイルおよびクラウド テンプレート デザインでのセキュリティ グループ設定の使用](#)および [vRealize Automation クラウド テンプレートでのセキュリティ グループ リソースの使用](#)を参照してください。

セキュリティ グループを含むクラウド テンプレート デザイン コードのサンプルについては、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

vRealize Automation のストレージ リソース

クラウド管理者は、ストレージ リソースとその機能を使用できます。いずれも、関連するクラウド アカウントから vRealize Automation でデータ収集を行うと検出されます。

ストレージ リソース機能は、ソース クラウド アカウントで通常発生するタグを介して公開されます。クラウド管理者は、vRealize Automation Cloud Assembly を使用して、ストレージ リソースに追加のタグを直接適用することもできます。追加のタグに特定の機能のラベルを付けて、プロビジョニング時に照合することもできます。

vRealize Automation は、標準ディスクおよび First Class Disk の機能をサポートしています。First Class Disk は vSphere でのみ使用できます。

- [vRealize Automation の標準ディスク ストレージで実行できる操作](#)
- [vRealize Automation の First Class Disk ストレージで実行できること](#)

ストレージ リソースの機能は、vRealize Automation Cloud Assembly ストレージ プロファイルの定義の一部として表示されます。[vRealize Automation でのストレージ プロファイルの詳細](#) を参照してください。

データ収集された First Class Disk は、[ボリューム リソース] 画面に表示されます。[vRealize Automation のボリューム リソース](#) を参照してください。

vRealize Automation のマシン リソース

vRealize Automation では、すべてのユーザーがデータ収集によって公開されているマシン リソースを確認できます。

プロジェクト内のすべてのマシンが表示されます。自分のマシンだけをリストに表示することや、フィルタを指定して表示されるマシンを制御することができます。

プロジェクト内のクラウド アカウントに関連付けられている管理対象外のマシンも、管理対象マシンと同じく、このリストに表示されます。[発生元] 列には、マシンのステータスが示されます。

- 検出 - まだオンボーディングされていないマシン。
- 展開済み - オンボーディングされた、または vRealize Automation からプロビジョニングされたマシンであり、管理対象マシンと見なされます。

ワークロード オンボーディング プランを使用して、管理対象外のマシンを vRealize Automation の管理対象にすることができます。

切断されたマシン NIC はリストに表示されません。これは、vRealize Automation がイーサネット カードを列挙するためにネットワーク スイッチまたはサブネット情報を必要とするためです。たとえば、展開からマシン NIC を削除済みの場合、その NIC はリストに表示されません。

オンボーディング プランを使用して管理対象外のマシンを vRealize Automation の管理対象にする方法については、[vRealize Automation Cloud Assembly でのオンボーディング プランについて](#)を参照してください。

vRealize Automation のボリューム リソース

vRealize Automation では、すべてのユーザーがボリューム リソースを確認できます。

vRealize Automation Cloud Assembly には、次の 2 つのソースから生成されるボリュームまたは論理ドライブが表示されます。

- ソース クラウド アカウントのデータ収集によって検出されたボリューム
- vRealize Automation Cloud Assembly によってプロビジョニングされたワークロードに関連付けられたボリューム

ボリュームまたは論理ドライブに基づいて容量と機能を確認できます。このリストには、ソース クラウド アカウントで発生した機能タグや、vRealize Automation Cloud Assembly 自体に追加された機能タグも公開されます。First Class Disk としてのボリュームのステータスも示されます。First Class Disk ストレージ ボリュームの詳細については、[vRealize Automation の First Class Disk ストレージで実行できることを参照してください](#)。

vRealize Automation Cloud Assembly のリソースの詳細

vRealize Automation Cloud Assembly では、価格設定カードなどのデータ収集されたリソースに関する追加情報を公開できます。

vRealize Automation でのデータ収集の仕組み

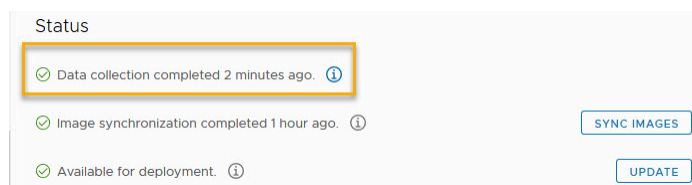
初期データ収集の後、リソース データ収集は 10 分ごとに自動的に実行されます。データ収集間隔は設定できません。また、データ収集を手動で開始することはできません。

既存のクラウド アカウントのリソース データ収集およびイメージ同期に関する情報は、その画面の [ステータス] セクションで確認できます。これを行うには、[インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、既存のクラウド アカウントの [開く] をクリックします。

既存のクラウド アカウントを開くと、関連付けられているエンドポイント バージョンを画面の [ステータス] セクションで確認できます。関連付けられているエンドポイントがアップグレード済みの場合は、データ収集時に新しいバージョンのエンドポイントが検出され、クラウド アカウントの画面の [ステータス] セクションに反映されます。

リソース データ収集

データ収集は 10 分ごとに実行されます。クラウド アカウントごとに、最新のデータ収集がいつ完了したか表示されます。

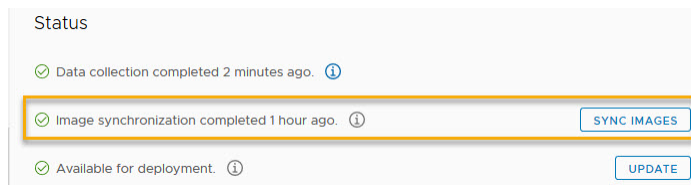


イメージ データ収集

イメージの同期は 24 時間ごとに実行されます。一部のクラウド アカウント タイプでは、イメージ同期を開始できます。イメージの同期を開始するには、クラウド アカウントを開き（[インフラストラクチャ] - [クラウド アカウント] の順に選択し、既存のクラウド アカウントを選択して開きます）、[イメージの同期] ボタンをクリックします。NSX クラウド アカウントのイメージ同期オプションはありません。

注： イメージは、内部でパブリックまたはプライベートのいずれかに分類されます。パブリック イメージは共有され、特定のクラウド サブスクリプションや組織に固有ではありません。プライベート イメージは共有されず、特定のサブスクリプションに固有です。パブリック イメージとプライベート イメージは 24 時間ごとに自動的に同期されます。[クラウド アカウント] 画面のオプションを使用すると、プライベート イメージの同期をトリガーできます。

[クラウド アカウント] 画面には、最新のイメージの同期がいつ完了したかが表示されます。



展開のフォルト トレランスと高可用性を簡素化するため、各 NSX-T データセンター エンドポイントは、3 つの NSX Manager を含むクラスタとなっています。関連情報については、[vRealize Automation での NSX-T クラウド アカウントの作成](#)を参照してください。

クラウド アカウントとオンボード プラン

クラウド アカウントを作成すると、アカウントに関連付けられたすべてのマシンでデータが収集され、[インフラストラクチャ] - [リソース] - [マシン] 画面に表示されます。クラウド アカウントに、vRealize Automation Cloud Assembly の外部に展開されたマシンがある場合は、オンボーディング プランを使用してマシンの展開を vRealize Automation Cloud Assembly で管理できます。

クラウド アカウントの追加については、[vRealize Automation Cloud Assembly へのクラウド アカウントの追加](#)を参照してください。

管理対象外のマシンのオンボードについては、[vRealize Automation Cloud Assembly でのオンボーディング プランについて](#)を参照してください。

vRealize Automation の標準ディスク ストレージで実行できる操作

標準ディスクは、パーシステントまたは非パーシステントにすることができます。

vRealize Automation では、標準ディスクと First Class Disk の 2 つのカテゴリのストレージがサポートされます。First Class Disk は vSphere でのみ使用できます。

■ vSphere

vSphere では、依存型（デフォルト）、独立型パーシステント、独立型非パーシステントの標準ディスクがサポートされます。関連情報については、[vRealize Automation のパーシステント ディスク ストレージで実行できる操作](#)を参照してください。

仮想マシンを削除すると、そのマシンの依存型および独立型非パーシステント ディスクも削除されます。

独立型パーシステント ディスクは、仮想マシンを削除しても削除されません。

依存型、および独立型非パースistent ディスクは、スナップショットを作成できます。独立型パースistent ディスクは、スナップショットを作成できません。

■ Amazon Web Services (AWS) EBS

EBS ボリュームを AWS コンピューティング インスタンスに接続すること、および AWS コンピューティング インスタンスから EBS ボリュームを接続解除することができます。

仮想マシンを削除すると、その仮想マシンに接続されている EBS ボリュームは接続解除されますが、削除はされません。

■ Microsoft Azure VHD

接続されたディスクは常にパースistent です。

仮想マシンを削除する場合は、接続されたストレージディスクを削除するかどうかを指定します。

■ Google Cloud Platform (GCP)

接続されたディスクは常にパースistent です。

パースistent ディスクは、仮想マシン インスタンスとは独立して配置されるため、パースistent ディスクを接続解除または移動することで、インスタンスを削除した後でもデータを保持することができます。

仮想マシンを削除すると、その仮想マシンに接続されているディスクは接続解除されますが、削除はされません。

関連情報については、[vRealize Automation でのストレージ プロファイルの詳細](#) を参照してください。

vRealize Automation の First Class Disk ストレージで実行できること

First Class Disk (FCD) により、仮想ディスクを Disk as-a-Service として、または EBS に似たディスク ストレージとしてストレージ ライフサイクルを管理できます。これにより、vSphere 仮想マシンとは別にディスクを作成して管理できます。

vRealize Automation は、標準ディスクと First Class Disk の 2 つのカテゴリのストレージ ディスクをサポートしています。First Class Disk 機能は vSphere でのみサポートされます。vRealize Automation では現在、First Class Disk 機能は API のみの機能として提供されています。

First Class Disk には、仮想マシンとは別に機能する独自のライフサイクル管理機能があります。First Class Disk が独立したパースistent ディスクと異なる点の 1 つは、First Class Disk を使用すると仮想マシンに依存せずにスナップショットを作成および管理できるということです。

新しい vRealize Automation ストレージ プロファイルを作成して、First Class Disk 機能と標準ディスク機能のどちらかをサポートすることができます。[vRealize Automation でのストレージ プロファイルの詳細](#) および [vRealize Automation のストレージ リソース](#) を参照してください。

vRealize Automation のクラウド テンプレートおよび展開に `Cloud.vSphere.Disk` の First Class Disk 要素を追加して、vSphere の First Class Disk をサポートすることもできます。データ収集された First Class Disk は、[ボリューム リソース] 画面に表示されます。[vRealize Automation のボリューム リソース](#) を参照してください。

vCenter では、First Class Disk は「IVD (Improved Virtual Disks)」または「管理対象仮想ディスク」とも呼ばれます。

機能

vRealize Automation API の機能を使用すると、次のことが可能です。

- First Class Disk の作成、リスト、削除
- First Class Disk のサイズ変更
- First Class Disk の接続/接続解除
- First Class Disk のスナップショット作成と管理
- 既存の標準ディスクから First Class Disk への変換

First Class Disk (FCD) 機能を使用するようにストレージ プロファイルを定義する方法など、vRealize Automation API を使用して First Class Disk (FCD) ストレージを作成および管理するための関連 API についての情報は、[code.vmware.com](https://code.vmware.com/docs/6516/what-are-the-vrealize-automation-cloud-apis-and-how-do-i-use-them) の [What are the vRealize Automation Cloud APIs and how do I use them](https://code.vmware.com/docs/6516/what-are-the-vrealize-automation-cloud-apis-and-how-do-i-use-them) にあります。

- FCD の API に関するドキュメントは、[Virtual Disk Development Kit Programming Guide](#) の [First Class Disk \(FCD\)](#) セクションにあります。
- vRealize Automation での FCD の API の使用事例に関するドキュメントへのリンクは、vRealize Automation リリースの [vRealize Automation API ドキュメント ページ](#) にあります。

考慮事項と制限事項

First Class Disk に関する考慮事項と制限事項は次のとおりです。

- First Class Disk は vSphere 仮想マシンでのみ使用できます。
- First Class Disk を使用するには、vSphere 6.7 Update 2 以降が必要です。
- データストア クラスタで First Class Disk のプロビジョニングはサポートされていません。
- First Class Disk では、ボリュームのマルチ接続はサポートされません。
- スナップショットが含まれている First Class Disk はサイズを変更できません。
- スナップショットが含まれている First Class Disk は削除できません。
- First Class Disk のスナップショット階層は、`createdAt` API オプションを使用してのみ構成できます。
- First Class Disk を接続する仮想マシンのハードウェアは、バージョンが vmx-13 (ESX 6.5 互換) 以降である必要があります。

vRealize Automation のパーシステント ディスク ストレージで実行できる操作

パーシステント ディスクを使用すると、重要なデータを過失による削除から保護できます。

クラウド テンプレートのボリュームの下に `persistent: true` プロパティを追加すると、ディスクは vRealize Automation Cloud Assembly または vRealize Automation Service Broker の削除の影響を受けなくなります。パーシステント ディスクは、展開の削除中または Day 2 の削除またはディスクの削除操作中に削除されません。

そのため、パーシステント ディスクは、展開の削除またはディスクの削除を行った後もインフラストラクチャに保持されます。これらを削除するには、次の方法を使用します。

- DELETE API を使用して、`per` フラグをクエリ パラメータとして明示的に渡します。

- クラウド エンドポイントから直接削除します。

これらを削除するための vRealize Automation Cloud Assembly または vRealize Automation Service Broker ユーザー インターフェイスはありません。

価格設定カードとは

vRealize Automation Cloud Assembly 価格設定カードは、クラウド管理者がリソースを管理する際に、展開ごとの金銭的影響に対して価格設定ポリシーを定義して割り当てするのに役立ちます。

価格設定カードの作成または割り当てを行う前に、vRealize Operations Manager の価格設定が vRealize Automation と連携するように構成して有効にする必要があります。vRealize Operations Manager と vRealize Automation の連携を構成するときは、必ず両方のアプリケーションを同じタイムゾーンに設定します。vRealize Operations でタイムゾーンを構成するには、SSH を有効にし、各 vRealize Operations Manager ノードにログインして、\$ALIVE_Base/user/conf/analytics/advanced.properties ファイルを編集し、timeZoneUseInMeteringCalculation = <time zone> を追加します。

注： マルチテナント環境で価格設定を有効にするには、vRealize Automation テナントごとに個別の vRealize Operations Manager インスタンスが必要です。

価格設定カードでは、価格設定ポリシーのレートを定義します。それにより、価格設定ポリシーを特定のプロジェクトに割り当てて、合計価格を定義することができます。vRealize Operations Manager エンドポイントを作成すると、事前定義されたデフォルトのレート カードが [インフラストラクチャ] - [価格設定カード] タブの価格設定と等しいコストで利用できるようになります。プロジェクトのみに適用される価格設定カードか、またはクラウドゾーンに適用される価格設定カードを作成できます。デフォルトでは、すべての新しい価格設定カードはプロジェクトに適用されます。

注： [すべての価格設定カードの適用先] の設定を変更すると、すべての既存の価格設定カードの割り当てが削除されます。また、vRealize Operations Manager エンドポイントが Cloud Assembly から削除された場合は、価格設定カードとその割り当てもすべて削除されます。

展開の価格が時系列で展開カードに表示されます。価格は毎月の初めにゼロにリセットされ、当日までの値が表示されます。展開の詳細にコンポーネント コストの内訳があります。この情報を展開レベルで提供すると、クラウド管理者にその情報が通知されるだけでなく、メンバーの作業が予算および長期的な開発にどのような影響を与えるのか、メンバー自身が理解できるようになります。

Cloud Assembly と Service Broker のユーザーに価格情報を表示するには、[価格情報を表示] ボタンを選択します。無効のままにすると、価格情報が Cloud Assembly および Service Broker ユーザーに表示されません。

価格の計算方法

コンピューティング リソースとストレージ リソースの展開レベルに表示される初期価格は、業界標準のベンチマーク レートに基づき、時系列で計算されたものです。レートがホストに適用され、サービスが CPU レートとメモリ レートを計算します。サーバは、24 時間ごとに価格を再計算します。

新しいポリシー、割り当て、および初期価格は、次に実行される vROps データ収集サイクルで価格設定されます。デフォルトでは、データ収集サイクルは 5 分ごとに実行されます。新しいポリシーまたは変更がプロジェクトと展開で更新されるまで、最長で 24 時間かかる場合があります。

また、vROps エンドポイント画面の [インフラストラクチャ] - [統合] - [vROps エンドポイント] - [] で、価格サーバを手動で更新することもできます。vCenter Server のセクションで、[同期] をクリックします。[同期] オプションを使用して価格サーバを手動で更新すると、組織内のすべてのプロジェクトに対して価格が再計算されます。組織のプロジェクトの数によっては、このプロセスに時間がかかる場合があります。

サポートされているリソースのリストについては、[vRealize Automation Cloud Assembly でのコストのかかるコンポーネント タイプのリスト](#)を参照してください。

vRealize Automation Cloud Assembly でのコストのかかるコンポーネント タイプのリスト

vRealize Automation Cloud Assembly では、以下のブループリント コンポーネント タイプに対するベンチマーク コスト情報を提供しています。

表 4-2. コスト算出コンポーネント タイプ

ブループリント コンポーネント タイプ	サービス名/オブジェクト タイプ	ブループリント リソース タイプ	コメント
クラウド非依存	マシン	Cloud.Machine	依存しないマシンが vSphere で構成されている場合は、展開コストを表示できます。
	ディスク	Cloud.Volume	依存しないディスクが vSphere で構成されている仮想マシンに接続されている場合は、展開コストを表示できます。
vSphere	vSphere マシン	Cloud.vSphere.Machine	クラウド固有のブループリントを使用して展開します。
	vSphere ディスク	Cloud.vSphere.Disk	仮想マシンに接続されたクラウド固有のブループリントを使用して展開します。
VMware 管理対象クラウド (VMC)	vSphere マシン	Cloud.vSphere.Machine	VMC でサポートされるのは、レートベースの価格カードのみです (コストベースの価格カードはサポートされません)。
	vSphere ディスク	Cloud.vSphere.Disk	

Cloud Assembly で価格設定カードを作成する方法

クラウド管理者が決定した価格設定戦略に従って価格設定カードを作成し、プロジェクトまたはクラウド ゾーンに割り当てることができます。

価格設定カードは、ユーザーが選択したパラメータに基づいてカスタマイズできます。価格設定カードを構成した後は、価格設定戦略によって決定される 1 つ以上のプロジェクトおよびクラウド ゾーンにこのカードを割り当てることができます。

前提条件

価格設定カードの作成または割り当てを行う前に、vRealize Operations で価格を設定して有効にし、通貨を設定して、vRealize Automation と連携させる必要があります。vRealize Operations と vRealize Automation の連携を設定するときは、必ず両方のアプリケーションを同じタイムゾーンに設定します。vRealize Operations でタイムゾーンを構成するには、SSH を有効にし、各 vRealize Operations ノードにログインして、`$ALIVE_Base/user/conf/analytics/advanced.properties` ファイルを編集し、`timeZoneUseInMeteringCalculation = <time zone>` を追加します。

価格設定カードを構成する前に、vRealize Operations エンドポイントを構成する必要があります。vRealize Operations エンドポイントを構成するには、[インフラストラクチャ] - [接続] - [統合] - [統合の追加] の順に移動します。

注： 複数の vRealize Operations エンドポイントを追加する場合、これらのエンドポイントで同じ vCenter Server を監視することはできません。

手順

- 1 [インフラストラクチャ] - [価格設定カード] - [新規価格設定カード] の順に移動します。
- 2 [サマリ] タブで価格設定カードの名前と説明を入力します。[価格設定] タブでポリシーを定義すると、概要テーブルに価格設定カードのレートが適用されます。

注： 通貨単位は、vRealize Operations で選択した値によって決まります。

- 3 オプション。価格設定カードが割り当てられていないすべてのプロジェクトに対して、デフォルトでこの価格設定カードを割り当てる場合は、[未割り当てプロジェクトにデフォルトを割り当てますか?] チェックボックスを選択します。

4 [価格設定] をクリックし、価格設定ポリシーの詳細を設定します。

表 4-3. 価格設定ポリシーの構成

パラメータ	説明
基本料金	<p>ポリシーの名前と説明を入力します。コスト ベースまたはレート ベースを選択します。</p> <ul style="list-style-type: none"> ■ コスト - コストは vRealize Operations で定義されます。選択した場合は、乗算係数が必要です。たとえば、係数として 1.1 を選択すると、コストに 1.1 が乗じられ、計算後のコストが 10% 増加します。コストを使用した価格計算式は、$\text{<コスト>} \times \text{<倍率>} = \text{価格}$ です。 ■ レート - 選択した場合は、絶対値を使用してコストを決定する必要があります。レートを使用した価格計算式は、$\text{<レート>} = \text{価格}$ です。このレートによる課金方法を指定するには、ドロップダウン リストからレートの間隔を選択します。 <p>CPU、メモリ、ストレージ、およびその他のコストは、[基本料金] セクションで、コストまたはレートを定義します。</p>
ゲスト OS	<p>ゲスト OS の料金を定義するには、[料金の追加] をクリックし、ゲスト OS 名を入力し、課金方式と基準レートを指定します。</p> <ul style="list-style-type: none"> ■ 繰り返し - 基準レートを入力し、料金期間として繰り返し間隔を定義します。絶対レート値が必要で、この値が全体価格に加算されます。 ■ 1 回限り - 1 回限りの基準レート料金を定義します。絶対値が必要で、この値が 1 回限りの価格として加算されます。 ■ レート係数 - 乗算係数が必要です。この係数は、選択した料金カテゴリに適用されます。たとえば、CPU 料金でレート係数 2 を選択した場合、ゲスト OS の CPU には、標準コストの 2 倍の料金が課金されます。 <p>レートが異なる複数のゲスト OS を追加する場合は、[料金の追加] をクリックし、追加の料金ポリシーを設定します。</p> <p>注： ゲスト OS の初期料金がポリシーに含まれていても、[サマリ] 画面には表示されません。</p>
タグ	<p>タグ料金を定義するには、[料金の追加] をクリックし、タグ名を選択し、課金方式と基準レートを指定します。</p> <ul style="list-style-type: none"> ■ 繰り返し - 基準レートを入力し、料金期間として繰り返し間隔を定義します。絶対レート値が必要で、この値が全体価格に加算されます。 ■ 1 回限り - 1 回限りの基準レート料金を定義します。絶対値が必要で、この値が 1 回限りの価格として加算されます。 ■ レート係数 - 乗算係数が必要です。この係数は、選択した料金カテゴリに適用されます。 <p>タグの課金方法は、パワーオン状態に基づいて選択します。</p> <p>レートが異なる複数のタグを追加する場合は、[料金の追加] をクリックして、追加料金ポリシーを構成します。</p> <p>注： 計算後の最終価格の追加料金には、仮想マシンのタグの料金は含まれますが、ディスクおよびネットワークのタグの料金は含まれません。</p>

表 4-3. 価格設定ポリシーの構成（続き）

パラメータ	説明
カスタム プロパティ	<p>カスタム プロパティ料金を定義するには、[料金の追加] をクリックし、</p> <p>プロパティの名前と値を入力し、課金方式と基準レートを指定します。</p> <ul style="list-style-type: none"> ■ 繰り返し - 基準レートを入力し、料金期間として繰り返し間隔を定義します。絶対レート値が必要で、この値が全体価格に加算されます。 ■ 1 回限り - 1 回限りの基準レート料金を定義します。絶対値が必要で、この値が 1 回限りの価格として加算されます。 ■ レート係数 - 乗算係数が必要です。この係数は、選択した料金カテゴリに適用されます。 <p>カスタム プロパティの課金方法は、パワーオン状態に基づいて選択します。</p> <p>レートが異なる複数のカスタム プロパティを追加する場合は、[料金の追加] をクリックして、追加料金ポリシーを構成します。</p>
全体料金	<p>価格設定ポリシーに追加したい料金があれば、その料金を定義します。1 回限りの料金と繰り返し料金の両方を追加できます。</p>

注： 1 回限りの料金は、カタログ アイテムの価格見積もりにも [サマリ] タブにも表示されません。該当するカタログ アイテムについては、日単位の価格見積もりのみが表示されます。

- 5 [割り当て] タブをクリックし、[プロジェクトの割り当て] をクリックします。価格設定カードを割り当てる 1 つ以上のプロジェクトを選択します。

注： デフォルトでは、価格設定カードはプロジェクトに適用されます。[インフラストラクチャ] - [価格設定カード] タブで、クラウド ゾーンに価格設定カードが適用されるように選択できます。クラウド ゾーンを選択した場合は、[割り当て] タブの [クラウド ゾーンの割り当て] をクリックします。

- 6 [作成] をクリックすると、保存されて価格設定ポリシーが作成されます。

結果

新規価格設定ポリシーが [価格設定カード] 画面に表示されます。ポリシーの詳細と構成を表示または編集するには、[開く] をクリックします。

展開の価格を見積もる方法

カタログ アイテムを展開する前に、展開の価格見積もりとして、初期価格を使用できます。

Daily Price Estimate



Guest OS and one time prices are excluded in this estimate.

	price-service-f309c00	\$0.54
	Cloud_vSphere_Machine_1	\$0.53
	Compute	\$0.39
	Storage	\$0.03
	Additional charges	\$0.11
	Cloud_vSphere_Disk_1	\$0.01
	Storage	\$0.01

CLOSE

初期価格の見積もりでは、仮想マシン 1 台あたりの起動ディスクのサイズは常に 8 GB です。

展開の初期価格は、展開前の特定のカタログ アイテムに関する、リソースの割り当てに基づいた毎日の価格見積もりです。カタログ アイテムの展開後に、今日までの月間価格を初期価格の集計として、[展開] タブと [インフラストラクチャ] - [プロジェクト] タブに表示できます。初期価格の計算は、vSphere マシン、vSphere ディスク、Cloud Assembly カatalog アイテム、および vCenter Server がプライベート クラウド用に構成されたクラウドに依存しないアイテムなどの、プライベート クラウド リソースでサポートされます。

注： パブリック クラウド リソースや、非 vSphere マシンまたはディスクなどのプライベート クラウド リソースでは、初期価格の計算はサポートされていません。

前提条件

vRealize Automation Cloud Assembly で価格を表示するには、vRealize Operations 統合エンドポイントで価格計算を有効にし、通貨を事前設定する必要があります。

手順

- 1 カタログからカタログ アイテムを選択し、[申請] をクリックします。

Daily Price Estimate
0.00

CALCULATE
DETAILS

- 2 カタログ アイテム申請の詳細を入力し、[計算] をクリックします。

Daily Price Estimate
\$0.54

UPDATE
DETAILS

- 3 (オプション) [日単位の価格見積もり] ウィンドウに価格の内訳を表示するには、[詳細] をクリックします。

次のステップ

毎日の価格見積もりの内容に問題がない場合は、[送信] をクリックして展開申請を続行します。

すべてのプロジェクトの価格を見積もる方法

クラウド管理者は、すべてのプロジェクトの合計価格を見積もることが必要になる場合があります。

ショーバックの目的で、プロジェクトの価格設定カードを使用して、すべてのプロジェクトの合計価格を見積もることができます。

手順

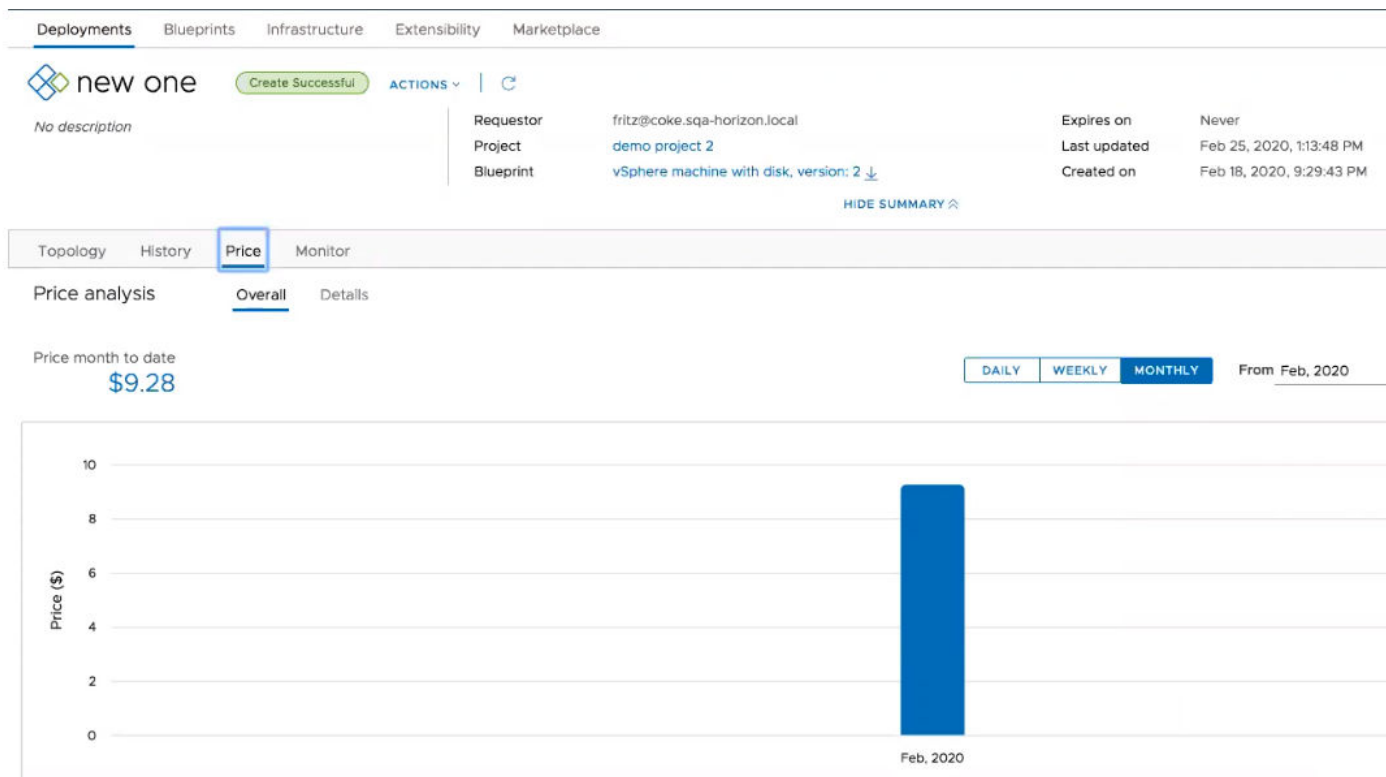
- 1 [インフラストラクチャ] - [価格設定カード] 画面で、[すべての価格設定カードの適用先] の横にある [編集] をクリックして、[プロジェクト] を選択します。

注： [すべての価格設定カードの適用先] の設定を変更すると、すべての既存の価格設定カードの割り当てが削除されます。

- 2 コストベースのアプローチを使用して、価格設定カードと割り当てを作成します。[Cloud Assembly で価格設定カードを作成する方法](#)を参照してください。

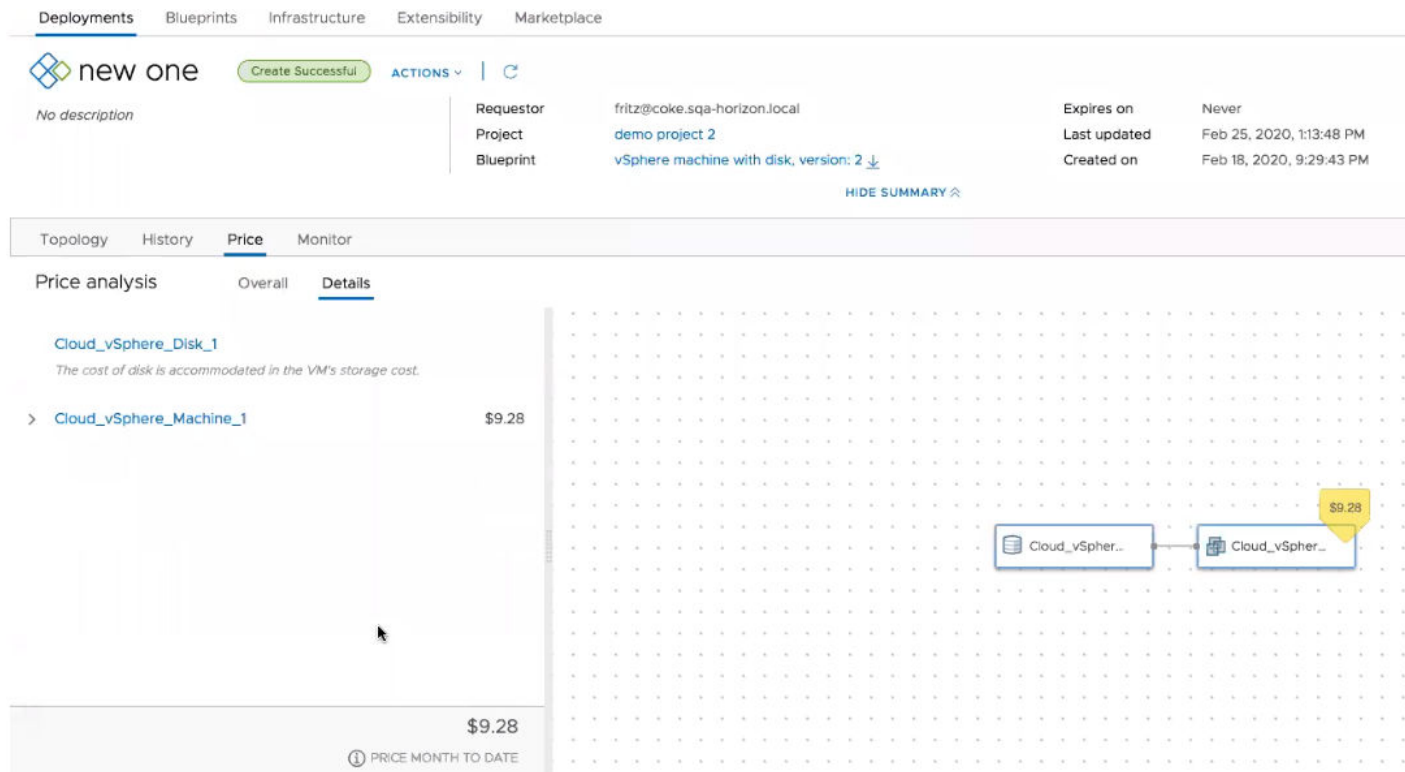
展開の価格履歴を表示する方法

価格カードを定義してプロジェクトに割り当てると、個々の展開の価格履歴を時系列で表示できるようになります。



価格履歴を表示するには、展開に移動して [価格] をクリックします。価格分析によって、展開の価格と過去 1 か月の価格の概要および詳細のビューが表示されます。グラフィカル表示を変更して、展開の価格を日単位、週単位、または月単位の値で表示できます。また、価格履歴の日付範囲または月を正確に指定することもできます。

コスト コンポーネントごとの価格内訳を表示するには、[詳細] をクリックします。



vRealize Automation を使用したマルチプロバイダ テナント リソースの構成

マルチテナント環境では、仮想プライベートゾーン (VPZ) を使用してテナントごとにリソースの割り当てを管理できます。

vRealize Automation 8.x では、VMware Lifecycle Manager と Workspace ONE Access を使用してマルチテナント環境を構成できます。これらのツールにより、ユーザーがマルチテナントを設定し、テナントを作成および構成できるようになります。テナントが構成されると、プロバイダ管理者は vRealize Automation Cloud Assembly 内に仮想プライベートゾーンを作成し、vRealize Automation Cloud Assembly のテナント管理機能を使用してゾーンをテナントに割り当てることができます。

マルチテナントは、次に示すように、3 つの異なる VMware 製品の調整と構成に依存します。

- Workspace ONE Access- この製品では、テナント組織内のユーザーおよびグループの管理を提供するマルチテナントおよび Active Directory ドメイン接続のインフラストラクチャ サポートを提供しています。
- vRealize Suite Lifecycle Manager- この製品では、vRealize Automation などのサポート対象製品のテナントの作成と構成がサポートされています。さらに、一部の証明書管理機能を提供します。
- vRealize Automation- プロバイダおよびユーザーは、vRealize Automation にログインして、展開を作成および管理するテナントにアクセスします。

マルチテナントを構成する場合、ユーザーはこれらの 3 つの製品のすべてと、関連するドキュメントに精通している必要があります。

Lifecycle Manager と Workspace ONE Access の使用方法の詳細については、[VMware Identity Manager を使用したユーザー管理](#)および[ユーザーとグループの管理](#)を参照してください。

vRealize Automation 用の仮想プライベート ゾーンの作成方法

プロバイダ管理者は、仮想プライベート ゾーン (VPZ) を作成して、インフラストラクチャ リソースをマルチ組織 vRealize Automation 環境内のテナントに割り当てることができます。管理者は、単一テナントの展開でのリソース割り当てを制御するために VPZ を使用することもできます。

VPZ を使用して、イメージ、ネットワーク、ストレージなどのリソースを割り当てることができます。これらはクラウド ゾーンとほぼ同じようにテナントごとに機能しますが、マルチテナント環境での使用を意図して設計されています。どのプロジェクトでもクラウド ゾーンまたは VPZ のいずれかを使用できますが、両方を使用することはできません。また、VPZ とテナントには 1 対 1 の関係があります。つまり、1 つの VPZ は一度に 1 つのテナントにのみ割り当てることができます。

NSX ありと NSX なしのどちらの VPZ も作成できます。NSX のないゾーンを作成する場合は、vSphere エンドポイントで NSX 関連の機能に関して制限があります。

- セキュリティ (グループ、ファイアウォール)
- ネットワーク コンポーネント (NAT)

前提条件

- VMware Lifecycle Manager と VMware Workspace ONE Access を使用して、vRealize Automation 展開でマルチテナントを有効にして構成します。
- テナントの構成の必要に応じて、テナント管理者を作成します。
- NSX を使用する場合は、適切な NSX クラウド アカウントをプロバイダ組織内に作成する必要があります。

手順

- 1 [インフラストラクチャ] - [構成] - [仮想プライベート ゾーン] の順に選択します。

VPZ 画面では、既存のすべてのゾーンが表示され、ゾーンを作成できます。

- 2 [新規仮想プライベート ゾーン] をクリックします。

画面の左側には 6 つの選択肢があり、これを使用してゾーンの概要情報とインフラストラクチャ コンポーネントを設定できます。

3 新規ゾーンの概要情報を入力します。

- a 名前と説明を追加します。
- b ゾーンが適用されるアカウントを選択します。
- c 配置ポリシーを選択します。

配置ポリシーにより、指定したクラウド ゾーンの展開環境内でホストを選択しやすくなります。

- デフォルト - コンピューティング リソースをクラスタおよびホスト全体にランダムに分散します。この選択肢は、個々のマシン単位で機能します。たとえば、ある特定の展開内のすべてのマシンが、要件を満たす使用可能なクラスタおよびホストにランダムに分散されます。
- binpack - 指定されたコンピューティングを実行するのに使用可能な十分なリソースのある、最も負荷の大きいホストにコンピューティング リソースを配置します。
- spread - 仮想マシンの数が最も少ないクラスタまたはホストに、展開コンピューティング リソースをプロビジョニングします。vSphere の場合、Distributed Resource Scheduler (DRS) が仮想マシンをホスト間で分散します。たとえば、ある展開環境のすべての要求されたマシンが同じクラスタに配置されても、次の展開では、その時点での負荷に応じて別の vSphere クラスタが選択されることがあります。

4 ゾーンのコンピューティング リソースを選択します。

クラウド ゾーンの必要に応じたコンピューティング リソースを追加します。初期状態で選択されているフィルタでは、すべてのコンピューティング リソースが含まれます。次のリストに示されているのが使用可能なすべてのコンピューティング リソースであり、これらは該当するゾーンに割り当てられます。クラウド ゾーンにコンピューティング リソースを追加する場合には、他に 2 つのオプションを選択できます。

- [コンピューティングの手動選択] - 下のリストからコンピューティング リソースを手動で選択する場合は、このメニュー項目を選択します。選択したら、[コンピューティングの追加] をクリックしてリソースをゾーンに追加します。
- [コンピューティングをタグによって動的に含める] - ゾーンに追加するコンピューティング リソースをタグに基づいて選択する場合は、このメニュー項目を選択します。適切なタグを追加する前の状態では、すべてのコンピューティング リソースが表示されています。このオプションでは、1 つまたは複数のタグを選択または入力できます。

どちらのコンピューティング オプションを選択した場合でも、表示されているコンピューティング リソースの右側のボックスを 1 つまたは複数選択して [削除] をクリックすることで削除できます。

5 必要に応じてタグを入力または選択します。

- 6 左側のメニューでフレーバーを選択し、ゾーンに対して 1 つ以上のフレーバーを定義します。フレーバーは、特定のクラウド アカウント/リージョンに対するターゲットの展開サイズを定義します。
- 7 左側のメニューでイメージを選択し、ゾーンに対して 1 つ以上のイメージを定義します。イメージは、ゾーンで使用する OS の仕様を定義するマシンテンプレートです。
- 8 左側のメニューでストレージを選択し、ゾーンに対してストレージ ポリシーとその他のストレージ構成を選択します。

- 9 左側のメニューで [ネットワーク] を選択し、ネットワークを定義します。さらに必要に応じて、このゾーンで使用するネットワーク ポリシーを定義します。また、選択したネットワーク ポリシーのロード バランサとセキュリティ グループを設定することもできます。

ネットワーク	<ul style="list-style-type: none"> ■ この VPZ に関連付けられているすべての既存のネットワークが [ネットワーク] タブのテーブルに表示されます。 ■ [ネットワークの追加] をクリックして、選択したリージョンに関連付けられているすべてのネットワークを表示します。このゾーンで使用するネットワークを追加します。 ■ ネットワークを選択し、[タグ] をクリックして、指定したネットワークに 1 つ以上のタグを追加します。 ■ [IP アドレス範囲の管理] を選択して、ユーザーがこのネットワークにアクセスする際に使用できる IP アドレス範囲を指定します。 ■ 必要に応じて、[ネットワーク ポリシー] タブをクリックし、隔離ポリシーを選択します。
ネットワーク ポリシー	<p>このゾーンで使用するネットワーク ポリシーが設定されている場合はそれを選択して、送信ネットワークとプライベート ネットワークに隔離ポリシーを適用します。</p> <ul style="list-style-type: none"> ■ 必要な場合は、隔離ポリシーを選択します。 ■ 必要な場合は、Tier-0 論理ルーターと Edge クラスタを選択します。
ロード バランサ	[ロード バランサの追加] をクリックして、アカウント/リージョン クラウド アカウントのロード バランサを構成します。
セキュリティ グループ	プロビジョニング済みのマシンにファイアウォール ルールを適用するためのセキュリティ グループを使用するには、[セキュリティ グループの追加] をクリックします。

結果

指定されたリソース割り当てを使用して、仮想プライベート ゾーンが作成されます。

次のステップ

クラウド管理者は、VPZ をプロジェクトに関連付けることができます。

- 1 Cloud Assembly で、[管理] - [プロジェクト] の順に選択します。
- 2 [プロビジョニング] タブを選択します。
- 3 [ゾーンの追加] をクリックして、[仮想プライベート ゾーンの追加] を選択します。
- 4 リストから目的の VPZ を選択します。
- 5 インスタンスの数、使用可能なメモリの量、使用可能な CPU の数について、プロビジョニングの優先順位と制限を設定できます。
- 6 [追加] をクリックします。

vRealize Automation テナントの VPZ 構成の管理

プロバイダ管理者は、vRealize Automation Cloud Assembly 内の仮想プライベート ゾーン (VPZ) を管理し、テナント単位でインフラストラクチャ リソースの割り当てを制御できます。管理者は、[テナント管理] 画面を使用して、テナントおよび VPZ ゾーンを確認したり、テナントの VPZ を有効または無効にしたりすることが可能です。

デフォルトでは、VPZ はテナントに割り当てられていません。テナントで使用するには、VPZ をこの画面で割り当てる必要があります。

初期作成時、VPZ はデフォルトで有効になります。有効になった VPZ は、指定されたテナントへの割り当ておよび使用が可能な状態です。VPZ を無効にすると、プロビジョニングでの使用や、テナントへの割り当てができなくなります。VPZ は無効にできますが、テナントへの割り当ては継続します。

プロバイダ管理者が [テナント管理] 画面に移動すると、管理者は、画面に表示された使用可能なすべてのテナントからいずれかを選択できます。テナントを選択すると、テナントに VPZ が割り当てられている場合は、現在の VPZ が画面に表示されます。管理者は、この画面を使用することで、選択したテナントに VPZ を割り当てることができます。

VPZ が割り当てられると、テナント管理者は VPZ をプロジェクトに追加することができます。これにより、テナント ユーザーによるプロビジョニングが可能になります。1 つのテナントに VPZ を割り当てると、別のテナントにも割り当てることができます。

