vRealize Automation Cloud Assembly の使用と管理

2022 年 10 月

vRealize Automation 8.0

vmware®

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

ヴイ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

目次

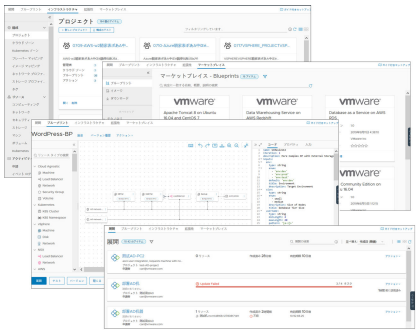
vRealize Automation Cloud Assembly の概要

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、コードとしてのブループリントを開発します。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、展開されたブループリン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、ブループリント ライブラリの構築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 のブループリントとイメージがあります。
- [ブループリント] 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 は、ブループリン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。

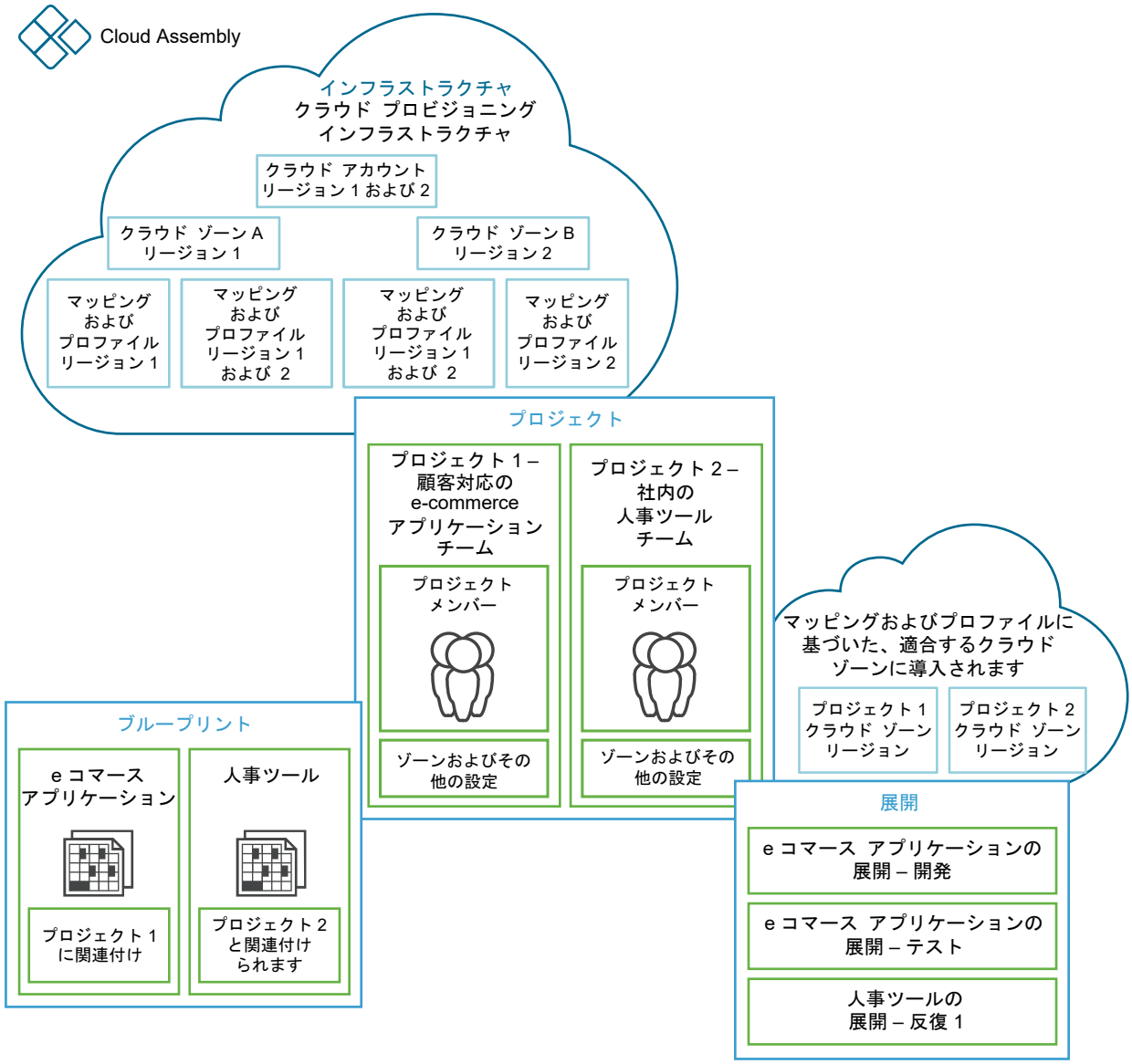
Cloud Assembly 管理者（一般的にはクラウド管理者と呼ばれる）として、プロビジョニング インフラストラクチャを設定し、ユーザーとリソースをグループ化するプロジェクトを作成します。

- 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構築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。

ブループリント開発者は、1 つまたは複数のプロジェクトのメンバーです。プロジェクトの 1 つに関連付けられているクラウド ゾーンにブループリントを作成して展開します。

- キャンバスを使用して、プロジェクトのブループリントを開発します。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 からブループリン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、ブループリン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ブループリントを作成し、展開する方法の例が必要な場合は、WordPress の使用事例を参照してください。


Cloud Assembly
インフラストラクチャ
クラウド プロビジョニング
インフラストラクチャ
クラウド アカウント
リージョン 1 および 2
クラウド ゾーン A
リージョン 1
クラウド ゾーン B
リージョン 2
マッピング
および
プロファイル
リージョン 1
マッピング
および
プロファイル
リージョン 1
および 2
マッピング
および
プロファイル
リージョン 1
および 2
マッピング
および
プロファイル
リージョン 2
プロジェクト
プロジェクト 1 –
顧客対応の
e-commerce
アプリケーション
チーム
プロジェクト
メンバー
ゾーンおよびその
他の設定
プロジェクト 2 –
社内の
人事ツール
チーム
プロジェクト
メンバー
ゾーンおよびその
他の設定
マッピングおよびプロファイルに
基づいた、適合するクラウド
ゾーンに導入されます
プロジェクト 1
クラウド ゾーン
リージョン
プロジェクト 2
クラウド ゾーン
リージョン
ブループリント
e コマース
アプリケーション
プロジェクト 1
に関連付け
人事ツール
プロジェクト 2
と関連付け
られます
展開
e コマース アプリケーションの
展開 – 開発
e コマース アプリケーションの
展開 – テスト
人事ツールの
展開 – 反復 1

組織での vRealize Automation Cloud Assembly の設定

2

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 Cloud Assembly 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 Cloud Assembly のユーザー ロールについて

ユーザー 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。組織レベルで定義されるロールと、vRealize Automation Cloud Assembly に固有のロールがあります。

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。

表 2-1. サービス ロール

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が必要です。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すべてのユーザー。 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。これらの 2 つのロールの権限は、以下で定義されています。

プロジェクトのロールと権限

プロジェクトのロール、プロジェクト管理者、およびプロジェクト メンバーは vRealize Automation Cloud Assembly で定義され、プロジェクトによって異なる場合があります。

次の表で権限が定義されている場合、クラウド管理者がユーザー インターフェイスのすべての領域に対して完全な権限を持っていることになります。

プロジェクト管理者は、クラウド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。

表 2-2. プロジェクト管理者の権限

タブ	ノードまたは領域	表示	作成	変更または削除
インフラストラクチャ	構成 - プロジェクト	はい（自分のプロジェクトのみ）	いいえ	はい（自分のプロジェクトのみ）
	構成 - クラウド ゾーン	いいえ	いいえ	いいえ
	構成 - フレーバー マッピング	はい	いいえ	いいえ
	構成 - イメージ マッピング	はい	いいえ	いいえ
	構成 - ネットワーク プロファイル	はい	いいえ	いいえ
	構成 - ストレージ プロファイル	はい	いいえ	いいえ
	構成 - タグ	はい	いいえ	いいえ
	リソース - コンピューティング	はい	いいえ	いいえ
	リソース - ネットワーク	はい	いいえ	いいえ
	リソース - ストレージ	はい	いいえ	いいえ
	リソース - マシン	はい（自分のプロジェクトのみ）	はい	はい（自分のプロジェクトのみ）