VPZ を有効にすると、指定されたテナントで使えるようになります。プロバイダ管理者は、VPZ を無効にしてメンテナンスやテナントの再構成を簡素化することができます。また、ユーザーに無効化についての通知を送信することもできます。テナントの VPZ を永続的に使用できないようにするには、割り当てを解除します。何らかの理由で既存の VPZ がテナントから割り当て解除された場合は、そのテナントから展開を作成する際に使用することはできません。

前提条件

- マルチテナントを設定し、展開環境に応じて適切な VPZ を作成していること。

手順

- 1 vRealize Automation Cloud Assembly で、[テナント管理] を選択します。
[テナント管理] 画面には、管理者の組織用に構成されたすべてのテナントがカード ビューで表示されます。
- 2 テナントをクリックして選択します。
- 3 [インフラストラクチャ管理] タブをクリックして、テナントに割り当てられているすべての VPZ を表示します。
- 4 [仮想プライベート ゾーン] を選択して、現在テナントに割り当てられていないすべてのゾーンを示すダイアログを開きます。ゾーンをテナントに割り当てます。
- 5 ダイアログで 1 つ以上のゾーンを選択し、[テナントに割り当て] をクリックします。

次のステップ

VPZ の割り当て後、テナント管理者がプロジェクトに割り当てることができます。

プロバイダ管理者は、テナントのカード ビューを使用して VPZ のステータスを監視および管理できます。

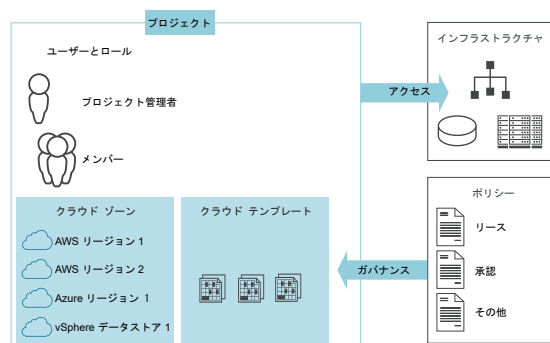
- テナントを無効にするには、テナントのカードで [無効化] をクリックします。
- テナントを有効にするには、テナントのカードで [有効化] をクリックします。
- テナントの割り当てを解除にするには、テナントのカードで [割り当て解除] をクリックします。

vRealize Automation Cloud Assembly プロジェクトの追加と管理

5

vRealize Automation Cloud Assembly のクラウド テンプレートにアクセスできるユーザーとテンプレートの展開先は、プロジェクトによって制御されます。プロジェクトを使用して、ユーザーが実行できる操作と、クラウド インフラストラクチャ内のクラウド テンプレートを展開できるクラウド ゾーンを編成および管理します。

クラウド管理者は、ユーザーとクラウド ゾーンを追加できるプロジェクトを設定します。クラウド テンプレートを作成および展開するすべてのユーザーは、1 つ以上のプロジェクトのメンバーである必要があります。



この章には、次のトピックが含まれています。

- [vRealize Automation Cloud Assembly 開発チームのプロジェクトを追加する方法](#)
- [vRealize Automation Cloud Assembly プロジェクトの詳細](#)

vRealize Automation Cloud Assembly 開発チームのプロジェクトを追加する方法

プロジェクトを作成し、そこにメンバーとクラウド ゾーンを追加すると、プロジェクト メンバーは関連付けられたゾーンにクラウド テンプレートを展開できるようになります。vRealize Automation Cloud Assembly 管理者が開発チームのプロジェクトを作成します。その後、プロジェクト管理者を割り当てるか、自身がプロジェクト管理者として運用します。

クラウド テンプレートを作成する場合は、まず、それを関連付けるプロジェクトを選択します。このプロジェクトは、クラウド テンプレートを作成する前に設定する必要があります。

プロジェクトが開発チームのビジネス ニーズをサポートしていることを確認します。

- プロジェクトは、チームの目標をサポートするリソースを提供しているか。インフラストラクチャ リソースとプロジェクトによるクラウド テンプレートのサポート方法の例については、[チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)を参照してください。
- プロジェクト メンバーは、展開が共有/プライベートにされることを要求または想定しているか。共有展開は、展開メンバーだけでなく、[展開] タブのすべてのプロジェクトメンバーが使用できます。展開の共有状態はいつでも変更できます。

展開をプロジェクト メンバーと共有する場合、メンバーは同じ Day 2 アクションを実行できます。Day 2 アクションを実行するメンバーの機能を管理するために、vRealize Automation Service Broker で Day 2 ポリシーを作成できます。ポリシーは vRealize Automation Cloud Assembly および vRealize Automation Service Broker の展開に適用されます。

Day 2 ポリシーの詳細については、[ポリシーを使用して展開ユーザーに Day 2 アクションの資格を付与する方法](#)を参照してください。

この手順は初期プロジェクトの作成に基づいており、基本的な設定のみが含まれます。開発チームがクラウド テンプレートを作成および展開するときに、プロジェクトを変更する場合があります。制約、カスタム プロパティ、およびその他のオプションを追加して、展開の効率を向上させることができます。[vRealize Automation Cloud Assembly プロジェクトの詳細](#)で利用可能な記事を参照してください。

前提条件

- クラウド ゾーンが設定されていることを確認します。[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。
- このプロジェクトのクラウド ゾーンとして含まれるリージョンについて、マッピングとプロファイルを設定していることを確認します。[4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド](#)を参照してください。
- このタスクを実行するために必要な権限があることを確認します。[vRealize Automation のユーザー ロールについて](#)を参照してください。
- プロジェクト管理者として指名するユーザーを決定します。プロジェクト管理者が vRealize Automation Cloud Assembly でできることを把握するには、[vRealize Automation のユーザー ロールについて](#)を参照してください。
- Active Directory グループをプロジェクトに追加する場合は、組織で Active Directory グループを設定していることを確認します。『vRealize Automation の管理』の[vRealize Automation でグループ ロールの割り当てを編集する方法](#)を参照してください。同期されていないグループはプロジェクトに追加できません。

手順

- 1 [インフラストラクチャ] - [管理] - [プロジェクト] の順に選択し、[新規プロジェクト] をクリックします。
- 2 プロジェクト名を入力します。

3 [ユーザー] タブをクリックします。

- a 申請したユーザーのみがプロジェクト メンバーによる展開にアクセスできるようにするには、[展開の共有] をオフにします。展開の所有権をプロジェクトの別のメンバーに確実に割り当てられるようにするには、[展開の共有] がオンになっていることを確認します。
- b 割り当てられたロールを持つユーザーを追加します。

4 [プロビジョニング] タブをクリックし、1つ以上のクラウド ゾーンを追加します。

プロジェクト ユーザーによって展開されるクラウド テンプレートをサポートするリソースが含まれる、任意のクラウド ゾーンと仮想プライベート ゾーンを追加します。

各ゾーンでは、ゾーンの優先順位を設定できます。また、プロジェクトで使用できるリソースの量を制限することもできます。制限できるものには、インスタンス、メモリ、および CPU の数があります。vSphere クラウド ゾーンの場合のみ、ストレージ制限を設定できます。

各ゾーンを追加して制限を適用する際は、メンバーがクラウド テンプレートを展開できなくなるほどプロジェクト リソースを制限しないでください。

ユーザーが展開の申請を送信すると、その展開をサポートするリソースを持つゾーンを判断するために、ゾーンが評価されます。複数のゾーンが展開をサポートする場合、優先度が評価され、優先度の値が高い（整数値が最も小さい）ゾーンにワークロードが配置されます。

5 [作成] をクリックします。

6 プロジェクトのクラウド ゾーンでプロジェクトをテストするには、[プロジェクト] 画面で [構成のテスト] をクリックします。

シミュレーションでは、プロジェクト クラウド ゾーンのリソースに対して、標準化された架空の展開テストが実行されます。失敗した場合は、詳細を確認し、リソース構成を修正します。

次のステップ

クラウド テンプレートの使用を開始します。6 章 [vRealize Automation Cloud Assembly 展開の設計](#) を参照してください。

vRealize Automation Cloud Assembly プロジェクトの詳細

プロジェクトは、クラウド テンプレートとリソース間のコネクタです。プロジェクトが機能する仕組みとその活用方法について理解が深まれば、vRealize Automation Cloud Assembly の開発および展開プロセスの効果も高まります。

vRealize Automation Cloud Assembly のプロジェクト タグとカスタム プロパティの使用

管理者は、プロジェクトの要件が vRealize Automation Cloud Assembly クラウド テンプレートと異なる場合、プロジェクト レベルのガバナンス制約またはカスタム プロパティを追加できます。制約タグに加えて、プロビジョニング プロセスによって展開されるリソースに追加されるリソース タグを指定することにより、リソースを管理できます。

プロジェクト リソース タグについて

プロジェクト リソース タグは、標準化された識別タグとして機能し、展開されたリソースの管理やコンプライアンスの徹底に使用できます。

プロジェクトで定義されたリソース タグは、そのプロジェクトの一部として展開されたすべてのコンポーネント リソースに追加されます。その後、標準的なタグ付けにより、他のアプリケーションを使用してリソースを管理できます。

たとえば、クラウド管理者が CloudHealth などのアプリケーションを使用してコストを管理する場合を考えます。欧州連合の人事ツールを開発するための専用プロジェクトに、`costCenter:eu-cc-1234` というタグを追加します。プロジェクト チームがこのプロジェクトから展開すると、展開されたリソースにはこのタグが追加されます。次に、このタグを含むリソースを識別して管理するようにコスト計算ツールを構成します。他のコスト センターに関連する他のプロジェクトでは、キーと共に別の値を使用することが考えられます。

プロジェクトの制約タグについて

プロジェクトの制約は、ガバナンス定義として運用します。これは、展開申請がプロジェクトのクラウド ゾーンで使用または回避するリソースを定義する `key:value` タグです。

展開プロセスでは、プロジェクトの制約に一致するネットワークとストレージのタグを検索し、一致するタグに基づいて展開します。

拡張性の制約は、拡張性ワークフローで使用する vRealize Orchestrator 統合インスタンスを指定するために使用されます。

プロジェクトの制約を設定するときは、次の形式を検討してください。

- `[key:value]` および `[key:value:hard]`。クラウド テンプレートを一致する機能タグが設定されたリソースでプロビジョニングする必要がある場合は、このタグを上記のいずれかの形式で使用します。一致するタグがないと、展開プロセスは失敗します。たとえば、プロジェクトのメンバーによって展開されたクラウド テンプレートは、PCI 準拠のネットワーク上でプロビジョニングする必要があります。`security:pci` を使用します。プロジェクトのクラウド ゾーンにネットワークがない場合、展開は失敗し、安全でない展開は確保されません。
- `[key:value:soft]`。このタグは、一致するリソースが必要であると同時に展開プロセスを失敗せずに続行し、タグが一致しないリソースを受け入れることができるようにする場合に使用します。たとえば、プロジェクト メンバーに対して、より安価なストレージにクラウド テンプレートを展開させながら、ストレージの可用性が展開の妨げにならないようにする場合を想定します。`tier:silver:soft` を使用します。プロジェクトのクラウド ゾーンに `tier:silver` でタグ付けされたストレージがない場合でも、クラウド テンプレートは他のストレージ リソースに展開されます。
- `[!key:value]`。このタグは、一致するタグを使用したリソースへの展開を回避する場合に、ハードまたはソフトで使用します。

重要なのは、プロジェクトの制約タグの優先順位はクラウド テンプレートの制約タグよりも高く、展開時にこのタグをオーバーライドすることです。この状況にならないクラウド テンプレートを使用している場合は、テンプレートで `failOnConstraintMergeConflict:true` を使用できます。たとえば、プロジェクトに `loc:london` というネットワーク制約があり、クラウド テンプレートは `loc:mumbai` が含まれているときに、プロジェクトの場所を優先するのではなく、制約が競合するというメッセージを表示して展開が失敗するように設定する場合は、次のサンプルのようなプロパティを追加します。

```
constraints:
  - tag: 'loc:mumbai'
failOnConstraintMergeConflict:true
```

プロジェクトのカスタム プロパティを使用する方法

プロジェクトのカスタム プロパティは、レポート作成、拡張性アクションおよびワークフローのトリガと設定、およびクラウド テンプレート レベルのプロパティのオーバーライドに使用できます。

カスタム プロパティを展開に追加すると、ユーザー インターフェイスの値を使用するか、API を使用してその値を取得することで、レポートを生成できるようになります。

また、拡張性により、拡張性サブスクリプションにカスタム プロパティも使用することができます。

クラウド テンプレートには、プロジェクトで変更が必要な特定のプロパティ値が設定されている場合があります。カスタム プロパティとして、代替の名前と値を指定できます。

展開時の vRealize Automation Cloud Assembly プロジェクトの動作

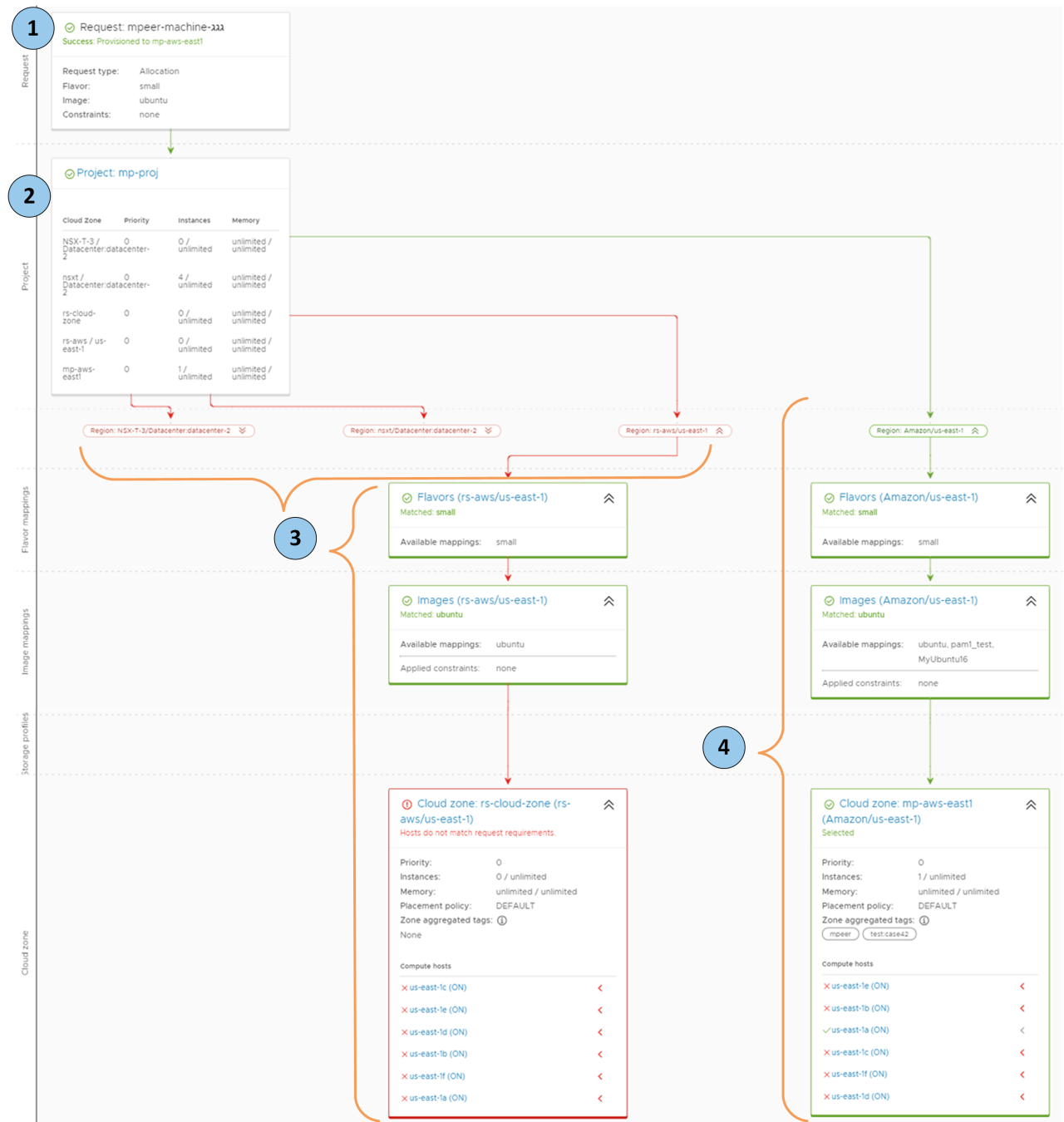
プロジェクトは、クラウド ゾーンへのユーザー アクセスと、プロビジョニングされたリソースのユーザー所有権を制御するものです。クラウド管理者もクラウド テンプレート開発者も、展開時にプロジェクトがどのように機能するかを理解しておく必要があります。これにより、展開を管理し、問題のトラブルシューティングを行えます。

クラウド管理者としてさまざまなチームのプロジェクトを設定するには、プロジェクトがクラウド テンプレート コンポーネントの展開場所をどのように決定するかを理解する必要があります。これを理解すると、クラウド テンプレート開発者をサポートするプロジェクトを作成し、失敗した展開をトラブルシューティングするのに役立ちます。

クラウド テンプレートを作成する場合は、まずプロジェクトに関連付けます。展開時に、クラウド テンプレートの要件がプロジェクト クラウド ゾーンに対して評価されて、最適な展開場所が決定されます。

次のワークフローは、このプロセスを示しています。

- 1 クラウド テンプレートの展開申請を送信します。
- 2 プロジェクトでは、フレーバー、イメージ、制約タグなどのテンプレート要件およびプロジェクト要件が評価されます。要件がプロジェクトのクラウド ゾーンと比較されて、要件をサポートするゾーンが特定されます。
- 3 そのゾーンには、申請をサポートするリソースがありませんでした。
- 4 このクラウド ゾーンが申請の要件をサポートしているため、テンプレートはこのクラウド ゾーンのアカунトリージョンに展開されます。



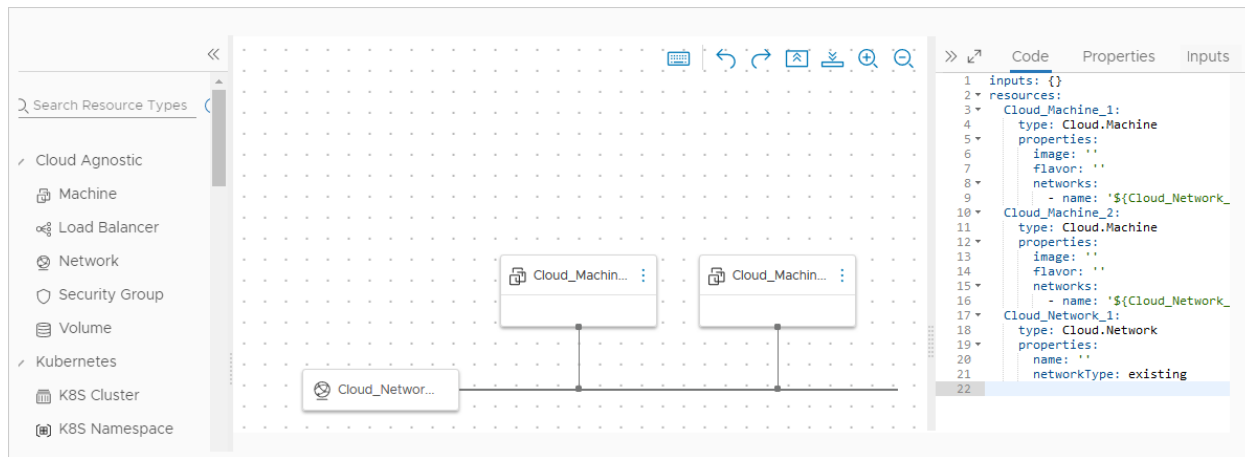
vRealize Automation Cloud Assembly 展開の設計

6

展開の基礎はクラウド テンプレート（旧称はブループリント）です。クラウド テンプレートとは、クラウド リソースに作成されるマシン、アプリケーション、およびサービスを、vRealize Automation Cloud Assembly を基準にして定義する仕様です。

クラウド テンプレート開発者は、特定のクラウド ベンダーを対象としたテンプレートを設計したり、テンプレートがクラウドに依存しないようにしたりできます。プロジェクトに割り当てられているクラウド ゾーンによって、どの方法を実行するかが決まります。クラウド ゾーンを構成するリソースの種類を理解していることを確認するには、クラウド管理者にお問い合わせください。

vRealize Automation Cloud Assembly テンプレートの作成は、infrastructure-as-code プロセスであることに注意してください。デザイン キャンパスでリソースを追加し、接続して、開始することができます。次に、キャンパスの右側にあるコード エディタを使用して詳細を設定します。コード エディタを使用すると、コードを直接入力することも、プロパティ値をフォームに入力することもできます。



クラウド テンプレートを作成する前に

vRealize Automation Cloud Assembly テンプレートはいつでも作成できますが、導入するには、最初にクラウド リソース インフラストラクチャを定義する必要があります。

■ 4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャのビルド

また、これらのインフラストラクチャ リソースをクラウド ゾーンとして含む vRealize Automation Cloud Assembly プロジェクトを作成する必要があります。

■ 5 章 vRealize Automation Cloud Assembly プロジェクトの追加と管理

この章には、次のトピックが含まれています。

- [クラウド テンプレートを作成する方法](#)
- [シンプルな vRealize Automation Cloud Assembly テンプレートをゼロから作成する方法](#)
- [シンプルな vRealize Automation Cloud Assembly テンプレートを拡張する方法](#)
- [vRealize Automation Cloud Assembly 設計に高度な機能を追加する方法](#)
- [vRealize Automation リソースのプロパティについて](#)
- [vRealize Automation Cloud Assembly コードの例](#)
- [vRealize Automation Cloud Assembly に Terraform 構成を含める方法](#)
- [vRealize Automation Cloud Assembly マーケットプレイスの使用方法](#)

クラウド テンプレートを作成する方法

vRealize Automation Cloud Assembly では、クラウド テンプレートを作成してコードとして保存することができ、クラウド テンプレートのデザインと再利用を簡単に実行できます。

クラウド テンプレートは、空のキャンバスからビルドすることも、既存のコードを利用することもできます。

vRealize Automation Cloud Assembly デザイン画面

クラウド テンプレートを最初から作成するには、[デザイン] - [クラウド テンプレート] の順に移動して、[新規作成元] - [空白のキャンバス] の順にクリックします。リソースをキャンバスにドラッグして接続し、コード エディタで構成を完了します。

コード エディタを使用すると、コードの入力、切り取り、コピー、貼り付けを直接行うことができます。コードの編集に慣れていない場合は、デザイン キャンバスでリソースを選択し、コード エディタの [プロパティ] タブをクリックして値を入力します。入力したプロパティ値は、コードに直接入力した場合と同じように表示されます。

The screenshot displays the vRealize Automation Cloud Assembly interface. On the left, a code editor shows a YAML configuration for a 'WebTier' resource. On the right, the 'Properties' tab is active, showing various configuration fields for the resource.

```

WebTier:
  type: Cloud.Machine
  properties:
    name: wordpress
    flavor: '${input.size}'
    image: ubuntu
    count: '${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
  
```

The Properties panel on the right includes the following sections:

- Count:** Set to "\${input.count}" with edit and info icons.
- Image Type:** Set to "ubuntu" with delete, edit, and info icons.
- Flavor *:** Set to "\${input.size}" with delete, edit, and info icons.
- Storage:** A section for configuring storage.
- Constraints:** A section for adding constraints, currently showing a "Tag" constraint with a dropdown menu and a list of 1-10 items.
- Maximum Capacity of the disk in GB:** Set to "1" with edit and info icons.
- Size of boot disk in GB:** Set to "1" with edit and info icons.
- Networks:** A section for adding networks, currently showing a "+" button and an info icon.

また、クラウド テンプレートから別のクラウド テンプレートにコードをコピー アンド ペーストできます。

クラウド テンプレートのクローン作成

テンプレートのクローンを作成するには、[デザイン] に移動し、ソースを選択して [クローン作成] をクリックします。クラウド テンプレートのクローンを作成して、ソースに基づくコピーを作成し、そのクローンを新しいプロジェクトに割り当てるか、新しいアプリケーションのスタート コードとして使用します。

アップロードとダウンロード

vRealize Automation Cloud Assembly マーケットプレイスでは、すぐに作業を開始できるように、完成したクラウド テンプレートが提供されています。[vRealize Automation Cloud Assembly マーケットプレイスの使用方法](#) を参照してください。

さらに、サイトに適した任意の方法でクラウド テンプレート YAML コードをアップロード、ダウンロード、および共有できます。外部のエディタと開発環境を使用して、テンプレート コードを変更することもできます。

注： 共有テンプレート コードを検証するには、デザイン画面で vRealize Automation Cloud Assembly コードエディタを使用することをお勧めします。

Cloud Templates 22 items						
NEW FROM SYNC REPOS CLONE DEPLOY DOWNLOAD DELETE Filter...						
<input type="checkbox"/>	Name	Description	Source Control	Source Control - Last Sync	Project	Last Updated
<input checked="" type="checkbox"/>	ESFSE				65-Project	Aug 31, 2020, 4:41:52 PM
<input type="checkbox"/>	demo-clone		demo-01/admin-templat...	✓ New draft, version(s) ci	62-Project	Aug 31, 2020, 4:39:47 PM
<input type="checkbox"/>	aws-with-network		demo-01/admin-templat...	✓ New draft, version(s) ci	62-Project	Aug 30, 2020, 5:01:59 PM
<input type="checkbox"/>	test1		demo-01/admin-templat...	✓ New draft, version(s) ci	62-Project	Aug 28, 2020, 3:38:19 PM
<input type="checkbox"/>	test2		demo-01/admin-templat...	✓ New draft, version(s) ci	62-Project	Aug 28, 2020, 3:14:57 PM
<input type="checkbox"/>	test3		demo-01/admin-templat...	✓ New draft, version(s) ci	62-Project	Aug 28, 2020, 1:35:22 PM

シンプルな vRealize Automation Cloud Assembly テンプレートをゼロから作成する方法

デザイン画面を使用して、プロビジョニングするマシンまたはアプリケーションの vRealize Automation Cloud Assembly クラウド テンプレート仕様を作成します。

- 1 リソースを見つけます。
- 2 リソースをキャンバスにドラッグします。
- 3 リソースを接続します。
- 4 クラウド テンプレート コードを編集して、リソースを構成します。

The screenshot displays the vRealize Automation Cloud Assembly design interface. On the left, a sidebar lists resource categories: Cloud Agnostic (Machine, Load Balancer, Network, Security Group, Volume) and Kubernetes (K8S Cluster, K8S Namespace). The main canvas shows a diagram with three resources: Cloud_Machine_1, Cloud_Machine_2, and Cloud_Network_1. Cloud_Machine_1 and Cloud_Machine_2 are connected to Cloud_Network_1. On the right, the 'Code' tab is active, showing the CloudFormation-style template code. The code defines inputs, resources, and their properties, including network configurations and machine specifications.

```

1 inputs: {}
2 resources:
3   Cloud_Machine_1:
4     type: Cloud.Machine
5     properties:
6       image: ''
7       flavor: ''
8       networks:
9         - name: '${Cloud_Network_1.name}'
10  Cloud_Machine_2:
11    type: Cloud.Machine
12    properties:
13      image: ''
14      flavor: ''
15      networks:
16        - name: '${Cloud_Network_1.name}'
17  Cloud_Network_1:
18    type: Cloud.Network
19    properties:
20      name: ''
21      networkType: existing
22

```

デザイン画面では、クラウド テンプレート名の変更、バージョン管理やバージョンの変更、またはテンプレートのクローン作成や展開を行うこともできます。

vRealize Automation Cloud Assembly リソースを選択してクラウド テンプレートに追加する方法

vRealize Automation Cloud Assembly リソースはクラウド テンプレートのビルディング ブロックです。デザイン画面では、クラウドに依存しないリソースやクラウド ベンダー固有のリソースを使用できます。

リソースは、デザイン画面の左側に選択可能な状態で表示されます。

クラウドに依存しないリソース

クラウドに依存しないリソースを任意のクラウド ベンダーに展開できます。プロビジョニング時は、展開に一致するクラウド固有のリソースを使用します。たとえば、クラウド テンプレートを AWS と vSphere の両方のクラウド ゾーンに展開する場合は、クラウドに依存しないリソースを使用します。

クラウド ベンダーのリソース

Amazon Web Services、Microsoft Azure、Google Cloud Platform、VMware vSphere などに固有のベンダー リソースは、一致する AWS、Azure、GCP、または vSphere クラウド ゾーンにのみ展開できます。

クラウドに依存しないリソースは、特定のベンダーのクラウド固有のリソースを含むクラウド テンプレートに追加できます。ベンダーに関しては、プロジェクトのクラウド ゾーンでサポートされる内容について理解しておきます。

構成管理リソース

構成管理リソースは、統合されたアプリケーションに依存します。たとえば、Puppet リソースでは、他のリソースの構成を監視し、適用することができます。

vRealize Automation Cloud Assembly でクラウド テンプレート リソースを接続する方法

vRealize Automation Cloud Assembly グラフィカル デザイン キャンバスを使用して、クラウド テンプレート リソースを接続します。

接続と互換性がある場合は、リソースを接続できます。例：

- ロード バランサをマシンのクラスタに接続します。
- マシンをネットワークに接続します。
- 外部ストレージをマシンに接続します。

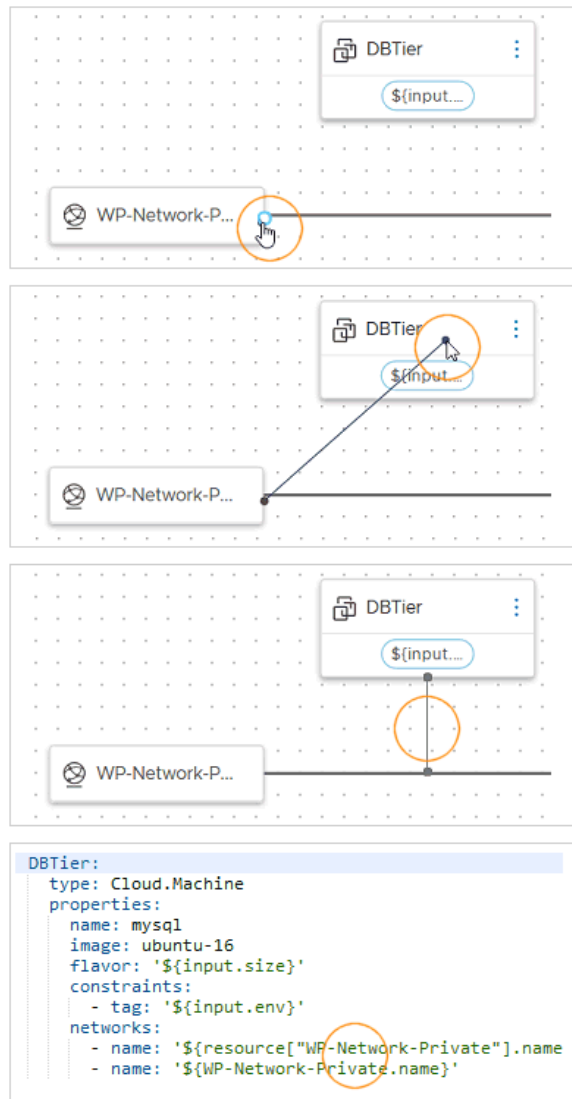
重要： 実線コネクタでは、2 つのリソースが同じクラウド ゾーンに展開されている必要があります。競合する制約をリソースに追加すると、展開が失敗することがあります。

たとえば、一方のリソースを us-west-1 のゾーンに、もう一方のリソースを us-east-1 のゾーンに配置する制約タグがある場合、接続されたリソースを展開することはできません。

実線または点線の矢印は、依存関係のみを示しています。接続されていることを示すものではありません。依存関係の詳細については、[vRealize Automation Cloud Assembly でリソースの展開順序を設定する方法](#)を参照してください。

接続するには、リソースのエッジの上にマウスを移動して、接続バブルを表示します。次に、バブルをクリックして、ターゲット リソースにドラッグしてから離します。

コード エディタで、ソース リソースの追加コードがターゲット リソース コードに表示されます。



この図では、SQL マシンとプライベート ネットワークが接続されているため、これらを同じクラウド ゾーンに展開する必要があります。

vRealize Automation Cloud Assembly で有効なクラウド テンプレート コードを作成する方法

キャンバスで vRealize Automation Cloud Assembly リソースを追加して接続すると、スタータ コードのみが作成されます。コンポーネントの詳細を設定するには、コードを編集します。

コード エディタを使用すると、コードを直接入力することも、プロパティ値をフォームに入力することもできます。コードを直接作成しやすくするために、vRealize Automation Cloud Assembly エディタには、構文補完機能とエラー チェック機能が搭載されています。

エディタのヒント

例

使用可能な値

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15     networks:
16       - name: small flavor
17       - name: large flavor
18 Cloud_Network_1:
19   type: Cloud.Network
20   properties:
21     name: ''
22     networkType: existing

```

使用できるプロパティ

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15     tags: array
16     storage: object
17     remoteAccess: object
18     name: string
19     imageRef: string
20     count: integer
21     constraints: array
22     cloudConfig: string

```

子プロパティ

```

10 Cloud_Machine_2:
11   type: Cloud.Machine
12   properties:
13     image: ''
14     flavor: ''
15     constraints:
16       - tag: string
17 Cloud_Network_1:
18   type: Cloud.Network

```

構文エラー

⚠ 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

```

Ctrl + F キーによる検索

```

1 inputs: {}
2 resources:
3   Cloud_Machine_2:
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}'

```

エディタのヒント

例

オプション
パラメータ

オプション パラメータの挿入

- + attachedDisks
- + autoScaleConfiguration
- + cloudConfig
- + cloudConfigSettings

```

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}'

```

スキーマヘルプ

すべてのカスタム プロパティで、vRealize Automation Resource Type Schema on VMware {code} を参照することもできます。

cloudConfig

タイプ

string

When provisioning an instance, machine cloud-init startup instructions from user data fields. Sample cloud config instructions:

```

#cloud-config
repo_update: true
repo_upgrade: all
packages:
- httpd
- mariadb-server

runcmd:
- [ sh, -c, "amazon-linux-extras install -y
- systemctl start httpd
- sudo systemctl enable httpd

```

```

Tier:
type: Cloud.Machine
properties:
  name: mysql
  image: ubuntu-16
  flavor: '${input.size}'
  constraints:
    - tag: '${input.env}'
  networks:
    - name: '${resource["WP-Network-Private"]}'
    - name: '${WP-Network-Private.name}'
  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
      - service mysql restart
      - mysql -e "GRANT ALL PRIVILEGES ON *.
      - mysql -e "FLUSH PRIVILEGES;"
  attachedDisks: []
bTier:
type: Cloud.Machine

```

vRealize Automation Cloud Assembly を使用して異なるバージョンを保存する方法

正常に機能している設計に変更を加えることにはリスクが伴うため、クラウド テンプレート開発者は、スナップショットを安全のために取得することができます。

展開時には、どのバージョンを展開するか選択できます。

クラウド テンプレート バージョンの取得

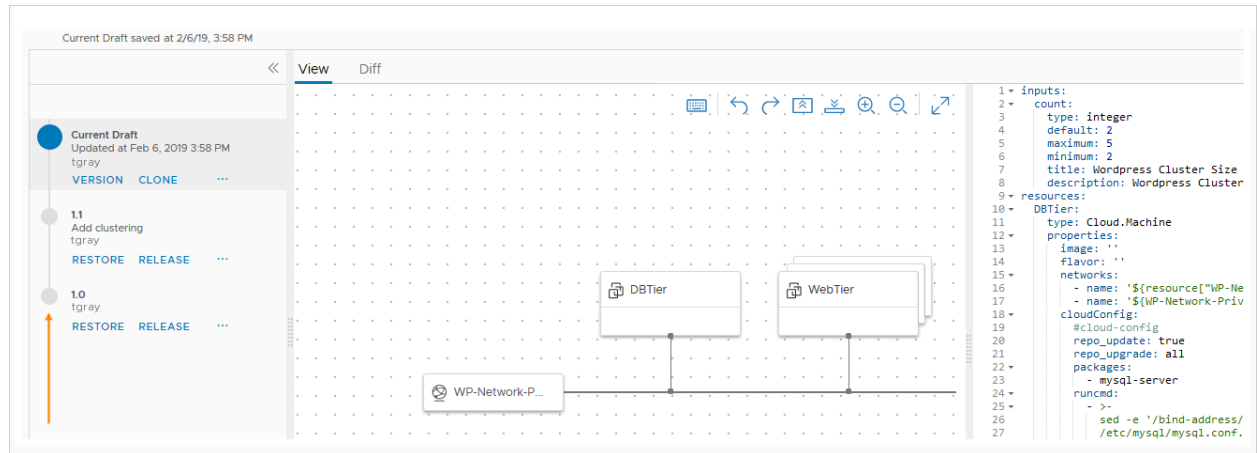
デザイン画面で、[バージョン] をクリックし、名前を入力します。

名前には、スペース以外の英数字と、特殊文字としてピリオド、ハイフン、アンダースコアのみを使用できます。

古いバージョンのリストア

デザイン画面で、[バージョン履歴] をクリックします。

左側で、以前のバージョンを選択して、キャンバスとコード エディタで検査します。目的のバージョンを見つけたら、[リストア] をクリックします。リストアすると、現在のドラフトが上書きされますが、名前の付けられたバージョンは削除されません。



vRealize Automation Service Broker へのバージョンのリリース

デザイン画面で、[バージョン履歴] をクリックします。

左側でバージョンを選択して、リリースします。

現在のドラフトは、バージョンを指定するまでリリースできません。

vRealize Automation Service Broker へのバージョンの再インポート

カタログ ユーザーに対して新しいバージョンを有効にするには、そのバージョンを再インポートします。

vRealize Automation Service Broker で、[コンテンツとポリシー] - [コンテンツ ソース] の順に選択します。

ソースのリストで、新しくリリースされたバージョンのクラウド テンプレートを含むプロジェクトのソースをクリックします。

[保存とインポート] をクリックします。

クラウド テンプレートのバージョンの比較

変更とバージョンの数が増えてくると、それらの間の相違点を特定することが必要になる場合があります。

vRealize Automation Cloud Assembly の [バージョン履歴] ビューでバージョンを選択し、[差分] をクリックします。次に、[比較対象] ドロップダウンから、比較対象となる別のバージョンを選択します。

コードの違いを確認するか、視覚的なトポロジの違いを確認するかを切り替えることができます。

図 6-1. コードの違い



ここで説明する手法には、インフラストラクチャ コードに関する知識が多少必要です。幸い、vRealize Automation Cloud Assembly コードは人間が判読できるため、かなり簡単に理解できます。

vRealize Automation でのユーザー入力によるクラウド テンプレートのカスタマイズ方法

クラウド テンプレート デザイナは、入力パラメータを使用して、ユーザーが申請時にカスタム選択を行えるようにできます。

ユーザーが入力するようにすると、違いのわずかなテンプレートの複数のコピーを保存する必要がなくなります。また、入力する場合は、インストール後の作業に対応したテンプレートを準備できます。[vRealize Automation の Day 2 更新でクラウド テンプレートの入力を使用する方法](#) を参照してください。

次の入力は、1つのクラウド テンプレートを MySQL データベース サーバ用に作成し、ユーザーがそのテンプレートを別のクラウド リソース環境に展開して、毎回異なる容量と認証情報を適用する方法を示しています。

クラウド テンプレートの入力パラメータを定義する方法

テンプレート コードに `inputs` セクションを追加し、選択可能な値を設定します。

次の例では、マシン サイズ、オペレーティング システム、およびクラスタ化されたサーバの数を選択できます。

```
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)

```

コードの編集に慣れていない場合は、コード エディタの [入力] タブをクリックして、設定を入力できます。次の例は、前述した MySQL データベースの入力の一部を示しています。

The screenshot shows the 'Inputs' tab of a Cloud Template. It contains a table with the following data:

<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

An 'Edit Cloud Template Input: size' modal is open, showing the following fields:

- Name: size
- Title: Tier Machine Size
- Description: Size of Nodes
- Type: string (dropdown menu)
- Encrypted: ☐

クラウド テンプレートの入力パラメータを参照する方法

次に、resources セクションで、`${input. property-name }` 構文を使用して入力パラメータを参照します。

プロパティ名にスペースが含まれている場合は、ドット表記を使用するのではなく、次のように角括弧と二重引用符で囲みます。 `${input["プロパティ名"]}`

重要： クラウド テンプレート コードでは、入力パラメータを指す目的以外に `input` という語を使用することはできません。

```

resources:
  WebTier:
    type: Cloud.Machine
    properties:

```

```
name: wordpress
flavor: '${input.wp-size}'
image: '${input.wp-image}'
count: '${input.wp-count}'
```

入力プロパティのリスト

プロパティ	説明
const	oneOf とともに使用されます。わかりやすいタイトルに関連付けられている実際の値。
default	入力に対して事前に設定される値。 デフォルトは、正しいタイプである必要があります。整数のデフォルトとして単語を入力しないでください。
description	入力のユーザー ヘルプ テキスト。
encrypted	ユーザーが指定した入力を暗号化するかどうかを True または False で指定します。 通常、パスワードは暗号化されます。
enum	使用可能な値のドロップダウン メニュー。 次の例をフォーマット ガイドとして使用します。 <pre>enum: - value 1 - value 2</pre>
format	入力する際に期待されるフォーマットを設定します。たとえば、(25/04/19) は日時をサポートします。 vRealize Automation Service Broker カスタム フォームで日付ピッカーを使用できるようにします。
items	アレイ内の項目を宣言します。数値、整数、文字列、ブール値、またはオブジェクトをサポートします。
maxItems	アレイ内の選択可能な項目の最大数。
maxLength	文字列の場合に許容される最大文字数。 たとえば、フィールドを 25 文字までに制限するには、 <code>maxLength: 25</code> と入力します。
maximum	数値または整数の場合に許容される最大値。
minItems	アレイ内の選択可能な項目の最小数。
minLength	文字列の場合に許容される最小文字数。
minimum	数値または整数の場合に許容される最小値。
oneOf	ユーザー入力フォームでわかりにくい値 (const) をわかりやすい名前 (タイトル) で表示できるようにします。デフォルト値を設定する場合は、タイトルではなく、定数を設定します。 文字列、整数、および数値のタイプとともに使用します。

プロパティ	説明
pattern	正規表現の構文で表した、文字列入力に使用可能な文字。 例: '[a-z]+' または '[a-z0-9A-Z@#&]+'
properties	オブジェクトの key:value プロパティ ブロックを宣言します。
readOnly	フォーム ラベルのみを指定する際に使用されます。
title	oneOf とともに使用されます。定数値のわかりやすい名前。タイトルは、展開時にユーザー入力フォームに表示されます。
type	数値、整数、文字列、ブール値、またはオブジェクトを表すデータ タイプ。
writeOnly	フォームのアスタリスクの後ろにあるキーストロークを非表示にします。enum とともに使用できません。vRealize Automation Service Broker カスタム フォームのパスワード フィールドとして表示されます。

その他の例

列挙型の文字列

```
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
```

最小値と最大値を持つ整数

```
count:
  type: integer
  title: Machine Count
  description: The number of machines that you want to deploy.
  maximum: 5
  minimum: 1
  default: 1
```

オブジェクトのアレイ

```
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
```

わかりやすい名前を持つ文字列

```
platform:
  type: string
  oneOf:
    - title: AWS
      const: platform:aws
    - title: Azure
      const: platform:azure
    - title: vSphere
      const: platform:vsphere
  default: platform:aws
```

パターン検証を持つ文字列

```
username:
  type: string
  title: Username
  description: The name for the user that will be created when the machine is provisioned.
  pattern: ^[a-zA-Z]+$
```

パスワードとしての文字列

```
password:
  type: string
  title: Password
  description: The initial password that will be required to logon to the machine.
  Configured to reset on first login.
  encrypted: true
  writeOnly: true
```

テキスト エリアとしての文字列

```
ssh_public_key:
  type: string
  title: SSH public key
  maxLength: 256
```

ブール値

```
public_ip:
  type: boolean
  title: Assign public IP address
  description: Choose whether your machine should be internet facing.
  default: false
```

カレンダーの日付と時刻のセレクト

```
leaseDate:
  type: string
  title: Lease Date
  format: date-time
```

vRealize Automation Cloud Assembly のリソース フラグによる申請のカスタマイズ方法

vRealize Automation Cloud Assembly には、申請時のリソースの処理方法を調整するクラウド テンプレート設定がいくつか含まれています。

リソース フラグの設定は、リソース オブジェクト プロパティ スキーマの一部ではありません。図のように、特定のリソースのフラグ設定を、プロパティ セクションの外部に追加します。

```
resources:
  Cloud_Machine_1:
    type: Cloud.Machine
    preventDelete: true
    properties:
      image: coreos
      flavor: small
      attachedDisks:
        - source: '${resource.Cloud_Volume_1.id}'
  Cloud_Volume_1:
    type: Cloud.Volume
    properties:
      capacityGb: 1
```

リソース フラグ	説明
createBeforeDelete	<p>一部の更新操作では、既存のリソースを削除してから、新しいリソースを作成する必要があります。デフォルトでは、削除が最初に行われます。このため、古いリソースが削除されたにもかかわらず、何らかの理由で新しいリソースが正常に作成されていないことがあります。</p> <p>前のリソースを削除する前に、新しいリソースが正常に作成されていることを確認する必要がある場合は、このフラグを <code>true</code> に設定します。</p>
createTimeout	<p>リソースの割り当て、作成、プランの申請に関する vRealize Automation Cloud Assembly のデフォルトのタイムアウトは、2 時間 (2h) です。また、プロジェクト管理者は、これらの申請に対してカスタムのデフォルト タイムアウトを設定して、このタイムアウトをプロジェクト全体に適用できます。</p> <p>このフラグを使用すると、任意のデフォルト設定をオーバーライドして、特定のリソース操作のタイムアウトを個別に設定することができます。updateTimeout および deleteTimeout も参照してください。</p>
deleteTimeout	<p>削除申請に対する vRealize Automation Cloud Assembly のデフォルト タイムアウトは 2 時間 (2h) です。また、プロジェクト管理者は、削除申請に対して別のデフォルト タイムアウトを設定して、このタイムアウトをプロジェクト全体に適用できます。</p> <p>このフラグを使用すると、任意のデフォルト設定をオーバーライドして、特定のリソース削除操作のタイムアウトを個別に設定することができます。updateTimeout および createTimeout も参照してください。</p>
dependsOn	<p>このフラグは、リソース間の明示的な依存関係を識別します。依存関係がある場合は、次のリソースを作成するには、その前のリソースが存在している必要があります。詳細については、vRealize Automation Cloud Assembly でリソースの展開順序を設定する方法を参照してください。</p>
dependsOnPreviousInstances	<p><code>true</code> に設定した場合は、クラスター リソースを順次作成します。デフォルトは <code>false</code> で、クラスター内のすべてのリソースが同時に作成されます。</p> <p>たとえば、順次作成は、プライマリ ノードとセカンダリ ノードを作成する必要があるデータベース クラスターに役立ちますが、セカンダリ ノードの作成にはノードを既存のプライマリ ノードに接続する設定が必要です。</p>
forceRecreate	<p>一部の更新操作では、既存のリソースを削除してから、新しいリソースを作成する必要があります。更新するときに、古いリソースを削除してから新しいリソースを作成するようにしたい場合は、この動作が更新時のデフォルトであるかどうかに関係なく、このフラグを <code>true</code> に設定します。</p>
ignoreChanges	<p>リソースのユーザーがリソースを展開された状態から変更して、再構成する場合があります。</p> <p>展開の更新を実行するが、変更したリソースをクラウド テンプレートの構成で上書きしない場合は、このフラグを <code>true</code> に設定します。</p>

リソース フラグ	説明
preventDelete	後続の削除要求からリソースを保護する必要がある場合は、このフラグを true に設定します。
updateTimeout	更新申請に対する vRealize Automation Cloud Assembly のデフォルト タイムアウトは 2 時間 (2h) です。また、プロジェクト管理者は、更新申請に対して別のデフォルト タイムアウトを設定して、このタイムアウトをプロジェクト全体に適用できます。 このフラグを使用すると、任意のデフォルト設定をオーバーライドして、特定のリソース更新操作のタイムアウトを個別に設定することができます。deleteTimeout および createTimeout も参照してください。

vRealize Automation Cloud Assembly でリソースの展開順序を設定する方法

vRealize Automation Cloud Assembly テンプレートを展開するときに、あるリソースの前に別のリソースを展開することが必要な場合があります。

重要： 実線または点線の矢印は、依存関係のみを示しています。接続されていることを示すものではありません。リソースを接続して通信する方法については、[vRealize Automation Cloud Assembly でクラウド テンプレート リソースを接続する方法](#)を参照してください。

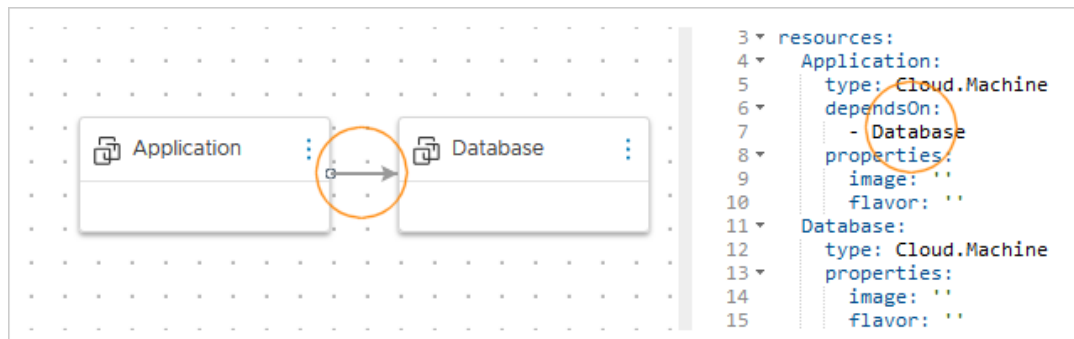
明示的な依存関係を作成する方法

あるリソースの前に別のリソースを展開することが必要な場合があります。たとえば、アプリケーション サーバを作成してアクセスするよう設定する前に、データベース サーバが展開されている必要があります。

明示的な依存関係は、導入時の、またはスケール インまたはスケール アウト アクションのビルド順を設定します。明示的な依存関係は、グラフィカル デザイン キャンバスまたはコード エディタを使用して追加できます。

- デザイン キャンバス オプション：依存リソースから始まり最初に展開するリソースで終わる接続を描画します。
- コード エディタ オプション：依存リソースに dependsOn プロパティを追加し、最初に展開するリソースを特定します。

明示的な依存関係では、キャンバスに実線の矢印が作成されます。



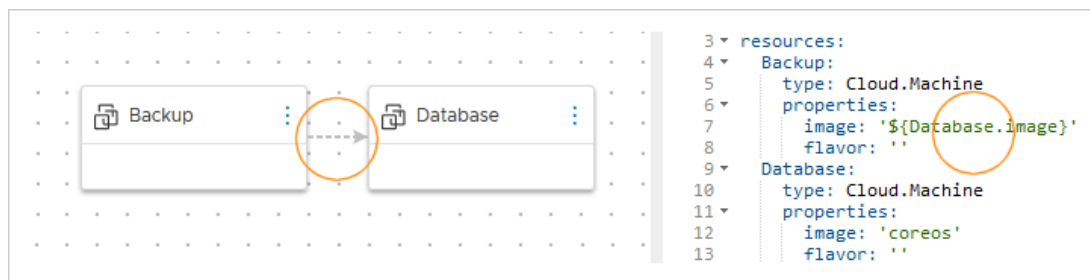
明示的な依存関係またはプロパティ バインドを作成する方法

リソースのプロパティに、別のリソースのプロパティの値が必要になることがあります。たとえば、バックアップ サーバはバックアップするデータベース サーバのオペレーティング システム イメージを必要とすることがあり、その場合はデータベース サーバが先に配置されている必要があります。

暗黙的な依存関係はプロパティ バインドとも呼ばれ、必要なプロパティが使用可能になるまで依存する側のリソースの展開を待つことによってビルドの順序を制御します。暗黙的な依存関係は、コード エディタを使用して追加します。

- 依存リソースを編集して、最初に配置されている必要があるリソースとプロパティを識別するためのプロパティを追加します。

暗黙的な依存関係またはプロパティ バインドでは、キャンバスに破線の矢印が作成されます。



vRealize Automation Cloud Assembly で式を使用してクラウド テンプレート コードの汎用性を高める方法

柔軟性を高めるため、式を vRealize Automation Cloud Assembly のクラウド テンプレート コードに追加できます。

式では、次の例に示すように、`${expression}` コンストラクトが使用されます。

これらの例では、重要な行のみを表示するよう省略しています。未編集のクラウド テンプレート全体が最後に表示されます。

例

展開時に、リモート アクセスに必要な暗号化キーをユーザーが貼り付けられるようにします。

```

inputs:
  sshKey:
    type: string
    maxLength: 500
resources:
  frontend:
    type: Cloud.Machine
    properties:
      remoteAccess:
        authentication: publicPrivateKey
        sshKey: '${input.sshKey}'

```

VMware Cloud on AWS への展開の場合、フォルダ名をワークロードの必要な名前に設定します。

```
inputs:
  environment:
    type: string
    enum:
      - AWS
      - vSphere
      - Azure
      - VMC
      - GCP
    default: vSphere
resources:
  frontend:
    type: Cloud.Machine
    properties:
      folderName: '${input.environment == "VMC" ? "Workload" : ""}'
```

展開時に、選択した環境に一致する env タグ（すべて小文字）でマシンをタグ付けします。

```
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)}'
```

フロントエンド クラスタ内のマシンの数を 1（小規模）または 2（大規模）に設定します。大規模クラスタは以下の除外プロセスによって設定される点に注目してください。

```
inputs:
  envsize:
    type: string
    enum:
      - Small
      - Large
resources:
  frontend:
    type: Cloud.Machine
    properties:
      count: '${input.envsize == "Small" ? 1 : 2}'
```

ネットワーク リソースで検出されたプロパティにバインドすることにより、マシンを同じデフォルトのネットワークに接続します。

```
resources:
  frontend:
    type: Cloud.Machine
    properties:
      networks:
        - network: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - network: '${resource.Cloud_Network_1.name}'
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      name: Default
      networkType: existing
```

API に送信されたアクセス認証情報を暗号化します。

```
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 マシンのアドレスを検出します。

```
resources:
  frontend:
    type: Cloud.Machine
    properties:
      cloudConfig: |
        runcmd:
          - echo ${resource.apitier.networks[0].address}
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - network: '${resource.Cloud_Network_1.name}'
```

完全なクラウド テンプレート

```

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:
        - network: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      folderName: '${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: '${"env:" + to_lower(input.environment)}'
networks:
  - network: '${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 のクラウド テンプレート式の構文

式の構文には、vRealize Automation Cloud Assembly テンプレートで使用可能な式のすべての機能が示されています。

この構文は、[vRealize Automation Cloud Assembly で式を使用してクラウド テンプレート コードの汎用性を高める方法](#)で、その例のほんの一部が記載されています。

リテラル

次のリテラルがサポートされています。

- ブール値 (True または False)
- 整数
- 浮動小数点
- 文字列

二重引用符、一重引用符、およびバックスラッシュは、バックスラッシュでエスケープします。

" は \" とエスケープします

' は \' とエスケープします

\ は \\ とエスケープします

引用符は、次の例に示すように、同じタイプの引用符で囲まれた文字列内でのみエスケープする必要があります。

```
"I am a \"double quoted\" string inside \"double quotes\"."
```

- Null

環境変数

環境名：

- orgId
- projectId
- projectName
- deploymentId

- deploymentName
- blueprintId
- blueprintVersion
- blueprintName
- requestedBy (ユーザー)
- requestedAt (時間)

構文：

```
env.ENV_NAME
```

例：

```
${env.blueprintId}
```

リソース変数

リソース変数を使用すると、他のリソースからリソース プロパティにバインドできます。

構文：

```
resource.RESOURCE_NAME.PROPERTY_NAME
```

例：

- \${resource.db.id}
- \${resource.db.networks[0].address}
- \${resource.app.id} (カウントが指定されていない場合、クラスタ化されていないリソースの文字列を返します。クラスタ化されたリソースのアレイを返します。)
- \${resource.app[0].id} (クラスタ化されたリソースの最初のエントリを返します)

リソースのセルフ変数

リソースのセルフ変数は、割り当てフェーズをサポートするリソースでのみ許可されます。リソースのセルフ変数は、割り当てフェーズが完了した後にのみ使用できます (または値が設定されます)。

構文：

```
self.property_name
```

例：

```
${self.address} (割り当てフェーズで割り当てられたアドレスを返します。)
```

resource_x という名前のリソースの場合、self.property_name と resource.resource_x.property_name は同じであり、両方とも自己参照とみなされます。

クラスタ数のインデックス

構文：

```
count.index
```

例：

`${count.index == 0 ? "primary" : "secondary"}` (クラスタ化されたリソースのノード タイプを返します。)

制限事項：

リソース割り当てに `count.index` を使用することはできません。たとえば、入力時に作成されたディスク アレイ内の位置を次のキャパシティ式で参照すると失敗します。

```
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)}'
```

条件

構文：

- 等価演算子は `==` および `!=` です。
- 関係演算子は、`<` `>` `<=` および `>=` です。
- 論理演算子は、`&&` `||` および `!` です。
- 条件は、次のパターンを使用します。

condition-expression ? true-expression : false-expression

例：

`${input.count < 5 && input.size == 'small'}`

`${input.count < 2 ? "small" : "large"}`

算術演算子

構文：

演算子は、`+` `-` `/` `*` および `%` です。

例：

```
${(input.count + 5) * 2}
```

文字列の連結

構文：

`${'ABC' + 'DEF'}` は ABCDEF と評価されます。

演算子 [] および .

式は、[] および . 演算子の処理を統一して、ECMAScript に従います：

このため、`expr.identifier` は `expr["identifier"]` に相当します。この識別子は、値が識別子であるリテラルを作成するために使用され、その値とともに [] 演算子が使用されます。

例：

```
${resource.app.networks[0].address}
```

また、プロパティにスペースが含まれている場合は、ドット表記を使用するのではなく、次のように角括弧と二重引用符で囲みます。

誤：

```
input.operating system
```

正：

```
input["operating system"]
```

マップの構築

構文：

```
${{'key1':'value1', 'key2':input.key2}}
```

アレイの構築

構文：

```
${['key1','key2']}
```

例：

```
${[1,2,3]}
```

関数

構文：

```
${function(arguments...)}
```

例：

```
${to_lower(resource.app.name)}
```


表 6-1. 関数

関数	説明
abs(数値)	絶対値の値
floor(数値)	引数以下で、かつ正確な整数と等しい最大値（正の無限大に最も近い値）を返します
ceil(数値)	引数以上で、かつ正確な整数と等しい最小値（負の無限大に最も近い値）を返します
to_lower(文字列)	文字列を小文字に変換します
to_upper(文字列)	文字列を大文字に変換します
contains(アレイ, 値)	アレイに値が含まれているかどうかを確認します
contains(文字列, 値)	文字列に値が含まれているかどうかを確認します
join(アレイ, 区切り文字)	文字列のアレイを区切り文字で結合し、文字列を返します
split(文字列, 区切り文字)	区切り文字を使用して文字列を分割し、文字列のアレイを返します
slice(アレイ, 開始, 終了)	開始インデックスから終了インデックスまでのアレイを切り出して返します
reverse(アレイ)	アレイの逆順のエントリを返します
starts_with(サブジェクト, プリフィックス)	サブジェクトの文字列がプリフィックスの文字列で始まるかどうかを確認します
ends_with(サブジェクト, サフィックス)	サブジェクトの文字列がサフィックスの文字列で終わるかどうかを確認します
replace(文字列, ターゲット, 置換)	ターゲットの文字列を含む文字列をターゲットの文字列に置換します
substring(文字列, 開始, 終了)	開始インデックスから終了インデックスまでの文字列の部分文字列を返します
format(フォーマット, 値...)	Java の クラスのフォーマット のフォーマットと値を使用して、フォーマット化された文字列を返します。
keys(マップ)	マップのキーを返します
values(マップ)	マップの値を返します
merge(マップ, マップ)	マージされたマップを返します
length(文字列)	文字列の長さを返します
length(アレイ)	アレイの長さを返します
max(アレイ)	数値のアレイから最大値を返します
min(アレイ)	数値のアレイから最小値を返します
sum(アレイ)	数値のアレイからすべての値の合計を返します
avg(アレイ)	数値のアレイからすべての値の平均を返します

表 6-1. 関数（続き）

関数	説明
digest(値, タイプ)	サポートされているタイプ（md5、sha1、sha256、sha384、sha512）を使用して値のダイジェストを返します。
to_string(値)	値の文字列表現を返します
to_number(文字列)	文字列を数値として解析します
not_null(アレイ)	null でない最初のエントリを返します
base64_encode(文字列)	base64 でエンコードした値を返します
base64_decode(文字列)	base64 でデコードした値を返します
now()	ISO-8601 フォーマットの現在時刻を返します
uuid()	ランダムに生成された UUID を返します
from_json(文字列)	JSON 文字列を解析します
to_json(値)	JSON 文字列として値をシリアル化します
json_path(値, パス)	JSON に対する XPath を使用して、パスを値に対して評価します。
matches(文字列, 正規表現)	文字列が正規表現と一致するかどうかを確認します
url_encode(文字列)	URL エンコード仕様を使用して文字列をエンコードします
trim(文字列)	先頭と末尾のスペースを削除します

vRealize Automation Cloud Assembly テンプレートでリモート アクセスを有効にする方法

vRealize Automation Cloud Assembly によって展開されたマシンにリモート アクセスするには、展開する前にそのマシンのクラウド テンプレートにプロパティを追加します。

リモート アクセスでは、次のいずれかの認証オプションを設定できます。

注： キーをコピーする必要がある場合は、クラウド テンプレートに cloudConfig セクションを作成して、プロビジョニング時にキーが自動的にコピーされるようにすることもできます。詳細はここでは説明しませんが、cloudConfig に関する一般的な情報が [vRealize Automation Cloud Assembly テンプレートでマシンを自動的に初期化する方法](#) に示されています。

vRealize Automation Cloud Assembly プロビジョニング時のキー ペアの生成

リモート アクセス認証用の独自のパブリック - プライベート キー ペアを持っていない場合は、vRealize Automation Cloud Assembly でキー ペアを生成できます。

次のコードをガイドラインとして使用してください。

- 1 例に示すように、vRealize Automation Cloud Assembly では、プロビジョニングの前に `remoteAccess` プロパティをクラウド テンプレートに追加します。

ユーザー名はオプションです。省略すると、システムによってランダムな ID がユーザー名として生成されます。

例：

```
type: Cloud.Machine
properties:
  name: our-vm2
  image: Linux18
  flavor: small
  remoteAccess: authentication: generatedPublicPrivatekey username: testuser
```

- 2 vRealize Automation Cloud Assembly で、マシンをクラウド テンプレートからプロビジョニングし、起動状態にします。

プロビジョニング プロセスによってキーが生成されます。

- 3 [展開] - [トポロジ] プロパティでキー名を特定します。

- 4 vSphere クライアントなどのクラウド プロバイダ インターフェイスを使用して、プロビジョニングするマシンのコマンド ラインにアクセスします。

- 5 プライベート キーに読み取り権限を付与します。

```
chmod 600 key-name
```

- 6 vRealize Automation Cloud Assembly 展開に移動し、マシンを選択して、[アクション] - [プライベート キーの取得] の順にクリックします。

- 7 プライベート キー ファイルをローカル マシンにコピーします。

一般的なローカル ファイル パスは、`/home/username/.ssh/key-name` です。

- 8 リモート SSH セッションを開き、プロビジョニングされたマシンに接続します。

```
ssh -i key-name user-name@machine-ip
```

vRealize Automation Cloud Assembly に対する独自のパブリック - プライベート キーペアの指定

多くの企業は、認証のために独自のパブリック - プライベート キー ペアを作成して配布します。

次のコードをガイドラインとして使用してください。

- 1 ローカル環境で、パブリック - プライベート キー ペアを取得または生成します。

ここでは、単にキーを生成してローカルに保存します。

- 2 例に示すように、vRealize Automation Cloud Assembly では、プロビジョニングの前に `remoteAccess` プロパティをクラウド テンプレートに追加します。

`sshKey` には、パブリック キー ファイル `key-name.pub` 内で見つかった長い英数字が含まれています。

ユーザー名はオプションで、ログインで使用するために作成されます。省略すると、システムによってランダムな ID がユーザー名として生成されます。

例：

```
type: Cloud.Machine
properties:
  name: our-vm1
  image: Linux18
  flavor: small
  remoteAccess:
    authentication: publicPrivateKey
    sshKey: ssh-rsa Iq+5aQgBP3ZNT4o1baP5Ii+dstIcowRRkyobbfpA1mj9ts1f
qGxvU66PX9IeZax5hZvNWFgJw6ag+Z1zndOLhVdVoW49f274/mIRi1d7Uuw...
    username: testuser
```

- 3 vRealize Automation Cloud Assembly で、マシンをクラウド テンプレートからプロビジョニングし、起動状態にします。
- 4 クラウド ベンダー クライアントを使用して、プロビジョニングされたマシンにアクセスします。
- 5 マシンのホーム フォルダにパブリック キー ファイルを追加します。remoteAccess.sshKey で指定したキーを使用します。
- 6 プライベート キー ファイルの相手側がローカル マシンに含まれることを確認します。

キーの一般的な形式は /home/username/.ssh/key-name で、拡張子 .pub はありません。

- 7 リモート SSH セッションを開き、プロビジョニングされたマシンに接続します。

```
ssh -i key-name user-name@machine-ip
```

vRealize Automation Cloud Assembly への AWS キー ペアの指定

AWS キー ペア名をクラウド テンプレートに追加することで、vRealize Automation Cloud Assembly が AWS に展開するマシンにリモートからアクセスできます。

AWS キー ペアはリージョン固有であることに注意してください。us-east-1 にワークロードをプロビジョニングする場合は、キー ペアも us-east-1 に含まれている必要があります。

次のコードをガイドラインとして使用してください。このオプションは、AWS クラウド ゾーンでのみ機能します。

```
type: Cloud.Machine
properties:
  image: Ubuntu
  flavor: small
  remoteAccess: authentication: keyPairName keyPair: cas-test
constraints:
  - tag: 'cloud:aws'
```

vRealize Automation Cloud Assembly へのユーザー名とパスワードの指定

クラウド テンプレートにユーザー名とパスワードを追加することで、vRealize Automation Cloud Assembly によって展開されるマシンにシンプルなりモート アクセスを実行できるようになります。

これによって安全性は低下しますが、ユーザー名とパスワードを使用してリモート ログインすることが必要な状況も考えられます。クラウド ベンダーや構成によっては、このセキュリティの低いオプションがサポートされない場合があることに注意してください。

- 1 例に示すように、vRealize Automation Cloud Assembly では、プロビジョニングの前に `remoteAccess` プロパティをクラウド テンプレートに追加します。

ユーザー名とパスワードを使用してログインするアカウントに、ユーザー名とパスワードを設定します。

例：

```
type: Cloud.Machine
properties:
  name: our-vm3
  image: Linux18
  flavor: small
  remoteAccess: authentication: usernamePassword username: testuser password: admin123
```

- 2 vRealize Automation Cloud Assembly で、マシンをクラウド テンプレートからプロビジョニングし、起動状態にします。
- 3 クラウド ベンダーのインターフェイスに移動し、プロビジョニングされたマシンにアクセスします。
- 4 プロビジョニングされたマシンで、アカウントを作成または有効にします。
- 5 ローカル マシンから、プロビジョニングされたマシンの IP アドレスまたは FQDN へのリモートセッションを開き、通常どおりユーザー名とパスワードを使用してログインします。

vRealize Automation Cloud Assembly 設計に高度な機能を追加する方法

設計のエンタープライズ対応を強化する高度な Infrastructure-as-Code (IaC) 技術と vRealize Automation Cloud Assembly の機能があります。

ここで説明するいくつかの機能は、vRealize Automation Cloud Assembly の設計機能を拡張しますが、その他の機能はクラウド テンプレート コーディング手法に直接適用されます。

vRealize Automation Cloud Assembly を使用して展開されたリソースの名前をカスタマイズする方法

クラウド管理者またはプロジェクト管理者は、環境内のリソースの命名規則を設定し、展開されるリソースがユーザーの関与なしでその規則に従うようにすることができます。vRealize Automation Cloud Assembly プロジェクトのすべての展開用に命名テンプレートを作成できます。

たとえば、ホストの命名規則でリソースのプリフィックスを *projectname-sitecode-costcenter-whereDeployed-identifier* にします。プロジェクトごとに、マシンのカスタム命名テンプレートを構成します。テンプレート変数の一部は、展開時にシステムから取得されます。それ以外の変数は、プロジェクトのカスタム プロパティに基づきます。上記のプリフィックスのカスタム命名テンプレートは、次の例のようになります。

```
${project.name}-${resource.siteCode}-${resource.costCenter}-${endpoint.name}-${#####}
```

テンプレートで `{#####}` と示されている識別子の部分には、6 桁の識別子が表示されます。この識別子は、一意性を確保するためのカウンタです。このカウンタは組織に対してグローバルであるため、現在のプロジェクトだけでなく、すべてのプロジェクトにわたって増加します。複数のプロジェクトがある場合、現在のプロジェクト内の展開の識別子は、たとえば 000123 から 000124 へは増加せず、000123 から 000127 のように増加します。

すべてのリソース名は一意である必要があります。増分値による番号のプロパティを使用して一意性を確保します。この番号は、vRealize Automation Cloud Assembly によって名前付けされる展開を含め、すべての展開に割り振られます。システムの堅牢性が向上し、また、仮想マシン、ロード バランサ、セキュリティ グループ、NAT、ゲートウェイ、リソース グループ、ディスクなど、多くのリソースにカスタム命名が適用されているため、番号付けがランダムに見える場合がありますが、値の一意性は依然として保証されます。また、テスト展開を実行したときも番号は増加します。

ここに記載されている例に加えて、ユーザー名、使用されるイメージ、その他の組み込みオプション、単純な文字列を追加することもできます。テンプレートの作成時には、使用可能なオプションに関するヒントが表示されます。

表示される値の一部は、使用事例での例に過ぎないことに注意してください。使用環境で、使用事例の一つ一つをそのまま使用することはできません。使用環境独自のクラウド インフラストラクチャや展開管理のニーズに合わせて、置き換える必要がある値を検討するか、使用事例の値を当てはめます。

前提条件

- プロジェクトからの展開に使用する命名規則が明確になっていることを確認します。
- この手順では、カスタム ホスト プリフィックスの名前付けをテストするために使用する単純なクラウド テンプレートがすでにあるか、作成できることを前提としています。

手順

- 1 [インフラストラクチャ] - [プロジェクト] の順に選択します。
- 2 既存のプロジェクトを選択するか、新しいプロジェクトを作成します。
- 3 [プロビジョニング] タブの [カスタム プロパティ] セクションで、サイト コード値とコスト センター値のプロパティを作成します。

ここで、表示されている値を環境に適した値に置き換えます。

カスタム プロパティ
このプロジェクトのすべての申請に追加するカスタム プロパティを指定します。 ①

カスタム プロパティの 定義	名前	値
	siteCode	BGL
	costCenter	IT-research

カスタム命名
このプロジェクトでプロビジョニングされるマシン、ネットワーク、セキュリティ グループ、およびディスクに使用する名前付けテンプレートを指定します。

テンプレート `${project.name}-${resource.siteCode}-${resource` ①
Hint: Avoid conflicting names by generating digits in names. \${#####}

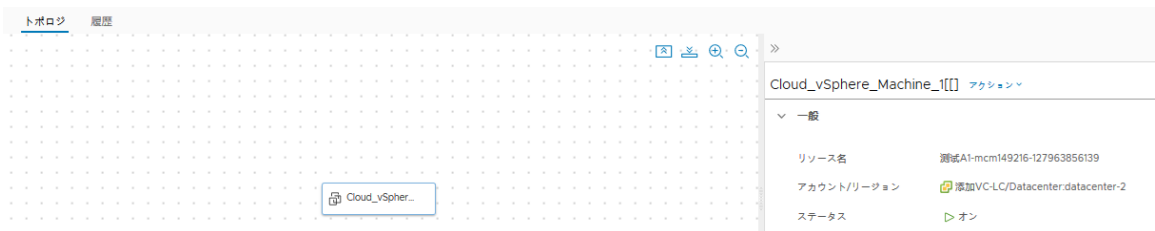
- 名前が **siteCode** で値が **BGL** であるカスタム プロパティを作成します。
- 名前が **costCenter** で値が **IT-research** である別のカスタム プロパティを追加します。

4 [カスタム命名] セクションで、次のテンプレートを追加します。

```
${project.name}-${resource.siteCode}-${resource.costCenter}-${endpoint.name}-${#####}
```

文字列をコピーすることはできますが、これが最初の命名テンプレートである場合は、ビルド時にヒント テキストとクイック選択を使用することを検討してください。

- プロジェクトに関連付けられているクラウド テンプレートを展開して、カスタム名がリソースに適用されていることを確認します。
 - [デザイン] タブをクリックし、プロジェクトに関連付けられているクラウド テンプレートをクリックします。
 - クラウド テンプレートを展開します。
[展開] タブが開き、進行中の展開が示されます。
 - 展開が完了したら、展開名をクリックします。
 - [トポロジ] タブの右側のペインで、カスタム名がリソース名になっていることを確認します。



- 命名規則を確認するためにテスト クラウド テンプレートを展開した場合は、その展開を削除します。

次のステップ

他のプロジェクト用にカスタム命名テンプレートを作成します。

vRealize Automation Cloud Assembly テンプレートでマシンを自動的に初期化する方法

vRealize Automation Cloud Assembly でマシンの初期化を適用するには、コマンドを直接実行するか、または vSphere ベースのクラウド ゾーンに展開する場合はカスタマイズ仕様を使用できます。

- コマンド：クラウド テンプレート コード内の cloudConfig セクションに、実行するコマンドが保持されます。
- カスタマイズ仕様：クラウド テンプレート コード内のプロパティから、vSphere のカスタマイズ仕様が名前前で参照されます。

コマンドとカスタマイズ仕様が共存できない可能性について

vSphere に展開する場合、cloudConfig とカスタマイズ仕様による初期化を組み合わせるときは作業に注意が必要です。これらは正式には互換性がなく、組み合わせて使用すると、一貫性のない結果、または望ましくない結果が生じる場合があります。

コマンドとカスタマイズ仕様の関係の例については、[vRealize Automation Cloud Assembly クラウド テンプレートでの vSphere 固定 IP アドレスの割り当て](#)を参照してください。

vRealize Automation Cloud Assembly テンプレートの vSphere カスタマイズ仕様

vSphere ベースのクラウド ゾーンに展開する場合、展開時にカスタマイズ仕様によってゲスト OS の設定を適用できます。

カスタマイズ仕様を有効にする方法

カスタマイズ仕様は、展開先の vSphere に配置する必要があります。

クラウド テンプレート コードを直接編集します。次の例では、vSphere 上の WordPress ホスト用の cloud-assembly-linux カスタマイズ仕様を参照しています。

```
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'
```

カスタマイズ仕様または cloudConfig コマンドを使用するかどうか

プロビジョニングの操作性を vSphere で現在実行しているものと同じようにするは、カスタマイズ仕様を引き続き使用することをお勧めします。ただし、ハイブリッドまたは複数のクラウド プロビジョニングに拡張する場合、cloudConfig 初期化コマンドを使用する方法がより無難です。

クラウド テンプレートの cloudConfig セクションの詳細については、[vRealize Automation Cloud Assembly テンプレートの設定コマンド](#)を参照してください。

コマンドとカスタマイズ仕様が共存できない可能性について

vSphere に展開する場合、組み込みの cloudConfig コマンドとカスタマイズ仕様による初期化を組み合わせるときは作業に注意が必要です。これらは正式には互換性がなく、組み合わせて使用すると、一貫性のない結果、または望ましくない結果が生じる場合があります。

コマンドとカスタマイズ仕様の関係の例については、[vRealize Automation Cloud Assembly クラウド テンプレートでの vSphere 固定 IP アドレスの割り当て](#)を参照してください。

vRealize Automation Cloud Assembly テンプレートの設定コマンド

cloudConfig セクションを vRealize Automation Cloud Assembly クラウド テンプレート コードに追加して、展開時に実行されるマシン初期化コマンドを追加できます。

cloudConfig コマンドの作成方法

- Linux：初期化コマンドはオープン スタンドardsの [cloud-init](#) に基づきます。
- Windows：初期化コマンドは [Cloudbase-init](#) を使用します。

Linux の [cloud-init](#) と Windows の [Cloudbase-init](#) は同じ構文を共有しません。一方のオペレーティング システムの cloudConfig セクションは、もう一方のオペレーティング システムのマシン イメージでは動作しません。

cloudConfig コマンドの機能

初期化コマンドを使用して、インスタンスの作成時にデータまたは設定の適用を自動化します。これにより、ユーザー、権限、インストール、または他のコマンドベースの操作をカスタマイズできます。次に例を示します。

- ホスト名の設定
- SSH プライベート キーの生成と設定
- パッケージのインストール

cloudConfig コマンドを追加できる場所

cloudConfig セクションはクラウド テンプレート コードに追加できますが、あらかじめ、インフラストラクチャの構成時にマシン イメージに追加することもできます。これにより、そのソース イメージを参照するすべてのクラウド テンプレートに同じ初期化が行われます。

イメージ マップとクラウド テンプレートの両方に初期化コマンドが含まれている場合があります。展開時、これらのコマンドはマージされ、vRealize Automation Cloud Assembly が統合されたコマンドを実行します。

どちらにも同じコマンドが含まれているものの、パラメータが異なる場合は、イメージ マップのコマンドのみが実行されます。

詳細については、[vRealize Automation でのイメージ マッピングの詳細](#)を参照してください。

cloudConfig コマンドの例

次の cloudConfig セクションの例は、Linux ベースの MySQL サーバ用の [基本的なクラウド テンプレートの作成](#) のクラウド テンプレート コードから取得されています。

注： コマンドが正しく解釈されるように、以下に示すように必ずパイプ文字 cloudConfig: | を含めてください。

```
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
```

cloud-init スクリプトが予期しない動作をする場合は、トラブルシューティングの際に /var/log/cloud-init-output.log に取得されるコンソールの出力を確認してください。cloud-init の詳細については、[cloud-init のドキュメント](#)を参照してください。

コマンドとカスタマイズ仕様が共存できない可能性について

vSphere に展開する場合、組み込みの cloudConfig コマンドとカスタマイズ仕様による初期化を組み合わせるときは作業に注意が必要です。これらは正式には互換性がなく、組み合わせると、一貫性のない結果、または望ましくない結果が生じる場合があります。

コマンドとカスタマイズ仕様の関係の例については、[vRealize Automation Cloud Assembly クラウド テンプレートでの vSphere 固定 IP アドレスの割り当て](#)を参照してください。

vRealize Automation Cloud Assembly クラウド テンプレートでの vSphere 固定 IP アドレスの割り当て

vSphere に展開する場合は固定 IP アドレスを割り当てることができますが、cloudConfig 初期化コマンドとカスタマイズ仕様の間に競合が発生しないように注意する必要があります。

デザインのサンプル

次に示すデザインでは、クラウド テンプレートの初期化コマンドとカスタマイズ仕様の間に競合が発生しないように、固定 IP アドレスを安全に適用しています。すべてのサンプルに、`assignment: static` ネットワーク設定が含まれています。

デザイン

クラウド テンプレート コードのサンプル

cloud-init コードのない
Linux マシンに固定 IP ア
ドレスを割り当てる場合

```
resources:
  wpnet:
    type: Cloud.Network
    properties:
      name: wpnet
      networkType: public
      constraints:
        - tag: sqa
  DBTier:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      image: linux-template
      networks:
        - name: '${wpnet.name}'
          assignment: static
          network: '${resource.wpnet.id}'
```

ネットワーク割り当てコ
マンドを含まない cloud-
init コードを使用して
Linux マシンに固定 IP ア
ドレスを割り当てる場合。
注 : vSphere カスタマイ
ズ仕様は、
customizeGuestOs プ
ロパティを True に設定
しても省略しても適用さ
れます。

Ubuntu のサンプル

```
resources:
  wpnet:
    type: Cloud.Network
    properties:
      name: wpnet
      networkType: public
      constraints:
        - tag: sqa
  DBTier:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      image: ubuntu-template
      customizeGuestOs: true
      cloudConfig: |
        #cloud-config
        ssh_pwauth: yes
        chpasswd:
          list: |
            root:Pa$$w0rd
            expire: false
        write_files:
          - path: /tmpFile.txt
            content: |
              ${resource.wpnet.dns}
      runcmd:
        - hostnamectl set-hostname --pretty $
        {self.resourceName}
        - touch /etc/cloud/cloud-init.disabled
      networks:
        - name: '${wpnet.name}'
          assignment: static
          network: '${resource.wpnet.id}'
```

CentOS のサンプル

```
resources:
  wpnet:
    type: Cloud.Network
    properties:
```

デザイン

クラウド テンプレート コードのサンプル

```
name: wpnet
networkType: public
constraints:
  - tag: sqa
DBTier:
  type: Cloud.vSphere.Machine
  properties:
    flavor: small
    image: centos-template
    customizeGuestOs: true
    cloudConfig: |
      #cloud-config
      write_files:
        - path: /test.txt
          content: |
            deploying in power off.
            then rebooting.
  networks:
    - name: '${wpnet.name}'
      assignment: static
      network: '${resource.wpnet.id}'
```

デザイン

クラウド テンプレート コードのサンプル

ネットワーク割り当てコマンドを含む cloud-init コードを使用して Linux マシンに固定 IP アドレスを割り当てる場合。customizeGuestOs プロパティは、False にする必要があります。

Ubuntu のサンプル

```
resources:
  wpnet:
    type: Cloud.Network
    properties:
      name: wpnet
      networkType: public
      constraints:
        - tag: sqa
  DBTier:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      image: ubuntu-template
      customizeGuestOs: false
      cloudConfig: |
        #cloud-config
        write_files:
          - path: /etc/netplan/99-installer-
            config.yaml
            content: |
              network:
                version: 2
                renderer: networkd
                ethernet:
                  ens160:
                    addresses:
                      - $
                        {resource.DBTier.networks[0].address}/$
                        {resource.wpnet.prefixLength}
                    gateway4: $
                        {resource.wpnet.gateway}
                    nameservers:
                      search: $
                        {resource.wpnet.dnsSearchDomains}
                      addresses: ${resource.wpnet.dns}
        runcmd:
          - netplan apply
          - hostnamectl set-hostname --pretty $
            {self.resourceName}
          - touch /etc/cloud/cloud-init.disabled
        networks:
          - name: '${wpnet.name}'
            assignment: static
            network: '${resource.wpnet.id}'
```

CentOS のサンプル

```
resources:
  wpnet:
    type: Cloud.Network
    properties:
      name: wpnet
      networkType: public
      constraints:
        - tag: sqa
  DBTier:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      image: centos-template
```

デザイン

クラウド テンプレート コードのサンプル

```

customizeGuestOs: false
cloudConfig: |
  #cloud-config
  ssh_pwauth: yes
  chpasswd:
    list: |
      root:VMware1!
    expire: false
  runcmd:
    - nmcli con add type ethernet con-name
      'custom ens192' ifname ens192 ip4 $
      {self.networks[0].address}/$
      {resource.wpnet.prefixLength} gw4 $
      {resource.wpnet.gateway}
    - nmcli con mod 'custom ens192' ipv4.dns "$
      {join(resource.wpnet.dns, ' ')}"
    - nmcli con mod 'custom ens192' ipv4.dns-
      search "${join(resource.wpnet.dnsSearchDomains, ',')}"
    - nmcli con down 'System ens192' ; nmcli
      con up 'custom ens192'
    - nmcli con del 'System ens192'
    - hostnamectl set-hostname --static `dig -x
      ${self.networks[0].address} +short | cut -d "." -f 1`
    - hostnamectl set-hostname --pretty $
      {self.resourceName}
    - touch /etc/cloud/cloud-init.disabled
  networks:
    - name: '${wpnet.name}'
      assignment: static
      network: '${resource.wpnet.id}'

```

参照イメージを基に展開を行うときに、ネットワーク割り当てコマンドを含む cloud-init コードを使用して Linux マシンに固定 IP アドレスを割り当てる場合。

customizeGuestOs プロパティは、False にする必要があります。

また、カスタマイズをブロックする ovfProperties プロパティをクラウド テンプレートに含めることはできません。

```

resources:
  wpnet:
    type: Cloud.Network
    properties:
      name: wpnet
      networkType: public
      constraints:
        - tag: sqa
  DBTier:
    type: Cloud.vSphere.Machine
    properties:
      flavor: small
      imageRef: 'https://cloud-images.ubuntu.com/
        releases/focal/release/ubuntu-20.04-server-cloudimg-
        amd64.ova'
      customizeGuestOs: false
      cloudConfig: |
        #cloud-config
        ssh_pwauth: yes
        chpasswd:
          list: |
            root:Pa$$w0rd
            ubuntu:Pa$$w0rd
          expire: false
        write_files:
          - path: /etc/netplan/99-netcfg-vrac.yaml
            content: |
              network:
                version: 2
                renderer: networkd

```

デザイン

クラウド テンプレート コードのサンプル

```

ethernets:
  ens192:
    dhcp4: no
    dhcp6: no
    addresses:
      - $
    {resource.DBTier.networks[0].address}/$
    {resource.wpnet.prefixLength}
    gateway4: $
    {resource.wpnet.gateway}
    nameservers:
      search: $
    {resource.wpnet.dnsSearchDomains}
    addresses: ${resource.wpnet.dns}
  runcmd:
    - netplan apply
    - hostnamectl set-hostname --pretty $
    {self.resourceName}
    - touch /etc/cloud/cloud-init.disabled
  networks:
    - name: '${wpnet.name}'
      assignment: static
      network: '${resource.wpnet.id}'

```

機能しない、または望ましくない結果が発生する可能性があるデザイン

- cloud-init コードにネットワーク割り当てコマンドが含まれておらず、customizeGuestOs プロパティが False である。

ネットワーク設定を構成するための初期化コマンドもカスタマイズ仕様もない。

- cloud-init コードにネットワーク割り当てコマンドが含まれておらず、ovfProperties プロパティが設定されている。

初期化コマンドはなく、ovfProperties がカスタマイズ仕様をブロックした。

- cloud-init コードにネットワーク割り当てコマンドが含まれており、customizeGuestOs プロパティが見つからないか True に設定されている。

カスタマイズ仕様を適用すると初期化コマンドと競合する。

cloud-init およびカスタマイズ仕様に関するその他の回避策

vSphere に展開する際には、cloud-init とカスタマイズ仕様の競合を回避するようにイメージをカスタマイズすることもできます。詳細については、次の外部リポジトリを参照してください。

- [vSphere イメージ準備スクリプト](#)

vRealize Automation Cloud Assembly の展開が初期化を待機する方法

vRealize Automation Cloud Assembly の展開に進む前に、1 台の仮想マシンを完全に起動することが必要な場合があります。

たとえば、展開しているマシンでパッケージのインストールと Web サーバの起動がまだ実行の途中である場合、アプリケーションが使用可能になる前に行動の速いユーザーがアプリケーションにアクセスしていることが考えられます。

この機能を使用する場合は、次の考慮事項に注意してください。

- この機能は、[cloud-init phone_home](#) モジュールを使用しており、Linux マシンを展開するときに使用できます。
- [Cloudbase-init](#) の制限のために、Windows では [phone_home](#) は使用できません。
- [phone_home](#) は、明示的な依存関係と同様に展開順序に影響することがありますが、タイミングと処理のオプションの点で明示的な依存関係よりも柔軟です。

[vRealize Automation Cloud Assembly でリソースの展開順序を設定する方法](#)を参照してください。

- [phone_home](#) を使用するには、クラウド テンプレート内に [cloudConfig](#) セクションが必要です。
- アイデア次第で用途が広がります。初期化コマンドには、[phone_home](#) と連携して使用できる、操作間の組み込み待機時間が含まれる場合があります。
- マシン テンプレートに [phone_home](#) モジュールの設定がすでに含まれている場合、クラウド テンプレートベースの [phone_home](#) は機能しません。
- マシンには、vRealize Automation Cloud Assembly への送信通信アクセスが必要です。

vRealize Automation Cloud Assembly で [phone_home](#) を使用してマシンの初期化を待機するには、クラウド テンプレートに [cloudConfigSettings](#) セクションを追加します。