表 2-2. プロジェクト管理者の権限 （続き）

タブ	ノードまたは領域	表示	作成	変更または削除
	リソース - ボリューム			
	アクティビティ - 申請	はい（自分のプロジェクトのみ）	該当なし	はい（自分のプロジェクトのみ）
	アクティビティ - イベント	はい（自分のプロジェクトのみ）	該当なし	はい（自分のプロジェクトのみ）
	接続 - クラウド アカウント	いいえ	いいえ	いいえ
	接続 - 統合		いいえ	いいえ
	接続 - クラウド プロキシ		いいえ	いいえ
	コスト - VMC の評価	はい	いいえ	いいえ
	コスト - プライベート クラウド	はい	いいえ	いいえ
	オンボーディング		いいえ	いいえ
ブループリント	ブループリント	はい（自分のプロジェクトのみ）	はい（自分のプロジェクトのみ）	はい（自分のプロジェクトのみ）
展開	展開	はい（自分のプロジェクトのみ）	該当なし	はい（自分のプロジェクトのみ）

プロジェクト メンバーは通常、ブループリントを作成して展開する開発者です。

表 2-3. プロジェクト メンバーの権限

タブ	ノードまたは領域	表示	作成	変更または削除
インフラストラクチャ	構成 - プロジェクト	はい（メンバーになっているプロジェクトのみ）	いいえ	いいえ
	構成 - クラウド ゾーン	いいえ	いいえ	いいえ
	構成 - フレーバー マッピング	はい	いいえ	いいえ
	構成 - イメージ マッピング	はい	いいえ	いいえ
	構成 - ネットワーク プロファイル	はい	いいえ	いいえ
	構成 - ストレージ プロファイル	はい	いいえ	いいえ
	構成 - タグ	はい	いいえ	いいえ
	リソース - コンピューティング	はい	いいえ	いいえ
	リソース - ネットワーク	はい	いいえ	いいえ

表 2-3. プロジェクト メンバーの権限 （続き）

タブ	ノードまたは領域	表示	作成	変更または削除
	リソース - ストレージ	はい	いいえ	いいえ
	リソース - マシン	はい（展開済みのもののみ）	はい	はい（展開済みのもののみ）
	リソース - ボリューム			
	アクティビティ - 申請	はい（展開済みのもののみ）	該当なし	はい（展開済みのもののみ）
	アクティビティ - イベント	はい（展開済みのもののみ）	該当なし	はい（展開済みのもののみ）
	接続 - クラウド アカウント	いいえ	いいえ	いいえ
	接続 - 統合			
	接続 - クラウド プロキシ			
	コスト - VMC の評価	はい	いいえ	いいえ
	コスト - プライベート クラウド	はい	いいえ	いいえ
	オンボーディング			
ブループリント	ブループリント	はい（自分のプロジェクトのみ）	はい（自分のプロジェクトのみ）	はい（自分のプロジェクトのみ）
展開	展開	はい（自分の展開のみ。プロジェクトの展開がすべてのプロジェクト メンバーと共有されている場合を除く。）	該当なし	はい（自分の展開のみ。プロジェクトの展開がすべてのプロジェクト メンバーと共有され、Day 2 アクションを実行する資格が付与されている場合を除く。）

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、その領域にブループリン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	VMware ID。 ■ 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	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。 ■ *.vmwareidentity.com ■ gaz.csp-vidm-prod.com ■ *.vmware.com 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。 必要なポートおよびプロトコルの関連情報については、次を参照してください。 ■ インストールに関するヘルプのポートおよびプロトコル ■ リファレンス アーキテクチャに関するヘルプのポートの要件

目的	必要なもの
Amazon Web Services (AWS) クラウド アカウントの追加	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。 ■ 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 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。 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。 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。 ■ 自動スケーリングのアクション： ■ 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： ■ * 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。 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。 ■ AWS STS リソース： ■ * すべての STS リソース権限を付与します。 EC2 機能を許可するには、次の AWS 権限が必要です。 ■ EC2 のアクション： ■ 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

- ec2:CreateSnapshot
- ec2:ModifyInstanceAttribute
- ec2:DescribeRouteTables
- ec2:DescribeInstanceStatus
- ec2:DetachVolume
- ec2:RebootInstances
- ec2:AuthorizeSecurityGroupEgress
- ec2:ModifyVolume
- ec2:TerminateInstances
- ec2:DescribeSpotFleetRequestHistory
- ec2:DescribeTags
- ec2:CreateTags
- ec2:RunInstances
- ec2:DescribeNatGateways
- ec2:StopInstances
- ec2:DescribeSecurityGroups
- ec2:CreateVolume
- ec2:DescribeSpotFleetRequests
- ec2:DescribeImages
- ec2:DescribeVpcs
- ec2:DeleteSecurityGroup
- ec2:DeleteVpc
- ec2:CreateSubnet
- ec2:DescribeSubnets
- ec2:RequestSpotFleet

  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。

- EC2 のリソース：
  - *

    すべての EC2 リソース権限を付与します。

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。

- ロード バランサのアクション：
  - elasticloadbalancing:DeleteLoadBalancer
  - elasticloadbalancing:DescribeLoadBalancers
  - elasticloadbalancing:RemoveTags
  - elasticloadbalancing:CreateLoadBalancer
  - elasticloadbalancing:DescribeTags
  - elasticloadbalancing:ConfigureHealthCheck
  - elasticloadbalancing:AddTags
  - elasticloadbalancing:CreateTargetGroup
  - elasticloadbalancing:DeleteLoadBalancerListeners
  - elasticloadbalancing:DeregisterInstancesFromLoadBalancer
  - elasticloadbalancing:RegisterInstancesWithLoadBalancer
  - elasticloadbalancing:CreateLoadBalancerListeners

目的	必要なもの
	■ ロード バランサ リソース： ■ * ロード バランサ リソースのすべての権限を付与します。 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。 ■ 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	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）。 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。 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。 次の情報をメモしておきます。 ■ サブスクリプション ID Microsoft Azure サブスクリプションへのアクセスを許可します。 ■ テナント ID 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。 ■ クライアント アプリケーションの ID 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。 ■ クライアント アプリケーションのプライベート キー 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。 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。 ■ Microsoft Compute ■ 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 ■ 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

- 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
  - Microsoft.Resources/subscriptions/resourcegroups/delete
  - Microsoft.Resources/subscriptions/resourcegroups/read
  - Microsoft.Resources/subscriptions/resourcegroups/write
- Microsoft Storage
  - Microsoft.Storage/storageAccounts/delete
  - Microsoft.Storage/storageAccounts/listKeys/action
  - Microsoft.Storage/storageAccounts/read
  - Microsoft.Storage/storageAccounts/write
- Microsoft Web
  - 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

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。

- 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

目的	必要なもの
	■ Microsoft.Web/sites/functions/masterkey/read 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。 ■ 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	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。 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。 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。 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。 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。 ■ roles/compute.admin 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。 ■ roles/iam.serviceAccountUser 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。 ■ compute.* ■ resourcemanager.projects.get ■ resourcemanager.projects.list ■ serviceusage.quotas.get ■ serviceusage.services.get ■ serviceusage.services.list ■ roles/compute.imageUser 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。 ■ 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 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。 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。 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。 ■ compute.acceleratorTypes ■ compute.addresses.get ■ compute.addresses.list ■ compute.addresses.use

目的	必要なもの
	■ 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.useExternalIp ■ compute.projects.get ■ compute.regionOperations.get ■ compute.regionOperations.list ■ compute.regions ■ compute.reservations.get ■ compute.reservations.list ■ compute.subnetworks.get ■ compute.subnetworks.list ■ compute.subnetworks.use ■ compute.subnetworks.useExternalIp ■ compute.targetPools.get ■ compute.targetPools.list

目的	必要なもの

- 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

  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。

  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。

  - 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
	■ 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.useExternalIp ■ 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.useExternalIp ■ 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
	■ 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	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。 ■ 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 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。
NSX-V クラウド アカウントを追加する	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。 ■ 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 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。

目的	必要なもの
vCenter クラウド アカウントを追加する	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。 ■ vCenter の IP アドレスまたは FQDN 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。
VMware Cloud on AWS (VMC) クラウド アカウントを追加する	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。 ■ 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 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。 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。

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 で実行されるときの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 2-4. vCenter Server インスタンスを管理するために vSphere エージェントで必要な権限

属性値	権限
データストア	■ 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	■ フォルダの作成 ■ フォルダの削除
グローバル	■ 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 2-4. vCenter Server インスタンスを管理するために vSphere エージェントで必要な権限 （続き）

属性値	権限
リソース	■ 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	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。 ■ 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	■ 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 2-4. vCenter Server インスタンスを管理するために vSphere エージェントで必要な権限 （続き）