```
cloudConfigSettings:
  phoneHomeShouldWait: true
  phoneHomeTimeoutSeconds: 600
  phoneHomeFailOnTimeout: true
```

プロパティ	説明
phoneHomeShouldWait	初期化を待機するかどうかを true または false で指定します。
phoneHomeTimeoutSeconds	初期化の実行中でも展開に進むかどうかを決定するタイミング。デフォルトは 10 分です。
phoneHomeFailOnTimeout	タイムアウト後に展開に進むかどうかを true または false で指定します。進んだ場合でも、別の理由で展開が失敗する可能性があります。

Windows ゲストのカスタマイズを実行する方法

vRealize Automation Cloud Assembly で展開時に Windows マシンを自動的に初期化するには、Cloudbase-Init をサポートするイメージを準備してから、適切なコマンドを含むクラウド テンプレートを準備します。

イメージの作成プロセスは、クラウド ベンダーによって異なります。ここで示す例は、vSphere 用です。

vSphere の初期化可能な Windows イメージを作成する方法

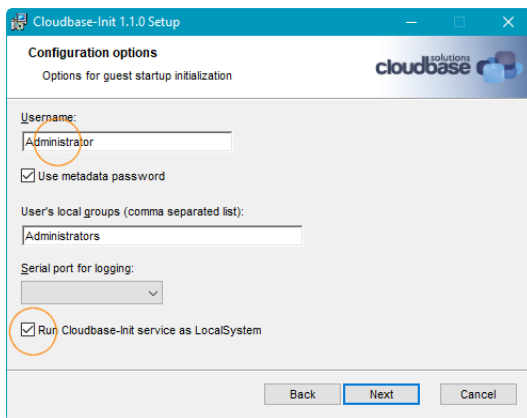
vRealize Automation Cloud Assembly で、vSphere に展開された Windows マシンを初期化するには、Cloudbase-Init をインストールして設定を行ったテンプレートが必要です。

- 1 vSphere を使用して Windows 仮想マシンを作成し、パワーオンします。
- 2 この仮想マシンで、Windows にログインします。
- 3 Cloudbase-Init をダウンロードします。

<https://cloudbase.it/cloudbase-init/#download>

- 4 Cloudbase-Init セットアップ用の .msi ファイルを起動します。

インストールの際、ユーザー名として [Administrator] を入力し、実行オプションとして LocalSystem を選択します。



他の設定項目はデフォルト値のままにすることができます。

- 5 インストールの実行を許可します。ただし、セットアップ ウィザードの最後の完了画面は閉じないでください。

重要： セットアップ ウィザードの最後の画面は閉じないでください。

- 6 セットアップ ウィザードの完了画面が開いた状態で、Windows を使用して Cloudbase-Init のインストールパスに移動し、次のファイルをテキスト エディタで開きます。

```
conf\cloudbase-init-unattend.conf
```

- 7 図に示すように、metadata_services を OvfService に設定します。

```
metadata_services=cloudbaseinit.metadata.services.ovfservice.OvfService
```

- 8 cloudbase-init-unattend.conf を保存して閉じます。

- 9 同じフォルダにある次のファイルをテキスト エディタで開きます。

```
conf\cloudbase-init.conf
```

- 10 図に示すように、`first_logon_behaviour`、`metadata_services`、および `plugins` を設定します。

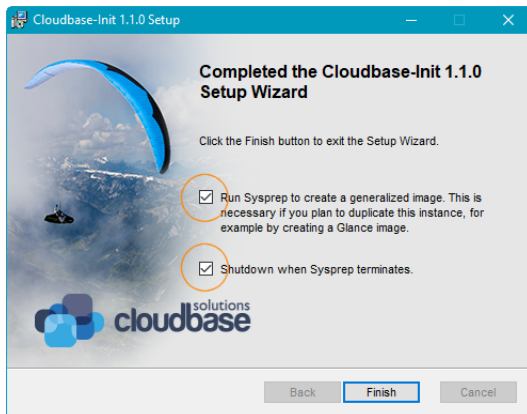
```
first_logon_behaviour=always
...
metadata_services=cloudbaseinit.metadata.services.ovfservice.OvfService
...
plugins=cloudbaseinit.plugins.windows.createuser.CreateUserPlugin,cloudbaseinit.plugins.win
dows.setuserpassword.SetUserPasswordPlugin,cloudbaseinit.plugins.common.sshpublickeys.SetUs
erSSHPublicKeysPlugin,cloudbaseinit.plugins.common.userdata.UserDataPlugin
...
```

- 11 `cloudbase-init.conf` を保存して閉じます。
- 12 セットアップ ウィザードの完了画面で、Sysprep を実行するオプションと、Sysprep の実行後にシャットダウンするオプションを選択し、[終了] をクリックします。

注： Sysprep を実行すると、イメージの展開が機能しなくなる場合があります。

展開するときに、vRealize Automation Cloud Assembly は動的に生成されるカスタマイズ仕様を適用します。これによって、ネットワーク インターフェイスが切断されます。イメージ内で Sysprep が保留中の状態になると、カスタマイズ仕様は失敗し、展開が切断されたままになることがあります。

環境内でこの問題が発生している可能性がある場合は、イメージを作成するときに Sysprep オプションを無効なままにしておきます。



- 13 仮想マシンをシャットダウンした後で、vSphere を使用してテンプレートに変換します。

設定についての説明

次の表は、セットアップ中に行った設定内容を説明したものです。

設定	目的
ユーザー名、CreateUserPlugin、および SetUserPasswordPlugin	Sysprep 実行後、初めて Cloudbase-Init を起動すると、CreateUserPlugin によって、ユーザー名の管理者アカウントが、パスワードが空の状態で作成されます。空のパスワードは、SetUserPasswordPlugin によって、クラウド テンプレートに含めるリモート アクセス パスワードに変更することができます。
最初のログイン動作	この設定により、初回ログイン時にパスワードの変更を求めるプロンプトがユーザーに表示されます。

設定	目的
メタデータ サービス	Cloudbase-Init は、OvfService のみを一覧表示し、vCenter Server でサポートされていないその他のメタデータ サービスの検出を行いません。その結果、よりクリーンなログ ファイルが生成されます。これは他のサービスの検出を行うと、検出に失敗したというエントリがログに記録されるためです。
プラグイン	Cloudbase-Init は、OvfService でサポートされている機能を備えたプラグインのみを一覧表示するため、ログはクリーンな状態になります。Cloudbase-Init は、指定された順序でプラグインを実行します。
LocalSystem として実行	一部の高度な初期化コマンドでは、Cloudbase-Init を専用の管理者アカウントで実行することが必要になります。この設定は、そのようなコマンドをサポートします。

クラウド テンプレートに Cloudbase-Init コマンドを含める方法

Windows マシンを初期化するには、vRealize Automation Cloud Assembly でインフラストラクチャとクラウド テンプレートを作成し、初期化可能な Windows イメージで必要なコマンドが実行されるようにします。

ここに示す例は vSphere に基づいていますが、ほかのクラウド ベンダーでも同様です。

前提条件

- インフラストラクチャを作成します。vRealize Automation Cloud Assembly で、vSphere クラウド アカウントおよび関連付けられているクラウド ゾーンを追加します。
- フレーバーとイメージ マッピングを追加し、さらにネットワークとストレージ プロファイルを追加します。
インフラストラクチャ内のイメージ マッピングでは、Cloudbase-Init をサポートするために作成した Windows テンプレートを参照する必要があります。[vSphere の初期化可能な Windows イメージを作成する方法](#) を参照してください。
このテンプレートがリストにない場合は、クラウド アカウントに移動して、イメージを同期します。同期を行わない場合でも、24 時間ごとに自動同期が実行されます。
- プロジェクトを追加し、ユーザーを追加します。ユーザーが自分のクラウド ゾーンにプロビジョニングできるようにしてください。

インフラストラクチャとプロジェクトの作成の詳細については、[チュートリアル：vRealize Automation Cloud Assembly でのマルチクラウド インフラストラクチャおよび展開のセットアップとテスト](#)の例を参照してください。

手順

- 1 vRealize Automation Cloud Assembly で [デザイン] タブに移動して、新規のクラウド テンプレートを作成します。
- 2 必要な Cloudbase-init コマンドを含む cloudConfig セクションを追加します。

次のコマンドの例では、Windows の c: ドライブに新しいファイルを作成し、ホスト名を設定しています。

```
resources:
  Cloud_Machine_1:
    type: Cloud.Machine
    properties:
```

```

image: cloudbase-init-win-2016
flavor: small
remoteAccess:
  authentication: usernamePassword
  username: Administrator
  password: Password1234@$$
cloudConfig: |
  #cloud-config
  write_files:
    content: Cloudbase-Init test
    path: C:\test.txt
  set_hostname: testname

```

詳細については、[Cloudbase-init のドキュメント](#)を参照してください。

- 3 Windows への初回ログイン用にマシンを構成できるように、remoteAccess プロパティを追加します。

テンプレートを作成した際に説明したように、メタデータ サービスはログイン認証情報を取得し、CreateUserPlugin と SetUserPasswordPlugin に公開します。パスワードは Windows のパスワードの要件を満たしている必要があります。

- 4 vRealize Automation Cloud Assembly から、クラウド テンプレートをテストして展開します。
- 5 展開後、Windows RDP およびテンプレートの認証情報を使用して新しい Windows マシンにログインし、カスタマイズを確認します。

上の例では、C:\test.txt ファイルを検索し、ホスト名のシステム プロパティをチェックします。

vRealize Automation Cloud Assembly クラウド テンプレートで使用するカスタム リソース タイプを作成する方法

vRealize Automation Cloud Assembly でクラウド テンプレートを作成する場合、リソース タイプ パレットには、サポートされているクラウド アカウントおよび統合エンドポイントのリソース タイプが含まれます。リソース タイプの拡張されたリストに基づいてクラウド テンプレートを作成することが必要な場合があります。カスタム リソースを作成してデザイン キャンバスに追加し、デザインおよび展開のニーズをサポートするクラウド テンプレートを作成できます。

vRealize Orchestrator を使用してカスタム リソースを作成する

各カスタム リソースは vRealize Orchestrator SDK インベントリ タイプに基づいており、必要な SDK タイプのインスタンスを出力する vRealize Orchestrator ワークフローによって作成されます。Properties、Date、string、number などのプリミティブ タイプは、カスタム リソースの作成ではサポートされていません。

注： SDK オブジェクト タイプは、プラグイン名とタイプ名の分離に使用されるコロン（「:」）によって、他のプロパティ タイプと区別できます。たとえば、AD:UserGroup は、Active Directory ユーザー グループの管理に使用される SDK オブジェクト タイプです。

vRealize Orchestrator の組み込みワークフローを使用することも、独自のワークフローを作成することもできます。Anything-as-a-Service (XaaS) ワークフローを作成するために vRealize Orchestrator を使用すると、展開時に Active Directory ユーザーをマシンに追加するクラウド テンプレートを作成したり、カスタムの F5 ロードバランサを展開に追加したりすることができます。

カスタム リソース名とリソース タイプ

クラウド テンプレート リソース タイプ パレット内では、カスタム リソース名でカスタム リソースを識別します。

カスタム リソースのリソース タイプは **Custom.** で始まる必要があり、各リソース タイプは一意にする必要があります。たとえば、Active Directory ユーザーを追加するカスタム リソースのリソース タイプとして `Custom.ADUser` を設定することができます。テキスト ボックスに **Custom.** が含まれているかどうかは検証されませんが、この文字列は削除しても自動的に追加されます。

外部タイプ

外部タイプ プロパティは、カスタム リソースのタイプを定義します。vRealize Automation Cloud Assembly のカスタム リソースで作成ワークフローを選択すると、その下に外部タイプのドロップダウンが表示されます。ドロップダウンには、vRealize Orchestrator ワークフローの出力パラメータから選択された外部タイプのプロパティが含まれています。ドロップダウンに含まれる選択されたワークフローの出力プロパティは、`VC:VirtualMachine` や `AD:UserGroup` などの非アレイの SDK オブジェクト タイプである必要があります。

注： 動的タイプのプラグインを使用するカスタム ワークフローを作成する場合は、`DynamicTypesManager.getObject()` メソッドを使用して変数が作成されていることを確認します。

カスタム リソースを定義するときは、選択した外部タイプの可用性の範囲も定義します。選択した外部タイプは次のように定義できます。

- プロジェクト間で共有。
- 選択したプロジェクトでのみ使用可能。

外部タイプは、定義された範囲ごとに1つのみ指定できます。たとえば、プロジェクトに外部タイプとして `VC:VirtualMachine` を使用するカスタム リソースを作成した場合、同じ外部タイプを使用する同じプロジェクトに対して別のカスタム リソースを作成することはできません。また、同じ外部タイプを使用する2つの共有カスタム リソースを作成することもできません。

ワークフローの入力/出力の検証

作成、削除、更新ワークフローをカスタム リソースにライフサイクル アクションとして追加すると、vRealize Automation Cloud Assembly は、選択したワークフローに適切な入力および出力プロパティ定義があることを検証します。

- 作成ワークフローには、`SSH:Host` や `SQL:Database` などの SDK オブジェクト タイプの出力パラメータが必要です。選択したワークフローが検証で正常と判定されなかった場合、更新または削除ワークフローを追加したり、変更をカスタム リソースに保存したりすることはできません。
- 削除ワークフローには、カスタム リソースの外部タイプと一致する SDK オブジェクト タイプの入力パラメータが必要です。
- 更新ワークフローには、カスタム リソースの外部タイプと一致する SDK オブジェクト タイプの入力および出力パラメータの両方が必要です。

カスタム リソース プロパティ スキーマ

vRealize Orchestrator ワークフローをカスタム リソースに追加すると、入力および出力パラメータがプロパティとして追加されます。カスタム リソース プロパティ スキーマを表示するには、[プロパティ] タブを選択します。スキーマには、名前、データ タイプ、プロパティ タイプに加え、利用できる場合は指定されたプロパティの説明も含まれています。スキーマは、指定されたプロパティが必須またはオプションのいずれであるかも定義します。

ユーザーを Active Directory に追加するクラウド テンプレートを vRealize Automation Cloud Assembly で作成する方法

クラウド テンプレートの作成時に使用する vRealize Automation Cloud Assembly クラウド テンプレート リソースの他に、独自のカスタム リソースを作成することもできます。

カスタム リソースは、定義されたメイン リソース操作ワークフローを使用して、vRealize Automation で管理する vRealize Orchestrator オブジェクトです。クラウド テンプレート サービスは、作成または削除操作がトリガされると、適切な vRealize Orchestrator ワークフローを自動で呼び出します。リソース タイプの機能は、Day 2 操作として使用可能な vRealize Orchestrator ワークフローを選択することでも拡張できます。

この使用事例では、vRealize Orchestrator ライブラリで提供されている組み込みのワークフローを使用します。この使用事例には、プロセスの実行方法を示すための規範的な値または文字列が含まれています。これらは環境に合わせて変更できます。

参考として、この使用事例では [DevOpsTesting] という名前のプロジェクトを使用します。このサンプル プロジェクトは、環境内の任意のプロジェクトに置き換えることができます。

前提条件

- vRealize Orchestrator 統合が構成されていることを確認します。[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。
- 作成、更新、破棄、およびインストール後アクションに使用しているワークフローが vRealize Orchestrator に存在し、そこから正常に実行されていることを確認します。
- vRealize Orchestrator で、ワークフローによって使用されるリソース タイプを特定します。このカスタム リソースに含まれているワークフローは、すべてが同じリソース タイプを使用する必要があります。この使用事例では、リソース タイプは AD:User です。リソース タイプの検証については、[vRealize Automation Cloud Assembly クラウド テンプレートで使用するカスタム リソース タイプを作成する方法](#)を参照してください。
- vRealize Orchestrator 統合で組み込みの Active Directory ワークフローを使用して、Active Directory サーバを構成します。
- マシン クラウド テンプレートを構成および展開する方法について理解していることを確認します。

手順

- 1 グループにユーザーを追加するための Active Directory カスタム リソースを作成します。

この手順では、カスタム リソースをリソース タイプとしてクラウド テンプレート デザイン キャンバスに追加します。

- a vRealize Automation Cloud Assembly で、[デザイン] - [カスタム リソース] の順に選択し、[新しいカスタム リソース] をクリックします。
- b 次の値を指定します。

ワークフロー名を除き、これらはサンプル値です。

設定	サンプルの値
名前	AD:User これは、クラウド テンプレートのリソース タイプ パレットに表示される名前です。
リソース タイプ	Custom.ADUser リソース タイプは Custom. で始まる必要があり、各リソース タイプは一意にする必要があります。 テキスト ボックスに Custom. が含まれているかどうかは検証されませんが、この文字列は削除しても自動的に追加されます。 このリソース タイプは、リソース タイプ パレットに追加されるため、クラウド テンプレート内で使用できます。

- c クラウド テンプレートのリソース タイプ リストでこのリソース タイプを有効にするには、[有効化] オプションのトグル ボタンがオンになっていることを確認します。
- d リソース タイプを任意のプロジェクトで使用できるようにする [範囲] 設定を選択します。

- e リソースと Day 2 アクションを定義するワークフローを構成します。

注： 選択した Day 2 ワークフローには、外部タイプと同じタイプの入力パラメータが必要です。外部タイプの入力、カスタム リソースに自動的にバインドされるため、ユーザーによって申請される Day 2 カスタム フォームには表示されません。

設定	サンプルの値
ライフサイクル アクション - 作成	<p>[パスワードを使用した組織単位のユーザーの作成] ワークフローを選択します。</p> <p>複数の vRealize Orchestrator 統合がある場合は、これらのカスタム リソースの実行に使用する統合インスタンスのワークフローを選択します。</p> <p>ワークフローを選択すると、外部タイプ ドロップダウンメニューが使用可能になり、自動的に AD:User に設定されます。</p> <p>注： 外部ソース タイプは、共有している場合に 1 回のみ使用することも、プロジェクトごとに 1 回ずつ使用することもできます。この使用事例では、すべてのプロジェクトに同じカスタム リソースを提供しています。これは、すべてのプロジェクトの他のすべてのリソース タイプに対して AD:User が使用できないことを意味します。AD:User タイプを必要とする他のワークフローがある場合は、プロジェクトごとに個別のカスタム リソースを作成する必要があります。</p>
ライフサイクル アクション - 破棄	[ユーザーの削除] ワークフローを選択します。
その他のアクション	<p>[ユーザー パスワードの変更] ワークフローを選択します。</p> <p>アクションの申請時にユーザーが応答するアクション申請フォームを変更するには、[要求パラメータ] 列のアイコンをクリックします。</p> <p>注： 追加のアクション ワークフローでは、ワークフローに外部タイプと同じタイプの入力パラメータがあることを確認します。</p>

この例では、更新ワークフローの適切なアプリケーションがありません。プロビジョニングされたカスタム リソースに変更を加える更新ワークフローの一般的な例として、展開のスケールインやスケールアウトがあります。

- f [プロパティ] タブでスキーマ キーとタイプの値を確認して、ワークフローの入力を理解し、クラウド テンプレートで構成できるようにします。

スキーマには、ワークフローで定義されている必須およびオプションの入力値が一覧表示されます。必須の入力値は、クラウド テンプレート YAML に含まれています。

ユーザーの作成ワークフローの場合、必須の値は、accountName、displayName、ouContainer です。他のスキーマ プロパティは必須ではありません。スキーマを使用して、他のフィールド値、ワークフロー、またはアクションへのバインドを作成する場所を決定することもできます。この使用事例には、バインドは含まれていません。

- g カスタムリソースの作成を完了するには、[作成] をクリックします。

2 展開時にユーザーをマシンに追加するクラウド テンプレートを作成します。

- a [デザイン] - [クラウド テンプレート] の順に選択し、[新規作成元] - [空白のキャンバス] の順にクリックします。
- b クラウド テンプレートに「**AD ユーザーを持つマシン**」という名前を付けます。
- c [DevOpsTesting] プロジェクトを選択し、[作成] をクリックします。
- d vSphere マシンを追加して構成します。
- e クラウド テンプレートのデザイン画面の左側にあるカスタム リソース リストから、[AD user] リソース タイプをキャンバスにドラッグします。

注： カスタム リソースは、スクロールして左側のペインで選択するか、[リソース タイプの検索] テキスト ボックスで検索して選択します。カスタム リソースが表示されない場合は、[リソース タイプの検索] テキスト ボックスの横にある更新ボタンをクリックします。

- f 右側で YAML コードを編集して、必須の入力値とパスワードを追加します。

追加するユーザーの名前をユーザーが指定できるように、`inputs` セクションをコードに追加します。次の例では、これらの値の一部はサンプル データです。実際の値は異なる場合があります。

```
inputs:
  accountName:
    type: string
    title: Account name
    encrypted: true
  displayName:
    type: string
    title: Display name
  password:
    type: string
    title: Password
    encrypted: true
  confirmPassword:
    type: string
    title: Password
    encrypted: true
  ouContainer:
    type: object
    title: AD OU container
    $data: 'vro/data/inventory/AD:OrganizationalUnit'
  properties:
    id:
      type: string
    type:
      type: string
```

- g `resources` セクションで、ユーザーの選択を求めるプロンプトを表示するために、`${input.input-name}` コードを追加します。

```
resources:
  Custom_ADUser_1:
    type: Custom.ADUser
    properties:
      accountName: '${input.accountName}'
      displayName: '${input.displayName}'
      ouContainer: '${input.ouContainer}'
      password: '${input.password}'
      confirmPassword: '${input.confirmPassword}'
```

3 クラウド テンプレートを展開します。

- a クラウド テンプレート デザイナ画面で、[展開] をクリックします。
- b [展開名] に「**AD User Scottt**」と入力します。
- c [クラウド テンプレートのバージョン] を選択し、[次へ] をクリックします。

- d 展開の入力を完了します。
 - e [[展開]] をクリックします。
- 4 プロビジョニング プロセスを監視して、ユーザーが Active Directory に追加されていることを確認します。
- a [展開] をクリックして、[AD User Scott] 展開を見つけます。
 - b 申請のステータスを監視し、展開が正常に完了したことを確認します。
 - c パスワードの変更アクションが使用可能で、動作していることを確認します。

次のステップ

テストしたクラウド テンプレートが動作していれば、他のクラウド テンプレートで [AD user] のカスタム リソースの使用を開始できます。

Cloud Assembly で SSH を含むクラウド テンプレートを作成する方法

vRealize Orchestrator ワークフローを使用してクラウド テンプレートを作成する際に使用できるカスタム リソースを作成できます。この使用事例では、SSH ホストを追加するカスタム リソースを追加します。後で、クラウド テンプレートにリソースを含めることができます。さらに、更新ワークフローを追加して、ユーザーが、Day 2 アクションを個別に実行するのではなく、展開後に SSH 構成に変更を加えることができるようにします。

カスタム リソースは、定義されたメイン リソース操作ワークフローを使用して、vRealize Automation で管理する vRealize Orchestrator オブジェクトです。クラウド テンプレート サービスは、作成または削除操作がトリガされると、適切な vRealize Orchestrator ワークフローを自動で呼び出します。リソース タイプの機能は、Day 2 操作として使用可能な vRealize Orchestrator ワークフローを選択することでも拡張できます。

この使用事例では、vRealize Orchestrator ライブラリで提供されている組み込みのワークフローを使用します。この使用事例には、プロセスの実行方法を示すための規範的な値または文字列が含まれています。これらは環境に合わせて変更できます。

参考として、この使用事例では [DevOpsTesting] という名前のプロジェクトを使用します。プロジェクトは、既存のプロジェクトに置き換えることができます。

前提条件

- vRealize Orchestrator 統合が構成されていることを確認します。[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。
- 作成、更新、破棄、およびインストール後アクションに使用しているワークフローが vRealize Orchestrator に存在し、そこから正常に実行されていることを確認します。
- vRealize Orchestrator で、ワークフローによって使用されるリソース タイプを特定します。このカスタム リソースに含まれているワークフローは、すべてが同じリソース タイプを使用する必要があります。この使用事例では、リソース タイプは SSH:Host です。リソース タイプの検証については、[vRealize Automation Cloud Assembly クラウド テンプレートで使用するカスタム リソース タイプを作成する方法](#)を参照してください。
- マシン クラウド テンプレートを構成および展開する方法について理解していることを確認します。

手順

- 1 SSH をクラウド テンプレートに追加するための SSH ホストのカスタム リソースを作成します。

この手順では、カスタム リソースをリソース タイプとしてクラウド テンプレート デザイン キャンバスに追加します。

- a vRealize Automation Cloud Assembly で、[デザイン] - [カスタム リソース] の順に選択し、[新しいカスタム リソース] をクリックします。
- b 次の値を指定します。

ワークフロー名を除き、これらはサンプル値です。

表 6-2.

設定	サンプルの値
名前	SSH Host - DevOpsTesting Project これは、クラウド テンプレートのリソース タイプ パレットに表示される名前です。
リソース タイプ	Custom.SSHHost リソース タイプは Custom. で始まる必要があり、各リソース タイプは一意にする必要があります。 テキスト ボックスに Custom. が含まれているかどうかは検証されませんが、この文字列は削除しても自動的に追加されます。 このリソース タイプは、デザイン キャンバスに追加されるため、クラウド テンプレート内で使用できます。

- c クラウド テンプレートのリソース タイプ リストでこのリソース タイプを有効にするには、[有効化] オプションのトグル ボタンがオンになっていることを確認します。
- d リソース タイプを [DevOpsTesting] プロジェクトで使用できるようにする [範囲] 設定を選択します。
- e リソースを定義するワークフローを選択します。

設定	設定
ライフサイクル アクション - 作成	[SSH ホストの追加] ワークフローを選択します。 複数の vRealize Orchestrator 統合がある場合は、これらのカスタム リソースの実行に使用する統合インスタンスのワークフローを選択します。 ワークフローを選択すると、外部タイプ ドロップダウンメニューが使用可能になり、自動的に SSH:Host に設定されます。外部ソース タイプは、共有している場合に 1 回のみ使用することも、プロジェクトごとに 1 回ずつ使用することもできます。この使用事例では、[DevOpsTesting] プロジェクトに対してのみカスタム リソースを提供しています。SSH:Host タイプを必要とする他のワークフローがある場合は、プロジェクトごとに個別のカスタム リソースを作成する必要があります。
ライフサイクル アクション - 更新	[SSH ホストの更新] ワークフローを選択します。
ライフサイクル アクション - 破棄	[SSH ホストの削除] ワークフローを選択します。

- f [プロパティ] タブでスキーマ キーとタイプの値を確認して、ワークフローの入力を理解し、クラウド テンプレートで構成できるようにします。

スキーマには、ワークフローで定義されている必須およびオプションの入力値が一覧表示されます。必須の入力値は、クラウド テンプレート YAML に含まれています。

[SSH ホストの追加] ワークフローでは、hostname、port、username が必須の入力値です。他のスキーマ プロパティは必須ではありません。スキーマを使用して、他のフィールド値、ワークフロー、またはアクションへのバインドを作成する場所を決定することもできます。この使用事例には、バインドは含まれていません。

- g カスタムリソースの作成を完了するには、[作成] をクリックします。

2 展開時に SSH ホストを追加するクラウド テンプレートを作成します。

- a [デザイン] - [クラウド テンプレート] の順に選択し、[新規作成元] - [空白のキャンバス] の順にクリックします。
- b クラウド テンプレートに「SSH ホストを持つマシン」という名前を付けます。
- c [DevOpsTesting] プロジェクトを選択し、[作成] をクリックします。
- d vSphere マシンを追加して構成します。
- e クラウド テンプレートのデザイン画面の左側にあるカスタム リソース リストから、[SSH Host - DevOpsTesting Project] リソース タイプをキャンバスにドラッグします。

注： カスタム リソースは、スクロールして左側のペインで選択するか、[リソース タイプの検索] テキスト ボックスで検索して選択します。カスタム リソースが表示されない場合は、[リソース タイプの検索] テキスト ボックスの横にある更新ボタンをクリックします。

このリソース タイプがこのプロジェクト用に設定されているため、使用可能であることを示すリマインダです。別のプロジェクト用のクラウド テンプレートを作成している場合、このリソース タイプは表示されません。

- f 右側で YAML コードを編集して、必須の入力値を追加します。

展開時にユーザーがユーザー名とホスト名を指定できるように、`inputs` セクションをコードに追加します。この例では、デフォルト ポートは 22 です。次の例では、これらの値の一部はサンプル データです。実際の値は異なる場合があります。

```
inputs:
  hostname:
    type: string
    title: The hostname of the SSH Host
  username:
    type: string
    title: Username
```

- g `resources` セクションで、ユーザーの選択を求めるプロンプトを表示するために、`${input.input-name}` コードを追加します。

```
resources:
  Custom_SSHTHost_1:
    type: Custom.SSHTHost
    properties:
      port: 22
      hostname: '${input.hostname}'
      username: '${input.username}'
```

3 クラウド テンプレートを展開します。

- a クラウド テンプレート デザイン画面で、[展開] をクリックします。
- b [展開名] に「SSH ホスト テスト」と入力します。
- c [クラウド テンプレートのバージョン] を選択し、[次へ] をクリックします。
- d 展開の入力を完了します。
- e [[展開]] をクリックします。

4 プロビジョニング プロセスを監視して、SSH ホストが展開に含まれていることを確認します。

- a [展開] をクリックして、[SSH ホスト] 展開を見つけます。
- b 申請のステータスを監視し、展開が正常に完了したことを確認します。

次のステップ

テストしたクラウド テンプレートが動作していれば、他のクラウド テンプレートで SSH Host のカスタム リソースの使用を開始できます。

Day 2 の変更に対する準備 vRealize Automation Cloud Assembly のための設計方法

vRealize Automation Cloud Assembly リソース タイプにすでに関連付けられている Day 2 アクションに加え、ユーザーが必要とする可能性があるカスタムの更新について事前に準備できる設計オプションがあります。

注意： 展開を変更するには、そのクラウド テンプレートを編集して再適用するか、Day 2 アクションを使用します。ただし、通常は、2 つの方法を混在させないでください。

通常、パワーオン/オフなどのライフサイクルの Day 2 変更は安全ですが、ディスクを追加するなどの他の変更を行う場合は注意が必要です。

たとえば、Day 2 アクションを行ってディスクを追加してから、クラウド テンプレートを再適用するという混在方式を実行すると、クラウド テンプレートによって Day 2 の変更が上書きされ、ディスクの削除やデータの消失が発生する可能性があります。

Day 2 の準備中に、クラウド テンプレート コードを直接使用することも、vRealize Automation Cloud Assembly 設計インターフェイスを使用することもできます。

- クラウド テンプレート コードの入力を使用すると、展開または展開されたリソースを更新する際に、インターフェイスで新しい値の入力を要求されます。
- vRealize Automation Cloud Assembly を使用して、vRealize Orchestrator のワークフローまたはアクションに基づいてカスタム アクションを設計することができます。カスタム アクションを実行すると、vRealize Orchestrator は展開または展開されたリソースに変更を加えます。

vRealize Automation の Day 2 更新でクラウド テンプレートの入力を使用する方法

クラウド テンプレートをデザインする場合、vRealize Automation の入力パラメータにより、Day 2 のユーザーは、最初の展開要求の選択項目を再入力できます。

注意： プロパティを変更すると、リソースが再作成される場合があります。たとえば、Cloud.Service.Azure.App.Service の下の `connection_string.name` を変更すると、既存のリソースが削除され、新しいリソースが作成されます。

インストール後の変更をサポートするように入力をデザインするときは、リソースを削除および再作成する入力を許可するかどうかを決定します。リソースを再作成するプロパティについては、[vRealize Automation リソースのプロパティについて](#) のスキーマ リンクにアクセスしてください。

入力の作成方法については、[vRealize Automation でのユーザー入力によるクラウド テンプレートのカスタマイズ方法](#)を参照してください。

特定のインストール後の例については、次のセクションを参照してください。

展開されたマシンを別のネットワークに移動する方法

展開とネットワークを維持しながら、vRealize Automation Cloud Assembly で展開したマシンを再配置することが必要になる場合があります。

たとえば、最初にテスト ネットワークに展開した後、本番ネットワークに移行することができます。ここで説明する手法を使用すると、クラウド テンプレートを事前に設計することで上記のような Day 2 アクションに対する準備ができます。マシンは移動されることに注意してください。削除された後で再展開されるものではありません。

この手順は、**Cloud.vSphere.Machine** リソースにのみ適用されます。vSphere に展開されているクラウドに依存しないマシンには使用できません。

前提条件

- vRealize Automation Cloud Assembly ネットワーク プロファイルには、マシンが接続するすべてのサブネットが含まれている必要があります。vRealize Automation Cloud Assembly でネットワークを確認するには、[インフラストラクチャ] - [構成] - [ネットワーク プロファイル] の順に移動します。

ネットワーク プロファイルは、ユーザーの適切な vRealize Automation Cloud Assembly プロジェクトの一部であるアカウントおよびリージョンに含まれている必要があります。

- 2 つのサブネットに異なるタグを付けます。次の例では、**test** と **prod** というタグ名を想定しています。
- 展開されたマシンの IP アドレス割り当てタイプは同じままにする必要があります。別のネットワークへの移行中に、固定から DHCP、またはその逆に変更することはできません。

手順

- 1 vRealize Automation Cloud Assembly で、[デザイン] に移動し、展開のクラウド テンプレートを作成します。
- 2 コードの inputs セクションに、ユーザーがネットワークを選択できるようにするエントリを追加します。

```
inputs:
  net-tagging:
    type: string
    enum:
      - test
      - prod
    title: Select a network
```

- 3 コードの resources セクションに、**Cloud.Network** を追加し、vSphere マシンを接続します。
- 4 **Cloud.Network** で、inputs の選択内容を参照する制約を作成します。

```
resources:
  ABCServer:
    type: Cloud.vSphere.Machine
    properties:
      name: abc-server
      . . .
      networks:
        - network: '${resource["ABCNet"].id}'
  ABCNet:
    type: Cloud.Network
    properties:
```

```
name: abc-network
...
constraints:
  - tag: '${input.net-tagging}'
```

- 5 設計を続行し、通常どおりに展開します。展開時、**test** または **prod** ネットワークの選択を求めるプロンプトが表示されます。
- 6 Day 2 の変更を行う必要がある場合は、[展開] に移動し、クラウド テンプレートに関連付けられている展開を特定します。
- 7 展開の右側で、[アクション] - [更新] の順にクリックします。
- 8 更新パネルでは、同じように **test** または **prod** ネットワークの選択を求めるプロンプトが表示されます。
- 9 ネットワークを変更するには、選択して [次へ] をクリックし、[送信] をクリックします。

仮想マシンを vMotion するための vRealize Automation Cloud Assembly カスタム アクションの作成方法

クラウド テンプレートを展開した後、展開を変更する Day 2 アクションを実行できます。vRealize Automation Cloud Assembly には多くのインストール後アクションが含まれていますが、他のアクションを提供することもできます。カスタム リソース アクションを作成し、ユーザーがインストール後アクションとして使用できるようにすることができます。

カスタム リソース アクションは、vRealize Orchestrator ワークフローに基づいています。

このカスタムのインストール後アクションの例は、作成プロセスを紹介することを目的としています。カスタム アクションを効果的に使用するには、必要なタスクを実行する vRealize Orchestrator ワークフローとアクションを作成する必要があります。

前提条件

- vRealize Orchestrator 統合が構成されていることを確認します。[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。
- インストール後アクションに使用するワークフローが vRealize Orchestrator に存在し、正常に実行されることを確認します。

手順

- 1 vMotion を使用してホストから別のホストに vSphere 仮想マシンを移動するカスタム リソース アクションを作成します。
 - a vRealize Automation Cloud Assembly で、[デザイン] - [リソース アクション] の順に選択し、[新しいリソース アクション] をクリックします。
 - b 次の値を指定します。

ワークフロー名を除き、これらはサンプル値です。

設定	サンプルの値
名前	vSphere_VM_vMotion これは、リソース アクションのリストに表示される名前です。
表示名	仮想マシンの移動 これは、展開アクション メニューに表示される名前です。

- c [有効化] オプションをクリックして、リソース タイプが一致するリソースのインストール後アクション メニューでこのアクションを有効にします。
- d インストール後アクションを定義するリソース タイプとワークフローを選択します。

設定	サンプルの値
リソース タイプ	[Cloud.vSphere.Machine] リソース タイプを選択します。 これは、クラウド テンプレート コンポーネントとして展開されるリソース タイプで、必ずしもクラウド テンプレートに含まれる内容ではありません。たとえば、クラウド テンプレート内にクラウドに依存しないマシンがある場合でも、そのマシンは、vCenter Server 上に展開されると、[Cloud.vSphere.Machine] となります。このアクションは展開されたタイプに適用されます。このため、カスタム アクションを定義するときは、クラウドに依存しないタイプを使用しないでください。 この例では、vMotion は vSphere マシンに対してのみ機能しますが、複数のリソース タイプに対してその他のアクションの実行が必要になる場合もあります。アクションは、リソース タイプごとに作成する必要があります。
ワークフロー	[vMotion での仮想マシンの移行] ワークフローを選択します。 複数の vRealize Orchestrator 統合がある場合は、これらのカスタム リソース アクションを実行するために使用する統合インスタンスのワークフローを選択します。

- 2 vRealize Orchestrator プロパティと vRealize Automation Cloud Assembly スキーマ プロパティのバインドを作成します。vRealize Automation Cloud Assembly の Day 2 アクションでは、次の 3 タイプのバインドがサポートされます。

バインド タイプ	説明
申請中	デフォルト値のバインド タイプ。選択すると、入力プロパティが申請フォームに表示され、その値はユーザーが申請時に指定する必要があります。
バインド アクション付き	<p>このオプションは、次のようなりファレンス タイプの入力に対してのみ使用できます。</p> <ul style="list-style-type: none"> ■ VC:VirtualMachine ■ VC:Folder <p>ユーザーはバインドを実行するアクションを選択します。選択したアクションは、必ず入力パラメータと同じタイプを返します。プロパティの定義の正しい形式は <code>\$(properties.someProperty)</code> です。</p>
直接	このオプションは、プリミティブ データ タイプを使用する入力プロパティで使用できます。選択すると、プロパティが適切なタイプで、入力プロパティのスキーマから直接マッピングされます。ユーザーはスキーマ ツリーからプロパティを選択します。タイプが異なるプロパティは無効になります。

この使用事例では、バインドは、ワークフローで使用される vRealize Orchestrator の VC:VirtualMachine 入力タイプと vRealize Automation Cloud Assembly の Cloud.vSphere.Machine リソース タイプ間の接続を確立する vRealize Orchestrator アクションです。バインドを設定することで、vSphere 仮想マシンへの vMotion アクションを要求するユーザーに対して、インストール後アクションがシームレスになるようにします。システムは、ユーザーが指定しなくて済むように、ワークフローで名前を提供します。

- a [vMotion での仮想マシンの移行] ワークフローを選択後、[プロパティ バインド] ペインに移動します。
- b vm 入力プロパティのバインドを選択します。
- c [バインド] で、[バインド アクション付き] を選択します。
[findVcVmByVcAndVmUuid] アクションが自動的に選択されます。このアクションは、vRealize Automation Cloud Assembly の vRealize Orchestrator 統合によって事前構成されています。
- d [保存] をクリックします。

- 3 Day 2 アクションへの変更を保存するには、[作成] をクリックします。

4 ワークフロー内の他の入力パラメータを考慮して、ユーザーがアクションを要求するときに表示される申請フォームをカスタマイズできます。

- a [リソース アクション] から、最近作成した Day 2 アクションを選択します。
- b [要求パラメータの編集] をクリックします。

ユーザーに要求画面を表示する方法をカスタマイズできます。

デフォルトのフィールド名	表示	値	制約
仮想マシンのターゲット リソース プール。デフォルトは、現在のリソース プールです。	<ul style="list-style-type: none"> ■ ラベル = ターゲット リソース プール ■ 表示タイプ = 値ピッカー 		
仮想マシンの移行先のホスト	<ul style="list-style-type: none"> ■ ラベル = ターゲット ホスト ■ 表示タイプ = 値ピッカー 		必須 = はい
移行タスクの優先順位	ラベル = タスクの優先順位	値のオプション <ul style="list-style-type: none"> ■ 値のソース = 定数 テキスト ボックスにカンマ区切りのリストを入力します。 <div> lowPriority Low,defaultPriority Default,highPriority High </div>	必須 = はい
(オプション) パワーオン状態が指定した状態と一致する場合にのみ、仮想マシンを移行します	このテキスト ボックスを削除します。 vMotion では、パワー状態にかかわらずマシンを移動できます。		

- c [保存] をクリックします。

5 アクションがいつ利用可能になるかを制限するには、条件を構成します。

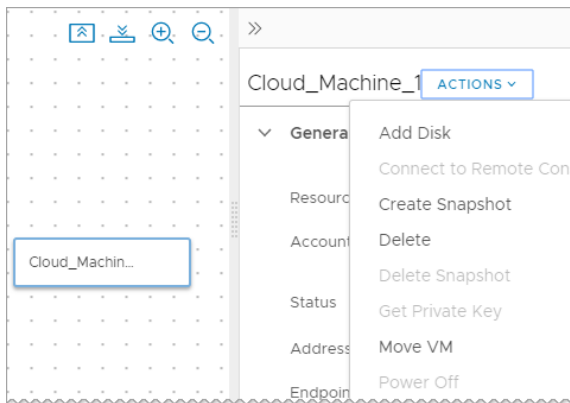
たとえば、CPU 数が 4 以下のマシンでのみ vMotion アクションを使用できるようにするといった状況が考えられます。

- a [条件が必要] をオンにします。
- b 条件を入力します。

Key	演算子	値
\${properties.cpuCount}	lessThan	4

- c [更新] をクリックします。

- 6 [仮想マシンの移動] アクションが、条件に一致する展開されたマシンに対して使用可能であることを確認します。
- [展開] を選択します。
 - 定義された条件に一致する展開済みのマシンを含む展開を探します。
 - 展開を開いて、マシンを選択します。
 - 右側のペインで [アクション] をクリックし、Move VM アクションが表示されることを確認します。



- アクションを実行します。

拡張性によりアプリケーションのライフサイクルを延長および自動化する方法

アプリケーションのライフサイクルは、拡張性アクションか、拡張性サブスクリプションを使用する vRealize Orchestrator ワークフローを使用して延長できます。

vRealize Automation Cloud Assembly 拡張性により、サブスクリプションを使用して拡張性アクションまたは vRealize Orchestrator ワークフローをイベントに割り当てることができます。指定されたイベントが発生すると、サブスクリプションによって実行されるアクションまたはワークフローが開始され、すべてのサブスクライバーに通知されます。

拡張性アクション

拡張性アクションは、アクションとそのアクションを実行する方法を指定するために使用される、軽量で小さなコード スクリプトです。拡張性アクションは、事前定義された vRealize Automation Cloud Assembly アクション テンプレートから直接インポートすることも、ZIP ファイルからインポートすることもできます。アクション エディタを使用して、拡張性アクション用のカスタム スクリプトを作成することもできます。複数のアクション スクリプトが1つのスクリプトにまとめてリンクされている場合、アクション フローを作成します。アクション フローを使用して、一連のアクションを作成できます。アクション フローの使用については、[アクション フローについて](#)を参照してください。

vRealize Orchestrator ワークフロー

vRealize Automation Cloud Assembly を既存の vRealize Orchestrator 環境と統合することにより、拡張性サブスクリプションでワークフローを使用できます。

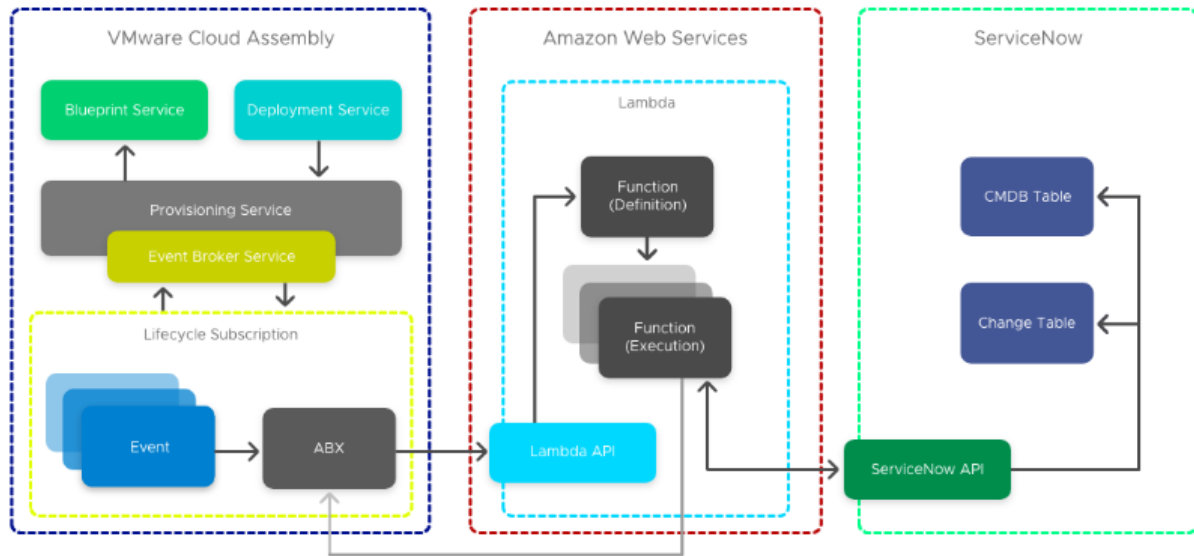
拡張性アクションのサブスクリプション

拡張性アクションを vRealize Automation Cloud Assembly サブスクリプションに割り当てると、アプリケーションのライフサイクルを延長できます。

注： 以下に、サブスクリプションの使用事例を示しますが、すべての拡張性アクションの機能を網羅しているわけではありません。

拡張アクションを使用して Cloud Assembly と ServiceNow を統合する方法

拡張性アクションを使用して、vRealize Automation Cloud Assembly を ServiceNow のような Enterprise ITSM と統合できます。

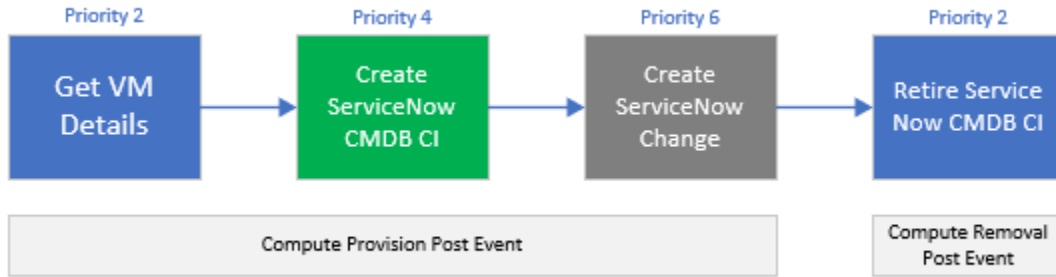


エンタープライズ ユーザーは、コンプライアンスを実現するために、一般的に Cloud Management Platform と IT サービス管理 (ITSM) および構成管理データベース (CMDB) プラットフォームを統合します。この例のとおり、拡張性アクション スクリプトを使用することにより、vRealize Automation Cloud Assembly を CMDB および ITSM 用の ServiceNow と統合できます。

注： また、vRealize Orchestrator ワークフローを使用することにより、ServiceNow を vRealize Automation Cloud Assembly と統合することもできます。ワークフローを使用して ServiceNow と統合する方法については、[vRealize Orchestrator ワークフローを使用して、ITSM 用の Cloud Assembly を ServiceNow と統合する方法を参照してください](#)。

この統合を作成するには、4 つの拡張性アクション スクリプトを使用します。最初の 3 つのスクリプトは、プロビジョニング中に、コンピューティング プロビジョニング後イベントで順番に開始されます。4 番目のスクリプトは、コンピューティング削除後イベントでトリガされます。

イベント トピックの詳細については、[vRealize Automation Cloud Assembly で提供されるイベント トピックを参照してください](#)。



[仮想マシンの詳細の取得]

仮想マシンの詳細の取得スクリプトは、CI の作成に必要な追加のペイロードの詳細と、Amazon Web Services Systems Manager パラメータ ストア (SSM) に保存されている ID トークンを取得します。また、このスクリプトは、後で使用するために、`customProperties` を追加プロパティで更新します。

[ServiceNow CMDB CI の作成]

ServiceNow CMDB CI の作成スクリプトは、ServiceNow インスタンス URL を入力として渡し、セキュリティ要件を満たすためにインスタンスを SSM に保存します。また、このスクリプトは、ServiceNow CMDB の一意のレコード識別子応答 (`sys_id`) を読み取ります。これを出力として渡し、作成中にカスタムプロパティ `serviceNowSysId` を書き込みます。この値は、インスタンスが破棄されたときに CI を使用中止としてマークするために使用されます。

注： Lambda が SSM パラメータ ストアにアクセスできるようにするために、vRealize Automation services Amazon Web Services ロールに追加権限を割り当てる必要がある場合があります。

[ServiceNow の変更の作成]

このスクリプトは、ServiceNow インスタンス URL を入力として渡し、ServiceNow 認証情報を SSM として保存してセキュリティ要件を満たすことで、ITSM の統合を完了します。

[ServiceNow の変更の作成]

ServiceNow CMDB CI の使用中止スクリプトは、ServiceNow に停止するように指示し、作成スクリプトで作成されたカスタムプロパティ `serviceNowSysId` に基づいて CI を使用中止とマークします。

前提条件

- この統合を設定する前に、次の条件付きクラウド テンプレート プロパティを使用してすべてのイベント サブスクリプションをフィルタリングします。

```
event.data["customProperties"]["enable_servicenow"] === "true"
```

注： このプロパティは、ServiceNow との統合が必要なクラウド テンプレートで設定できます。

- Python をダウンロードしてインストールします。

サブスクリプションのフィルタリングの詳細については、[拡張サブスクリプションの作成](#)を参照してください。

手順

- 1 仮想マシンからコマンドライン プロンプトを開きます。

2 仮想マシンの詳細の取得スクリプトを実行します。

```
from botocore.vendored import requests
import json
import boto3
client = boto3.client('ssm','ap-southeast-2')

def handler(context, inputs):
    baseUrl = inputs['url']
    casToken = client.get_parameter(Name="casToken",WithDecryption=True)

    url = baseUrl + "/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 = baseUrl + "/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 構成アイテムの作成アクションを実行します。

```
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 作成アクション スクリプトを実行します。

```

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)

```

結果

vRealize Automation Cloud Assembly は ITSM ServiceNow と正常に統合されました。

次のステップ

必要に応じて、CMDB 構成アイテムの削除アクションを使用することにより、CI の使用を中止できます。

```

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+"/"+{0}".format(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)

```

vRealize Automation Cloud Assembly で、拡張性アクションを使用して ServiceNow を統合する方法の詳細については、[Extending Cloud Assembly with Action Based Extensibility for ServiceNow Integration](#) を参照してください。

拡張性アクションを使用してプロビジョニング中に仮想マシンにタグを付ける方法

拡張性アクションをサブスクリプションと組み合わせて使用すると、仮想マシンのタグ付けを自動化および簡素化できます。

クラウド管理者は、拡張性アクションと拡張性サブスクリプションを使用することにより、指定した入力および出力で自動的にタグ付けされる展開を作成できます。タグ仮想マシン サブスクリプションを含むプロジェクトに対して新しい展開を作成すると、展開イベントによってタグ仮想マシン スクリプトがトリガされ、タグが自動的に適用されます。これにより、時間の節約になり効率が向上し、展開の管理も容易になります。

前提条件

- クラウド管理者の認証情報へのアクセス。
- Lambda 機能に対する Amazon Web Services ロール。

手順

- 1 [拡張性] - [ライブラリ] - [アクション] - [新規アクション] の順に移動し、次のパラメータを使用してアクションを作成します。

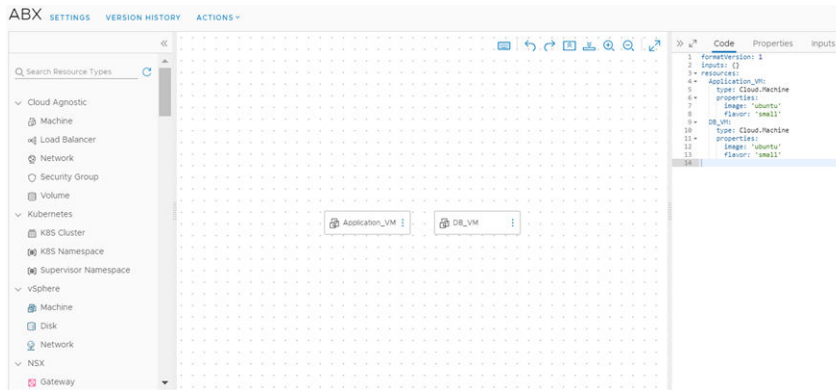
パラメータ	説明
アクション名	拡張性アクション名。「 TagVM 」をプリフィックスまたはサフィックスとして使用することが推奨されます。
プロジェクト	拡張性アクションをテストするプロジェクト。
アクション テンプレート	タグ仮想マシン
ランタイム	Python
スクリプトの送信元	スクリプトの記述

- 2 [メインの関数] として **Handler** と入力します。
- 3 拡張性アクションをテストするには、タグ付け入力を追加します。
たとえば、`resourceNames = ["DB_VM"]` や `target = world` のようにします。
- 4 アクションを保存するには、[保存] をクリックします。
- 5 アクションをテストするには、[テスト] をクリックします。
- 6 アクション エディタを終了するには、[閉じる] をクリックします。
- 7 [拡張性] - [サブスクリプション] の順に移動します。
- 8 [新しいサブスクリプション] をクリックします。
- 9 以下のサブスクリプションの詳細を入力します。

詳細	設定
イベント トピック	仮想マシンのタグ付けフェーズに関連するイベント トピックを選択します（例：コンピューティング割り当て）。 注： タグは、選択したイベント トピックのイベント パラメータの一部である必要があります。
ブロック	サブスクリプションのタイムアウトを 1 分に設定します。
アクション/ワークフロー	拡張性アクションの実行可能なタイプを選択し、カスタムの拡張性アクションを選択します。

- 10 カスタムの拡張性アクションのサブスクリプションを保存するには、[保存] をクリックします。
- 11 [デザイン] - [クラウド テンプレート] に移動し、空のキャンバスからクラウド テンプレートを作成します。

12 2 台の仮想マシン (Application_VM と DB_VM) をクラウド テンプレートに追加します。



13 仮想マシンを展開するには、[展開] をクリックします。

14 展開時に、イベントが開始されて拡張性アクションが実行されていることを確認します。

15 タグが正しく適用されたことを確認するには、[インフラストラクチャ] - [リソース] - [マシン] の順に移動します。

拡張性アクションの詳細

アクションベースの拡張性により、vRealize Automation Cloud Assembly 内のコードの簡素化されたスクリプトを使用して、拡張性アクションを自動化します。

アクションベースの拡張性により、軽量で柔軟なランタイム エンジン インターフェイスが利用できるようになります。これにより、小さなスクリプト実行可能アクションを定義して、拡張性サブスクリプションで指定されたイベントが発生したときに実行するように設定できます。

コードのこれらの拡張性アクション スクリプトを vRealize Automation Cloud Assembly 内またはローカル環境に作成して、サブスクリプションに割り当てることができます。拡張性アクション スクリプトは、タスクと手順をより軽量で簡単に自動化するために使用されます。vRealize Automation Cloud Assembly と vRealize Orchestrator サーバの統合の詳細については、[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。

アクションベースの拡張性により、次の機能が利用できるようになります。

- vRealize Orchestrator ワークフローの代替機能。再利用可能な小さいスクリプト実行可能アクションを使用して、軽量の統合とカスタマイズを実現します。
- アクションテンプレートの再利用。再利用可能なパラメータ化されたアクションを実行できます。

拡張性アクションを作成するには、ユーザー定義のアクション スクリプト コードを記述するか、または事前定義されたスクリプト コードを .ZIP パッケージとしてインポートします。アクションベースの拡張性では、Node.js、Python、および PowerShell のランタイム環境がサポートされています。Node.js および Python のランタイムは、Amazon Web Services Lambda に依存しています。そのため、Amazon Web Services の ID およびア

アクセス権の管理 (IAM) を備えたアクティブなサブスクリプションを用意し、vRealize Automation Cloud Assembly で Amazon Web Services をエンドポイントとして設定する必要があります。Amazon Web Services Lambda の使用を開始する方法については、[ABX: Serverless Extensibility of Cloud Assembly Services](#) を参照してください。

注： 拡張性アクションは、プロジェクト固有のアクションです。

拡張性アクションの作成方法

vRealize Automation Cloud Assembly を使用すると、拡張性サブスクリプションで使用する拡張性アクションを作成できます。

拡張性アクションは、ユーザー定義のスクリプト コードとアクション テンプレートを使用してアプリケーション ライフサイクルを拡張するための、高度にカスタマイズ可能な、軽量で、柔軟な方法です。アクション テンプレートには、拡張性アクションの基盤を設定するのに役立つ事前定義済みパラメータが含まれています。

拡張性アクションを作成するには、次の 2 つの方法があります。

- 拡張性アクション スクリプトのユーザー定義コードを記述する。

注： 拡張性アクション エディタのユーザー定義コードを記述すると、アクティブなインターネット接続が必要になることがあります。

- 拡張性アクションの展開パッケージを ZIP パッケージとしてインポートする。拡張性アクション用の ZIP パッケージの作成の詳細については、[Python ランタイムの拡張性アクション用の ZIP パッケージの作成](#)、[Node.js ランタイムの拡張性アクション用の ZIP パッケージの作成](#)または [PowerShell ランタイムの拡張性アクション用の ZIP パッケージの作成](#)を参照してください。

Amazon Web Services を FaaS プロバイダとして使用する拡張性アクションを作成するための手順を以下で説明します。

前提条件

- 有効かつアクティブなプロジェクトのメンバーシップであること。
- Lambda 機能に Amazon Web Services ロールを設定済みであること。たとえば、`AWSLambdaBasicExecutionRole` です。
- クラウド管理者ロールまたは `iam:PassRole` 権限が有効になっていること。

手順

- 1 [拡張性] - [ライブラリ] - [アクション] の順に選択します。
- 2 [新しいアクション] をクリックします。
- 3 アクションの名前を入力し、プロジェクトを選択します。
- 4 [次へ] をクリックします。

- 5 アクション テンプレートを検索して選択します。

注： アクション テンプレートを使用せずにカスタム アクションを作成する場合は、[カスタム スクリプト] を選択します。

設定可能な新規パラメータが表示されます。

- 6 [スクリプトの記述] または [パッケージのインポート] を選択します。
- 7 アクションのランタイムを選択します。
- 8 アクションのエントリ ポイントに [メインの関数] の名前を入力します。

注： ZIP パッケージからインポートされたアクションの場合、メインの関数には、エントリ ポイントを含むスクリプト ファイルの名前も含める必要があります。たとえば、メイン スクリプト ファイルのタイトルが `main.py` で、エントリ ポイントが `handler (context, inputs)` の場合、メインの関数の名前は `main.handler` にする必要があります。

- 9 アクションの [入力] パラメータと [出力] パラメータを定義します。
- 10 (オプション) アクションにアプリケーションの依存関係を追加します。

注： PowerShell スクリプトでは、アプリケーションの依存関係を定義して、PowerShell Gallery リポジトリに対して解決できるようにします。アプリケーションの依存関係を定義して、パブリック リポジトリから解決できるようにするには、次の形式を使用します。

```
@{
    Name = 'Version'
}

e.g.

@{
    Pester = '4.3.1'
}
```

注： ZIP パッケージからインポートされたアクションの場合は、アプリケーションの依存関係は自動的に追加されます。

- 11 タイムアウトとメモリ制限を定義するには、[カスタムのタイムアウトと制限の設定] オプションを有効にします。
- 12 アクションをテストするには、[保存] をクリックしてから [テスト] をクリックします。

次のステップ

拡張性アクションが作成されて検証されたら、それをサブスクリプションに割り当てることができます。

注： 拡張性サブスクリプションでは、拡張性アクションの最新のリリース バージョンが使用されます。新しいバージョンのアクションを作成したら、エディタ ウィンドウの右上にある [バージョン] をクリックします。サブスクリプションで使用するバージョンのアクションをリリースするには、[リリース] をクリックします。

Python ランタイムの拡張性アクション用の ZIP パッケージの作成

vRealize Automation Cloud Assembly の拡張性アクションで使用する Python スクリプトと依存関係を含む ZIP パッケージを作成できます。

拡張性アクション用のスクリプトを作成するには、次の 2 つの方法があります。

- vRealize Automation Cloud Assembly の拡張性アクション エディタでスクリプトを直接記述する。
- ローカル環境でスクリプトを作成し、関連する依存関係とともに ZIP パッケージに追加する。

ZIP パッケージを使用すると、vRealize Automation Cloud Assembly にインポートして拡張性アクションで使える、アクション スクリプトと依存関係の事前構成済みテンプレートをカスタムで作成できます。

さらに、ZIP パッケージは、環境でインターネット アクセスが利用できない場合など、アクション スクリプトの依存関係に関連付けられているモジュールを vRealize Automation Cloud Assembly サービスで解決できないシナリオで使用できます。

また、ZIP パッケージを使用して、複数の Python スクリプト ファイルを含む拡張性アクションを作成することもできます。複数のスクリプト ファイルを使用すると、拡張性アクション コードの構造を編成するのに役立ちます。

前提条件

Python 3.3 以前を使用している場合は、PIP パッケージ インストーラをダウンロードして構成します。[「Python パッケージ インデックス」](#)を参照してください。

手順

- 1 ローカル マシンで、アクション スクリプトと依存関係用のフォルダを作成します。
たとえば、`/home/user1/zip-action` です。
- 2 メインの Python アクション スクリプトをフォルダに追加します。
たとえば、`/home/user1/zip-action/main.py` です。
- 3 (オプション) Python スクリプトの依存関係をフォルダに追加します。
 - a 依存関係を含む `requirements.txt` ファイルを作成します。[「要件ファイル」](#)を参照してください。
 - b Linux シェルを開きます。

注： vRealize Automation Cloud Assembly でのアクションベースの拡張性のランタイムは、Linux ベースです。したがって、Windows 環境でコンパイルされた Python の依存関係によって、生成された ZIP パッケージが拡張性アクションの作成に使用できなくなる可能性があります。そのため、Linux シェルを使用する必要があります。

- c 次のコマンドを実行して、`requirements.txt` ファイルをスクリプト フォルダにインストールします。

```
pip install -r requirements.txt --target=home/user1/zip-action
```


- 4 割り当てられたフォルダで、スクリプト要素と requirements.txt ファイル（該当する場合）を選択して、ZIP パッケージに圧縮します。

注： スクリプトおよび依存関係要素は両方とも、ZIP パッケージのルート レベルに保存する必要があります。Linux 環境で ZIP パッケージを作成すると、パッケージのコンテンツがルート レベルに保存されないという問題が発生することがあります。この問題が発生した場合は、コマンドライン シェルで `zip -r` コマンドを実行して、パッケージを作成してください。

```
cd your_script_and_dependencies_folder
zip -r ../your_action_ZIP.zip *
```

次のステップ

ZIP パッケージを使用して、拡張性アクション スクリプトを作成します。[拡張性アクションの作成方法](#)を参照してください。

Node.js ランタイムの拡張性アクション用の ZIP パッケージの作成

vRealize Automation Cloud Assembly の拡張性アクションで使用される Node.js スクリプトと依存関係を含む ZIP パッケージを作成できます。

拡張性アクション用のスクリプトを作成するには、次の 2 つの方法があります。

- vRealize Automation Cloud Assembly の拡張性アクション エディタでスクリプトを直接記述する。
- ローカル環境でスクリプトを作成し、関連する依存関係とともに ZIP パッケージに追加する。

ZIP パッケージを使用すると、vRealize Automation Cloud Assembly にインポートして拡張性アクションで利用できる、アクション スクリプトと依存関係の事前構成済みテンプレートをカスタムで作成できます。

さらに、ZIP パッケージは、環境でインターネット アクセスが利用できない場合など、アクション スクリプトの依存関係に関連付けられているモジュールを vRealize Automation Cloud Assembly サービスで解決できないシナリオで使用できます。

また、パッケージを使用して、複数の Node.js スクリプト ファイルを含む拡張性アクションを作成することもできます。複数のスクリプト ファイルを使用すると、拡張性アクション コードの構造を編成するのに役立ちます。

手順

- 1 ローカル マシンで、アクション スクリプトと依存関係用のフォルダを作成します。

たとえば、`/home/user1/zip-action` です。

- 2 メインの Node.js アクション スクリプトをフォルダに追加します。

たとえば、`/home/user1/zip-action/main.js` です。

3 (オプション) Node.js スクリプトの依存関係をフォルダに追加します。

- a スクリプト フォルダに依存関係を含む `package.json` ファイルを作成します。「[package.json ファイルの作成](#)」および「[package.json ファイルでの dependencies と devDependencies の指定](#)」を参照してください。
- b コマンドライン シェルを開きます。
- c アクション スクリプトと依存関係用に作成したフォルダに移動します。

```
cd /home/user1/zip-action
```

- d 次のコマンドを実行して、`package.json` ファイルをスクリプト フォルダにインストールします。

```
npm install --production
```

注： このコマンドにより、フォルダ内に `node_modules` ディレクトリが作成されます。

- 4 割り当てられたフォルダで、スクリプト要素と `node_modules` ディレクトリ（該当する場合）を選択して、ZIP パッケージに圧縮します。

注： スクリプトおよび依存関係要素は両方とも、ZIP パッケージのルート レベルに保存する必要があります。Linux 環境で ZIP パッケージを作成すると、パッケージのコンテンツがルート レベルに保存されないという問題が発生することがあります。この問題が発生した場合は、コマンドライン シェルで `zip -r` コマンドを実行して、パッケージを作成してください。

```
cd your_script_and_dependencies_folder
zip -r ../your_action_ZIP.zip *
```

次のステップ

ZIP パッケージを使用して、拡張性アクション スクリプトを作成します。[拡張性アクションの作成方法を参照](#)してください。

PowerShell ランタイムの拡張性アクション用の ZIP パッケージの作成

拡張性アクションで使用する PowerShell スクリプトと依存関係モジュールを含む ZIP パッケージを作成できます。

拡張性アクション用のスクリプトを作成するには、次の 2 つの方法があります。

- vRealize Automation Cloud Assembly の拡張性アクション エディタでスクリプトを直接記述する。
- ローカル環境でスクリプトを作成し、関連する依存関係とともに ZIP パッケージに追加する。

ZIP パッケージを使用すると、vRealize Automation Cloud Assembly にインポートして拡張性アクションで利用できる、アクション スクリプトと依存関係の事前構成済みテンプレートをカスタムで作成できます。

注： PowerCLI コマンドレットを依存関係として定義したり、ZIP パッケージにバンドルしたりする必要はありません。PowerCLI コマンドレットは、vRealize Automation Cloud Assembly サービスの PowerShell ランタイムと共に事前構成されて提供されます。

さらに、ZIP パッケージは、環境でインターネット アクセスが利用できない場合など、アクション スクリプトの依存関係に関連付けられているモジュールを vRealize Automation Cloud Assembly サービスで解決できないシナリオで使用できます。

また、ZIP パッケージを使用して、複数の PowerShell スクリプト ファイルを含む拡張性アクションを作成することもできます。複数のスクリプト ファイルを使用すると、拡張性アクション コードの構造を編成するのに役立ちます。

前提条件

PowerShell および PowerCLI について理解していることを確認します。[Docker Hub](#) では、PowerShell Core、PowerCLI 10、PowerNSX、および複数のコミュニティ モジュールとスクリプト サンプルを含む Docker イメージを見つけることができます。

手順

- 1 ローカル マシンで、アクション スクリプトと依存関係用のフォルダを作成します。

たとえば、`/home/user1/zip-action` です。

- 2 拡張子が `.psm1` のメイン PowerShell スクリプトをフォルダに追加します。

次のスクリプトは、`main.psm1` というシンプルな PowerShell 関数を示しています。

```
function handler($context, $payload) {  
  
    Write-Host "Hello " $payload.target  
  
    return $payload  
}
```

注： PowerShell 拡張性アクションの出力は、関数の本体に表示されている最後の変数に基づいています。含まれている関数内のその他の変数はすべて破棄されます。

- 3 (オプション) `context` パラメータを使用して、メイン PowerShell スクリプトにプロキシ設定を追加します。[PowerShell スクリプトでの、コンテキスト パラメータを使用したプロキシ設定の追加](#)を参照してください。

4 (オプション) PowerShell スクリプトのすべての依存関係を追加します。

注： PowerShell 依存関係スクリプトには、.psm1 拡張子を使用する必要があります。スクリプトと、スクリプトが保存されているサブフォルダには、同じ名前を使用します。

- a Linux PowerShell シェルにログインします。

注： vRealize Automation Cloud Assembly でのアクションベースの拡張性のランタイムは、Linux ベースです。Windows 環境でコンパイルされた PowerShell 依存関係によって、生成された ZIP パッケージが使用できなくなることがあります。インストールされるすべてのサードパーティ依存関係は、PowerShell スクリプトが Photon OS で実行されるため、VMware Photon OS と互換性がある必要があります。

- b /home/user1/zip-action フォルダに移動します。

- c Save-Module コマンドレットを実行して、依存関係が含まれている PowerShell モジュールをダウンロードして保存します。

```
Save-Module -Name <module name> -Path ./
```

- d 追加の依存関係モジュールについては、前のサブステップを繰り返します。

重要： 各依存関係モジュールが個別のサブフォルダに配置されていることを確認します。PowerShell モジュールの記述と管理の詳細については、[PowerShell スクリプト モジュールを記述する方法](#)を参照してください。

5 割り当てられたフォルダで、スクリプト要素と依存関係モジュール サブフォルダ（該当する場合）を選択して、ZIP パッケージに圧縮します。

注： スクリプトと依存関係モジュール サブフォルダは両方とも、ZIP パッケージのルート レベルに保存する必要があります。Linux 環境で ZIP パッケージを作成すると、パッケージのコンテンツがルート レベルに保存されないという問題が発生することがあります。この問題が発生した場合は、コマンドライン シェルで zip -r コマンドを実行して、パッケージを作成してください。

```
cd your_script_and_dependencies_folder
zip -r ../your_action_ZIP.zip *
```

次のステップ

ZIP パッケージを使用して、拡張性アクション スクリプトを作成します。[拡張性アクションの作成方法](#)を参照してください。

PowerShell スクリプトでの、コンテキスト パラメータを使用したプロキシ設定の追加

context パラメータを使用して、PowerShell スクリプトでネットワーク プロキシ通信を有効にすることができます。

PowerShell コマンドレットによっては、PowerShell 関数の環境変数としてネットワーク プロキシを設定しなければならない場合があります。プロキシ設定は、`$context.proxy.host` と `$context.proxy.port` のパラメータを使用して PowerShell 関数に提供されます。

これらの `context` パラメータは、PowerShell スクリプトの先頭に追加できます。

```
$proxyString = "http://" + $context.proxy.host + ":" + $context.proxy.port
$Env:HTTP_PROXY = $proxyString
$Env:HTTPS_PROXY = $proxyString
```

コマンドレットで `-Proxy` パラメータがサポートされている場合は、特定の PowerShell コマンドレットにプロキシの値を直接渡すこともできます。

クラウド固有の拡張性アクションの設定

クラウド アカウントを操作するように拡張性アクションを設定できます。

拡張性アクションを作成するときには、次のように設定して、さまざまなクラウドベースのアカウントにリンクできます。

- Microsoft Azure
- Amazon Web Services

前提条件

有効なクラウド アカウントが必要です。

手順

- 1 [拡張性] - [ライブラリ] - [アクション] の順に選択します。
- 2 [新しいアクション] をクリックします。
- 3 必要に応じてアクション パラメータを入力します。
- 4 [FaaS プロバイダ] ドロップダウン メニューで、クラウド アカウントプロバイダを選択するか、[自動] を選択します。

注： [自動] を選択すると、FaaS プロバイダが自動的に定義されます。

- 5 [保存] をクリックします。

結果

拡張性アクションが、設定したクラウド アカウントで使用できるようにリンクされます。

オンプレミスの拡張性アクションの設定

Amazon Web Service または Microsoft Azure クラウド アカウントではなくオンプレミスの FaaS プロバイダを使用するように、拡張性アクションを設定できます。

拡張性アクション用のオンプレミスの FaaS プロバイダを使用すると、vRealize Automation Cloud Assembly 拡張性サブスクリプションで LDAP、CMDB、または vCenter データセンターなどのオンプレミス サービスを使用できます。

手順

- 1 [拡張性] - [ライブラリ] - [アクション] の順に選択します。
- 2 [新しいアクション] をクリックします。
- 3 拡張性アクションの名前とプロジェクトを入力します。
- 4 (オプション) 拡張性アクションの説明を入力します。
- 5 [次へ] をクリックします。
- 6 拡張性アクション スクリプトを作成またはインポートします。
- 7 [FaaS プロバイダ] ドロップダウン メニューをクリックし、[オンプレミス] を選択します。
- 8 新しい拡張性アクションを保存するには、[保存] をクリックします。

次のステップ

vRealize Automation Cloud Assembly 拡張性サブスクリプションで、作成された拡張性アクションを使用します。

共有拡張性アクションの作成

vRealize Automation Cloud Assembly 管理者は、アクションをエクスポートおよびインポートせずにプロジェクト間で共有できる拡張性アクションを作成します。

拡張性アクションのエクスポートおよびインポートの詳細については、[拡張性アクションのエクスポートとインポート](#)を参照してください。

前提条件

vRealize Automation Cloud Assembly 組織内に 2 つ以上のプロジェクトを作成します。

手順

- 1 [拡張性] - [ライブラリ] - [アクション] の順に選択します。
- 2 [新しいアクション] をクリックします。
- 3 拡張性アクションの名前を入力します。
- 4 (オプション) 拡張性アクションの説明を入力します。
- 5 拡張性アクションを作成するプロジェクトを選択します。
- 6 [この組織内のすべてのプロジェクトに共有] チェックボックスを選択します。
- 7 [次へ] をクリックします。
- 8 アクション スクリプトを作成またはインポートし、拡張性アクションを保存します。

注： [設定] から共有を有効または無効にできます。拡張性アクションがサブスクリプションで使用されている場合、共有を無効にすることはできません。共有を無効にするには、サブスクリプションから拡張性アクションを削除する必要があります。

- 9 拡張性サブスクリプションを作成し、共有拡張性アクションを追加して、サブスクリプション範囲を [任意のプロジェクト] に設定します。

注： 拡張性サブスクリプションの作成の詳細については、[拡張サブスクリプションの作成](#)を参照してください。

拡張性サブスクリプションは、任意のプロジェクトの一致するイベントによってトリガされます。

次のステップ

共有拡張性アクションをコンテンツ ソースとして vRealize Automation Service Broker カタログにインポートすることもできます。ソース プロジェクトを選択するときに、拡張性アクションが作成されたプロジェクトを入力します。vRealize Automation Service Broker への拡張性アクションの追加の詳細については、[Service Broker カタログへの拡張性アクションの追加](#)を参照してください。

拡張性アクションのエクスポートとインポート

vRealize Automation Cloud Assembly を使用すると、拡張性アクションをエクスポートおよびインポートして、さまざまなプロジェクトで使用できます。

前提条件

既存の拡張性アクション。

手順

- 1 拡張性アクションをエクスポートします。
 - a [拡張性] - [ライブラリ] - [アクション] の順に移動します。
 - b 拡張性アクションを選択し、[エクスポート] をクリックします。

アクション スクリプトとその依存関係が ZIP ファイルとしてローカル環境に保存されます。
- 2 拡張性アクションをインポートします。
 - a [拡張性] - [ライブラリ] - [アクション] の順に移動します。
 - b [インポート] をクリックします。
 - c エクスポートされた拡張性アクションを選択し、プロジェクトに割り当てます。
 - d [インポート] をクリックします。

注： 指定したプロジェクトに、インポートされた拡張性アクションがすでに割り当てられている場合は、競合の解決ポリシーを選択するように求められます。

代替 アクションエディタから [パッケージのインポート] オプションを直接選択してアクション スクリプトをインポートすることもできます。

アクション フローについて

アクション フローは、ライフサイクルと自動化をさらに拡張するために使用される一連の拡張性アクション スクリプトです。

すべてのアクション フローは flow_start で始まり、flow_end で終了します。次のアクション フロー要素を使用して、複数の拡張性アクション スクリプトを同時にリンクできます。

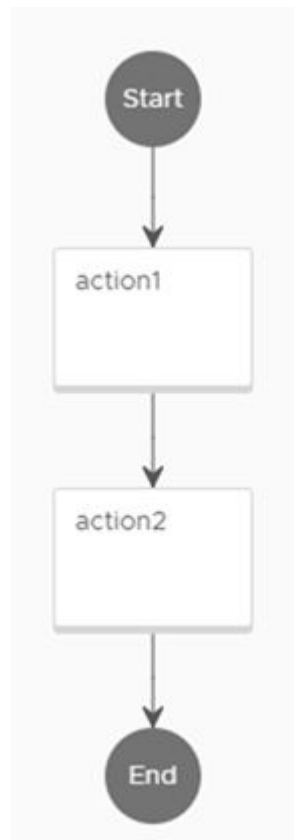
- **順次アクション フロー** - 連続して実行する複数の拡張性アクション スクリプト。
- **フォーク アクション フロー** - パスを分割して同じ出力に使用する複数の拡張性アクション スクリプトまたはフロー。
- **結合アクション フロー** - 結合して同じ出力に使用する複数の拡張性アクション スクリプトまたはフロー。
- **条件付きアクション フロー** - 条件が満たされると実行される複数の拡張性アクション スクリプトまたはフロー。

順次アクション フロー

順次実行される複数の拡張性アクション スクリプト。

```
version: "1"
flow:
  flow_start:
    next: action1
  action1:
    action: <action_name>
    next: action2
  action2:
    action: <action_name>
    next: flow_end
```

注： アクションを next: アクションとして割り当てると、以前のアクションにループ バックできます。たとえば、この例では next: flow_end ではなく next: action1 を入力して、action1 を再実行し、一連のアクションを再起動できます。



フォーク アクション フロー

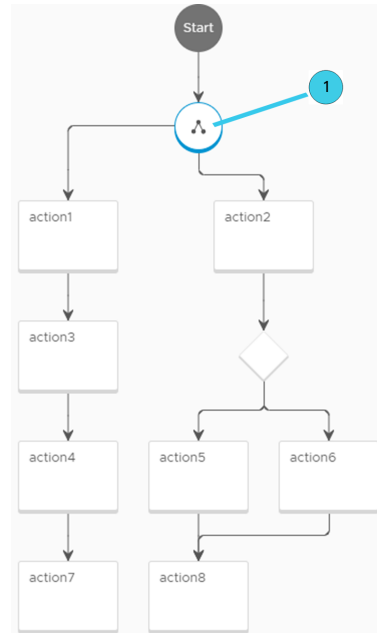
いくつかのパスに分かれながら同じ出力に寄与する複数の拡張性アクション スクリプトやアクション フロー。


```

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>

```

注： アクションを next: アクションとして割り当てると、以前のアクションにループバックできます。たとえば、アクション フローを終了する next: flow_end の代わりに next: action1 と入力して action1 を再実行し、アクションのシーケンスを再起動します。



① フォーク要素

結合アクション フロー

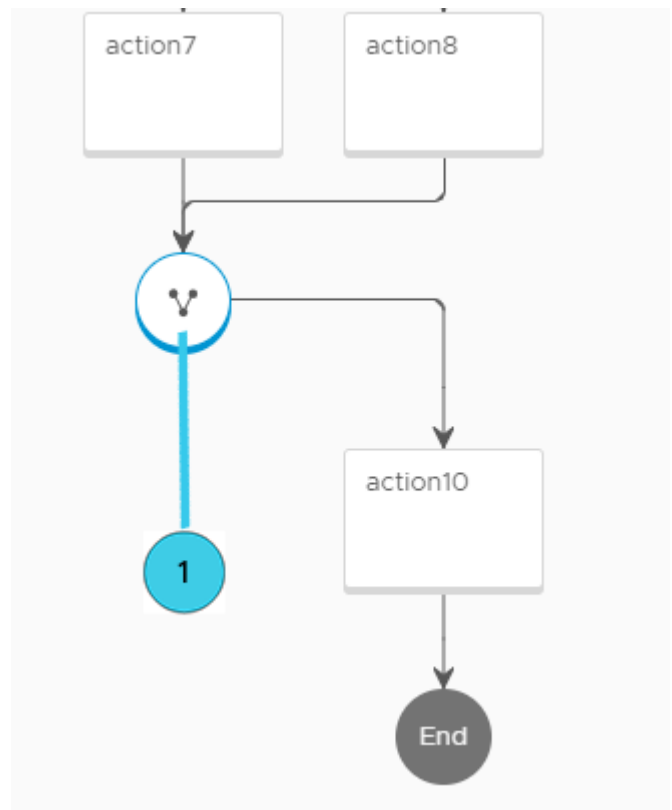
いくつかのパスが結合して同じ出力に寄与する複数の拡張性アクション スクリプトやアクション フロー。

```

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

```

注： アクションを next: アクションとして割り当てると、以前のアクションにループバックできます。たとえば、この例では next: flow_end ではなく next: action1 を入力して、action1 を再実行し、一連のアクションを再起動できます。



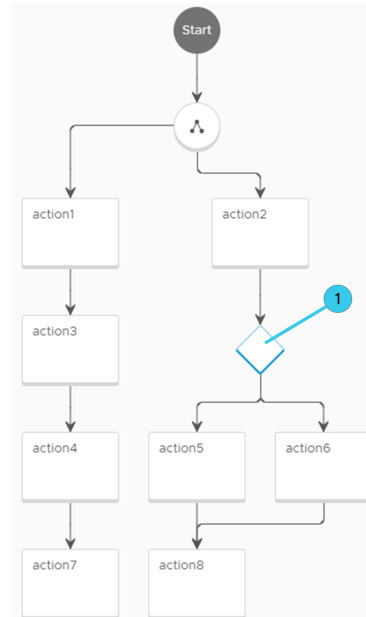
① 結合要素

条件付きアクション フロー

切り替え要素を使用して条件が満たされた場合に実行される、複数の拡張性アクション スクリプトまたはフロー。

アクションを実行するために、条件を `true` と等しくする必要が生じる場合もあります。この例に示すように、他にもアクションを実行するためには事前にパラメータ値の条件を満たしておく必要が生じる場合もあります。どの条件にも満たない場合、アクション フローは失敗します。

```
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 切り替え要素

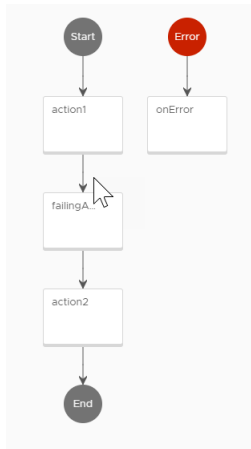
注： アクションを `next: アクション` として割り当てると、以前のアクションにループ バックできます。たとえば、アクション フローを終了する `next: flow_end` の代わりに `next: action1` と入力して `action1` を再実行し、アクションのシーケンスを再起動します。

アクション フローでエラー ハンドラを使用する方法

エラー ハンドラ要素を使用すると、フローの所定のステージでエラーを発行するようにアクション フローを設定できます。

エラー ハンドラ要素には、次の 2 つの入力が必要です。

- 機能しないアクションの所定のエラー メッセージ。
- アクション フロー入力。



フローのアクションが機能せず、アクション フローにエラー ハンドラ要素が含まれている場合、アクションが機能しなかったことを警告するエラー メッセージが発行されます。エラー ハンドラは、それ自体がアクションです。アクション フローに使用できるエラー ハンドラには、次のようなスクリプトがあります。

```
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
```

[アクション実行] ウィンドウで成功した実行と機能しなかった実行を確認できます。

Cloud Assembly

展開 ブループリント インフラストラクチャ **拡張性** マーケットプレイス

ガイド付きセットアップ

イベント サブスクリプション ライブラリ イベント トピック アクション ワークフロー アクティビティ

アクションの実行

ワークフローの実行

アクションの実行

489 個のアイテム

☐ キャンセル ☐ 削除

ユーザーの実行 フィルタリングしています...

ステータス	アクション	アクション ID
<input type="checkbox"/> 完了	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/> 失敗	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/> 完了	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6
<input type="checkbox"/> 完了	AWS-ABX	8a769ecc6df809c7016e01a83fe204e6

この例の flow-with-handler アクション フローにはエラー ハンドラ要素が含まれており、正常に実行されました。ただし、フロー内のアクションの 1 つが機能せず、そのためにエラー ハンドラが起動してエラーが発行されました。

アクションの実行を追跡する方法

[アクションの実行] タブには、サブスクリプションのトリガ済み拡張性アクションのログとそのステータスが表示されます。

[拡張性] - [アクティビティ] - [アクションの実行] を使用して、アクションの実行のログを表示できます。また、一度に 1 つまたは複数のプロパティを使用して、アクションの実行のリストをフィルタすることもできます。個々のアクションの実行の詳細を表示するには、[実行 ID] をクリックします。

拡張性アクションの実行失敗のトラブルシューティング

拡張性アクションの実行が失敗した場合は、トラブルシューティング手順を実行して修正できます。

アクションの実行が失敗したときに、エラー メッセージ、失敗した状態、および失敗したログが表示されることがあります。アクションの実行が失敗した場合、原因は展開の失敗かコードの失敗です。

問題	解決方法
展開の失敗	原因は、クラウド アカウント設定やアクション展開をはじめ依存関係に関連する問題のためにアクションの展開が阻止されたことです。使用したプロジェクトが、設定済みのクラウド アカウント内に定義され、機能を実行する権限が付与されていることを確認してください。アクションを再開する前に、アクションの詳細画面内で特定のプロジェクトに対するアクションをテストできます。
コードの失敗	原因は、スクリプトやコードが無効であることです。アクションの実行ログを使用して、無効なスクリプトをトラブルシューティングを行って修正してください。

拡張性ワークフロー サブスクリプション

vRealize Orchestrator でホストされているワークフローを vRealize Automation Cloud Assembly で使用して、アプリケーションのライフサイクルを拡張できます。

vRealize Orchestrator ワークフロー サブスクリプションを使用して仮想マシンのプロパティを変更する方法

既存の vRealize Orchestrator ワークフローを使用して仮想マシンのプロパティを変更し、仮想マシンを Active Directory に追加することができます。

イベント トピック パラメータで、イベント ブローカ サービス (EBS) メッセージのペイロード形式を定義します。ワークフロー内で EBS メッセージ ペイロードを受信して使用するには、inputProperties ワークフローの入力パラメータを定義する必要があります。

前提条件

- クラウド管理者のユーザー ロール
- 既存の vRealize Orchestrator オンプレミス ワークフロー。
- vRealize Orchestrator クライアント サーバを正常に統合して接続しています。

手順

- 1 [拡張性] - [サブスクリプション] の順に選択します。
- 2 [新しいサブスクリプション] をクリックします。

3 次のパラメータを使用してサブスクリプションを作成します。

パラメータ	値
名前	RenameVM
イベント トピック	目的の vRealize Orchestrator 統合に適したイベント トピックを選択します (例: コンピューティング割り当て)。
ブロック/非ブロック	非ブロック
アクション/ワークフロー	vRealize Orchestrator の実行可能なタイプを選択します。目的のワークフローを選択します (例: 仮想マシン名の設定)。

4 サブスクリプションを保存するには、[保存] をクリックします。

5 クラウド テンプレートを作成するか、既存のクラウド テンプレートを展開して、サブスクリプションを割り当て、有効にします。

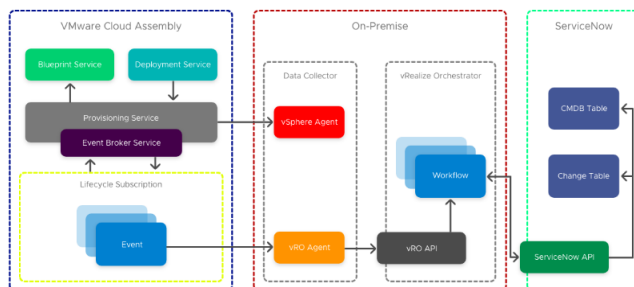
次のステップ

次のいずれかの方法でワークフローが正常に開始されたことを確認します。

- [拡張性] - [アクティビティ] - [ワークフローの実行] で、ワークフローの実行ログを確認する。
- vRealize Orchestrator クライアントを開き、ワークフローに移動してステータスを確認するか、特定のログのタブを開いてワークフローのステータスを確認する。

vRealize Orchestrator ワークフローを使用して、ITSM 用の Cloud Assembly を ServiceNow と統合する方法

vRealize Orchestrator でホストされたワークフローを使用して、ITSM のコンプライアンスのために vRealize Automation Cloud Assembly を ServiceNow と統合することができます。



エンタープライズ ユーザーは、コンプライアンスを実現するために、一般的に Cloud Management Platform と IT サービス管理 (ITSM) および構成管理データベース (CMDB) プラットフォームを統合します。この例のとおり、vRealize Orchestrator がホストされているワークフローを使用して、vRealize Automation Cloud Assembly を CMDB と ITSM 用に ServiceNow と統合できます。vRealize Orchestrator 統合とワークフローを使用する際に、環境ごとに複数のインスタンスがある場合は、機能タグが特に便利です。機能タグの詳細については、[vRealize Automation Cloud Assembly での機能タグの使用](#)を参照してください。

注: また、拡張性アクション スクリプトを使用して、ServiceNow と vRealize Automation Cloud Assembly を統合することもできます。拡張性アクション スクリプトを使用した ServiceNow の統合の詳細については、[拡張アクションを使用して Cloud Assembly と ServiceNow を統合する方法](#)を参照してください。

この例では、ServiceNow の統合は 3 つの最上位レベルのワークフローで構成されます。各ワークフローには独自のサブスクリプションがあるため、各コンポーネントを個別に更新および反復処理できます。

- イベント サブスクリプション エントリ ポイント - 基本ログは、要求しているユーザーと vCenter Server 仮想マシン（該当する場合）を識別します。
- 統合ワークフロー - オブジェクトを分離し、テクニカル ワークフローに入力を取り入れて、ログ、プロパティ、出力の更新を処理します。
- テクニカル ワークフロー - ServiceNow API のダウンストリーム システム統合により、ペイロードの外部に仮想マシンのプロパティを追加して、CMDB CI、CR、および vRealize Automation Cloud Assembly IaaS API を作成できます。

前提条件

- スタンドアローンまたはクラスタ化された vRealize Orchestrator 環境。
- vRealize Automation Cloud Assembly での vRealize Orchestrator の統合。スタンドアローン vRealize Orchestrator と vRealize Automation Cloud Assembly の統合の詳細については、[Cloud Assembly で vRealize Orchestrator との統合を設定する](#)を参照してください。

手順

- 1 vRealize Orchestrator で、複数のワークフローで使用される共通の構成を含む構成ファイルを作成して保存します。
- 2 手順 1 の構成ファイルと同じ場所に vRealize Automation Cloud Assembly API トークンを保存します。