属性値	権限
vApp	■ インポート ■ vApp アプリケーションの設定 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。 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。
仮想マシン - インベントリ	■ 既存から作成 ■ 新規作成 ■ 移動 ■ 削除
仮想マシン - 相互作用	■ 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	■ 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	■ 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	■ 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、これらを後で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 ブループリン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、「WordPress の使用事例」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 Cloud Assembly 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 アカウントを設定します。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 プロキシを設定します。vRealize Automation のインターネット プロキシ サーバの設定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。

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 ブループリン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。

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 Cloud Assembly 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 サーバの設定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。

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 ブループリン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 Cloud Assembly 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 プロキシを設定します。vRealize Automation のインターネット プロキシ サーバの設定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。

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 Cloud Assembly に vCenter クラウド アカウントを作成します

vRealize Automation Cloud Assembly ブループリントをデプロイするアカウント領域に、vCenter クラウド アカウントを追加します。

ネットワークとセキュリティのために、NSX-T または NSX-V のクラウド アカウントを vCenter のクラウド アカウントに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 Cloud Assembly 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データセンターが収集されます。

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 Cloud Assembly 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 Cloud Assembly 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。

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。

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。マシンや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 リソースを設定します。4 章 vRealize Automation Cloud Assembly 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。

vRealize Automation Cloud Assembly 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。

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 Cloud Assembly 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 Cloud Assembly 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。
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。
6. [保存] をクリックします。

次のステップ

vCenter クラウド アカウントを作成または編集して、この NSX クラウド アカウントと関連付けることができます。vRealize Automation Cloud Assembly 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構築を参照してください。

vRealize Automation Cloud Assembly 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、vCenter クラウド アカウントに関連付けることができます。

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。この 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。

  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 Cloud Assembly 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 Cloud Assembly 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（上記参照）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-T クラウド アカウントと関連付ける vCenter Server 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。

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。

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 Cloud Assembly 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構築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 ブループリン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、VMware Cloud on AWS の使用事例 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 Cloud Assembly 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 サーバの設定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必要がある場合は、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。

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、VMware Cloud on AWS の使用事例 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。

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 サーバの設定方法を参照してください。

vRealize Automation Cloud Assembly での GitLab および GitHub 統合の使用方法

vRealize Automation Cloud Assembly は GitLab および GitHub リポジトリとの統合をサポートしているため、ソース管理でブループリントおよびアクション 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 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、Git でブループリントを検出するために、特定の構造でブループリン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、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ブループリントを使用できるようになります。これらのブループリントは、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。ブループリントの場合は、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 のブループリント管理ページで Git から取得します。

はじめに

GitLab または GitHub によってブループリントが検出されるようにするには、ブループリン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ブループリントを設定して保存します。有効なブループリントのみが GitLab にインポートされます。
  - ブループリント用として指定するフォルダを 1 つ以上作成します。
  - すべてのブループリントは、 `blueprint.yaml` ファイル内に格納する必要があります。
  - ブループリントの最上部に、`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ブループリントについて、次のガイドラインを遵守する必要があります。

- 各ブループリントを個別のフォルダに配置する必要があります。
- すべてのブループリントに `blueprint.yaml` という名前を付ける必要があります。
- すべてのブループリント 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。
- 有効なブループリントのみがインポートされます。
- Git からインポートされたドラフトのブループリントを更新し、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。ブループリントを更新し、今後 Git と同期する場合は、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 ブループリント統合の設定

  この手順では、vRealize Automation Cloud Assembly で GitLab 統合を設定することで、リポジトリのブループリン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ブループリントを自動的にダウンロードできるようにします。GitLab でブループリントを使用するには、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ブループリン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 GitLab ブループリント統合の設定

この手順では、vRealize Automation Cloud Assembly で GitLab 統合を設定することで、リポジトリのブループリン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ブループリントを自動的にダウンロードできるようにします。GitLab でブループリントを使用するには、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ブループリン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ブループリントを使用できるようになります。これらのブループリントは、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ブループリントを 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、新しいブループリントを 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。リポジトリにブループリントを追加するには、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ブループリントを更新し、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。ブループリントを更新し、今後 Git と同期する場合は、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 で使用するようにブループリントを設定し、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、指定したブループリントを 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 によってブループリントが検出されるようにするには、ブループリン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ブループリントを設定して保存します。有効なブループリントのみが GitLab にインポートされます。vRealize Automation Cloud Assembly での GitLab および GitHub 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ブループリントを受け入れるように GitLab 接続を設定します。

   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、ブループリン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、ブループリン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 からブループリン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ブループリントを設定して保存します。有効なブループリントのみが GitHub にインポートされます。vRealize Automation Cloud Assembly での GitLab および GitHub 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。

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 からブループリントを取得できるようになりました。

vRealize Automation での外部 IP アドレス管理統合ポイントの構成

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、ブループリン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 のブループリン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 での外部 IP アドレス管理統合ポイントの追加 を参照してください。

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 Cloud Assembly 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。

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 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 での外部 IP アドレス管理統合ポイントの追加 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 Cloud Assembly 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 Cloud Assembly 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（ブループリント用の 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、ブループリントで使用できます。

次のステップ

目的のブループリントに 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 AutomationCloud 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 Cloud Assembly のユーザー ロールについてを参照してください。
- vRealize Orchestrator7.x インスタンスをバージョン 8.0 に移行します。『VMware vRealize Orchestrator のインストール、構成、および移行』の「スタンドアローン vRealize Orchestrator の外部 vRealize Orchestrator 8.0 への移行」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使用できる複数の 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ブループリントで使用される 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。詳細については、vRealize Automation Cloud Assembly 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 統合を設定します。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。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。

手順

1 [インフラストラクチャ] - [設定] - [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

ブループリン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される 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使用できる複数の 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される 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。詳細については、vRealize Automation Cloud Assembly 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 統合を設定します。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。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。

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

ブループリン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

Pivotal Container Service (PKS) または Red Hat OpenShift を vRealize Automation Cloud Assembly と統合して、Kubernetes リソースを管理および展開することができます。vRealize Automation Cloud Assembly で外部 Kubernetes リソースを統合することもできます。

PKS 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。

- 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 Kubernetes クラスタと名前空間の操作

  vRealize Automation Cloud Assembly で Kubernetes 展開の基盤として機能する Kubernetes クラスタと名前空間の構成を追加、表示、および管理できます。

- 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 展開で使用される Kubernetes クラスタおよび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できます。さらに、どのプロパティがクラスタに対して許容されるかも指定できます。

- vRealize Automation Cloud Assembly のブループリントへの Kubernetes コンポーネントの追加

  Kubernetes コンポーネントを vRealize Automation Cloud Assembly ブループリン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でプロビジョニングされた Kubernetes クラスタを管理できます。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 Cloud Assembly 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 クラスタを検出してそれらのクラスタ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、ブループリン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 クラスタと名前空間の操作

vRealize Automation Cloud Assembly で Kubernetes 展開の基盤として機能する Kubernetes クラスタと名前空間の構成を追加、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に構成されている、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。

- 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 Kubenetes 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 Kubconfig ファイルへのリンクが名前空間に含まれます。

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 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。

2 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 PKS 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。	PKS 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の適切な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、Kubernentes 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	PKS 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 プロキシと証明書情報を選択する必要があります。

3 [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。

1 [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。

2 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。

3 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。

4 この名前空間に関連付ける [クラスタ] を指定します。

5 [作成] をクリックして、名前空間を vRealize Automation Cloud Assembly に追加します。

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 クラスタおよび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できます。さらに、どのプロパティがクラスタに対して許容されるかも指定できます。

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。

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。

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、[追加] ボタンをクリックして、Kubernetes 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] ページで、クラスタに Kubenetes 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に構成されます。

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] - [構成] - [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、[ブループリント] ページを使用して、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。

vRealize Automation Cloud Assembly のブループリントへの Kubernetes コンポーネントの追加

Kubernetes コンポーネントを vRealize Automation Cloud Assembly ブループリン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 のブループリントに Kubernetes コンポーネントを追加するには、ブループリントをクリックし、[新規] を選択してから、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 クラスタをブループリン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 PKS 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。

ブループリントの例

最初のブループリントの例は、タグ付けによって制御される単純な Kubernetes 展開のブループリントを示しています。Kubernetes ゾーンのページで構成した 2 つの展開プランで Kubernetes ゾーンが作成されました。この場合、`placement:tag` というタグがゾーンの機能として追加され、ブループリントの同様の制約と一致させるために使用されました。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 番目のブループリントの例では、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ブループリン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 PKS 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、ブループリントに *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、ブループリン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_KBS_Namespace_1:
8        type: Cloud.KBS.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。

Kubernetes での vRealize Automation Cloud Assembly 拡張性の使用

vRealize Automation Cloud Assembly には、Kubernetes クラスタ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 クラスタの割り当て
- Kubernetes クラスタのプロビジョニング後
- Kubernetes クラスタの削除後
- Kubernetes クラスタのプロビジョニング
- Kubernetes クラスタの削除

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 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 および Ansible Open Source との連携がサポートされています。

Puppet との連携

Puppet ベースの設定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 の有効なインスタンスが必要です。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 設定管理を、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 テンプレートを設定します。

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、設定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。

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 `` `~/var/tmp/vmware/provider/user_defined_script/$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ | head -1)/` `` にあります。 完全なログを確認するには、[log.txt] ファイルを参照してください。詳細な Puppet エージェント ログを確認するには、https://puppet.com/docs/puppet/4.8/services_agent_unix.html#logging を参照してください。
Puppet の削除および実行に関連するタスクのログ	ログは、PE の `` `~/var/tmp/vmware/provider/user_defined_script/$(ls -t ~/var/tmp/vmware/provider/user_defined_script/ | head -1)/` `` 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 はブループリントで使用できます。

次のステップ

目的のブループリントに Puppet コンポーネントを追加します。

1 ブループリント メニューの [コンテンツ管理] 見出しの下で [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	このブループリントで使用する 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	このブループリントで使用する Puppet のロールを選択します。
エージェント実行間隔	このブループリン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。ブループリントに、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。ブループリントで 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 winm に設定されます。

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、ブループリン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 はブループリントで使用できます。

次のステップ

目的のブループリントにコンポーネントを追加します。

1 [ブループリント 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 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 プロパティを設定します。

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、ブループリン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 統合ではブループリントの 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、プレイブックの実行時に Ansible コントロール マシンでエラーが発生します。

次のブループリントの YAML スニペットは、hostVariables プロパティの使用方法の例を示しています。

```
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}
```

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 統合では、Active Directory サーバへの LDAP 接続のみがサポートされます。

前提条件

- vCenter オンプレミスと Active Directory の統合を設定する場合は、Active Directory 統合用の拡張クラウド プロキシを設定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。
- Active Directory サーバは、LDAP サーバ接続を使用する必要があります。
- 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ブループリントに関連付けることができるようになりました。このブループリン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。

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、ブループリントを作成でき、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ブループリントを作成します。

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。

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 Cloud Assembly 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。WordPress の使用事例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。WordPress の使用事例：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。WordPress の使用事例：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。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。

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、[ブループリント...] をクリックします。

7 [Cloud Assembly フォーマットでブループリントを作成する] をクリックします。

8 ブループリン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ブループリン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、結果のブループリントにはブループリント コードの `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、ブループリン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ブループリントを見つけて更新しやすくするには、展開の [ブループリント構成] 画面で [ブループリント名] オプションを使用します。自動生成されたブループリント名を記録するか、指定したブループリン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、ブループリントを見つけて開き、`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、付属するブループリン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 [ブループリント] タブをクリックし、ブループリント名をクリックすることにより、ブループリン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 ブループリントと展開を作成します。

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。

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 Cloud Assembly 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。WordPress の使用事例：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。WordPress の使用事例：クラウド アカウントの追加 を参照してください。
- [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。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。

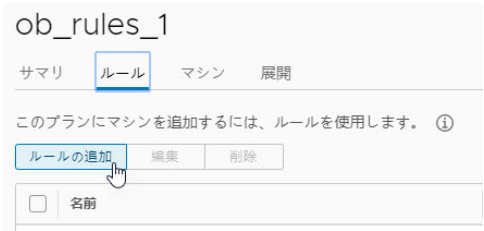
2 [新しいオンボーディング プラン] をクリックして、値を入力します。

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

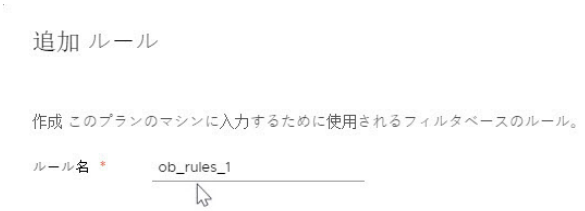
3 [作成] をクリックします。

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。

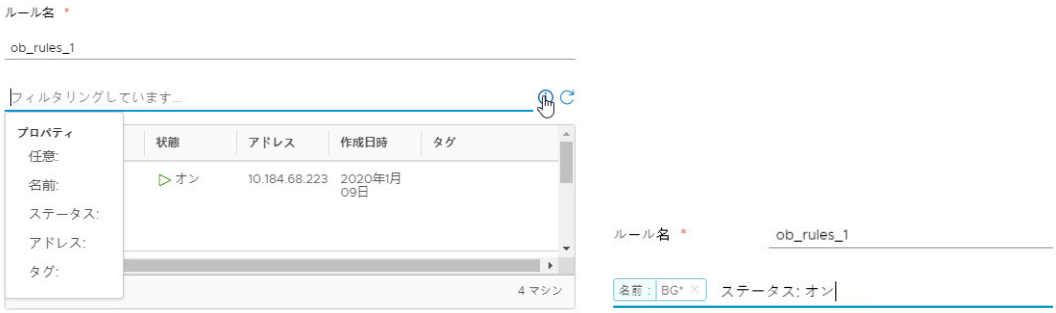


5 ルール名（**ob_rules_1** など）を入力します。



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。

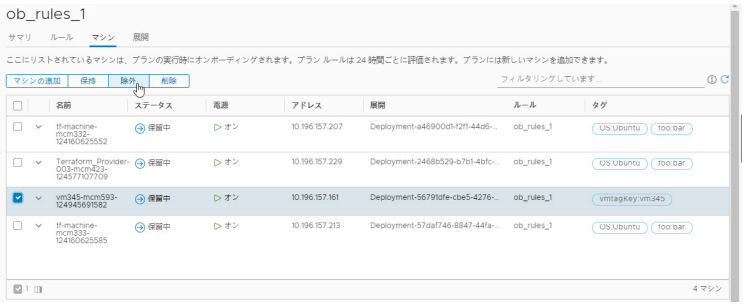


7 [保存] をクリックします。

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。

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。

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。



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る状態です。デフォルトでは、各マシンに対して個別のブループリン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、[ブループリント] > [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ブループリン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、結果のブループリントにはブループリント コードの `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、ブループリン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ブループリントを見つけて更新しやすくするには、展開の [ブループリント構成] 画面で [ブループリント名] オプションを使用します。自動生成されたブループリント名を記録するか、指定したブループリン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、ブループリントを見つけて開き、`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。

プランが実行され、マシンが vRealize Automation Cloud Assembly の管理下に入ります。マシンごとに個別のブループリン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 サーバの設定方法

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 8.0.1 フォワード 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.\nacl proxy-exclude dstdomain
192.168.\nacl proxy-exclude dstdomain 172.16.\nacl proxy-exclude dstdomain kubernetes\nacl
proxy-exclude dstdomain 10.192.204.9\nacl proxy-exclude dstdomain .eng.vmware.com\nacl
proxy-exclude dstdomain 10.192.213.146\nacl proxy-exclude dstdomain
```

```
10.192.213.151\nalways_direct allow proxy-exclude\nhttp_access allow mylan\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 での cloud-init または cloudbase-init を使用した Windows テンプレートの設定方法

Windows 展開環境でカスタム イメージを作成および使用するために、vRealize Automation 環境で cloud-init および cloudbase-init スクリプトがサポートされるように構成することができます。

イメージ マッピングおよびブループリント コードでは、クラウド設定スクリプトがサポートされています。クラウド設定スクリプトは、cloud-init および cloudbase-init のルールと形式に準拠しています。cloud-init および cloudbase-init の関連情報については、https://cloudbase.it/cloudbase-init を参照してください。

注： Windows での cloud-init または cloudbase-init の構成については、次の VMware ブログを参照してください。

- Windows Cloud-Init solution ブログ記事
- Windows guest initialization with Cloudbase-Init in vCenter ブログ記事

Linux 用の cloud-init の構成の関連情報については、ブログ投稿 Building a vRealize Automation Cloud Ready Ubuntu Template for vSphere を参照してください。

vRealize Automation での cloud-init およびクラウド設定スクリプトの使用方法については、次を参照してください。

- vRealize Automation Cloud Assembly でのイメージ マッピングの詳細
- vRealize Automation Cloud Assembly ブループリントでマシンを自動的に初期化する方法