注： vRealize Automation Cloud Assembly API トークンには有効期限があります。

- 3 指定したスクリプト要素を使用して、vRealize Orchestrator にワークフローを作成します。このスクリプトは、REST ホストを参照して特定します。また、トークンのオプション パラメータを使用する REST アクションを標準化します。これは追加の認証ヘッダーとして追加されます。

```
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"];

```

このスクリプトは、出力 `cpuCount` および `memoryMB` を親ワークフローに送信し、既存の `customProperties` プロパティを更新します。これらの値は、CMDB を作成するときに後続のワークフローで使用できます。

- 4 ServiceNow CMDB CI の作成スクリプトをワークフローに追加します。この要素は、構成アイテムを使用して ServiceNow REST ホストを特定し、cmdb_ci_vmware_instance テーブルの REST 操作を作成します。さらに、POST データのワークフロー入力に基づいてコンテンツ オブジェクトの文字列を作成して、返された sys_id を出力します。

```

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 子ワークフローの出力を使用して、既存の `customProperties` を使用するプロパティ オブジェクトを作成し、`serviceNowSysId` プロパティを ServiceNow の値で上書きします。この一意の ID は、インスタンスが破棄されたときに使用中止としてマークするために CMDB で使用されます。

結果

vRealize Automation Cloud Assembly は ITSM ServiceNow と正常に統合されました。vRealize Automation Cloud Assembly で、ワークフローを使用して ServiceNow を統合する方法の詳細については、[Extending Cloud Assembly with vRealize Orchestrator for ServiceNow Integration](#) を参照してください。

ワークフロー サブスクリプションの詳細

vRealize Automation Cloud Assembly との vRealize Orchestrator の統合を使用すると、ワークフローでアプリケーションのライフサイクルを延長できます。

vRealize Automation には vRealize Orchestrator 展開が組み込まれています。この組み込みの vRealize Orchestrator 展開のワークフロー ライブラリをサブスクリプションで使用できます。ワークフローの作成、変更、および削除を行うには、vRealize Orchestrator クライアントを使用します。

外部の vRealize Orchestrator 展開を vRealize Automation Cloud Assembly に統合することもできます。『組み込みの vRealize Orchestrator クライアントの使用』の「外部の vRealize Orchestrator クライアントを統合する方法」を参照してください。

vRealize Orchestrator ワークフローの作成に関するベスト プラクティス

ワークフロー サブスクリプションは、特定のイベント トピックとそのトピックのイベント パラメータに基づいています。サブスクリプションが vRealize Orchestrator ワークフローを確実に開始できるようにするために、正しい入力パラメータを使用してサブスクリプションを設定し、イベント データと連携できるようにする必要があります。

ワークフローの入力パラメータ

カスタム ワークフローには、すべてのパラメータまたはペイロード内のすべてのデータを使用する単一のパラメータを含めることができます。

単一のパラメータを使用するには、1つのパラメータを `Properties` のタイプで設定し、`inputProperties` と名前を指定します。

ワークフローの出力パラメータ

カスタム ワークフローには、応答イベント トピック タイプに必要な後続のイベントに関連する出力パラメータを含めることができます。

イベント トピックで応答が見込まれている場合、ワークフローの出力パラメータは、応答スキーマのパラメータと一致する必要があります。

ワークフローの実行を追跡する方法

[ワークフローの実行] ウィンドウには、サブスクリプションによってトリガされたワークフローとそのステータスのログが表示されます。

ワークフロー実行のログを表示するには、[拡張性] - [アクティビティ] - [ワークフローの実行] の順に選択します。

失敗したワークフロー サブスクリプションのトラブルシューティング

ワークフロー サブスクリプションが失敗した場合は、トラブルシューティングの手順を実行して修正することができます。

ワークフローの実行に失敗すると、ワークフロー サブスクリプションが正常に起動または完了しなくなる可能性があります。ワークフローの実行の失敗には、いくつかの一般的な原因が考えられます。

問題	原因	解決方法
vRealize Orchestrator のワークフロー サブスクリプションが正常に開始または完了しなかった。	イベント メッセージの受信時にカスタム ワークフローを実行するようにワークフロー サブスクリプションを設定したが、ワークフローが正常に実行または完了しない。	<ol style="list-style-type: none"> 1 ワークフロー サブスクリプションが正しく保存されていることを確認します。 2 ワークフロー サブスクリプションの条件が正しく設定されていることを確認します。 3 vRealize Orchestrator に、指定したワークフローがあることを確認します。 4 ワークフローが vRealize Orchestrator 内で正しく設定されていることを確認します。
承認申請の vRealize Orchestrator ワークフロー サブスクリプションが実行されなかった。	vRealize Orchestrator のワークフローを実行するための事前承認または事後承認のワークフロー サブスクリプションを設定したが、定義された条件に一致するマシンがサービス カタログで申請されても、ワークフローが実行されない。	<p>承認ワークフロー サブスクリプションを正常に実行するには、すべてのコンポーネントが正しく設定されていることを確認する必要があります。</p> <ol style="list-style-type: none"> 1 承認ポリシーがアクティブで正しく適用されていることを確認します。 2 ワークフロー サブスクリプションが正しく設定および保存されていることを確認します。 3 承認に関連するメッセージのイベント ログを確認します。
承認申請の vRealize Orchestrator ワークフロー サブスクリプションが却下された。	指定した vRealize Orchestrator のワークフローを実行する事前承認または事後承認のワークフロー サブスクリプションを設定したが、申請が外部の承認レベルで却下された。考えられる原因の 1 つは、vRealize Orchestrator の内部ワークフローの実行エラーです。たとえば、ワークフローが見つからなかったり、vRealize Orchestrator サーバが実行されていなかったりすることが考えられます。	<ol style="list-style-type: none"> 1 承認に関連するメッセージのログを確認します。 2 vRealize Orchestrator サーバが実行されていることを確認します。 3 vRealize Orchestrator に、指定したワークフローがあることを確認します。

拡張性サブスクリプションの詳細

アプリケーションのライフサイクルは、拡張性アクションか、拡張性サブスクリプションを使用する vRealize Orchestrator でホストされたワークフローを使用して延長できます。

お使いの環境内でトリガ イベントが発生すると、サブスクリプションが開始され、指定したワークフローまたは拡張性アクションが実行されます。イベント ログのシステムイベント、[ワークフローの実行] ウィンドウでのワークフローの実行、および [アクションの実行] ウィンドウでのアクションの実行を表示できます。サブスクリプションはプロジェクトに固有のものであり、指定したプロジェクトを通じてクラウド テンプレートおよび展開にリンクされます。

拡張性に関する用語

vRealize Automation Cloud Assembly 内で拡張性サブスクリプションを扱っていると、サブスクリプションおよびイベント ブローカ サービスに固有のいくつかの用語に出会うことがあります。

表 6-3. 拡張性に関する用語

用語	説明
イベント トピック	同じ論理的な意図と構造を持つ一連のイベントを表します。各イベントは、イベント トピックの1つのインスタンスです。 ブロック パラメータを特定のイベント トピックに割り当てることができます。詳細については、 ブロックに関するイベント トピック を参照してください。
イベント	プロデューサまたはプロデューサによって管理されるエンティティのいずれかで、状態が変更されたことを示します。このイベントは、イベントの発生に関する情報を記録するエンティティです。
イベント ブローカ サービス	プロデューサが公開したメッセージを、登録した利用者へ送るサービスです。
ペイロード	そのイベント トピックに関連するすべてのプロパティが含まれたイベント データ。
サブスクリプション	イベントに関する通知を受け取る意思があるサブスクライバが、イベント トピックに登録し、通知をトリガする基準を定義します。イベントをトリガする拡張性アクションまたはワークフローのいずれかにサブスクリプションを関連付けて、アプリケーションのライフサイクルの一部を自動化します。
サブスクライバ	サブスクリプションの定義に基づいて、イベント ブローカ サービスに公開されたイベントごとに通知されるユーザー。サブスクライバは、利用者と呼ばれることもあります。
システム管理者	vRealize Automation Cloud Assembly を使用して、テナント ワークフロー サブスクリプションとシステム ワークフロー サブスクリプションの作成、読み取り、更新、および削除を行うための権限を持ったユーザー。
ワークフロー サブスクリプション	vRealize Orchestrator ワークフローをトリガするイベント トピックと条件を指定します。
アクション サブスクリプション	拡張性アクションの実行をトリガするイベント トピックと条件を指定します。
ワークフロー	vRealize Automation Cloud Assembly 内で統合された vRealize Orchestrator ワークフロー。これらのワークフローをサブスクリプション内のイベントにリンクできます。
拡張性アクション	サブスクリプション内でイベントがトリガされた後に実行できる、簡素化されたコードのスクリプト。拡張性アクションはワークフローと似ていますが、ワークフローより軽量です。拡張性アクションは、vRealize Automation Cloud Assembly でカスタマイズできます。
アクションの実行	[アクションの実行] タブに表示されます。[アクションの実行] は、イベントのトリガに応答して実行された拡張性アクションの詳細なログです。

ブロックに関するイベント トピック

一部のイベント トピックはブロック イベントをサポートします。拡張性サブスクリプションの動作は、トピックでこれらのイベント タイプがサポートされるかどうか、またサブスクリプションの設定方法によって異なります。

vRealize Automation Cloud Assembly 拡張性サブスクリプションでは、非ブロック イベント トピックとブロック イベント トピックの 2 種類のイベント トピックを使用できます。イベント トピック タイプにより、拡張性サブスクリプションの動作が定義されます。

非ブロック イベント トピック

非ブロック イベント トピックでは、非ブロック サブスクリプションの作成のみが許可されます。非ブロック サブスクリプションは非同期的にトリガされるため、サブスクリプションのトリガが、設定された順序どおりに処理されないことがあります。

ブロックに関するイベント トピック

一部のイベント トピックはブロックをサポートします。サブスクリプションがブロックとしてマークされている場合、設定された条件を満たすすべてのメッセージは、ブロック サブスクリプションの実行可能な項目が実行されるまで、条件に一致する他のサブスクリプションによって受信されることはありません。

ブロック サブスクリプションは優先順位のとおりに実行されます。最も高い優先順位の値は 0（ゼロ）です。1 つのイベント トピックに優先順位レベルが同じ 2 つ以上のブロック サブスクリプションがある場合、サブスクリプションの名前に基づいてアルファベットの逆順に実行されます。すべてのブロック サブスクリプションが処理されたら、メッセージは、すべての非ブロック サブスクリプションに同時に送信されます。ブロック サブスクリプションは同期的に実行されるため、後続のサブスクリプションが通知される際には、変更されたイベント ペイロードに更新済みのイベントが含まれます。

ブロック イベント トピックを使用して、相互に依存する複数のサブスクリプションを管理できます。

たとえば、2 つのプロビジョニング ワークフロー サブスクリプションがあり、2 つ目のサブスクリプションが 1 つ目のサブスクリプションの結果を使用する場合について考えます。1 つ目のサブスクリプションは、プロビジョニング時にプロパティを変更し、2 つ目のサブスクリプションはファイル システム内の新しいプロパティ（マシン名など）を記録します。ChangeProperty サブスクリプションの優先順位は 0 で、RecordProperty は、2 つ目のサブスクリプションが 1 つ目のサブスクリプションの結果を使用するため優先順位が 1 になります。マシンのプロビジョニングが開始されると、ChangeProperty サブスクリプションが実行されます。RecordProperty サブスクリプションの条件はプロビジョニング後の条件に基づくため、RecordProperty サブスクリプションはイベントによってトリガされます。ただし、ChangeProperty ワークフローはブロック ワークフローであるため、完了するまでイベントは受信されません。マシン名が変更されて最初のワークフロー サブスクリプションが終了したら、2 つ目のワークフロー サブスクリプションが実行され、ファイル システムのマシン名が記録されます。

実行可能な項目のリカバリ

ブロック イベント トピックの場合は、実行可能な項目のリカバリをサブスクリプションに追加できます。サブスクリプション内の実行可能な項目のリカバリは、プライマリの実行可能な項目に障害が発生した場合に実行されます。たとえば、プライマリの実行可能な項目が ServiceNow などの CMDB システムにレコードを作成するワークフローであるワークフロー サブスクリプションを作成できます。ワークフロー サブスクリプションが失敗した場合でも、一部のレコードが CMDB システム内に作成される可能性があります。この場合、実行可能な項目のリカバリを使用することにより、障害が発生した実行可能な項目によって CMDB システムに残されたレコードをクリーンアップできます。

相互に依存する複数のサブスクリプションを含むユースケースでは、実行可能な項目のリカバリーに `ebs.recover.continuation` プロパティを追加することが考えられます。このプロパティを使用して、現在のサブスクリプションが失敗した場合に拡張性サービスがチェーン内の次のサブスクリプションを続行する必要があるかどうかを指定できます。

vRealize Automation Cloud Assembly で提供されるイベント トピック

vRealize Automation Cloud Assembly には、事前定義済みのイベント トピックが含まれています。

イベント トピック

イベント トピックは、同種のイベントをグループ化するためのカテゴリです。イベント トピックをサブスクリプションに割り当てると、そのサブスクリプションをトリガするイベントが定義されます。デフォルトでは、次のイベント トピックが vRealize Automation Cloud Assembly によって提供されます。すべてのトピックを使用して、リソースのカスタム プロパティやタグを追加または更新できます。vRealize Orchestrator ワークフローまたは拡張性アクションが失敗すると、対応するタスクも失敗します。

表 6-4. Cloud Assembly のイベント トピック

イベント トピック	ブロック可能	説明
Cloud template configuration	いいえ	クラウド テンプレートの作成や削除などのクラウド テンプレート構成イベントが発生したときに発行されます。このイベント トピックは、このようなイベントを外部システムに通知するのに役立ちます。
Cloud template version configuration	いいえ	新しいクラウド テンプレートのバージョン管理イベント（バージョンの作成、リリース、リリース解除、またはリストアなど）が発生したときに発行されます。このイベント トピックは、サードパーティのバージョン管理システムの統合に役立つ場合があります。
Compute allocation	はい	<code>resourcenames</code> および <code>hostselections</code> の割り当て前に発行されます。これらのプロパティは、いずれもこの段階で変更できます。
Compute post provision	はい	リソースが正常にプロビジョニングされた後に発行されます。
Compute post removal	はい	コンピューティング リソースが削除された後に発行されます。
Compute provision	はい	リソースがハイパーバイザー レイヤーでプロビジョニングされる前に発行されます。 注： 割り当てられた IP アドレスは変更できません。
Compute removal	はい	リソースが削除される前に発行されます。
Compute reservation	はい	予約時に発行されます。 注： 配置順序を変更できます。
Deployment action completed	はい	展開アクションが完了した後に発行されます。

表 6-4. Cloud Assembly のイベント トピック （続き）

イベント トピック	ブロック可能	説明
Deployment action requested	はい	展開アクションが完了する前に発行されます。
Deployment completed	はい	クラウド テンプレートまたはカタログ申請の展開後に発行されます。
Deployment onboarded	いいえ	新しい展開がオンボーディングされたときに発行されます。
Deployment requested	はい	クラウド テンプレートまたはカタログ申請の展開前に発行されます。
Deployment resource action completed	はい	リソース アクションの展開後に発行されます。
Deployment resource action requested	はい	リソース アクションの展開前に発行されます。
Deployment resource completed	はい	展開リソースのプロビジョニング後に発行されます。
Deployment resource requested	はい	展開リソースのプロビジョニング前に発行されます。
Disk allocation	はい	ディスク リソースの事前割り当てに対して発行されます。
Disk attach	はい	<p>ディスクがマシンに接続される前に発行されます。Disk attach は読み取りと書き込みのイベントです。書き戻しでサポートされるディスク プロパティは次のとおりです。</p> <ul style="list-style-type: none"> ■ diskFullPaths ■ diskDatastoreNames ■ diskParentDirs <p>更新のためには、vSphere 固有の 3 つのディスク プロパティすべてが必要です。他のすべてのプロパティは読み取り専用です。</p> <p>注： 書き戻しは、vSphere First Class Disk ではオプションです。</p>
Disk detach	はい	ディスクがマシンから切断された後に発行されます。Disk detach は読み取り専用のイベントです。
Disk post removal	はい	ディスク リソースが削除された後に発行されます。
Disk post resize	はい	ディスク リソースのサイズが変更された後に発行されます。
EventLog	はい	ログ関連のイベントに対して発行されます。
Kubernetes cluster allocation	はい	Kubernetes クラスタのリソースの事前割り当てに対して発行されます。
Kubernetes cluster post provision	はい	Kubernetes クラスタがプロビジョニングされた後に発行されます。

表 6-4. Cloud Assembly のイベント トピック (続き)

イベント トピック	ブロック可能	説明
Kubernetes cluster post removal	はい	Kubernetes クラスタが削除された後に発行されます。
Kubernetes cluster provision	はい	Kubernetes クラスタがプロビジョニングされる前に発行されます。
Kubernetes cluster removal	はい	Kubernetes クラスタを削除するプロセスが開始される前に発行されます。
Load balancer post provision	はい	ロード バランサのプロビジョニング後に発行されます。
Load balancer post removal	はい	ロード バランサの削除後に発行されます。
Load balancer provision	はい	ロード バランサのプロビジョニング前に発行されます。
Load balancer removal	はい	ロード バランサの削除前に発行されます。
Network Configure	はい	コンピューティングの割り当て中にネットワークが構成されたときに発行されます。 注： ネットワーク構成のトピックでは、複数の IP アドレスまたは NIC がサポートされています。
Network post provisioning	はい	ネットワーク リソースがプロビジョニングされた後に発行されます。
Network post removal	はい	ネットワーク リソースが削除された後に発行されます。
Network provisioning	はい	ネットワーク リソースがプロビジョニングされる前に発行されます。
Network removal	はい	ネットワーク リソースが削除される前に発行されます。
Security group post provisioning	はい	セキュリティ グループがプロビジョニングされた後に発行されます。
Security group post removal	はい	セキュリティ グループが削除された後に発行されます。
Security group provisioning	はい	セキュリティ グループがプロビジョニングされる前に発行されます。
Security group removal	はい	セキュリティ グループが削除される前に発行されます。
Project Lifecycle	いいえ	プロジェクトが作成、更新、または削除されたときに発行されるイベント。

イベントのパラメータ

イベント トピックを追加すると、追加したイベント トピックのパラメータが表示されます。これらのイベント パラメータは、イベントのパayload構造、つまり `inputProperties` を定義します。特定のイベント パラメータは変更できず、読み取り専用としてマークされます。これらの読み取り専用のパラメータを識別するには、パラメータの右側にある情報アイコンをクリックします。

拡張性イベント ログ

[拡張性イベント] 画面には、使用している環境内で発生したすべてのイベントのリストが表示されます。

拡張性イベント ログを表示するには、[拡張性] - [イベント] の順に選択します。また、1 つまたは複数のプロパティを使用して、イベントのリストをフィルタすることもできます。個々のイベントの詳細をさらに表示するには、イベントの ID を選択します。

ID	Timestamp	Event Topic	User Name	Target ID	Description
cb156ce-a324-f5ae-5dd1-66d1e59f1fa6	04/28/20, 1:10 PM	N/A	N/A	endpoints	CREATE
efe21151-2906-dce2-14ab-68c17132d756	03/25/20, 4:22 PM	N/A	N/A	endpoints	CREATE
468e8e55-cf27-e77e-0179-1b5b736717b3	03/25/20, 10:12 AM	N/A	N/A	endpoints	CREATE
d9482883-d1ae-5899-fb06-852c202ec178	03/20/20, 2:41 PM	N/A	N/A	endpoints	CREATE
38584d40-p663-631f-7098-3747aa528d12	01/30/20, 5:35 PM	N/A	N/A	endpoints	CREATE

拡張サブスクリプションの作成

vRealize Automation Cloud Assembly で vRealize Orchestrator 統合または拡張性アクションを使用すると、アプリケーションを拡張するためのサブスクリプションを作成できます。

拡張性サブスクリプションを使用すると、特定のライフサイクル イベントでワークフローまたはアクションをトリガして、アプリケーションを拡張できます。また、フィルタをサブスクリプションに適用して、指定したイベントのブール条件を設定することもできます。たとえば、ブール式が `'true'` の場合にのみ、イベントおよびワークフローまたはアクションがトリガされるようにします。これは、イベント、アクション、またはワークフローがトリガされるタイミングを制御するのに役立ちます。

前提条件

- クラウド管理者のユーザー ロール
- vRealize Orchestrator のワークフローを使用している場合：
 - 組み込みの vRealize Orchestrator クライアントのライブラリ、または任意の統合された外部 vRealize Orchestrator インスタンスのライブラリ。
- 拡張性アクションを使用している場合：
 - 既存の拡張性アクション スクリプト。詳細については、[拡張性アクションの作成方法を参照してください](#)。

手順

- 1 [拡張性] - [サブスクリプション] の順に選択します。
- 2 [新しいサブスクリプション] をクリックします。

- 3 サブスクリプションの詳細を入力します。
- 4 [イベント トピック] を選択します。
- 5 (オプション) イベント トピックの条件を設定します。

注： 条件は、javascript 構文式を使用して作成できます。この式には、"&&" (AND)、"||" (OR)、"^" (XOR)、"!" (NOT) などのブール演算子を使用できます。"==" (equal to)、"!=" (not equal to)、">=" (greater than or equal)、"<=" (less than or equal)、">" (greater than)、"<" (lesser than) などの算術演算子も使用できます。シンプルな式を基礎として複雑なブール式を構築できます。指定されたトピック パラメータに従ってイベントのペイロード（データ）にアクセスするには、eventType、またはイベントのヘッダー プロパティ (eventId、correlationType、correlationId、description、targetType、targetId、userName、orgId、'event.data'、 sourceType、sourceIdentity、timestamp) のいずれかを使用します。

- 6 [アクション/ワークフロー] で、拡張性サブスクリプションに対して実行可能な項目を選択します。
- 7 (オプション) 必要に応じて、イベント トピックのブロック動作を構成します。
- 8 (オプション) 拡張性サブスクリプションのプロジェクトの範囲を定義するには、[任意のプロジェクト] を無効にして、[プロジェクトの追加] をクリックします。
- 9 サブスクリプションを保存するには、[保存] をクリックします。

結果

サブスクリプションが作成されます。選択したイベント トピックで分類されたイベントが発生すると、リンク先の vRealize Orchestrator ワークフローまたは拡張性アクションが開始され、すべてのサブスクリバに通知されます。

次のステップ

サブスクリプションを作成したら、そのサブスクリプションをリンクして使用するためのクラウド テンプレートを作成または展開できます。また、vRealize Automation Cloud Assembly 内の [拡張性] タブで、実行されたワークフローのステータスを確認できます。サブスクリプションに vRealize Orchestrator ワークフローが含まれている場合には、vRealize Orchestrator クライアントから実行とワークフロー ステータスを監視することもできます。

拡張性サブスクリプションのトラブルシューティング

拡張サブスクリプションの障害のトラブルシューティングを実行します。

サブスクリプションが失敗した場合、よくある原因にワークフローや拡張性アクション スクリプトのエラーがあります。

トピックのパラメータとペイロードの表示

サブスクリプション トピック パラメータのダンプ スクリプトを使用して、特定のイベント ステージで、仮想マシンの特定のパラメータとペイロードを表示できます。

主にこのスクリプトは、vRealize Orchestrator ワークフローで使用可能な入力をデバッグおよび確認する際に役立ちます。仮想マシンのすべてのパラメータを表示するには、ワークフローで次のスクリプトを使用します。

```
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")
```

サブスクリプション バージョン履歴

サブスクリプションが失敗した場合は、バージョン履歴を表示できます。

サブスクリプション バージョン履歴の表示

[バージョン履歴] タブには、サブスクリプションの変更履歴が、変更したユーザーと変更日付とともに表示されます。サブスクリプションが失敗した場合や正しく動作しない場合は、バージョン履歴を見ると、原因を特定できることがあります。

The screenshot shows the vRealize Orchestrator interface. In the sidebar, the 'サブスクリプション' (Subscriptions) option is selected, indicated by a blue circle with the number 1. The main area displays the 'Test subscription' version history, indicated by a blue circle with the number 2. The version history table shows three entries, with the most recent one selected, indicated by a blue circle with the number 3. The selected entry shows a date of 20/01/13 15:08 and a user of rishi@vmware.com. To the right of the table, a JSON object is displayed, showing the configuration details for the subscription, including its ID, type, event topic ID, name, org ID, owner ID, subscriber ID, blocking status, description, criteria, constraints, project ID, timeout, and broadcast status.

1

[サブスクリプション] タブからサブスクリプションを開きます。

2

バージョン履歴を表示するには、[バージョン履歴] をクリックします。

3

各変更エントリをクリックすると、その変更に関連付けられているサブスクリプション コードを表示できます。

vRealize Automation リソースのプロパティについて

vRealize Automation infrastructure-as-code エディタでは、クリックするかカーソルを合わせることで構文とコードの補完ヘルプを利用できます。クラウド テンプレート リソース プロパティの完全なセット（カスタム プロパティとも呼ばれる）を確認するには、統合リソース スキーマを参照します。

スキーマは VMware {code} サイトで利用可能です。リンク先にアクセスし、[Models] をクリックし、クラウド テンプレート（旧称はブループリント）で使えるリソース オブジェクトを一覧表示します。

- [VMware {code} の vRealize Automation Resource Type Schema](#)

vRealize Automation Cloud Assembly コードの例

vRealize Automation Cloud Assembly のクラウド テンプレートコードは、組み合わせとアプリケーションにおいてほとんど制限がありません。

多くの場合、コードの成功例は、今後の開発のための最適な開始点になります。例を使用する場合、リソース名、値などについては、サイトで設定されているものに置き換えてください。

vRealize Automation Cloud Assembly クラウド テンプレートの vSphere リソースの例

以下は、vRealize Automation Cloud Assembly クラウド テンプレート内の vSphere マシン リソースを示すコード例です。

リソース	クラウド テンプレートの例
CPU、メモリ、オペレーティング システムを含む vSphere 仮想マシン	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine cpuCount: 1 totalMemoryMB: 1024 image: ubuntu</pre>
データストア リソースを含む vSphere マシン	<pre>resources: demo-vsphere-disk-001: type: Cloud.vSphere.Disk properties: name: DISK_001 type: 'HDD' capacityGb: 10 dataStore: 'datastore-01' provisioningType: thick</pre>
ディスクが接続された vSphere マシン	<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>

リソース	クラウド テンプレートの例
<p>動的な数のディスクを持つ vSphere マシン</p>	<pre>inputs: disks: type: array title: disks items: title: disk type: object properties: size: type: integer title: size maxItems: 15 resources: Cloud_Machine_1: type: Cloud.vSphere.Machine properties: image: centos7 flavor: small attachedDisks: '\$ {map to object (resource.Cloud_Volume_1[*].id, "source")}' Cloud_Volume_1: type: Cloud.Volume allocatePerInstance: true properties: capacityGb: '\${input.disks[count.index].size}' count: '\${length(input.disks)}'</pre>
<p>スナップショット イメージからの vSphere マシン。スラッシュとスナップショット名を付加します。スナップショット イメージはリンク クローンにすることができます。</p>	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: imageRef: 'demo-machine/snapshot-01' cpuCount: 1 totalMemoryMB: 1024</pre>
<p>vCenter 内の特定のフォルダ内の vSphere マシン</p>	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine cpuCount: 2 totalMemoryMB: 1024 imageRef: ubuntu resourceGroupName: 'myFolder'</pre>

リソース	クラウド テンプレートの例
複数の NIC を含む vSphere マシン	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: image: ubuntu flavor: small networks: - network: '\${network-01.name}' deviceIndex: 0 - network: '\${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>
vCenter にタグが接続された vSphere マシン	<pre> resources: demo-machine: type: Cloud.vSphere.Machine properties: flavor: small image: ubuntu tags: - key: env value: demo </pre>

リソース	クラウド テンプレートの例
カスタマイズ仕様を含む vSphere マシン	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: name: demo-machine image: ubuntu flavor: small customizationSpec: Linux</pre>
リモート アクセスでの vSphere マシン	<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>

ドキュメント化された vRealize Automation Cloud Assembly テンプレートの例

この例には、完全なコメントのセットが含まれています。これにより、vRealize Automation Cloud Assembly テンプレート（旧称はブループリント）の各セクションの構造と目的を確認できます。

```
# *****
#
# This WordPress cloud template 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 cloud template deploys a minimum of 3 virtual machines and runs scripts
# to install packages.
#
# *****
#
# -----
# Templates need a descriptive name and version if
# source controlled in git.
# -----
name: WordPress Template 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 cloud template 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:
#       - 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 extensiblity 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:
#       - 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 template useful. Note that
# you can also access an API to create templates, 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 クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例

クラウド テンプレートのデザインおよび展開では、ネットワーク、セキュリティ、およびロード バランサのリソースと設定を使用できます。

クラウド テンプレート デザイン コードのオプションの概要については、[vRealize Automation Resource Type Schema](#) を参照してください。

関連情報については、次を参照してください。

- [vRealize Automation クラウド テンプレートでのネットワーク リソースの使用](#)
- [vRealize Automation クラウド テンプレートでのセキュリティ グループ リソースの使用](#)
- [vRealize Automation クラウド テンプレートでのロード バランサ リソースの使用](#)

これらの例では、基本的なクラウド テンプレート デザイン内のネットワーク、セキュリティ グループ、およびロード バランサのサンプル リソースを示しています。

リソース シナリオ	クラウド テンプレート デザイン コードの例
NSX ネットワーク リソースに複数の NIC が関連付けられている vSphere マシン。	<pre>resources: demo-machine: type: Cloud.vSphere.Machine properties: image: ubuntu flavor: small networks: - network: '\$ {resource.Cloud_vSphere_Network_1.id}' Cloud_vSphere_Network_1: type: Cloud.vSphere.Network properties: networkType: existing Cloud_vSphere_Network_2: type: Cloud.NSX.Network properties: networkType: existing</pre>
送信ネットワーク上で Cloud.NSX.Gateway クラウド テンプレート リソースを使用して、NAT ポート転送を有効にします。	<pre>... gateway: type: Cloud.NSX.Gateway properties: networks: - \${resource.out.id} natRules: - index: 1 translatedInstance: \$ {resource.jumpbox.networks[0].id} destinationPorts: 2200 translatedPorts: 22 description: inbound ssh - index: 2 ...</pre>

リソース シナリオ	クラウド テンプレート デザイン コードの例
<p>ロード バランシングのログ レベル、アルゴリズム、およびサイズを指定します。</p>	<p>ログ レベル、アルゴリズム、およびサイズの使用法を示すサンプル NSX ロード バランサ :</p> <pre>resources: Cloud_LoadBalancer_1: type: Cloud.NSX.LoadBalancer properties: name: myapp-lb network: '\${appnet-public.name}' instances: '\${wordpress.id}' routes: - protocol: HTTP port: '80' loggingLevel: CRITICAL algorithm: LEAST_CONNECTION type: MEDIUM</pre>
<p>指定されたマシンまたは指定されたマシン NIC にロード バランサを関連付けます。machine ID または machine network ID を指定して、ロード バランサ プールにマシンを追加できます。instances プロパティは、マシン (machine by ID) と NIC (machine by network ID) の両方をサポートします。</p> <p>1 番目の例では、machine by ID 設定を使用することにより、マシンがどのネットワークに展開されてもロード バランシングを行います。</p> <p>2 番目の例では、machine by network ID 設定を使用することにより、マシンが指定されたマシン NIC に展開されている場合にのみロード バランシングを行います。</p> <p>3 番目の例は、同じ instances オプションで使われる両方の設定を示しています。</p>	<p>instances プロパティを使用して、マシン ID またはマシン ネットワーク ID を定義できます。</p> <p>■ マシン ID</p> <pre>Cloud_LoadBalancer_1: type: Cloud.LoadBalancer properties: network: '\${resource.Cloud_Network_1.id}' instances: '\$ {resource.Cloud_Machine_1.id}'</pre> <p>■ マシン ネットワーク ID</p> <pre>Cloud_LoadBalancer_1: type: Cloud.LoadBalancer properties: network: '\${resource.Cloud_Network_1.id}' instances: '\$ {resource.Cloud_Machine_1.networks[0].id}'</pre> <p>■ ロード バランサに含めるように指定された 1 台のマシンと、ロード バランサに含めるように指定された別のマシン NIC :</p> <pre>instances: - resource.Cloud_Machine_1.id - resource.Cloud_Machine_2.networks[2].id</pre>
<p>パブリック IP アドレスではなく内部 IP アドレスを使用するためのパブリック クラウド マシン。この例では、特定のネットワーク ID を使用します。</p> <p>注：ターゲット ネットワーク ID を指定するには、networks: 設定で network: オプションを使用します。networks: 設定の name: オプションは廃止されているため、使用しないでください。</p>	<pre>resources: wf_proxy: type: Cloud.Machine properties: image: ubuntu 16.04 flavor: small constraints: - tag: 'platform:vsphere' networks: - network: '\${resource.wf_net.id}' assignPublicIpAddress: false</pre>

リソース シナリオ	クラウド テンプレート デザイン コードの例
<p>NSX ネットワーク リソース タイプを使用した NSX-V または NSX-T 用のルーティング ネットワーク。</p>	<pre>Cloud_NSX_Network_1: type: Cloud.NSX.Network properties: networkType: routed</pre>
<p>クラウド テンプレート内のマシン NIC リソースにタグを追加します。</p>	<pre>formatVersion: 1 inputs: {} resources: Cloud_Machine_1: type: Cloud.vSphere.Machine properties: flavor: small image: ubuntu networks: - name: '\${resource.Cloud_Network_1.name}' deviceIndex: 0 tags: - key: 'nic0' value: null - key: internal value: true - name: '\${resource.Cloud_Network_2.name}' deviceIndex: 1 tags: - key: 'nic1' value: null - key: internal value: false</pre>
<p>送信ネットワークに NSX-T 論理スイッチをタグ付けします。</p> <p>NSX-T および VMware Cloud on AWS ではタグ付けがサポートされています。</p> <p>このシナリオの詳細については、コミュニティブログの記事 Creating Tags in NSX with Cloud Assembly を参照してください。</p>	<pre>Cloud_NSX_Network_1: type: Cloud.NSX.Network properties: networkType: outbound tags: - key: app value: opencart</pre>

リソース シナリオ

クラウド テンプレート デザイン コードの例

マシン NIC に適用された制約タグのある既存のセキュリティ グループ。

既存のセキュリティ グループを使用するには、コンポーネントの securityGroupType プロパティに *existing* と入力します。

タグを Cloud.SecurityGroup リソースに割り当て、タグ制約を使用して、既存のセキュリティ グループを割り当てることができます。タグを含まないセキュリティ グループはクラウド テンプレート デザインで使用することはできません。

制約タグは、securityGroupType: existing セキュリティ グループのリソースに対して設定する必要があります。これらの制約は、既存のセキュリティ グループに対して設定されているタグと一致する必要があります。

securityGroupType: new セキュリティ グループのリソースに制約タグを設定することはできません。

```
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
```

リソース シナリオ

クラウド テンプレート デザイン コードの例

Allow と Deny アクセス オプションを示す 2 つのファイアウォール ルールが設定されているオンデマンド セキュリティ グループ。

```
resources:
  Cloud_SecurityGroup_1:
    type: Cloud.SecurityGroup
    properties:
      securityGroupType: new
      rules:
        - ports: 5000
          source:
            'fc00:10:000:000:000:56ff:fe89:48b4'
            access: Allow
            direction: inbound
            name: allow_5000
            protocol: TCP
        - ports: 7000
          source:
            'fc00:10:000:000:000:56ff:fe89:48b4'
            access: Deny
            direction: inbound
            name: deny_7000
            protocol: TCP
  Cloud_vSphere_Machine_1:
    type: Cloud.vSphere.Machine
    properties:
      image: photon
      cpuCount: 1
      totalMemoryMB: 256
      networks:
        - network: '$
{resource.Cloud_Network_1.id}'
          assignIPv6Address: true
          assignment: static
          securityGroups:
            - '$
{resource.Cloud_SecurityGroup_1.id}'
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      networkType: existing
```

リソース シナリオ

クラウド テンプレート デザイン コードの例

以下を含む、2 つのセキュリティ グループがある複雑なクラウド テンプレート

- 1 つの既存のセキュリティ グループ
- 複数のファイアウォール ルールの例を含む 1 つのオンデマンド セキュリティ グループ
- 1 台の vSphere マシン
- 1 つの既存のネットワーク

このサンプルでは、プロトコルとポート、サービス、送信元と宛先としての IP CIDR、送信元または宛先としての IP アドレス範囲、および any、IPv6、(::/0) のオプションのさまざまな組み合わせを示しています。

マシン NIC の場合は、接続されているネットワークと、セキュリティ グループを指定できます。NIC インデックスまたは IP アドレスを指定することもできます。

```
formatVersion: 1
inputs: {}
resources:
  DEMO_ESG : existing security group - security
group 1)
    type: Cloud.SecurityGroup
    properties:
      constraints:
        - tag: BlockAll
      securityGroupType: existing (designation
of existing for security group 1)
  DEMO_ODSG: (on-demand security group -
security group 2))
    type: Cloud.SecurityGroup
    properties:
      rules: (multiple firewall rules in this
section)
      - name: IN-ANY (rule 1)
        source: any
        service: any
        direction: inbound
        access: Deny
      - name: IN-SSH (rule 2)
        source: any
        service: SSH
        direction: inbound
        access: Allow
      - name: IN-SSH-IP (rule 3)
        source: 33.33.33.1-33.33.33.250
        protocol: TCP
        ports: 223
        direction: inbound
        access: Allow
      - name: IPv-6-ANY-SOURCE (rule 4)
        source: '::/0'
        protocol: TCP
        ports: 223
        direction: inbound
        access: Allow
      - name: IN-SSH-IP (rule 5)
        source: 44.44.44.1/24
        protocol: UDP
        ports: 22-25
        direction: inbound
        access: Allow
      - name: IN-EXISTING-SG (rule 6)
        source: '${resource["DEMO_ESG"].id}'
        protocol: ICMPv6
        direction: inbound
        access: Allow
      - name: OUT-ANY (rule 7)
        destination: any
        service: any
        direction: outbound
        access: Deny
      - name: OUT-TCP-IPv6 (rule 8)
        destination:
'2001:0db8:85a3::8a2e:0370:7334/64'
        protocol: TCP
        ports: 22
        direction: outbound
        access: Allow
```

リソース シナリオ

クラウド テンプレート デザイン コードの例

```

- name: IPv6-ANY-DESTINATION (rule 9)
  destination: '::/0'
  protocol: UDP
  ports: 23
  direction: outbound
  access: Allow
- name: OUT-UDP-SERVICE (rule 10)
  destination: any
  service: NTP
  direction: outbound
  access: Allow
securityGroupType: new (designation of on-
demand for security group 2)
DEMO_VC_MACHINE: (machine resource)
  type: Cloud.vSphere.Machine
  properties:
    image: PHOTON
    cpuCount: 1
    totalMemoryMB: 1024
    networks: (Machine network NICs)
  - network: '${resource.DEMO_NW.id}'
securityGroups: - '${resource.DEMO_ODSG.id}' -
                '${resource.DEMO_ESG.id}'
DEMO_NETWORK: (network resource)
  type: Cloud.vSphere.Network
  properties:
    networkType: existing
  constraints:
    - tag: nsx62

```

リソース シナリオ

クラウド テンプレート デザイン コードの例

1 アームのロード バランサを備えた
オンデマンド ネットワーク。

```
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
```

ロード バランサを使用する既存の
ネットワーク。

```
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:
```

リソース シナリオ

クラウド テンプレート デザイン コードの例

```

- 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

```

詳細情報

ネットワークとセキュリティの実装シナリオについては、次の VMware ブログを参照してください。

- [\[vRealize Automation Cloud Assembly Load Balancer with NSX-T Deep Dive\]](#)
- [Network Automation with Cloud Assembly and NSX – Part 1](#) (NSX-T と vCenter のクラウド アカウントおよびネットワーク CIDR の使用を含む場合)
- [Network Automation with Cloud Assembly and NSX – Part 2](#) (既存の送信ネットワーク タイプの使用を含む場合)
- [Network Automation with Cloud Assembly and NSX – Part 3](#) (既存およびオンデマンドのセキュリティ グループの使用を含む場合)
- [Network Automation with Cloud Assembly and NSX – Part 4](#) (既存およびオンデマンドのロード バランサの使用を含む場合)

vRealize Automation クラウド テンプレートでのネットワーク リソースの使用

vRealize Automation クラウド テンプレート デザインを作成または編集するときには、目的に最適なネットワーク リソースを使用します。NSX と、クラウド テンプレートで使用可能なクラウドに依存しないネットワーク オプションについて確認します。

vRealize Automation クラウド テンプレートで、マシンおよび関連する条件に基づいて、適用可能なネットワーク リソース タイプから 1 つを選択します。

クラウドに依存しないネットワーク リソース

クラウドに依存しないネットワークを追加するには、クラウド テンプレートの [デザイン] 画面で、[クラウドに依存しない] - [ネットワーク] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.Network` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_Network_1:
  type: Cloud.Network
  properties:
    networkType: existing
```

NSX ネットワークに接続されていない、または接続されない可能性があるターゲット マシン タイプのネットワーク特性を指定する場合は、クラウドに依存しないネットワークを使用します。

クラウドに依存しないネットワーク リソースは、次のリソース タイプで使用できます。

- クラウドに依存しないマシン
- vSphere
- Google Cloud Platform (GCP)
- Amazon Web Services (AWS)
- Microsoft Azure
- VMware Cloud on AWS (VMC)

クラウドに依存しないネットワーク リソースは、次のネットワーク タイプ (`networkType`) 設定で使用できます。

- パブリック
- プライベート
- 送信
- 既存

vSphere ネットワーク リソース

vSphere ネットワークを追加するには、クラウド テンプレートの [デザイン] 画面で [vSphere] - [ネットワーク] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.vSphere.Network` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_vSphere_Network_1:
  type: Cloud.vSphere.Network
  properties:
    networkType: existing
```

vSphere マシン タイプ (`Cloud.vSphere.Machine`) のネットワーク特性を指定する場合は、vSphere ネットワークを使用します。

vSphere ネットワーク リソースは、`Cloud.vSphere.Machine` マシン タイプでのみ使用できます。

vSphere リソースは、次のネットワーク タイプ (`networkType`) 設定で使用できます。

- パブリック

- プライベート
- 既存

ネットワーク タイプの詳細については、[vRealize Automation のネットワーク プロファイルおよびクラウド テンプレートでのネットワーク設定の使用](#)を参照してください。

NSX ネットワーク リソース

NSX ネットワークを追加するには、クラウド テンプレートの [デザイン] 画面で [NSX] - [ネットワーク] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.NSX.Network` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_NSX_Network_1:
  type: Cloud.NSX.Network
  properties:
    networkType: existing
```

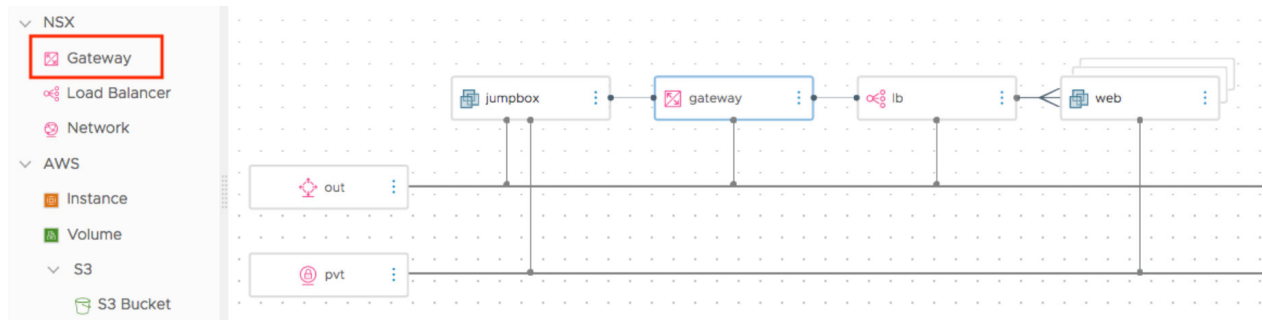
NSX-V または NSX-T クラウド アカウントに関連付けられている 1 台以上のマシンにネットワーク リソースを接続する場合は、NSX ネットワークを使用します。NSX ネットワーク リソースを使用すると、NSX-V または NSX-T クラウド アカウントに関連付けられている vSphere マシン リソースの NSX ネットワーク特性を指定できます。

`Cloud.NSX.Network` リソースは、次のネットワーク タイプ (`networkType`) 設定で使用できます。

- パブリック
- プライベート
- 送信
- 既存
- ルーティング - ルーティング ネットワークは、NSX-V と NSX-T でのみ使用できます。

各オンデマンド NSX-T ネットワークによって、新しい Tier-1 論理ルーターが作成されます。各オンデマンド NSX-V ネットワークによって、新しい Edge が作成されます。

NAT ルールと NAT ポート転送をサポートするには、`Cloud.NSX.Gateway` クラウド テンプレート リソースを追加して、送信 NSX-V または NSX-T ネットワークに接続されているゲートウェイ/ルーターに DNAT ルールを指定することができます。ゲートウェイは単一の送信ネットワークに接続する必要があり、同じ送信ネットワークに接続されている複数のマシンまたはロード バランサに接続できます。ゲートウェイ内で指定された DNAT ルールは、これらのマシンまたはロード バランサをターゲットとして参照します。NAT ルールは、クラスタ化されたマシンに指定することはできません。ただし、Day 2 操作としてクラスタ内のマシンに個別に指定することができます。



関連情報については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

外部 IP アドレス管理統合オプション

クラウド テンプレートのデザインおよび展開で Infoblox IP アドレス管理統合と共に使用できるプロパティの詳細については、[vRealize Automation での IP アドレス管理統合における Infoblox 固有のプロパティと拡張属性の使用](#)を参照してください。

利用可能な Day 2 操作

クラウド テンプレートおよび展開リソースで使用できる一般的な Day 2 操作のリストについては、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)を参照してください。

あるネットワークから別のネットワークに移動する方法の例については、[展開されたマシンを別のネットワークに移動する方法](#)を参照してください。

詳細情報

ネットワーク リソースの定義の詳細については、[vRealize Automation のネットワーク リソース](#)を参照してください。

ネットワーク プロファイルの定義の詳細については、[vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください。

サンプルのネットワーク リソースと設定を示すクラウド テンプレート デザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

vRealize Automation クラウド テンプレートでのセキュリティ グループ リソースの使用

vRealize Automation クラウド テンプレートを作成または編集するときには、目的に最適なセキュリティ グループ リソースを使用します。クラウド テンプレートで使用可能なセキュリティ グループ オプションについて確認します。

クラウドに依存しないセキュリティ グループ リソース

現在、セキュリティ グループ リソースのタイプは1つのみです。セキュリティ グループ リソースを追加するには、クラウド テンプレートの [デザイン] 画面で、[クラウドに依存しない] - [セキュリティ グループ] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.SecurityGroup` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_SecurityGroup_1:
  type: Cloud.SecurityGroup
  properties:
    constraints: []
    securityGroupType: existing
```

クラウド テンプレート デザインで、既存 (`securityGroupType: existing`) またはオンデマンド (`securityGroupType: new`) のいずれかとして、セキュリティ グループ リソースを指定します。

既存のセキュリティ グループをクラウド テンプレート デザインに直接追加することも、ネットワーク プロファイルに追加されている既存のセキュリティ グループを使用することもできます。既存のセキュリティ グループは、さまざまなタイプのクラウド アカウントでサポートされています。

NSX-V および NSX-T については、クラウド テンプレートのデザインまたは変更時に、既存のセキュリティ グループを追加するか、新しいセキュリティ グループを定義することができます。オンデマンド セキュリティ グループは、NSX-T と NSX-V でのみサポートされます。

Microsoft Azure を除くすべてのクラウド アカウント タイプでは、1つ以上のセキュリティ グループをマシン NIC に関連付けることができます。Microsoft Azure の仮想マシン NIC (`machineName`) は、1つのセキュリティ グループにのみ関連付けることができます。

デフォルトでは、セキュリティ グループ プロパティ `securityGroupType` は `existing` に設定されています。オンデマンド セキュリティ グループを作成するには、`securityGroupType` プロパティに `new` と入力します。オンデマンド セキュリティ グループのファイアウォール ルールを指定するには、セキュリティ グループ リソースの `Cloud.SecurityGroup` セクションにある `rules` プロパティを使用します。

既存のセキュリティ グループ

既存のセキュリティ グループは、NSX-T や Amazon Web Services などのソース クラウド アカウント リソース内に作成されます。これらは、vRealize Automation によってソースから収集されるデータです。vRealize Automation ネットワーク プロファイルの一部として、使用可能なリソースのリストから既存のセキュリティ グループを選択できます。クラウド テンプレート デザインでは、既存のセキュリティ グループを、指定したネットワーク プロファイルのメンバーシップによって暗黙的に指定することも、セキュリティ グループ リソースの `securityGroupType: existing` 設定を使用して名前で指定することもできます。ネットワーク プロファイルにセキュリティ グループを追加する場合は、少なくとも1つの機能タグをネットワーク プロファイルに追加します。オンデマンド セキュリティ グループ リソースをクラウド テンプレート デザインで使用するときは、制約タグが必要です。

クラウド テンプレート デザインのセキュリティ グループ リソースを 1 つ以上のマシン リソースに関連付けることができます。

注： クラウド テンプレート デザインでマシン リソースを使用して Microsoft Azure の仮想マシン NIC (machineName) にプロビジョニングする場合、マシン リソースを 1 つのセキュリティ グループにのみ関連付ける必要があります。

オンデマンドの NSX-V および NSX-T セキュリティ グループ

クラウド テンプレート デザインを定義または変更するときに、セキュリティ グループ リソース コードの `securityGroupType: new` 設定を使用して、オンデマンド セキュリティ グループを定義できます。

オンデマンドの NSX-V または NSX-T セキュリティ グループを使用して、特定のファイアウォール ルールのセットをネットワーク マシン リソースまたはグループ化されたリソースのセットに適用することができます。各セキュリティ グループには、複数の名前付きファイアウォール ルールを含めることができます。オンデマンド セキュリティ グループを使用して、サービスまたはプロトコル、およびポートを指定することができます。指定できるのはサービスまたはプロトコルのどちらかとなり、両方を指定することはできないことに注意してください。プロトコルを指定した場合、ポートも指定できます。サービスを指定した場合は、ポートを指定することはできません。ルールにサービスもプロトコルも含まれていない場合、デフォルトのサービス値は [任意] です。

また、ファイアウォール ルールで IP アドレスと IP アドレス範囲を指定することもできます。ファイアウォール ルールの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

NSX-V または NSX-T オンデマンド セキュリティ グループでファイアウォール ルールを作成する場合、デフォルトでは、指定したネットワーク トラフィックが許可されますが、他のネットワーク トラフィックも許可されます。ネットワーク トラフィックを制御するには、各ルールのアクセス タイプを指定する必要があります。ルールのアクセス タイプは次のとおりです。

- 許可 (デフォルト) - このファイアウォール ルールで指定されているネットワーク トラフィックを許可します。
- 拒否 - このファイアウォール ルールで指定されているネットワーク トラフィックをブロックします。接続が拒否されたことをクライアントにアクティブに通知します。
- ドロップ - このファイアウォール ルールで指定されているネットワーク トラフィックを拒否します。リスナーがオンラインでない場合と同様に、パケットをサイレントにドロップします。

`access: Allow` と `access: Deny` のファイアウォール ルールを使用するデザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

注： クラウド管理者は、NSX オンデマンド セキュリティ グループのみを含むクラウド テンプレート デザインを作成できます。また、そのデザインを展開して、再利用可能な既存のセキュリティ グループ リソースを作成することができます。このリソースは、組織のメンバーがネットワーク プロファイルおよびクラウド テンプレート デザインに既存のセキュリティ グループとして追加できます。

ファイアウォール ルールは、送信元と宛先の IP アドレスとして IPv4 または IPv6 形式の CIDR 値をサポートします。ファイアウォール ルールで IPv6 CIDR 値を使用するデザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

オンデマンド セキュリティ グループ ファイアウォール ルールでのアプリケーションの隔離ポリシーの使用

アプリケーションの隔離ポリシーを使用すると、クラウド テンプレートによってプロビジョニングされたリソース間の内部トラフィックのみを許可することができます。アプリケーションの隔離を行うと、クラウド テンプレートによってプロビジョニングされたマシンは相互に通信することができますが、ファイアウォールの外部に接続することはできなくなります。アプリケーションの隔離ポリシーは、ネットワーク プロファイル内に作成できます。クラウド テンプレート デザイン内にアプリケーションの隔離を指定することもできます。それには、拒否のファイアウォールルールを含む、またはプライベート ネットワークや送信ネットワークを含むオンデマンド セキュリティ グループを使用します。

作成したアプリケーションの隔離ポリシーは、比較的低い優先度となります。複数のポリシーを適用した場合、重み付けが大きいポリシーが優先されます。

アプリケーションの隔離ポリシーを作成すると、ポリシーには自動生成されたポリシー名が割り当てられます。このポリシーは、関連付けられているリソース エンドポイントおよびプロジェクトに固有の、他のクラウド テンプレート デザインおよびデザイン イテレーションでも再利用できます。アプリケーションの隔離ポリシーの名前は、クラウド テンプレート デザイン コードには表示されませんが、クラウド テンプレート デザインが展開された後に、プロジェクト画面（[インフラストラクチャ] - [管理] - [プロジェクト]）のカスタム プロパティとして表示されます。

プロジェクト内の関連付けられた同じエンドポイントの場合、アプリケーションの隔離のためにオンデマンド セキュリティ グループが必要な展開では、同じアプリケーションの隔離ポリシーを使用できます。ポリシーは、作成されると、削除されません。アプリケーションの隔離ポリシーを指定すると、vRealize Automation は、プロジェクト内で、関連付けられているエンドポイントに関してポリシーを検索します。ポリシーが見つければ再利用し、ポリシーが見つからない場合は作成します。アプリケーションの隔離ポリシー名は、プロジェクトのカスタム プロパティのリストに、初期導入の後にのみ表示されます。

反復的なクラウド テンプレート開発でのセキュリティ グループの使用

反復的な開発で、セキュリティ グループの制約を変更すると、セキュリティ グループがクラウド テンプレートのマシンに関連付けられていない場合は、セキュリティ グループが指定されたとおりにイテレーション内で更新されます。ただし、セキュリティ グループがすでにマシンに関連付けられている場合は、再展開が失敗します。クラウド テンプレートの反復的な開発中や、各再展開間の再関連付け中に、関連付けられたマシンから既存のセキュリティ グループまたは `securityGroupType` リソースのプロパティを接続解除する必要があります。必要なワークフローは次のとおりです。クラウド テンプレートを最初に展開したと想定としています。

- 1 Cloud Assembly テンプレート デザイナで、クラウド テンプレート内の関連付けられているすべてのマシンからセキュリティ グループを接続解除します。
- 2 [既存の展開の更新] をクリックしてテンプレートを再展開します。
- 3 テンプレートで、既存のセキュリティ グループの制約タグまたは `securityGroupType` プロパティを削除します。
- 4 新しいセキュリティ グループの制約タグまたは `securityGroupType` プロパティをテンプレートに追加します。
- 5 新しいセキュリティ グループの制約タグまたは `securityGroupType` プロパティのインスタンスをテンプレート内のマシンに関連付けます。
- 6 [既存の展開の更新] をクリックしてテンプレートを再展開します。

利用可能な Day 2 操作

クラウド テンプレートおよび展開リソースで利用できる一般的な Day 2 操作のリストについては、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)を参照してください。

詳細情報

ネットワークの隔離にセキュリティ グループを使用する方法の関連情報については、[vRealize Automation のセキュリティ リソース](#)を参照してください。

ネットワーク プロファイルでセキュリティ グループ設定を使用する方法については、[vRealize Automation でのネットワーク プロファイルの詳細と vRealize Automation Cloud Assembly のネットワーク プロファイルおよびクラウド テンプレート デザインでのセキュリティ グループ設定の使用](#)を参照してください。

サンプルのセキュリティ リソースと設定を示すクラウド テンプレート デザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

vRealize Automation クラウド テンプレートでのロード バランサ リソースの使用

vRealize Automation クラウド テンプレートを作成または編集するときには、目的に最適なロード バランサ リソースを使用します。

クラウド テンプレートで NSX およびクラウドに依存しないロード バランサ リソースを使用して、展開内のロード バランシングを制御できます。

クラウドに依存しないロード バランサは、複数のクラウドにまたがって展開できます。クラウド固有のロード バランサでは、特定のクラウドやトポロジでのみ使用できる高度な設定や機能を指定できます。クラウド固有のプロパティは、NSX ロード バランサ (Cloud.NSX.LoadBalancer) リソース タイプで使用できます。これらのプロパティをクラウドに依存しないロード バランサ (Cloud.LoadBalancer) に追加すると、Amazon Web Services や Microsoft Azure などのロード バランサがプロビジョニングされている場合、プロパティは無視されますが、NSX-V または NSX-T のロード バランサがプロビジョニングされている場合は認識されます。vRealize Automation クラウド テンプレートの条件に基づいて、使用可能なロード バランサ リソース タイプから 1 つを選択します。

デザイン キャンバスでロード バランサ リソースをセキュリティ グループ リソースに直接接続することはできません。

クラウドに依存しないロード バランサ リソース

すべてのタイプのターゲット マシンにネットワーク特性を指定する場合は、クラウドに依存しないロード バランサを使用します。

クラウドに依存しないロード バランサを追加するには、クラウド テンプレートのデザイン画面で [クラウドに依存しない] - [ロード バランサ] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.LoadBalancer` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_LoadBalancer_1:
  type: Cloud.LoadBalancer
  properties:
    routes: []
    network: ''
    instances: []
    internetFacing: false
```

NSX ロード バランサ リソース

クラウド テンプレートに NSX-V または NSX-T 固有の特性（ポリシー API メソッドまたは Manager API メソッド）が含まれている場合は、NSX ロード バランサを使用します。1 つ以上のロード バランサを NSX-V または NSX-T ネットワークに、あるいは NSX-V または NSX-T ネットワークに関連付けられたマシンに接続できます。

NSX ロード バランサを追加するには、[NSX] - [ロード バランサ] リソースを使用します。リソースは、クラウド テンプレート コードでは `Cloud.NSX.LoadBalancer` リソース タイプとして表示されます。デフォルトのリソースは、次のように表示されます。

```
Cloud_NSX_LoadBalancer_1:
  type: Cloud.NSX.LoadBalancer
  properties:
    routes: []
    network: ''
    instances: []
```

クラウド テンプレート コードのロード バランサ オプション

1 つ以上のロード バランサ リソースをクラウド テンプレートに追加すると、次の設定を指定できるようになります。[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)にいくつかの例を示します。

■ マシンの仕様

名前付きマシン リソースを指定して、ロード バランサ プールに参加させることができます。または、特定のマシン NIC をロード バランサ プールに参加させるように指定することもできます。

このオプションは、[NSX] ロード バランサ リソース (`Cloud.NSX.LoadBalancer`) でのみ使用できます。

このオプションは、現在、`existing` および `public` のネットワーク タイプで使用できます。オンデマンドの `private`、`routed`、`outbound` のネットワークタイプもサポートされています。

■ `resource.Cloud_Machine_1.id`

クラウド テンプレート コードで `Cloud_Machine_1` と指定されているマシンがロード バランサに含まれるように指定します。

■ `resource.Cloud_Machine_2.networks[2].id`

マシン NIC `Cloud_Machine_2.networks[2]` に展開されたとき、クラウド テンプレート コードで `Cloud_Machine_2` と指定されているマシンのみがロード バランサに含まれるように指定します。

■ ログ レベル

ログ レベルの値には、エラー ログの重要度レベルを指定します。オプションは、NONE、EMERGENCY、ALERT、CRITICAL、ERROR、WARNING、INFO、DEBUG、NOTICE です。ログ レベルの値は、クラウド テンプレートのすべてのロード バランサに適用されます。このオプションは、NSX に固有です。親が設定されているロード バランサの場合、親ログ レベルの設定は子のログ レベル設定をオーバーライドします。

関連情報については、NSX 製品ドキュメントの[ロード バランサの追加](#)などのトピックを参照してください。

■ タイプ

ロード バランサのタイプを使用して、スケーリング サイズを指定します。デフォルトは小です。このオプションは、NSX に固有です。親が設定されているロード バランサ場合、親のタイプ設定は子のタイプ設定をオーバーライドします。

■ 小

NSX-V ではコンパクト、NSX-T では小規模に関連付けられます。

■ 中

NSX-V では大、NSX-T では中規模に関連付けられます。

■ 大

NSX-V では特大、NSX-T では大規模に関連付けられます。

■ 特大

NSX-V では超特大、NSX-T では大規模に関連付けられます。

関連情報については、NSX 製品ドキュメントの[ロード バランサ リソースのスケーリング](#)などのトピックを参照してください。

このオプションは、[NSX] ロード バランサ リソース (Cloud.NSX.LoadBalancer) でのみ使用できます。

■ アルゴリズム (サーバ プール)

アルゴリズム バランシング メソッドを使用して、サーバ プール メンバー間で受信接続を分配する方法を制御します。アルゴリズムはサーバ プールで使用することも、サーバで直接使用することもできます。次の条件のいずれかを満たすサーバは、すべてのロード バランシング アルゴリズムでスキップされます。

- 管理状態が「DISABLED」に設定されている
- 管理状態が「GRACEFUL_DISABLED」に設定されていて、一致するパーシステンス エントリがない
- アクティブまたはパッシブ健全性チェックの状態が「DOWN」になっている
- サーバ プールの最大同時接続数の上限に達した

このオプションは、NSX に固有です。

■ IP_HASH

ソース IP アドレスのハッシュ、および実行されているすべてのサーバの重みの合計に基づいて、サーバを選択します。

NSX-V および NSX-T で IP-HASH に対応します。

■ LEAST_CONNECTION

サーバ上の既存の接続数に基づき、クライアント リクエストを複数のサーバに分散します。新しい接続は、接続数が最も少ないサーバに送信されます。サーバ プール メンバーに重みが設定されている場合でも、重みは無視されます。

NSX-V では LEASTCONN、NSX-T では LEAST_CONNECTION に対応します。

■ ROUND_ROBIN

受信したクライアント要求を、要求処理能力がある利用可能なサーバのリストに基づいて持ち回りで処理します。サーバ プール メンバーの重みは設定されていた場合でも無視されます。デフォルト。

NSX-V および NSX-T で ROUND_ROBIN に対応します。

■ WEIGHTED_LEAST_CONNECTION

各サーバには、プール内の他のサーバを基準としてこのサーバの実行方法を示す重み値が割り当てられています。この値を基に、プール内の他のサーバと比較して、サーバに送信されるクライアント リクエストの数が決定します。このロード バランシング アルゴリズムは、重み値を使用して、使用可能なサーバ リソース間で負荷を均等に分散する処理に特化しています。重みの値を設定せず、スロー スタートを有効にしている場合、値はデフォルトで1になります。

NSX-T では WEIGHTED_LEAST_CONNECTION に対応します。NSX-V では、対応するものではありません。

■ WEIGHTED_ROUND_ROBIN

各サーバには、プール内の他のサーバを基準としてこのサーバの実行方法を示す重み値が割り当てられています。この値を基に、プール内の他のサーバと比較して、サーバに送信されるクライアント リクエストの数が決定します。このロード バランシング アルゴリズムは、使用可能なサーバ リソース間で負荷をに分散することを目的としています。

NSX-T では WEIGHTED_ROUND_ROBIN に対応します。NSX-V では、対応するものではありません。

■ URI

URI の左側の部分がハッシュされ、実行中のサーバの合計重みによって除算されます。結果により、要求を受信するサーバが指定されます。これにより、サーバが起動または停止しない限り、URI は常に同一のサーバに送信されます。URI アルゴリズム パラメータには 2 つのオプション (uriLength=<len>、uriDepth=<dep>) があります。長さのパラメータの範囲は、1<=len<256 にする必要があります。深さのパラメータの範囲は、1<=dep<10 にする必要があります。長さおよび深さのパラメータには、正の整数が続きます。これらのオプションでは、URI の最初の部分のみに基づいてサーバのバランシングを行うことができます。長さのパラメータは、アルゴリズムが URI の最初の部分に定義された文字だけを対象としてハッシュを計算することを示します。深さのパラメータは、ハッシュの計算に使用されるディレクトリの最大の深さを示します。要求に含まれる各スラッシュが 1 つのレベルとして数えられます。両方のパラメータが指定されている場合は、いずれかに達したときに評価が停止します。

NSX-V では URI に対応します。NSX-T では、対応するものではありません。

■ HTTPHEADER

各 HTTP 要求で HTTP ヘッダー名の検索が行われます。括弧内のヘッダー名では、大文字と小文字が区別されません。ヘッダーがない、または値を含んでいない場合は、ラウンド ロビン アルゴリズムが適用されます。HTTPHEADER アルゴリズム パラメータのオプションは 1 つ (headerName=<name>) です。

NSX-V では HTTPHEADER に対応します。NSX-T では、対応するものではありません。

■ URL

引数に指定される URL パラメータは、各 HTTP GET 要求のクエリ文字列内で検索されます。パラメータの後ろに等号の = と値が続く場合、その値がハッシュされ、実行されるサーバの重みの合計で割られます。結果により、要求を受信するサーバが指定されます。このプロセスを使用して要求に含まれるユーザー ID が追跡され、サーバの起動または停止が起こらない限り、常に同一のユーザー ID が同一のサーバに送信されます。値またはパラメータが検出されない場合は、ラウンド ロビン アルゴリズムが適用されます。URL アルゴリズム パラメータのオプションは 1 つ (urlParam=<url>) です。

NSX-V では URL に関連付けられます。NSX-T では、対応するものではありません。

関連情報については、NSX 製品ドキュメントの[ロード バランシング用サーバ プールの追加](#)などのトピックを参照してください。

NSX-V ネットワークと NSX-T ネットワークおよびロード バランサのオプション

ロード バランサのオプションは、クラウド テンプレートでロード バランサ リソースが関連付けられているネットワークによって異なります。ロード バランサは、ネットワーク タイプとネットワーク条件に応じて構成できます。

■ オンデマンド送信ネットワーク

ロード バランサ コンピューティングがオンデマンド outbound ネットワークに接続されている場合、オンデマンド ネットワークの Tier-1 ルーターに対してロード バランサが作成されます。

■ オンデマンド プライベート ネットワーク

ロード バランサ コンピューティングがオンデマンド private ネットワークに接続されている場合、新しい Tier-1 ルーターが作成されて、ネットワーク プロファイルに指定されている Tier-0 ルーターに接続されます。その後、ロード バランサは Tier-1 ルーターに接続されます。Tier-1 ルーター VIP アドバタイズは、仮想マシン IP アドレスが existing ネットワーク上にあれば有効です。DHCP を使用するよう private ネットワークが構成されている場合、プライベート ネットワークとロード バランサは Tier-1 ルーターを共有します。

■ 既存のネットワーク

ロード バランサは、existing ネットワークに接続されている場合、既存のネットワークの Tier-1 ルーターで作成されます。Tier-1 ルーターに接続されたロード バランサがない場合は、新しいロード バランサが作成されます。ロード バランサがすでにある場合は、そのロード バランサに新しい仮想サーバが接続されます。

existing ネットワークが Tier-1 ルーターに接続されていない場合は、新しい Tier-1 ルーターが作成されて、ネットワーク プロファイルに定義されている Tier-0 ルーターに接続されます。Tier-1 ルーター VIP アドバタイズは有効ではありません。

■ ネットワーク プロファイルに定義されるネットワークの隔離

ネットワーク タイプが `outbound` または `private` の場合は、ネットワーク プロファイルでネットワークの隔離設定を指定して、新しいセキュリティ グループをエミュレートできます。マシンが既存のネットワークに接続され、隔離の設定がプロファイルで定義されるため、このオプションは既存のネットワークに作成されるロード バランサと似ています。違いは、データパスを有効にするために、Tier-1 のアップリンク ポート IP アドレスが隔離セキュリティ グループに追加されていることです。

NSX に関連付けられたネットワークでロード バランサ設定を指定するには、クラウド テンプレート デザインで NSX ロード バランサ リソースを使用します。

詳細については、VMware ブログの [vRA Cloud Assembly Load Balancer with NSX-T Deep Dive](#) を参照してください。

複数のロード バランサで NSX-T Tier-1 または NSX-V Edge が共有されている場合のログ レベルまたはタイプ の設定の再構成

複数のロード バランサを含むクラウド テンプレートを使用し、これらのロード バランサで NSX-T エンドポイントの Tier-1 ルーターまたは NSX-V エンドポイントの Edge ルーターを共有している場合、いずれかのロード バランサ リソースでログ レベルまたはタイプ の設定を再構成しても、他のロード バランサの設定は更新されません。設定が一致しない場合、NSX に不整合が生じます。ログのレベルまたはタイプ の設定を再構成するときに不整合が生じないようにするには、関連付けられた NSX エンドポイントの Tier-1 ルーターまたは Edge ルーターを共有するクラウド テンプレート内のすべてのロード バランサ リソースに同じ再構成値を使用します。

利用可能な Day 2 操作

ロード バランサを含む展開をスケール インまたはスケール アウトすると、そのロード バランサは、新しく追加されたマシンを含めるように、または撤去対象のロード バランシング マシンを停止するように構成されます。

クラウド テンプレートおよび展開で使用できる一般的な Day 2 操作のリストについては、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)を参照してください。

詳細情報

ネットワーク プロファイルでのロード バランサ設定の定義については、[vRealize Automation でのネットワーク プロファイルの詳細](#)を参照してください

ロード バランサを含むクラウド テンプレート デザインの例については、[vRealize Automation クラウド テンプレートのネットワーク、セキュリティ、およびロード バランサの例](#)を参照してください。

ユーザー名とパスワードでアクセスできる Puppet 対応のクラウド テンプレート

この例では、vCenter コンピューティング リソースに展開され、かつユーザー名とパスワードでアクセスできるクラウド テンプレートに Puppet 構成管理を追加します。

この手順では、Puppet 対応の展開可能なリソースを作成し、しかもユーザー名とパスワードの認証がないと使用できないようにするにはどうすればよいか、その例を示します。ユーザー名とパスワードによるアクセスとは、Puppet 構成管理を呼び出すためには、ユーザーがコンピューティング リソースから Puppet プライマリ マシンに手動でログインする必要があることを意味します。

必要に応じて、コンピューティング リソースが Puppet プライマリ マシンで認証を処理するように、クラウド テンプレートで構成管理を設定するリモート アクセス認証を設定できます。リモート アクセスが有効な場合、コンピューティング リソースはパスワード認証を満たすキーを自動的に生成します。有効なユーザー名が引き続き必要です。

vRealize Automation Cloud Assembly ブループリントにさまざまな Puppet シナリオを設定する例については、[AWS での Puppet 構成管理のクラウド テンプレートの例](#)と [vCenter Puppet 構成クラウド テンプレートの例](#)を参照してください。

前提条件

- 有効なネットワークに Puppet Enterprise インスタンスをセットアップします。
- 統合機能を使用して、Puppet Enterprise インスタンスを vRealize Automation Cloud Assembly に追加します。[vRealize Automation Cloud Assembly での Puppet Enterprise 統合の設定](#)を参照してください。
- vSphere アカウントと vCenter コンピューティング リソースを設定します。

手順

- 1 目的のクラウド テンプレートのキャンバスにある vSphere コンピューティング リソースに、Puppet 構成管理コンポーネントを追加します。
 - a [インフラストラクチャ] - [管理] - [統合] の順に選択します。
 - b [統合の追加] をクリックし、[Puppet] を選択します。
 - c Puppet 設定画面で適切な情報を入力します。

設定	説明	値の例
ホスト名	Puppet プライマリ マシンのホスト名または IP アドレス	Puppet-Ubuntu
SSH ポート	vRealize Automation Cloud Assembly と Puppet プライマリ マシン間の通信用の SSH ポート。(オプション)	なし
自動署名シークレット	証明書要求の自動署名をサポートするためにノードが提供する、Puppet プライマリ マシンに設定された共有シークレット。	ユーザー固有
場所	<p>Puppet プライマリ マシンがプライベート クラウドとパブリック クラウドのどちらにあるかを示します。</p> <p>注: クロス クラウド展開は、展開コンピューティング リソースと Puppet プライマリ マシンとの間に接続がある場合にのみサポートされます。</p>	
Cloud proxy	Microsoft Azure や Amazon Web Services などのパブリック クラウド アカウントには必要ありません。vCenter ベースのクラウド アカウントを使用している場合は、そのアカウントに適切な cloud proxy を選択します。	なし
Username	Puppet プライマリ マシンの SSH および RBAC ユーザー名。	ユーザー固有です。YAML 値は「\${input.username}」です。
パスワード	Puppet プライマリ マシンの SSH および RBAC パスワード。	ユーザー固有の YAML 値は「\${input.password}」です。
このユーザーに sudo コマンドを使用	procidd に sudo コマンドを使用する場合に選択します。	true
名前	Puppet プライマリ マシン名。	PEMasterOnPrem
説明		

- 2 次の例のように、ユーザー名とパスワードのプロパティを Puppet YAML に追加します。
- 3 次の例に示すように、Puppet クラウド テンプレート YAML に対する remoteAccess プロパティの値が authentication: username and password に設定されていることを確認します。

例：vCenter ユーザー名とパスワード YAML コード

次の例は、vCenter コンピューティング リソースにユーザー名とパスワードの認証を追加するための典型的な YAML コードを示しています。

```
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 での Puppet 構成管理のクラウド テンプレートの例

AWS コンピューティング リソースの構成管理に基づいて Puppet をサポートするためのクラウド テンプレートの構成には、いくつかのオプションがあります。

ユーザー名とパスワードを使用した AWS での Puppet の管理

例...

ブループリント YAML のサンプル

サポート対象の Amazon マシン イメージでのクラウド設定の認証。

```
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
```

既存のユーザーを使用したカスタム Amazon マシン イメージでのクラウド設定の認証。

```
inputs:
  username:
    type: string
    title: Username
    default: puppet
  password:
    type: string
    title: Password
    encrypted: true
    default: VMware@123
```

例...

ブループリント YAML のサンプル

```
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
```

生成済みのパブリック キーまたはプライベート キーを使用した AWS での Puppet の管理

例...

ブループリント YAML のサンプル

生成済みのパブリック キーまたはプライベート キーでアクセスする AWS での remoteAccess.authentication 認証。

```
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
```

vCenter Puppet 構成クラウド テンプレートの例

vCenter コンピューティング リソースで Puppet ベースの構成管理をサポートするようにクラウド テンプレートを設定するためのオプションがいくつか用意されています。

vSphere 上の Puppet でユーザー名とパスワードによる認証を使用する

次の例は、vSphere OVA 上の Puppet でユーザー名とパスワードによる認証を使用する場合の YAML コードを示しています。

表 6-5.

例...	ブループリント YAML のサンプル
<p>vSphere OVA 上の Puppet でユーザー名とパスワードによる認証を使用する場合の YAML コード。</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>vSphere OVA 上の Puppet でユーザー名とパスワードによる認証を使用してコンピューティング リソースを管理する場合の YAML コード。</p>	<pre>inputs: username: type: string title: Username default: puppet</pre>

表 6-5. (続き)

例...	ブループリント YAML のサンプル
<p>vCenter 上の Puppet でリモート アクセス対応のパスワード認証を使用してコンピューティング リソースを管理する場合の YAML コード。</p>	<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> <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>

表 6-5. (続き)

例...	ブループリント YAML のサンプル
	<pre> 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}' </pre>

vSphere 上の Puppet で生成済みのパブリック キーまたはプライベート キーによる認証を使用する

表 6-6.

例...	ブループリント YAML のサンプル
<p>vSphere OVA 上の Puppet で生成済みのパブリック キーまたはプライベート キーによる認証を使用してコンピューティング リソースを管理する場合の YAML コード。</p>	<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 に Terraform 構成を含める方法

Terraform 構成をリソースとして vRealize Automation Cloud Assembly のクラウド テンプレートに組み込むことができます。

vRealize Automation Cloud Assembly の Terraform ランタイム環境の準備

Terraform 構成を含むデザインでは、vRealize Automation Cloud Assembly オンプレミス製品と統合する Terraform ランタイム環境にアクセスできることが必須です。

Terraform ランタイムを追加する方法

ランタイム環境は、Terraform CLI コマンドを実行することで要求された操作を実行する Kubernetes クラスターで構成されています。また、ランタイムはログを収集し、Terraform CLI コマンドの結果を返します。

vRealize Automation オンプレミス製品では、ユーザーが独自の Terraform ランタイム Kubernetes クラスターを構成する必要があります。組織あたり 1 つの Terraform ランタイムのみがサポートされます。組織内のすべての Terraform 展開は同じランタイムを使用します。

1 Terraform CLI を実行する Kubernetes クラスターがあることを確認します。

- すべてのユーザーは、Kubeconfig ファイルを指定して、管理対象外の Kubernetes クラスターで Terraform CLI を実行できます。
- Enterprise ライセンス ユーザーは、vRealize Automation によって管理されている Kubernetes クラスターで Terraform CLI を実行することができます。

vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [リソース] - [Kubernetes] の順に移動して、Kubernetes クラスターがあることを確認します。追加する必要がある場合は、[vRealize Automation Cloud Assembly で Kubernetes を使用する方法](#)を参照してください。


2 Kubernetes クラスターを新たに追加した場合、または変更した場合は、データ収集が完了するまで待機します。

データ収集は名前空間などの情報のリストを取得するため、プロバイダによっては最大で 5 分かかることもあります。

3 データ収集の完了後、[インフラストラクチャ] - [統合] - [統合の追加] の順に移動し、[Terraform ランタイム] カードを選択します。

4 設定を入力します。

図 6-3. Terraform ランタイム統合の例



New Integration

Name *

OurOrg TF Runtime

Description

Terraform Runtime Integration

Kubernetes cluster *


OurK8Cluster

Kubernetes namespace *

OurK8Namespace

Runtime Container Settings

Image

docker.io/hashicorp/terraform:0.12.24 

CPU request (Millicores)

250

CPU limit (Millicores)

250

Memory request (MB)

512

Memory limit (MB)

512

VALIDATE

設定	説明
名前	ランタイム統合に一意の名前を付けます。
説明	統合の目的を入力します。
Terraform ランタイム統合：	
ランタイム タイプ (Enterprise のみ)	Enterprise ライセンス ユーザーは、vRealize Automation によって管理されている Kubernetes クラスタと管理対象外の Kubernetes クラスタのどちらで Terraform CLI を実行するか選択できます。
Kubernetes kubeconfig (すべてのユーザー)	<p>管理対象外の Kubernetes クラスタの場合、外部クラスタの kubeconfig ファイルの内容全体を貼り付けます。</p> <p>プロキシ サーバで外部 Kubernetes ランタイムを使用する方法については、プロキシ サポートを追加する方法を参照してください。</p> <p>このオプションは、すべてのユーザーが使用できます。</p>
Kubernetes クラスタ (Enterprise のみ)	<p>vRealize Automation によって管理されている Kubernetes の場合は、Terraform CLI を実行するクラスタを選択します。</p> <p>クラスタおよびそのクラスタの kubeconfig ファイルにアクセスすることが必要です。kubeconfig へのアクセスは、/cmx/api/resources/k8s/clusters/{clusterId}/kube-config に対する GET によって確認できます。</p> <p>このオプションは、Enterprise ライセンスでのみ使用できます。</p>
Kubernetes 名前空間	Terraform CLI を実行するポッドを作成するために、クラスタ内で使用する名前空間を選択します。

設定	説明
ランタイム コンテナ設定 :	
イメージ	<p>実行する Terraform バージョンのコンテナ イメージへのパスを入力します。</p> <p>注: [検証] ボタンをクリックしても、コンテナ イメージはチェックされません。</p>
CPU 要求	コンテナの実行に必要な CPU の量を入力します。デフォルトは 250 ミリコアです。
CPU リミット	コンテナの実行で許容される CPU の最大量を入力します。デフォルトは 250 ミリコアです。
メモリ要求	コンテナの実行に必要なメモリの量を入力します。デフォルトは 512 MB です。
メモリ リミット	コンテナの実行で許容されるメモリの最大量を入力します。デフォルトは 512 MB です。

5 [検証] をクリックし、必要に応じて設定を調整します。

6 [追加] をクリックします。

設定はキャッシュされます。統合の追加後、クラスタや名前空間などの設定を変更することはできますが、変更が検出されるまでと、新しい設定で Terraform CLI が実行されるまでに最大 5 分かかることがあります。

Terraform ランタイムのトラブルシューティング

Terraform 構成の展開の問題は、ランタイム統合に関連している場合があります。

問題	原因	解決方法
名前空間が無効であることを示すエラーが表示され、検証が失敗する。	クラスタが変更されても、ユーザー インターフェイスに以前の名前空間が残っています。	クラスタの選択を変更した後は、名前空間を必ず再選択します。
名前空間のドロップダウンが空であるか、新しく追加された名前空間がリストされていません。	クラスタのデータ収集が完了しない。データ収集は、クラスタの入力/変更後は最大で 5 分かかります。また、名前空間の入力/変更時には最大で 10 分かかります。	<p>既存の名前空間を含む新しいクラスタの場合、データ収集が完了するまで最大 5 分間待機します。</p> <p>既存のクラスタ内の新しい名前空間の場合、データ収集が完了するまで最大 10 分間待機します。</p> <p>問題が解決しない場合は、クラスタを削除して [インフラストラクチャ] - [リソース] - [Kubernetes] で再度追加します。</p>
統合アカウントを更新しても、Terraform CLI コンテナが、以前のクラスタ、以前の名前空間、以前のランタイム設定で作成される。	vRealize Automation によって使用される Kubernetes API クライアントは、5 分間キャッシュされます。	変更が有効になるまでに最大で 5 分かかる場合があります。
kubeconfig が使用できないことを示すエラーが表示され、検証または Terraform の展開操作に失敗する。	<p>これらのエラーは、クラスタに vRealize Automation からアクセスできないことが原因で発生する場合があります。</p> <p>それ以外の場合では、ユーザーの認証情報、トークン、証明書が無効になります。</p>	kubeconfig エラーはさまざまな理由で発生する可能性があります。トラブルシューティングには、テクニカルサポートにとの連携が必要になる場合があります。

プロキシ サポートを追加する方法

外部 Kubernetes ランタイム クラスタをプロキシ サーバ経由で接続するには、次の手順を実行します。

- 1 Kubernetes クラスタ サーバにログインします。
- 2 空のフォルダを作成します。
- 3 新しいフォルダで、Dockerfile という名前の新しいファイルに次の行を追加します。

```
FROM projects.registry.vmware.com/vra/terraform:latest as final
ENV https_proxy=protocol://username:password@proxy_host:proxy_port
ENV http_proxy=protocol://username:password@proxy_host:proxy_port
ENV no_proxy=.local,.localdomain,localhost
```

- 4 プレースホルダ値を変更して、インターネットへのアクセスに使用するプロキシ サーバ設定を `https_proxy` および `http_proxy` 環境変数に含めます。

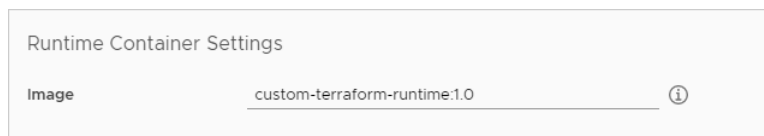
`protocol` は、プロキシ サーバでの使用に応じて `http` または `https` になります。これは、`https_proxy` または `http_proxy` という環境変数名と一致しない可能性があります。

- 5 Dockerfile を保存して閉じます。
- 6 空のフォルダで次のコマンドを実行します。アカウントの権限によっては、コマンドを `sudo` モードで実行することが必要な場合があります。

```
docker build --file Dockerfile --tag custom-terraform-runtime:1.0 .
```

コマンドを実行すると、ローカルに `custom-terraform-runtime:1.0` Docker イメージが作成されます。

- 7 vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [接続] - [統合] の順に選択して Terraform ランタイム統合に移動します。
- 8 `custom-terraform-runtime:1.0` イメージを使用するようにランタイム コンテナ設定を作成または編集します。



Runtime Container Settings

Image ⓘ

インターネット アクセスがない状況での vRealize Automation Cloud Assembly Terraform ランタイム

vRealize Automation Cloud Assembly ユーザーが、インターネットから切断されている間に Terraform 統合を設計および実行する必要がある場合は、次の例に従ってランタイム環境を設定します。

注： 設定時、インターネットに一時的に接続する必要があります。

次の手順では、ユーザーが独自の [Docker レジストリ](#) を持っており、インターネットに接続することなくそのリポジトリにアクセスできることを想定しています。

カスタム コンテナ イメージの作成

- 1 Terraform プロバイダ プラグイン バイナリを含むカスタム コンテナ イメージを作成します。

次の Dockerfile は、Terraform GCP プロバイダを使用してカスタム イメージを作成する例を示しています。

Dockerfile で基本イメージ `projects.registry.vmware.com/vra/terraform:latest` をダウンロードするには、`projects.registry.vmware.com` にある VMware Harbor レジストリにインターネット経由で一時的にアクセスする必要があります。

ファイアウォール設定またはプロキシ設定により、イメージのビルドが失敗する可能性があります。また、Terraform プロバイダ プラグイン バイナリをダウンロードするには、`releases.hashicorp.com` への一時的なアクセス権が必要になる場合があります。ただし、必要な場合は、プライベート レジストリを使用してプラグイン バイナリを提供することもできます。

```
FROM projects.registry.vmware.com/vra/terraform:latest as final

# Create provider plug-in directory
ARG plugins=/tmp/terraform.d/plugin-cache/linux_amd64
RUN mkdir -m 777 -p $plugins

# Download and unzip all required provider plug-ins from hashicorp to provider directory
RUN cd $plugins \
    && wget -q https://releases.hashicorp.com/terraform-provider-google/3.58.0/terraform-provider-google_3.58.0_linux_amd64.zip \
    && unzip *.zip \
    && rm *.zip

# For "terraform init" configure terraform CLI to use provider plug-in directory and not
download from internet
ENV TF_CLI_ARGS_init="-plugin-dir=$plugins -get-plugins=false"
```

- 2 カスタム コンテナ イメージを作成、タグ付けして、独自の Docker リポジトリにプッシュします。
- 3 vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [接続] - [統合] の順に選択して Terraform ランタイム統合に移動します。
- 4 カスタム コンテナ イメージのリポジトリを追加するために、ランタイム コンテナ設定を作成または編集します。作成したカスタム コンテナ イメージの名前は、たとえば `registry.ourcompany.com/project1/image1:latest` のようになります。

Runtime Container Settings

Image
registry.ourcompany.com/project1/image1:latest ⓘ

Terraform CLI のローカルでのホスト

- 1 Terraform CLI バイナリをダウンロードします。
- 2 Terraform CLI バイナリをローカル Web サーバにアップロードします。
- 3 vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [構成] - [Terraform バージョン] の順に選択します。

- 4 ローカル Web サーバでホストされる Terraform CLI バイナリの URL が含まれるように Terraform バージョンを作成または編集します。

0.12.29 DELETE	
Version *	0.12.29 ⓘ
Description	<input type="text"/>
Enabled	<input checked="" type="checkbox"/> ⓘ
URL *	http://host1.ourcompany.com:8080/tf/0.12.29/terraform_0.12.29_linux_amd64.zip ⓘ
SHA256 Checksum *	872245d9c6302b24dc0d98a1e010aef60865a2df60102c8ad03e9d5a1d ⓘ

Terraform 構成の設計および展開

ランタイムが設定されたら、Terraform 構成ファイルを git に追加して、そのクラウド テンプレートを設計し、展開できます。

最初に、[vRealize Automation Cloud Assembly](#) での [Terraform 構成の準備](#) を参照してください。

トラブルシューティング

展開時は、vRealize Automation Cloud Assembly で展開を開きます。[履歴] タブで Terraform イベントを探し、右側にある [ログの表示] をクリックします。ローカル Terraform プロバイダが機能している場合は、ログに次のメッセージが示されます。

```
Initializing provider plugins
```

```
Terraform has been successfully initialized
```

ログをより堅牢にするには、クラウド テンプレート コードを手動で編集して、次の例に示すように `TF_LOG: DEBUG` を追加します。

```
resources:
  terraform:
    type: Cloud.Terraform.Configuration
    properties:
      providers:
        - name: google
          # List of available cloud zones: gcp/us-west1
          cloudZone: gcp/us-west1
      environment:
        # Configure terraform CLI debug log settings
        TF_LOG: DEBUG
      terraformVersion: 0.12.29
      configurationSource:
        repositoryId: fc569ef7-f013-4489-9673-6909a2791071
        commitId: 3e00279a843a6711f7857929144164ef399c7421
        sourceDirectory: gcp-simple
```

vRealize Automation Cloud Assembly での Terraform 構成の準備

vRealize Automation Cloud Assembly テンプレートに Terraform 構成を追加する前に、バージョン管理リポジトリのセットアップと統合を実行します。

- 1 前提条件
- 2 バージョン管理リポジトリへの Terraform 構成ファイルの保存
- 3 クラウド ゾーン マッピングの有効化
- 4 リポジトリと vRealize Automation Cloud Assembly の統合

前提条件

vRealize Automation オンプレミス製品で Terraform 操作を実行するには、Terraform ランタイム統合が必要です。vRealize Automation Cloud Assembly の Terraform ランタイム環境の準備を参照してください。

バージョン管理リポジトリへの Terraform 構成ファイルの保存

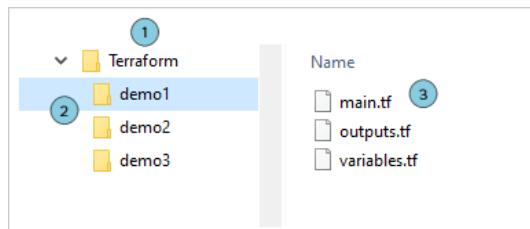
vRealize Automation Cloud Assembly では、Terraform 構成に対して次のバージョン管理リポジトリをサポートしています。