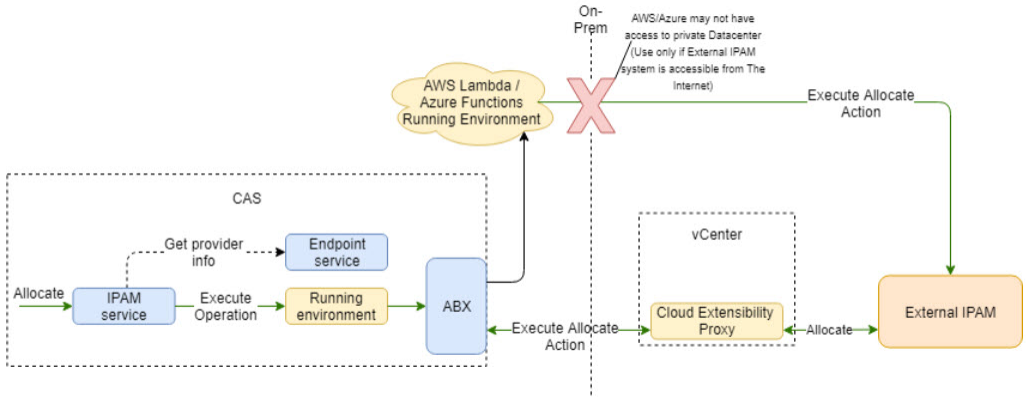
また、次の cloud-Init および cloudbase-init ベンダーのページも参照してください。

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

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 Solutions Exchange のマーケットプレイスから『VMware vRealize Automation Third-Party IPAM SDK』をダウンロードできます。



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 Solution Exchange のマーケットプレイス、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。

プロバイダ固有の外部 IP アドレス管理統合の使用事例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 の使用事例

3

これらの使用事例では、vRealize Automation Cloud Assembly 内にリソース インフラストラクチャを作成し、アプリケーションを設計して、そのインフラストラクチャに展開する例を示します。

使用事例で示される値は、単なる例です。構造と命名規則はそれぞれの環境によって異なります。

この章には、次のトピックが含まれています。

- WordPress の使用事例
- VMware Cloud on AWS の使用事例
- プロバイダ固有の外部 IP アドレス管理統合の使用事例

WordPress の使用事例

このエンドツーエンドの vRealize Automation Cloud Assembly の使用事例では、インフラストラクチャを作成し、そのインフラストラクチャに WordPress サイトを導入する例を示しています。

一連のセットアップを順番に確認して、WordPress サイトを完成させるプロセスを理解します。

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。

手順

1 WordPress の使用事例：インフラストラクチャの作成

   まずクラウド管理者は、エンジニアリング部門が後で WordPress サイトを開発、テストし、本番環境に移行できるようにリソースを構成する必要があります。

2 WordPress の使用事例：プロジェクトの作成

   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。

3 WordPress の使用例：ブループリントの作成および拡張

   開発者は、任意のクラウド ベンダーに導入できる汎用的な vRealize Automation Cloud Assembly ブループリントの形式で WordPress サイトを定義します。

WordPress の使用事例：インフラストラクチャの作成

まずクラウド管理者は、エンジニアリング部門が後で WordPress サイトを開発、テストし、本番環境に移行できるようにリソースを構成する必要があります。

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。

手順

1 WordPress の使用事例：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。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。

2 WordPress の使用事例：クラウド ゾーンの追加

この手順では、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。

3 WordPress の使用事例：フレーバー マッピングの追加

この手順で、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。

4 WordPress 使用事例：イメージ マッピングの追加

この手順では、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。

5 WordPress の使用事例：ネットワーク プロファイルの追加

この手順では、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。

6 WordPress の使用事例：ストレージ プロファイルの追加

この手順では、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。

WordPress の使用事例：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。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	SZXAINXU4UHNAQ1E156S
名前	OurCo-AWS

設定	サンプルの値
説明	WordPress
機能	cloud:aws

すべての値は、使用事例でのみ使用されるサンプルです。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-i8ep2jwp4c9arbfe
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-e52e4nu5fig4pmdd
名前	OurCo-Azure
説明	WordPress
機能	cloud:az

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。

次のステップ

プロジェクトで WordPress サイトを展開するクラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加を参照してください。

WordPress の使用事例：クラウド ゾーンの追加

この手順では、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。

前提条件

クラウド アカウントを追加します。WordPress の使用事例：クラウド アカウントの追加 を参照してください。

手順

1 [インフラストラクチャ] - [設定] - [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、使用事例でのみ使用されるサンプルです。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

次のステップ

フレーバー マッピングを追加して、異なるサイズのマシン展開のアカウントを検討します。WordPress の使用事例：フレーバー マッピングの追加を参照してください。

WordPress の使用事例：フレーバー マッピングの追加

この手順で、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、非公式には「T シャツのサイズ決め」とも呼ばれます。

前提条件

クラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加を参照してください。

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、使用事例でのみ使用されるサンプルです。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
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

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.large
アカウント/リージョン 値	OurCo-Azure/East US Standard_A7

次のステップ

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。WordPress 使用事例：イメージ マッピングの追加を参照してください。

WordPress 使用事例：イメージ マッピングの追加

この手順では、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。

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。

前提条件

クラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加を参照してください。

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-16
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、使用事例でのみ使用されるサンプルです。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。

次のステップ

ネットワークを追加します。WordPress の使用事例：ネットワーク プロファイルの追加を参照してください。

WordPress の使用事例：ネットワーク プロファイルの追加

この手順では、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。

前提条件

クラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加を参照してください。

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
機能タグ	env:dev

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、使用事例でのみ使用されるサンプルです。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 ゾーンのネットワーク 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
機能タグ	env:test

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

ネットワーク プロファイルの設定	値
説明	WordPress
機能タグ	env:prod

次のステップ

ストレージを追加します。WordPress の使用事例：ストレージ プロファイルの追加を参照してください。

WordPress の使用事例：ストレージ プロファイルの追加

この手順では、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。

前提条件

クラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加を参照してください。

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	usage:general

すべての値は、使用事例でのみ使用されるサンプルです。

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

ストレージ プロファイルの設定	サンプルの値
ボリューム タイプ	汎用 SSD
機能タグ	usage: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
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	usage: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。WordPress の使用事例：プロジェクトの作成を参照してください。

WordPress の使用事例：プロジェクトの作成

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

前提条件

クラウド ゾーンを追加します。WordPress の使用事例：クラウド ゾーンの追加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。

ここに記載されているアドレスは、使用事例でのみ使用されるサンプルです。

- 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、WordPress の使用事例：クラウド ゾーンの追加で作成されたゾーンを開きます。

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 WordPress の使用事例：クラウド ゾーンの追加で作成された他のゾーンを確認します。

次のステップ

基本的なブループリントを作成します。

WordPress の使用例：ブループリントの作成および拡張

開発者は、任意のクラウド ベンダーに導入できる汎用的な vRealize Automation Cloud Assembly ブループリントの形式で WordPress サイトを定義します。

使用事例のブループリントは、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 AWS、Azure、または vSphere ベースのクラウドに導入可能なサポート コンポーネントで構成されています。ブループリントは数個のコンポーネントから開始して、既存のコンポーネントを変更し、コンポーネントを追加しながら拡張します。

WordPress の使用事例：インフラストラクチャの作成の例には、クラウド管理者によって設定されたインフラストラクチャが含まれていました。

- 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 16 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：devnets、testnets、prodnets。
- アーカイブ ディスクをサポートするストレージ、開発およびテスト用の一般的なストレージ、本番環境用の高速ストレージ。
- WordPress プロジェクトには、3 つのすべてのクラウド ゾーン環境と、使用事例を試行できるユーザーが含まれています。

前提条件

インフラストラクチャの値について理解しておきます。たとえば、使用事例の例では、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。独自のブループリン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 WordPress の使用事例：基本的なブループリントの作成

開発者は、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 コンポーネントのみを含む vRealize Automation Cloud Assembly ブループリントから開始します。

2 WordPress の使用事例：基本的なブループリントのテスト

開発中は、通常、基幹部分から開始し、vRealize Automation Cloud Assembly ブループリントの拡張に伴って展開およびテストすることにより、ブループリントを構築します。

3 WordPress 使用事例：ブループリントの拡張

基本的な vRealize Automation Cloud Assembly ブループリントを作成してテストした後、それをマルチティア 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。

WordPress の使用事例：基本的なブループリントの作成

開発者は、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 コンポーネントのみを含む vRealize Automation Cloud Assembly ブループリントから開始します。

vRealize Automation Cloud Assembly は、infrastructure-as-code ツールです。開始するには、コンポーネントをデザイン 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。

前提条件

インフラストラクチャについて理解しておきます。ここに示す例では、WordPress の使用事例：インフラストラクチャの作成のインフラストラクチャ値を使用していますが、独自の値に置き換えることができます。

手順

1 [ブループリント] に移動し、[新規] をクリックします。

2 ブループリントに **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 ブループリント デザイン画面の左側にあるコンポーネントから 2 台のクラウドに依存しない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。

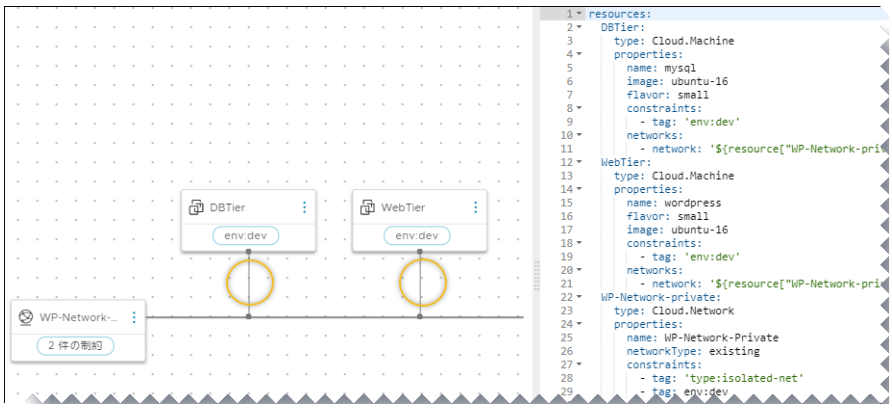
```
resources:
  DBTier:
    type: Cloud.Machine
    properties:
        name: mysql
        image: 'ubuntu-16'
        flavor: 'small'
        constraints: - tag: env:dev
  WebTier:
    type: Cloud.Machine
    properties:
        name: wordpress
        image: 'ubuntu-16'
        flavor: 'small'
  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

6 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
        constraints: - tag: 'type:isolated-net' - tag: 'env:dev'
```

7 マシンをネットワークに接続します。

線がネットワーク ブロックに重なる場所をクリックして、そのままマシン ブロックまでドラッグし、離します。



エディタで、ネットワーク コードが 2 台のマシンに追加されていることを確認します。

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

8 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、使用事例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
- 一般的な使用および高速使用のためのストレージ ディスク速度

ブループリン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、ブループリン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ブループリントを多用せず、さまざまな方法で導入可能なブループリントを 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:
  DBTier:
    type: Cloud.Machine
    properties:
        name: mysql
        image: 'ubuntu-16'
        flavor: '${input.size}'
        constraints:
            -   tag: '${input.env}'
        networks:
            - network: '${resource["WP-Network-Private"].id}'
  WebTier:
    type: Cloud.Machine
    properties:
        name: wordpress
        image: 'ubuntu-16'
        flavor: '${input.size}'
        constraints:
            - tag: '${input.env}'
        networks:
            - network: '${resource["WP-Network-Private"].id}'
  WP-Network-Private:
    type: Cloud.Network
```

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

9 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、データベース ディスク、および導入時の 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
	databaseDiskSize:
	  type: number
	  default: 4
	  maximum: 10
	  title: MySQL Data Disk Size
	  description: Database Disk Size</pre>

DBTier リソース	<pre>DBTier:
	  type: Cloud.Machine
	  properties:
	  name: mysql
	  image: ubuntu-16
	  flavor: '${input.size}'
	  constraints:
	  - tag: '${input.env}'
	  networks:
	  - network: '${resource["WP-Network-Private"].id}'
	  assignPublicIpAddress: true
	  remoteAccess:
	  authentication: usernamePassword
	  username: '${input.username}'
	  password: '${input.userpassword}'
	  cloudConfig: |
	  #cloud-config
	  repo_update: true
	  repo_upgrade: all
	
	  packages:
	  - mysql-server
	
	  runcmd:
	  - sed -e '/bind-address/ s/^#*/#/' -i /etc/mysql/mysql.conf.d/mysqld.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
	  flavor: '${input.size}'</pre>

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

例： 基本的なブループリント コードの完成例

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

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

        packages:
         - mysql-server

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

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

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

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

次のステップ

構文を確認して展開することにより、ブループリントをテストします。

WordPress の使用事例：基本的なブループリントのテスト

開発中は、通常、基幹部分から開始し、vRealize Automation Cloud Assembly ブループリントの拡張に伴って展開およびテストすることにより、ブループリントを構築します。

展開が想定どおりに動作することを確認するために、ブループリン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、その途中でコンポーネントの追加、再テスト、再展開を行います。

前提条件

基本的なブループリントを作成します。WordPress の使用事例 : 基本的なブループリントの作成を参照してください。

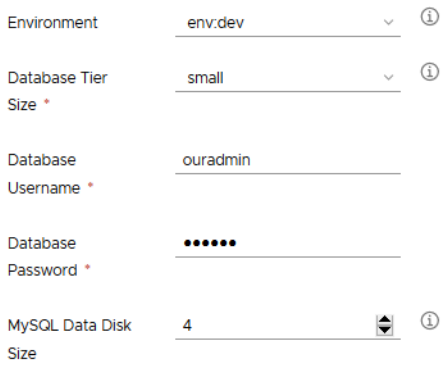
手順

1 [ブループリント] をクリックし、WordPress-BP ブループリントを開きます。

基本的なブループリントがデザイン キャンバスとコード エディタに表示されます。

2 ブループリントの構文、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。



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。シミュレーションでは、ブループリン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。

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、エラーが発生していないかを確認できます。

シミュレーションで正常に実行されても、ブループリントをエラーなしで展開できるとは限りません。

4 ブループリン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。

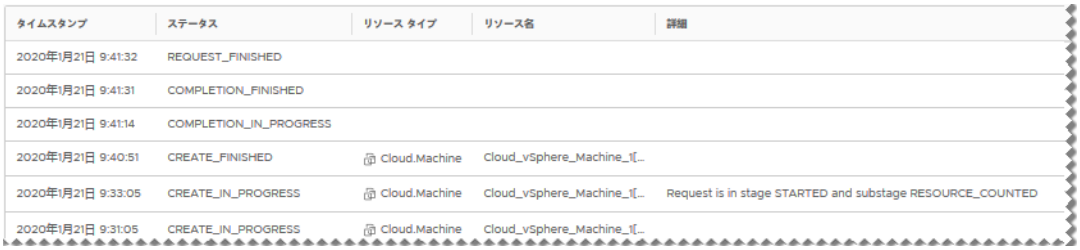
5 [新規展開の作成] を選択します。

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 ブループリントが正常に展開されたことを確認するには、[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。



タイムスタンプ	ステータス	リソース タイプ	リソース名	詳細
2020年1月21日 9:41:32	REQUEST_FINISHED			
2020年1月21日 9:41:31	COMPLETION_FINISHED			
2020年1月21日 9:41:14	COMPLETION_IN_PROGRESS			
2020年1月21日 9:40:51	CREATE_FINISHED	Cloud.Machine	Cloud_vSphere_Machine_1[...	
2020年1月21日 9:33:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[...	Request is in stage STARTED and substage RESOURCE_COUNTED
2020年1月21日 9:31:05	CREATE_IN_PROGRESS	Cloud.Machine	Cloud_vSphere_Machine_1[...	

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] - [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。

この使用事例では、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な場合は、ブループリントを変更して、[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 ブループリン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 ブループリント 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。同じブループリントを使用して、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

コンポーネントを追加して、ブループリントを本番環境に適したアプリケーションに拡張します。

WordPress 使用事例：ブループリントの拡張

基本的な vRealize Automation Cloud Assembly ブループリントを作成してテストした後、それをマルチティア 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。

ブループリントを拡張するには、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

基本的なブループリントを作成してテストします。WordPress の使用事例 : 基本的なブループリントの作成および WordPress の使用事例：基本的なブループリントのテストを参照してください。

手順

1 [ブループリント] をクリックし、WordPress-BP ブループリントを開きます。

基本的なブループリントがデザイン 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。

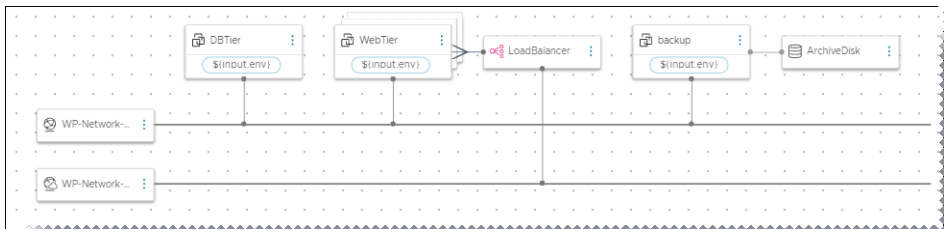
g ロード バランサをパブリック ネットワークに接続します。

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 ストレージ ディスク速度用に `archiveusage` 入力プロンプトを追加します。

k ストレージ ディスク サイズ用に `archiveDiskSize` 入力プロンプトを追加します。



3 基本的なブループリン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ブループリン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： ブループリントの拡張を終えたコードの例