- GitHub クラウド、GitHub Enterprise オンプレミス
- GitLab クラウド
- Bitbucket オンプレミス

バージョン管理リポジトリで、Terraform 構成ファイルが格納されたサブディレクトリを 1 レイヤーに含むデフォルトのディレクトリを作成します。Terraform 構成ごとに 1 つのサブディレクトリを作成します。

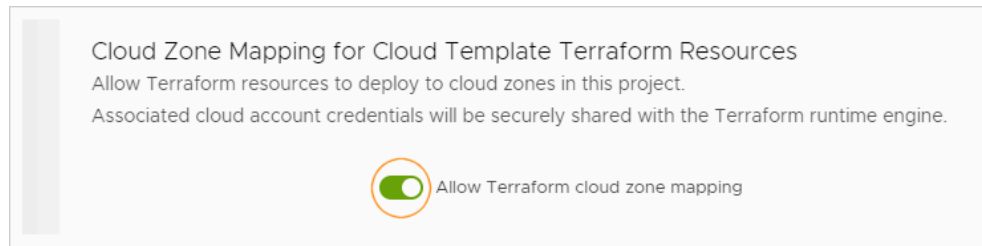
- 1 デフォルト ディレクトリ
- 2 単一のサブディレクトリ レイヤー
- 3 展開対応の Terraform 構成ファイル

Terraform 構成ファイルには、状態ファイルを含めないでください。terraform.tfstate が存在する場合、展開中にエラーが発生します。



クラウド ゾーン マッピングの有効化

クラウド アカウントに展開する場合は、Terraform ランタイム エンジンにクラウド ゾーンの認証情報が必要です。プロジェクトの [プロビジョニング] タブで、[Terraform のクラウド ゾーン マッピングを許可] を有効にします。



認証情報が安全に転送される場合でも、プロジェクト ユーザーがクラウド アカウントへの展開を必要としていない場合は、セキュリティの強化のため、このオプションを無効のままにする必要があります。

リポジトリと vRealize Automation Cloud Assembly の統合

vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [接続] - [統合] に移動します。

Terraform 構成を保存したリポジトリ サービス タイプ (GitHub、GitLab または Bitbucket) に統合を追加します。

統合にプロジェクトを追加する場合は、[Terraform 構成] タイプを選択し、リポジトリとブランチを特定します。

[フォルダ] は、以前の構造のデフォルト ディレクトリです。

vRealize Automation Cloud Assembly での Terraform 構成のデザイン

リポジトリおよび Terraform の構成ファイルを配置することで、構成ファイル用の vRealize Automation Cloud Assembly テンプレートをデザインできます。

- 1 前提条件
- 2 Terraform ランタイム バージョンの有効化
- 3 デザインへの Terraform リソースの追加
- 4 クラウド テンプレートの展開

前提条件

バージョン管理リポジトリのセットアップと統合を実行します。[vRealize Automation Cloud Assembly での Terraform 構成の準備](#)を参照してください。

Terraform ランタイム バージョンの有効化

Terraform 構成を展開するときに、ユーザーが使用できる Terraform ランタイム バージョンを定義できます。Terraform 構成に内部コードによるバージョンの制約が含まれる場合もあることに注意してください。

許可されているバージョンのリストを作成するには、[インフラストラクチャ] - [設定] - [Terraform バージョン] の順に移動します。バージョン 0.12.x のみがサポートされます。

デザインへの Terraform リソースの追加

Terraform 構成を含むクラウド テンプレートを作成します。

- 1 vRealize Automation Cloud Assembly で、[デザイン] - [クラウド テンプレート] の順に移動し [新規作成元] - [Terraform] の順にクリックします。

Terraform 構成ウィザードが表示されます。

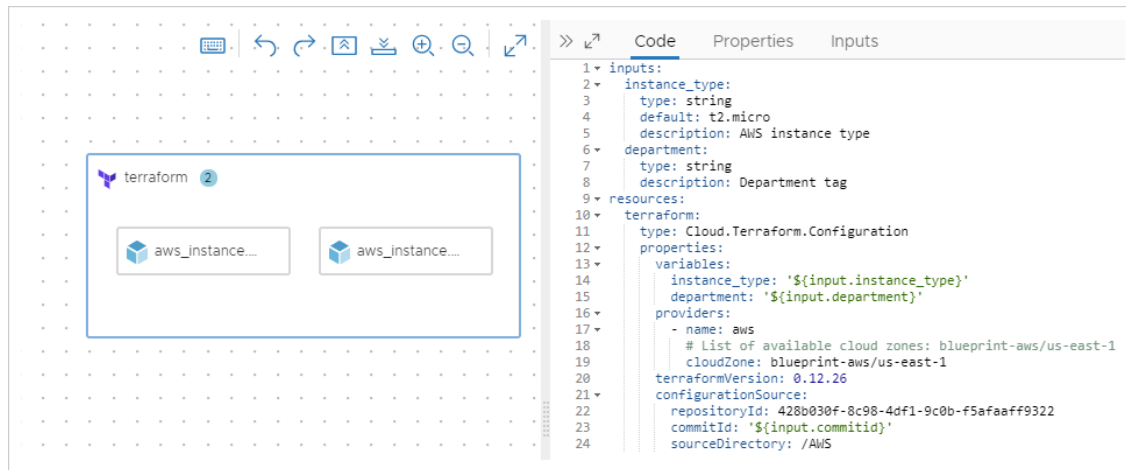
- 2 プロンプトの指示に従います。

ウィザード ページ	設定	値
[クラウド テンプレートの新規作成]	名前	デザインに識別しやすい名前を付けます。
	説明	デザインの目的を入力します。
	プロジェクト	Terraform 構成が保存されているリポジトリ統合を含むプロジェクトを選択します。
[構成ソース]	リポジトリ	Terraform 構成を保存した統合リポジトリを選択します。
	コミット	リポジトリ コミットを選択するか、エントリを空白のままにして、リポジトリ ヘッドから Terraform 構成を使用します。 Bitbucket の制限 : Bitbucket のリポジトリ サーバの設定により、選択可能なコミットの数に切り詰められることがあります。
	ソース ディレクトリ	作成したリポジトリ構造からサブディレクトリを選択します。前のセットアップで示したサブディレクトリの例は、demo1、demo2、demo3 です。
[構成の確定]	リポジトリ	正しいリポジトリが選択されていることを確認します。
	ソース ディレクトリ	正しいディレクトリが選択されていることを確認します。
	Terraform バージョン	Terraform 構成を展開するときに実行する Terraform ランタイム バージョンを選択します。
	プロバイダ	Terraform 構成にプロバイダ ブロックが含まれていた場合、このクラウド テンプレートの展開先となるプロバイダとクラウド ゾーンを確認します。 プロバイダがないことは問題にはなりません。ウィザードの終了後、テンプレートのプロパティでプロバイダとクラウド ゾーンを編集し、展開ターゲットを追加または変更します。

ウィザード ページ	設定	値
	変数	パスワードなど、暗号化を行う機密性の高い値を選択します。
	出力	Terraform 構成からの出力を確認します。これにより、デザイン コードで参照できる式に変換されます。

3 [作成] をクリックします。

Terraform リソースと、展開する Terraform 構成が反映された vRealize Automation Cloud Assembly コードが、クラウド テンプレート キャンバスに表示されます。



必要に応じて、他の vRealize Automation Cloud Assembly リソースをクラウド テンプレートに追加し、Terraform コードと Terraform 以外のコードをハイブリッド デザインに統合することができます。

注： リポジトリ内の Terraform 構成を更新しても、変更はクラウド テンプレートと同期されません。自動同期により、新たに追加された機密変数などのセキュリティ リスクが生じる可能性があります。

Terraform 構成の変更を取得するには、ウィザードを再実行し、新しいコミットを選択して、新しい機密変数を指定します。

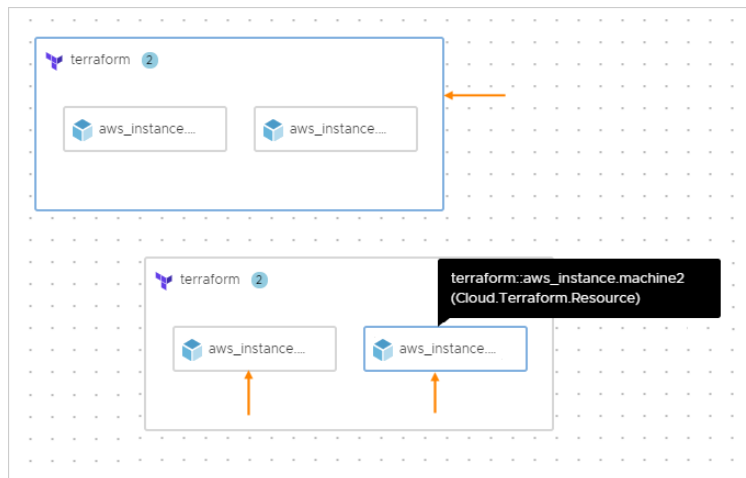
クラウド テンプレートの展開

クラウド テンプレートを展開する際に、展開の [履歴] タブで割り当てフェーズや作成フェーズなどのイベントを展開し、Terraform CLI からのメッセージのログを調べることができます。

承認—PLAN、ALLOCATE、CREATE など、通常の Terraform フェーズに加え、vRealize Automation Cloud Assembly では承認フェーズによるガバナンスを導入しています。申請の承認の詳細については、[Service Broker 承認ポリシーの構成方法](#)を参照してください。

Timestamp	Status	Resource type	Resource name	Details
Aug 3, 202...	PLAN_FINISHED	Cloud.Terraform.Configurati...	terraform	Creating 2 Terraform resources, updating 0 Terraform resources, deleting 0 Terraform resources
Aug 3, 202...	PLAN_IN_PROGRESS	Cloud.Terraform.Configurati...	terraform	Hide Logs
<pre> 2:24:23 PM * provider.random: version = "~> 2.3" 2:24:23 PM 2:24:23 PM Terraform has been successfully initialized! 2:24:28 PM Refreshing Terraform state in-memory prior to plan... 2:24:28 PM The refreshed state will be used to calculate this plan, but will not be 2:24:28 PM persisted to local or remote state storage. </pre> View as plain text				
Aug 3, 202...	INITIALIZATION_FINISH...			
Aug 3, 202...	INITIALIZATION_IN_PRO...			

展開後は、Terraform コンポーネント全体を表す外部リソースが、Terraform で作成された別のコンポーネントに対する内部の子リソースとともに表示されます。子リソースのライフサイクルは、親の Terraform リソースによって制御されます。



vRealize Automation での Terraform 構成に関する詳細

vRealize Automation で Terraform 構成をリソースとして組み込む際は、特定の制限事項やトラブルシューティングについて考慮する必要があります。

Terraform 構成の制限事項

- Terraform 構成を使用したデザインを検証する場合、テスト ボタンでは vRealize Automation Cloud Assembly 構文はチェックされますが、ネイティブの Terraform コード構文はチェックされません。
また、テスト ボタンでは、Terraform 構成に関連付けられたコミット ID は検証されません。
- Terraform 構成を含むクラウド テンプレートでは、テンプレートのクローンを別のプロジェクトに作成する場合に次の回避策が必要になります。
 - a 新しいプロジェクトの [統合] タブで、統合の repositoryId をコピーします。

b クローンのテンプレートを開きます。コード エディタで、`repositoryId` をコピーした値に置き換えます。

- バージョン管理リポジトリでは、Terraform 構成ファイルに状態ファイルを含めないでください。
`terraform.tfstate` が存在する場合、展開中にエラーが発生します。

Terraform の親リソースに対する Day 2 アクションをサポート

Terraform の親リソースでは、Terraform の状態ファイルを表示または更新できます。状態ファイルのアクションの詳細については、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)のアクションの包括的なリストを参照してください。

子リソースに対する Day 2 アクションをサポート

Terraform 構成を展開した後、子リソースで Day 2 アクションが使用可能になるまでに最大 20 分かかる場合があります。

Terraform 構成の子リソースでは、次の Day 2 アクションのサブセットのみがサポートされます。アクションの詳細については、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)のアクションの包括的なリストを参照してください。

プロバイダ	Terraform リソース タイプ	Day 2 アクションのサポート
AWS	aws_instance	パワーオン
		パワーオフ
		再起動
		リセット
Azure	azurerm_virtual_machine	パワーオン
		パワーオフ
		再起動
		中断
vSphere	vsphere_virtual_machine	パワーオン
		パワーオフ
		再起動
		リセット
		シャットダウン
		中断
		スナップショットの作成
		スナップショットの削除
		スナップショットに戻す

プロバイダ	Terraform リソース タイプ	Day 2 アクションのサポート
GCP	google_compute_instance	パワーオン
		パワーオフ
		スナップショットの作成
		スナップショットの削除

Day 2 アクションの可用性に関するトラブルシューティング

特別な設定は不要の（OOTB、Out-of-the-box）Day 2 アクションが不足しているか、無効になっている場合は、トラブルシューティングが必要になる場合があります。

問題	原因	解決方法
Terraform リソースで [アクション] メニューに OOTB Day 2 アクションが表示されません。	このアクションは、前述のリストに記載されているように、プロバイダおよびリソース タイプでサポートされていない可能性があります。 他の可能性として、リソースの検出とリソースのキャッシュのタイミングにより、このアクションの表示に最大で 20 分かかることもあります。	デザインのプロバイダとリソース タイプを確認します。 データ収集が完了するまで、最大 20 分待機します。
データ収集を考慮した 20 分間が経過した後も Terraform リソースに Day 2 アクションが表示されない。	リソース検出の問題により、アクションが表示されません。 リソースが誤ってプロジェクト外のクラウドゾーンに作成されている可能性があります。たとえば、プロジェクトにはクラウドアカウントとリージョン us-east-1 のクラウドゾーンのみが含まれていますが、Terraform 構成には us-west-1 に対するプロバイダブロックが含まれているため、デザイン時には変更していない場合などです。 もう 1 つの可能性は、データ収集が機能していないことです。	プロジェクトのクラウドゾーンをデザインのクラウドゾーンと対比して確認します。 [インフラストラクチャ] - [接続] - [クラウド アカウント] の順に選択し、データ収集ステータスと、クラウドアカウントの収集が最後に成功した時間を確認します。
リソース状態とデータ収集について明らかな問題がないにもかかわらず、Day 2 アクションは無効になっている（グレーアウトしている）。	断続的に発生するタイミングの問題と、データ収集の失敗が時折発生することが知られています。	この問題は、20 分以内に解決します。
リソースの状態に基づいて有効になるはずの、Day 2 アクションが誤って無効になっている。 たとえば、プロバイダのインターフェイスを使用してリソースをパワーオフしたにもかかわらず、パワーオフが有効になり、パワーオンが無効になる。	データ収集のタイミングによって一時的な不一致が発生する可能性があります。電源状態を vRealize Automation の外部から変更すると、変更が正しく反映されるまでに時間がかかります。	最大 20 分待機します。

vRealize Automation でのカスタム Terraform プロバイダの使用

カスタムの Terraform プロバイダを作成して使用する場合は、次の手順を実行します。

- 1 Git バージョン管理リポジトリ内のデフォルトの Terraform ディレクトリに、次のサブディレクトリ構造を追加します。

```
terraform.d/plugins/linux_amd64
```

2 カスタムの Terraform プロバイダ Go バイナリを `linux_amd64` ディレクトリに追加します。

デフォルトでは、`terraform init` はカスタム プロバイダ プラグインのディレクトリを検索します。

注： VMware は、カスタムの Terraform プロバイダの実行に失敗し、`no such file or directory` メッセージが送信される場合があることを認識しています。

この問題が発生した場合は、CGO が無効な（ゼロに設定された）状態でカスタム プロバイダ Go バイナリを再コンパイルしてください。CGO は、C コードを呼び出す Go パッケージのことです。

vRealize Automation Cloud Assembly マーケットプレイスの使用方法

リソース ライブラリの使用をすぐに開始するには、vRealize Automation Cloud Assembly マーケットプレイスからファイルをダウンロードします。マーケットプレイスからは、完成したクラウド テンプレートとオープンな仮想化イメージが得られます。

マーケットプレイスへのアクセス方法

vRealize Automation Cloud Assembly で、[インフラストラクチャ] - [接続] - [統合] の順に選択します。[統合の追加] をクリックし、[My VMware] をクリックして、My VMware アカウントの認証情報を入力します。

マーケットプレイスのクラウド テンプレート ファイルをダウンロードして使用する方法

[マーケットプレイス] タブで [取得] をクリックし、クラウド テンプレートのエンド ユーザー使用許諾契約書 (EULA) に同意します。次に、テンプレートを vRealize Automation Cloud Assembly プロジェクトに追加するか、単にダウンロードします。クラウド テンプレートは [デザイン] タブでアップロードできます。

プロジェクト ベースの例としては、ビッグ データを扱うプロジェクトの管理者になった場合が考えられます。チームを支援するために、チーム プロジェクトに追加する Hadoop テンプレートをマーケットプレイスで特定します。次に、そのクラウド テンプレートをリソース環境に合わせてカスタマイズし、リリースします。その後、チームが展開できるようにテンプレートを vRealize Automation Service Broker カタログにインポートします。

マーケットプレイス上のイメージ ファイルをダウンロードして使用する方法

[マーケットプレイス] タブで [取得] をクリックし、OVF または OVA イメージのエンド ユーザー使用許諾契約書 (EULA) に同意します。その後、OVF または OVA イメージをダウンロードして、クラウド テンプレート コードから参照できます。

前の例の続きを実行すると、チームは Hadoop 自体へのアクセスが必要になる可能性があります。Hadoop OVF をダウンロードし、クラウド アカウント リソース (vCenter Server コンテンツ ライブラリなど) に追加します。次に、その OVF イメージを参照する必要があるテンプレート コードを更新します。

vRealize Automation Cloud Assembly 展開の管理

7

vRealize Automation Cloud Assembly クラウド テンプレート開発者は、[展開] タブを使用して展開を管理します。失敗したプロビジョニング プロセスのトラブルシューティング、変更の実施、使用しない展開の削除を行うことができます。

展開とは、クラウド テンプレートをプロビジョニングしたインスタンスです。[展開] タブには、成功した展開と失敗した展開が表示されます。この画面を使用して、正常に完了した展開を管理できるほか、失敗した申請のトラブルシューティングを開始できます。

展開カードの操作

カード リストを使用して、展開を検索および管理できます。特定の展開をフィルタまたは検索してからこの展開でアクションを実行できます。

- 1 属性に基づいて申請をフィルタします。
- 2 キーワードまたは申請者に基づいて展開を検索します。
- 3 リストを時間または名前で並べ替えます。
- 4 使用されていない展開を削除してリソースを再利用するなど、展開レベルのアクションを実行します。

また、展開コスト、有効期限、およびステータスを表示することもできます。



この章には、次のトピックが含まれています。

- vRealize Automation Cloud Assembly で展開を監視する方法
- vRealize Automation Cloud Assembly の展開に失敗した場合の対処
- 完了した vRealize Automation Cloud Assembly 展開のライフサイクルを管理する方法
- vRealize Automation Cloud Assembly 環境で実行できるアクション

vRealize Automation Cloud Assembly で展開を監視する方法

vRealize Automation Cloud Assembly クラウド テンプレートを展開した後、要求を監視して、リソースがプロビジョニングされ実行されていることを確認できます。展開カードから、リソースのプロビジョニングを確認できます。次に、展開の詳細を確認できます。最後に、削除された展開を表示できます。

手順

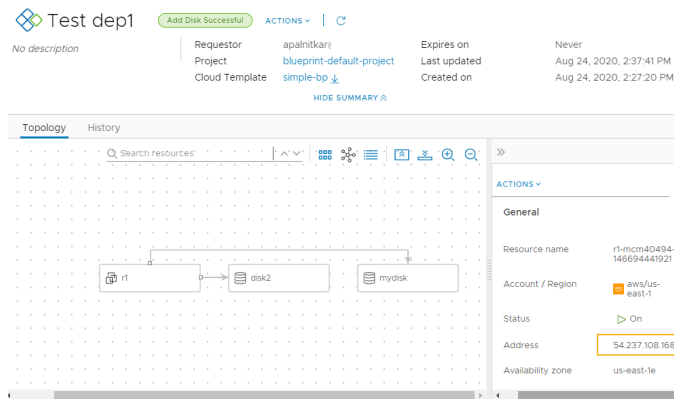
- 1 [展開] をクリックし、必要に応じてフィルタと検索を使用して、進行中の展開カードを特定します。
- 2 カードのステータスを確認します。

展開が進行中の場合、プロセス バーに残りのタスクの数が表示されます。展開が正常に完了すると、カードには展開に関する基本情報が表示されます。



- 3 リソースが展開された場所を調べるには、展開名をクリックし、[トポロジ] 画面で詳細を確認します。

主要なコンポーネントの IP アドレスが必要になることがあります。各コンポーネントをクリックすると、そのコンポーネントに固有の情報が表示されます。この例では、IP アドレスが強調表示されています。



外部リンクの可用性は、クラウド プロバイダによって異なります。使用可能な場合は、コンポーネントにアクセスするための認証情報がそのプロバイダで必要となります。

次のステップ

- 展開に変更を加えることができます。完了した [vRealize Automation Cloud Assembly 展開のライフサイクルを管理する方法](#)を参照してください。
- 展開が失敗する場合は、[vRealize Automation Cloud Assembly の展開に失敗した場合の対処](#)を参照してください。

vRealize Automation Cloud Assembly の展開に失敗した場合の対処

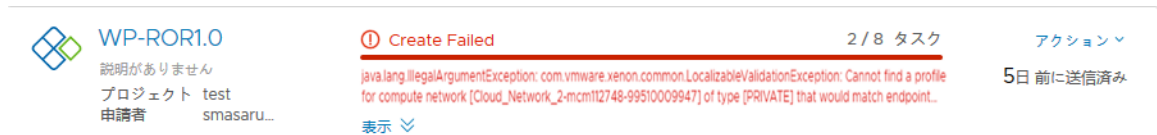
展開申請は、さまざまな理由で失敗することがあります。ネットワークトラフィック、ターゲットクラウドプロバイダのリソース不足、または展開仕様の不備が原因となる可能性があります。また、展開は成功しても、それが機能していないように見えることもあります。vRealize Automation Cloud Assembly を使用して展開を検証し、エラーメッセージを確認して、問題が環境や申請されたワークロードの仕様にあるのか、または他に理由があるのかを判断できます。

このワークフローを使用して、調査を開始します。このプロセスによって、失敗の原因が一時的な環境の問題であったことが判明する場合があります。条件が改善されたことを確認してから申請を再展開することで、このような問題は解決されました。他にも、調査で他の領域を詳しく確認する必要がある場合があります。

プロジェクトメンバーとして、vRealize Automation Cloud Assembly で申請の詳細を確認できます。

手順

- 1 申請が失敗したかどうかを判断するには、[展開] タブをクリックし、展開カードを見つけます。



失敗した展開はカードに示されます。

- a エラー メッセージを確認します。
- b 展開名をクリックすると、展開の詳細が表示されます。

2 展開の詳細画面で、[履歴] タブをクリックします。

The screenshot shows the vRealize Automation Cloud Assembly interface. At the top, there's a card for 'WP - ROR2' with a 'Create Failed' status. Below the card, there's a 'History' tab selected. The 'History' tab shows a table of events. The first event is 'REQUEST_FAILED' with a status of 'Failed'. The details of this event are expanded, showing an error message: 'Could not find any profile to match network 'WP-Network-Private' of type 'EXISTING' with constraints '[type:isolated-net, env:dev]'.'.

Annotations in the image:

- 2.a: Points to the 'CREATE' button in the 'History' tab.
- 2.b: Points to the 'Details' column in the 'Events' table.
- 2.c: Points to the 'Provisioning diagram' link in the 'History' tab.

- a イベント ツリーを確認して、プロビジョニング プロセスが失敗した場所を調べます。このツリーは、展開を変更する場合に便利ですが、変更は失敗します。

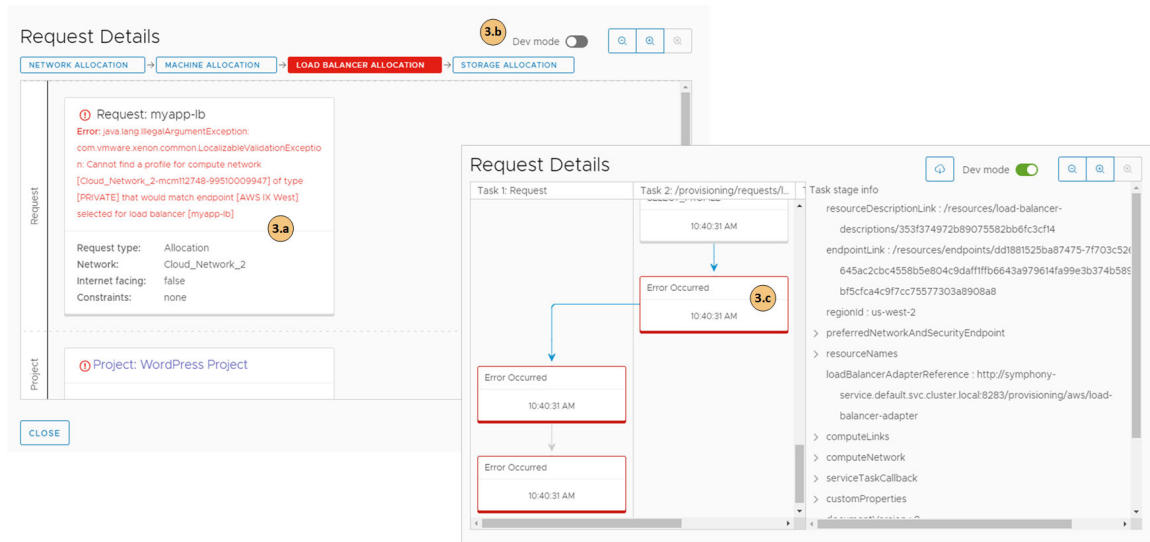
ツリーには、いつ展開アクションを実行したかも示されます。ツリーを使用して、失敗した変更のトラブルシューティングを行うことができます。

- b [詳細] には、詳細なエラー メッセージが表示されます。

- c 申請された項目が vRealize Automation Cloud Assembly のクラウド テンプレートの場合は、メッセージの右側にあるリンクをクリックすると、vRealize Automation Cloud Assembly が開き、[申請の詳細] を確認できます。

3 [申請の詳細] には、失敗したコンポーネントのプロビジョニング ワークフローが示されるため、問題を調査することができます。

申請履歴は 1 週間保持されます。



a エラー メッセージを確認します。

b [開発モード] をオンにして、単純なプロビジョニング ワークフローとより詳細なフローチャートを切り替えることができます。

c カードをクリックして展開スクリプトを確認します。

4 エラーを解決し、クラウド テンプレートを再展開します。

エラーは、テンプレートの構造内に含まれている場合もあれば、インフラストラクチャの構成方法に関係している場合もあります。

次のステップ

エラーを解決し、クラウド テンプレートが展開されると、[申請の詳細] に次の例のような情報が表示されます。申請の詳細を表示するには、[インフラストラクチャ] - [アクティビティ] - [申請] の順に選択します。



完了した vRealize Automation Cloud Assembly 展開のライフサイクルを管理する方法

展開がプロビジョニングされて実行が開始されたら、展開を管理するために実行できるアクションがいくつかあります。ライフサイクル管理には、展開のパワーオンまたはパワーオフ、サイズ変更、削除などがあります。個々のコンポーネントに対してさまざまなアクションを実行して管理することもできます。

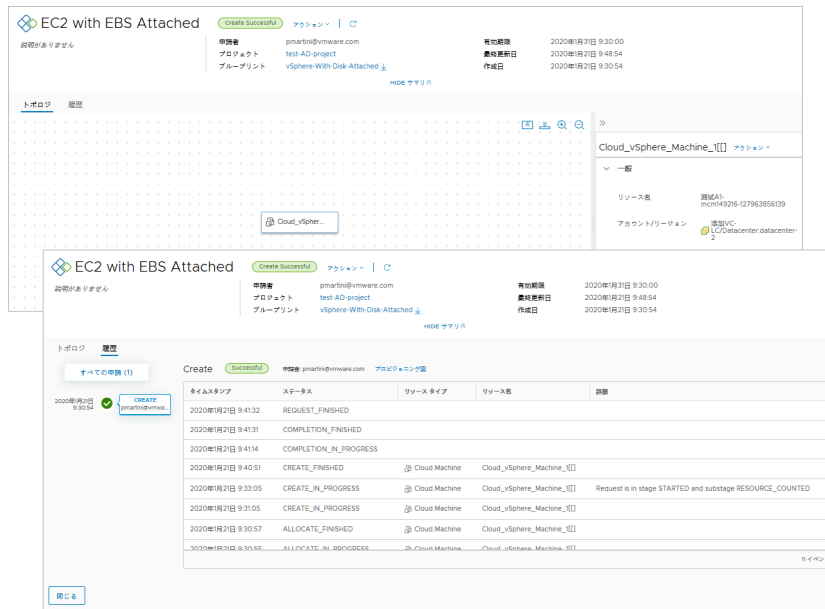
手順

- 1 [展開] をクリックして展開を見つけます。
- 2 展開の詳細にアクセスするには、展開の名前をクリックします。

[トポロジ] タブを使用して、展開の構造とリソースを視覚化できます。

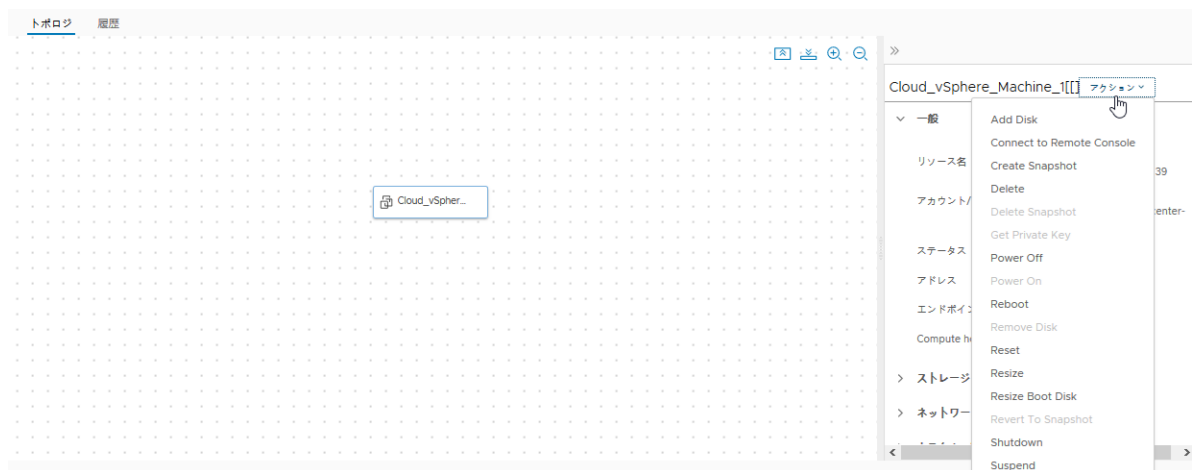
[履歴] タブには、すべてのプロビジョニング イベントと、申請された項目が展開された後に実行するアクションに関連するイベントがすべて表示されます。プロビジョニングのプロセスに問題がある場合は、[履歴] タブのイベントを使用して、障害のトラブルシューティングを行うことができます。

[コスト] タブには、一部のコンポーネントが展開されてからの現在のコストが表示されます。



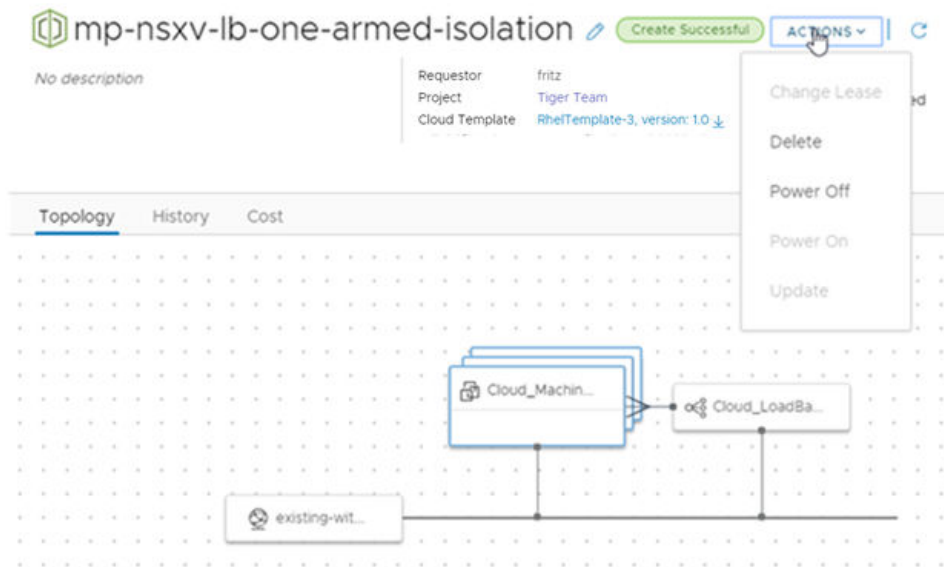
- 3 現在の設定では展開にコストがかかりすぎると判断し、コンポーネントのサイズを変更する場合は、[トポロジ] 画面でコンポーネントを選択してから、[アクション] - [サイズ変更] の順に選択します。

使用可能なアクションは、コンポーネント、クラウド アカウント、および権限によって異なります。



- 4 開発ライフサイクルの一環として、展開の1つが不要になったとします。デプロイを削除してリソースを再利用するには、[アクション] - [削除] の順に選択します。

使用可能なアクションは、展開の状態によって異なります。



次のステップ

実行可能な操作の詳細については、[vRealize Automation Cloud Assembly 環境で実行できるアクション](#)を参照してください。

vRealize Automation Cloud Assembly 環境で実行できるアクション

クラウド テンプレートを展開した後、リソースを管理するアクションを vRealize Automation Cloud Assembly で実行できます。使用可能なアクションは、リソースのタイプと、特定のクラウド アカウントまたは統合プラットフォームでアクションがサポートされているかどうかによって異なります。

使用可能なアクションは、管理者から付与されている実行資格によっても異なります。

管理者またはプロジェクト管理者は、Day 2 アクション ポリシーを vRealize Automation Service Broker で設定できます。[利用者に Service Broker の Day 2 アクション ポリシーの資格を付与する方法](#)を参照してください。

以下のリストに含まれていないアクションも表示される場合があります。多くの場合、これらは管理者によって追加されたカスタム アクションです。たとえば、[仮想マシンを vMotion するための vRealize Automation Cloud Assembly カスタム アクションの作成方法](#)です。

表 7-1. 実行可能なアクションのリスト

アクション	適用されるリソースタイプ	対象のクラウド アカウントまたは統合	説明
ディスクの追加	マシン	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	既存の仮想マシンにディスクを追加します。
リースの変更	展開	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	<p>リースの有効期限の日時を変更します。</p> <p>リースの有効期限が切れると、展開が破棄され、リソースが回収されます。</p> <p>リース ポリシーは vRealize Automation Service Broker で設定されます。</p>
セキュリティ グループの変更	マシン	<ul style="list-style-type: none"> ■ VMware vSphere 	<p>セキュリティ グループは、展開内のマシン ネットワークと関連付けたり、関連付けを解除したりできます。変更アクションは、NSX-V および NSX-T の既存のセキュリティ グループおよびオンデマンド セキュリティ グループに適用されます。このアクションは、マシン クラスターではなく、単一のマシンに対してのみ使用できます。</p> <p>セキュリティ グループをマシン ネットワークに関連付けるには、そのセキュリティ グループが展開に含まれている必要があります。</p> <p>展開内のすべてのマシンを含むすべてのネットワークからセキュリティ グループの関連付けを解除しても、セキュリティ グループは展開から削除されません。</p> <p>これらの変更は、ネットワーク プロファイルの一部として適用されるセキュリティ グループには影響しません。</p> <p>このアクションにより、マシンを再作成することなくマシンのセキュリティ グループ設定が変更されます。これは、非破壊的な変更です。</p> <p>マシンのセキュリティ グループの変更</p> <ul style="list-style-type: none"> ■ マシンのセキュリティ グループ設定を変更するには、[トポロジ] ペインでマシンを選択し、右ペインの [アクション] メニューをクリックして、[セキュリティ グループの変更] を選択します。これで、セキュリティ グループとマシン ネットワークの関連付けを追加または削除できます。
リモート コンソール への接続	マシン	<ul style="list-style-type: none"> ■ VMware vSphere 	<p>選択したマシンでリモート セッションを開きます。</p> <p>正常に接続できるように、次の要件を確認します。</p> <ul style="list-style-type: none"> ■ 展開の利用者として、プロビジョニングされたマシンがパワーオン状態であることを確認します。
スナップショットの作成	マシン	<ul style="list-style-type: none"> ■ Google Cloud Platform ■ VMware vSphere 	<p>仮想マシンのスナップショットを作成します。</p> <p>vSphere で 2 つのスナップショットのみが許可されていて、すでに 2 つある場合は、スナップショットを 1 つ削除しない限りこのコマンドを使用できません。</p>

表 7-1. 実行可能なアクションのリスト（続き）

アクション	適用されるリソースタイプ	対象のクラウド アカウントまたは統合	説明
削除	展開	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	<p>展開を削除します。</p> <p>すべてのリソースが削除され、再要求されます。</p> <p>削除が失敗した場合は、展開で削除アクションを再度実行できます。2 回目の試行では、[削除の失敗を無視] を選択できます。このオプションを選択すると、展開は削除されますが、リソースは再利用されない可能性があります。展開がプロビジョニングされたシステムを確認して、すべてのリソースが削除されていることを確認する必要があります。削除されていない場合は、それらのシステム上の残留リソースを手動で削除する必要があります。</p>
	NSX ゲートウェイ	<ul style="list-style-type: none"> ■ NSX 	NSX-T または NSX-V ゲートウェイから NAT ポート転送ルールを削除します。
	マシンとロード バランサ	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere ■ VMware NSX 	展開からマシンまたはロード バランサを削除します。このアクションを実行すると、展開が使用できなくなる可能性があります。
	セキュリティ グループ	<ul style="list-style-type: none"> ■ NSX-T ■ NSX-V 	<p>展開のどのマシンにも関連付けられていないセキュリティ グループは、展開から削除されます。</p> <ul style="list-style-type: none"> ■ セキュリティ グループがオンデマンドの場合、エンドポイントで破棄されます。 ■ セキュリティ グループが共有されている場合、このアクションは失敗します。
スナップショットの削除	マシン	<ul style="list-style-type: none"> ■ VMware vSphere ■ Google Cloud Platform 	仮想マシンのスナップショットを削除します。
タグの編集	展開	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	個々の展開リソースに適用されるリソース タグを追加または変更します。

表 7-1. 実行可能なアクションのリスト（続き）

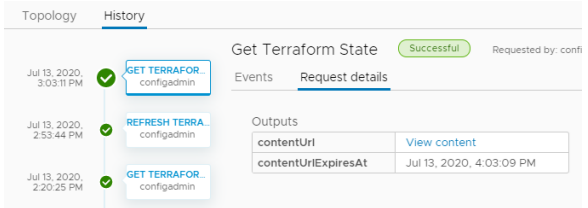
アクション	適用されるリソースタイプ	対象のクラウド アカウントまたは統合	説明
Terraform 状態を取得します	Terraform 構成	<ul style="list-style-type: none"> Amazon Web Service Google Cloud Platform Microsoft Azure VMware vSphere 	<p>Terraform 状態ファイルを表示します。</p> <p>クラウド プラットフォームに展開された Terraform マシンに加えられたすべての変更を表示したり、展開を更新したりするには、最初に Refresh Terraform State アクションを実行し、次にこの Get Terraform State アクションを実行します。</p> <p>ファイルがダイアログ ボックスに表示されたとき、ファイルを使用できるようにしてから約 1 時間後に、新しい更新アクションを実行する必要があります。ファイルが後で必要な場合はコピーできます。</p> <p>展開の [履歴] タブでファイルを表示することもできます。[イベント] タブで [Get Terraform State] イベントを選択し、[申請の詳細] をクリックします。ファイルの有効期限が切れていない場合は、[コンテンツの表示] をクリックします。ファイルの有効期限が切れている場合は、Refresh アクションと Get アクションを再度実行します。</p> 
パワーオフ	展開	<ul style="list-style-type: none"> Amazon Web Service Microsoft Azure VMware vSphere 	ゲスト OS をシャットダウンせずに展開を終了します。
	マシン	<ul style="list-style-type: none"> Amazon Web Service Google Cloud Platform Microsoft Azure VMware vSphere 	ゲスト OS をシャットダウンせずにマシンをパワーオフします。
パワーオン	展開	<ul style="list-style-type: none"> Amazon Web Service Microsoft Azure VMware vSphere 	展開を開始します。リソースがサスペンド中だった場合は、リソースがサスペンドされた時点から通常の処理が再開されます。
	マシン	<ul style="list-style-type: none"> Amazon Web Service Google Cloud Platform Microsoft Azure VMware vSphere 	マシンをパワーオンします。マシンがサスペンド中だった場合は、マシンがサスペンドされた時点から通常の処理が再開されます。
再起動	マシン	<ul style="list-style-type: none"> Amazon Web Service VMware vSphere 	<p>仮想マシンのゲスト OS を再起動します。</p> <p>vSphere マシンの場合、このアクションを使用するには、VMware Tools をマシンにインストールしておく必要があります。</p>
再構成	ロード バランサ	<ul style="list-style-type: none"> Amazon Web Service Microsoft Azure VMware NSX 	<p>ロード バランサのサイズとログ レベルを変更します。</p> <p>また、ルートの追加または削除、プロトコル、ポート、健全性構成、メンバー プール設定の変更もできます。</p>

表 7-1. 実行可能なアクションのリスト（続き）

アクション	適用されるリソースタイプ	対象のクラウド アカウントまたは統合	説明
	NSX ゲートウェイ ポート転送	<ul style="list-style-type: none"> ■ NSX-T ■ NSX-V 	NSX-T または NSX-V ゲートウェイから NAT ポート転送ルールを追加、編集、削除します。
Terraform 状態の更新	Terraform 構成	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	<p>Terraform 状態ファイルの最新のイテレーションを取得します。</p> <p>クラウド プラットフォームに展開された Terraform マシンに加えられたすべての変更を取得したり、展開を更新したりするには、最初にこの Refresh Terraform State アクションを実行します。</p> <p>ファイルを表示するには、構成に対して [Get Terraform State] アクションを実行します。</p> <p>展開の [履歴] タブを使用して、更新プロセスを監視します。</p>
ディスクの削除	マシン	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	既存の仮想マシンからディスクを削除します。
リセット	マシン	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ VMware vSphere 	ゲスト OS をシャットダウンせずにマシンを強制再起動します。
サイズ変更	マシン	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ Google Cloud Platform ■ VMware vSphere 	仮想マシンの CPU とメモリを増加または減少させます。
起動ディスクのサイズ変更	マシン	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere 	起動ディスク メディアのサイズを大きく、または小さくします。
ディスクのサイズ変更	ストレージ ディスク	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Google Cloud Platform 	ストレージ ディスクの容量を増やします。
再起動	マシン	<ul style="list-style-type: none"> ■ Microsoft Azure 	実行中のマシンをシャットダウンして再起動します。
スナップショットに戻す	マシン	<ul style="list-style-type: none"> ■ Google Cloud Platform ■ VMware vSphere 	<p>マシンの以前のスナップショットに戻ります。</p> <p>このアクションを使用するには、既存のスナップショットが必要です。</p>
Puppet タスクの実行	管理対象リソース	<ul style="list-style-type: none"> ■ Puppet Enterprise 	<p>環境内のマシンで選択したタスクを実行します。</p> <p>タスクは、Puppet インスタンスで定義されています。タスクを識別し、入力パラメータを指定できる必要があります。</p>
シャットダウン	マシン	<ul style="list-style-type: none"> ■ VMware vSphere 	ゲスト OS をシャットダウンして、マシンをパワーオフします。このアクションを使用するには、VMware Tools をマシンにインストールしておく必要があります。
中断	マシン	<ul style="list-style-type: none"> ■ Microsoft Azure ■ VMware vSphere 	マシンを使用できないように一時停止して、使用しているストレージ以外のシステム リソースが使用されないようにします。

表 7-1. 実行可能なアクションのリスト（続き）

アクション	適用されるリソースタイプ	対象のクラウド アカウントまたは統合	説明
更新	展開	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	<p>入力パラメータに基づいて展開を変更します。</p> <p>例については、展開されたマシンを別のネットワークに移動する方法を参照してください。</p>
タグの更新	マシンとディスク	<ul style="list-style-type: none"> ■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere 	個々のリソースに適用されるタグを追加、変更、または削除します。