```
inputs:
  env:
    type: string
    enum:
```

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

count: type: integer default: 2 maximum: 5 minimum: 2 title: WordPress Cluster Size
description: WordPress Cluster Size (Number of Nodes) archiveDiskSize: type: number default:
4 maximum: 10 title: WordPress Archive Disk Size description: Archive Storage Disk Speed
archiveusage: type: string enum: - 'usage:general' - 'usage:fast' description: Archive
Storage Disk Speed title: Archive Disk Speed
resources:
  DBTier:
    type: Cloud.Machine
    properties:
      name: mysql
      image: ubuntu-16
      flavor: '${input.size}'
      constraints:
        - tag: '${input.env}'
      networks:
        - network: '${resource["WP-Network-Private"].id}'
          assignPublicIpAddress: true
      remoteAccess:
        authentication: usernamePassword
        username: '${input.username}'
        password: '${input.userpassword}'
```

```
      cloudConfig: |
        #cloud-config
        repo_update: true
        repo_upgrade: all

        packages:
         - mysql-server

        runcmd:
         - sed -e '/bind-address/ s/^#*/#/' -i /etc/mysql/mysql.conf.d/mysqld.cnf
         - service mysql restart
         - mysql -e "GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'mysqlpassword';"
         - mysql -e "FLUSH PRIVILEGES;"
      attachedDisks: []
  WebTier:
    type: Cloud.Machine
    properties:
      name: wordpress
      flavor: '${input.size}'
      image: 'ubuntu-16'
      count: '${input.count}'
      constraints:
        - tag: '${input.env}'
      networks:
        - network: '${resource["WP-Network-Private"].id}'
          assignPublicIpAddress: true
      storage: disks: - capacityGb: '${input.archiveDiskSize}' name: ArchiveDisk
      cloudConfig: |
        #cloud-config
        repo_update: true
        repo_upgrade: all

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

        runcmd:
         - mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
         - i=0; while [ $i -le 10 ]; do mysql --connect-timeout=3 -h $
{DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
         - mysql -u root -pmysqlpassword -h ${DBTier.networks[0].address} -e "create database
wordpress_blog;"
         - mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
         - sed -i -e s/"define( 'DB_NAME', 'database_name_here' );"/"define( 'DB_NAME',
'wordpress_blog' );"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define( 'DB_USER', 'username_here' );"/"define( 'DB_USER', 'root' );"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define( 'DB_PASSWORD',
```

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

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

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

次のステップ

独自のインフラストラクチャを定義し、独自のブループリントを作成します。

4 章 vRealize Automation Cloud Assembly リソース インフラストラクチャの構築および 6 章 vRealize Automation Cloud Assembly 展開の設計を参照してください。

VMware Cloud on AWS の使用事例

この vRealize Automation Cloud Assembly の使用事例では、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ブループリン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 Cloud Assembly での基本的な VMware Cloud on AWS ワークフローの構成

この vRealize Automation Cloud Assembly 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 環境へのデプロイ用のブループリントを定義するプロセスを示します。

2 vRealize Automation Cloud Assembly 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 Cloud Assembly の VMware Cloud on AWS 展開に追加します。

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

この vRealize Automation Cloud Assembly 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 環境へのデプロイ用のブループリン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 環境内のリソースにブループリント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設定するには、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 Cloud Assembly 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 Cloud Assembly で VMware Cloud on AWS を使用するときに 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly プロジェクトを定義します。

6 vRealize Automation Cloud Assembly で VMware Cloud on AWS 展開をサポートするためのブループリント 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、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 3-1. 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 Cloud Assembly 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 プロキシを設定します。vRealize Automation のインターネット プロキシ サーバの設定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	1 [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。 2 [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。 3 ***myinitials_mytoken*** などの新しいトークン名を入力します。 4 [トークン TTL] を [無期限] に設定します。 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。 5 [範囲の定義] セクションで、[すべてのロール] を選択します。  6 [生成] をクリックします。 7 [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。 8 [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。 	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。 [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。 ■ [組織のロール] ■ [組織のメンバー] ■ [組織の所有者] ■ [サービス ロール - VMware Cloud on AWS] ■ [管理者] ■ [NSX Cloud 管理者] ■ [NSX Cloud 監査者] **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。 生成されたトークンまたは指定されているトークンを適用して、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。 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。 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。
SDDC 名	この例では、[Datacenter:Datacenter-abz] を選択します。 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。	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。 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。

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

vRealize Automation Cloud Assembly 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 Cloud Assembly で VMware Cloud on AWS を使用するときに 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 ブループリントでマシン、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 Cloud Assembly 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。

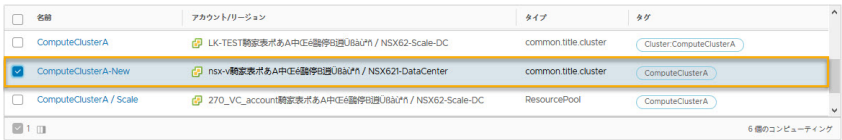
4 下のエリア 1 に示すように、CloudAdmin ユーザーが使用できるコンピューティング リソースを検索して選択します。この例では、`Cluster 1/ Compute-ResourcePool` という名前のリソースを使用します。

`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。

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。



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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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、ブループリン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構築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 Cloud Assembly 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 Cloud Assembly 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 **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。
名前	vmc-network1
説明	VMware Cloud on AWS CloudAdmin 認証情報を持つブループリント管理者がアクセスできるネットワークが含まれます。

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

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 プロファイルを設定します。

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 認証情報を持つブループリン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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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] - [設定] - [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 Cloud Assembly 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 環境に展開するブループリントを作成します。vRealize Automation Cloud Assembly で VMware Cloud on AWS 展開をサポートするためのブループリント 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 Cloud Assembly で VMware Cloud on AWS 展開をサポートするためのブループリント 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、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ブループリント 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、ブループリント 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 Cloud Assembly 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設定します。

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）ブループリント 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、ブループリント デザイン コードに必ず `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、上記のようにブループリント デザイン 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、ブループリント デザイン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。
- ブループリント 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必要がない場合をのぞき、ブループリント デザインで `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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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、vRealize Automation Cloud Assembly ブループリン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、VMware Cloud on AWS の使用事例 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 Cloud Assembly 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 Cloud Assembly で VMware Cloud on AWS のネットワークの隔離をサポートするようにブループリントのネットワーク コンポーネントを定義する

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 Cloud Assembly のブループリント キャンバスにドラッグし、隔離される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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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設定します。その後、ブループリントからネットワークにアクセスし、このブループリントを 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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したネットワーク プロファイル（`vmc-network1` など）を開きます。vRealize Automation Cloud Assembly 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

ブループリントでネットワーク コンポーネントを設定します。vRealize Automation Cloud Assembly で VMware Cloud on AWS のネットワークの隔離をサポートするようにブループリントのネットワーク コンポーネントを定義するを参照してください。

vRealize Automation Cloud Assembly 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 Cloud Assembly 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設定します。その後、ブループリントでネットワークを指定し、そのブループリントを 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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したネットワーク プロファイル（`vmc-network1` など）を開きます。vRealize Automation Cloud Assembly 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 Cloud Assembly 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

ブループリントでネットワーク コンポーネントを設定します。vRealize Automation Cloud Assembly で VMware Cloud on AWS のネットワークの隔離をサポートするようにブループリントのネットワーク コンポーネントを定義するを参照してください。

vRealize Automation Cloud Assembly で VMware Cloud on AWS のネットワークの隔離をサポートするようにブループリントのネットワーク コンポーネントを定義する

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 Cloud Assembly のブループリント キャンバスにドラッグし、隔離される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。

以前に作成したブループリントにネットワーク隔離を追加します。ブループリントは、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 Cloud Assembly 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 Cloud Assembly 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、ブループリント 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 Cloud Assembly 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ブループリントを開きます。vRealize Automation Cloud Assembly で VMware Cloud on AWS 展開をサポートするためのブループリント 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 ブループリント デザイン画面の左側にあるコンポーネントから、ネットワーク コンポーネントをキャンバスにドラッグします。

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

ブループリントを展開できる状態にするか、閉じます。

プロバイダ固有の外部 IP アドレス管理統合の使用事例

外部 IP アドレス管理プロバイダを使用して、ブループリント展開の IP アドレス割り当てを管理できます。この手順の例では、vRealize Automation Cloud Assembly で外部 IP アドレス管理統合を設定する方法と、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するブループリントを展開する方法を説明します。この例では、外部 IP アドレス管理プロバイダとして Infoblox を使用して IP アドレス管理統合を設定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ブループリントを作成し、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。

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 Cloud Assembly 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 Cloud Assembly 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 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 Marketplace からダウンロードできます。

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 での外部 IP アドレス管理統合ポイントの追加

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。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、ブループリント展開の 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。

5 vRealize Automation Cloud Assembly で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 IP アドレス管理プロバイダの値を使用する

ネットワークを定義して、IP アドレスの値を内部の vRealize Automation Cloud Assembly からではなく、外部 IP アドレス管理プロバイダで管理されている値を取得して使用するようにできます。

6 IP アドレス管理プロバイダ範囲の割り当てを使用する vRealize Automation Cloud Assembly ブループリントの定義と展開

ブループリントを定義して、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。

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

Infloblox 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、Infloblox 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 アプライアンス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 Infloblox 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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 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 Marketplace からダウンロードできます。

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。

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、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 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

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、マーケットプレイスに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 Cloud Assembly 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 Cloud Assembly 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 Infobox 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。

手順

1. VMware Solution Exchange のマーケットプレイスの vRA Cloud Infoblox Plug-in バージョン 0.1 (vRealize Automation 8.0) または vRA Cloud Infoblox Plug-in バージョン 1.0 (vRealize Automation 8.0.1) 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 での外部 IP アドレス管理統合ポイントの追加 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 Cloud Assembly 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 Cloud Assembly 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 での外部 IP アドレス管理統合ポイントの追加

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。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、ブループリント展開の 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が、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 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 Cloud Assembly 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 Cloud Assembly 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 アプリケーションで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 サーバの設定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 [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 Cloud Assembly 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 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]（*Infloblox_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 Cloud Assembly で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 IP アドレス管理プロバイダの値を使用する

ネットワークを定義して、IP アドレスの値を内部の vRealize Automation Cloud Assembly からではなく、外部 IP アドレス管理プロバイダで管理されている値を取得して使用するようにできます。

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、vCenterServer 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 Cloud Assembly 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、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 Cloud Assembly ネットワーク プロファイルを追加する方法および vRealize Automation Cloud Assembly 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。

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。プロバイダ固有の外部 IP アドレス管理統合の使用事例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 Cloud Assembly 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 での外部 IP アドレス管理統合ポイントの追加 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 vCenterServer から vRealize Automation Cloud Assembly 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 *Infloblox_ 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 *Infloblox_ 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 Cloud Assembly 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 Cloud Assembly 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。

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。

    機能タグはブループリントのプロビジョニングの関連付けの際にブループリン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

これにより、ネットワークおよびネットワーク プロファイルの設定が外部 IP アドレス管理統合をサポートするようになったため、ブループリント内で使用できるようになりました。

IP アドレス管理プロバイダ範囲の割り当てを使用する vRealize Automation Cloud Assembly ブループリントの定義と展開

ブループリントを定義して、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。

外部 IP アドレス管理の統合ワークフローの最後の手順では、ブループリン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。

ブループリントを展開して仮想マシンを起動すると、[リソース] - [ネットワーク] ページで、展開内の各仮想マシンに使用される IP アドレスは、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。プロバイダ固有の外部 IP アドレス管理統合の使用事例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 Cloud Assembly 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 での外部 IP アドレス管理統合ポイントの追加 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 Cloud Assembly で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 IP アドレス管理プロバイダの値を使用する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、ブループリン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、WordPress の使用事例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 [ブループリント] - [新規作成] の順にクリックし、[新規ブループリント] ページ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ブループリント
   - [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 コンポーネントをブループリント 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 ブループリント 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 ブループリント 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。

  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。*static* の割り当て設定は必須ですが、*address* 値は必須ではありません。特定のアドレスで外部 IP アドレスの選択を開始することは可能ですが、必須ではありません。*address* の値を指定しない場合、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 ブループリント 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}'
```

注： 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。

6 ブループリント ページで [展開] をクリックし *Infoblox-1* という名前を指定し、[展開の種類] ページの [展開] をクリックします。

7 ブループリントの展開中に、[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、*Infloblox_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 Infloblox 固有のプロパティと拡張属性の使用

Infloblox 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、Infloblox 固有のプロパティを使用できます。

Infloblox の IP アドレス管理統合では、次の Infloblox プロパティを使用できます。これらを vRealize Automation で使用することにより、ブループリントの展開時に 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。

vRealize Automation 8.0 用の vRA Cloud Infoblox Plug-in バージョン 0.1 パッケージでは次の Infloblox プロパティをすべて使用できますが、`Infoblox.IPAM.Network.dnsView` プロパティは vRealize Automation 8.0.1 用の vRA Cloud Infoblox Plug-in バージョン 1.0 パッケージおよび vRealize Automation 8.1 用の vRA Cloud Infoblox Plug-in バージョン 1.1 パッケージのみで使用できます。

注： これらのプロパティの使用方法は、プロバイダ固有の外部 IP アドレス管理統合の使用事例のサンプル ワークフローには含まれていません。

- `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：False。

- `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。デフォルト値：デフォルト。

- `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：True。

- `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：なし（空の文字列）。

`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] - [構成] - [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。
- ブループリントで各マシン コンポーネントにプロパティを指定できます。`Infoblox.IPAM.Network.dnsView` プロパティの使用方法を示すサンプル ブループリント 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。

vRealize Automation Cloud Assembly リソース インフラストラクチャの構築

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、ブループリン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 Cloud Assembly で一般的なマシン サイズを作成するための vRealize Automation Cloud Assembly フレーバー マッピングを追加する方法
- 一般的なオペレーティング システムを作成するための vRealize Automation Cloud Assembly イメージ マッピングを追加する方法
- vRealize Automation Cloud Assembly 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 Cloud Assembly でリソースを操作する方法

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、ブループリン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 タイプ内のセクションです。クラウド ゾーンは、プロジェクトに固有です。

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

  配置ポリシーによって、指定された領域内でホストの選択を行います。

  - default - コンピューティング リソースをホスト上にランダムに配置します。
  - 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 - コンピューティング リソースをホスト間で均等に配置します。

- 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

  このクラウド ゾーンにプロビジョニングできるコンピューティングを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）。

  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用さ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 Cloud Assembly で一般的なマシン サイズを作成するための vRealize Automation Cloud Assembly フレーバー マッピングを追加する方法

vRealize Automation Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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）を含めることできます。ブループリン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。

ブループリン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 リソースを含むブループリント内のイメージ 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、ブループリントで 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 マッピングはブループリントで vSphere 固有の構成の制限として機能します。

基本的なフレーバー マッピングの例については、WordPress の使用事例：フレーバー マッピングの追加を参照してください。

一般的なオペレーティング システムを作成するための vRealize Automation Cloud Assembly イメージ マッピングを追加する方法

vRealize Automation Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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。ブループリントをビルドして最終的に展開し、繰り返し使用する場合は、ニーズに最も適した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。

基本のイメージ マッピングを定義する方法の例については、WordPress 使用事例：イメージ マッピングの追加を参照してください。

ブループリン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。

ブループリン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。

ブループリント 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 ドキュメントの『vSphere の仮想マシン管理』にあるコンテンツ 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。

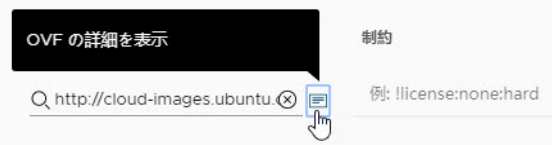
3 アクションが完了したら、[インフラストラクチャ] - [構成] - [イメージ マッピング] をクリックします。イメージ マッピングを新しく定義するか、既存のものを編集して、手順 1 のクラウド アカウント/リージョンを選択します。

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。

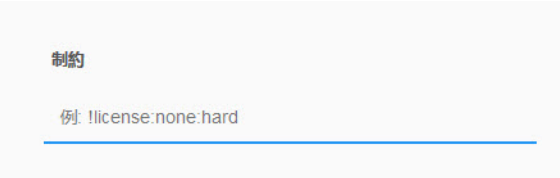
OVF 詳細の表示

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 ブループリント 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

ブループリン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、ブループリントに `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` タグがブループリン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。



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 マップとブループリン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、ブループリントをパブリック 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 ブループリント 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 マッピングとブループリン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ブループリン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、次にブループリント 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ブループリン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、ブループリント 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、ブループリント 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は、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実行され、イメージ マッピング スクリプトの終了後にブループリント 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

vRealize Automation Cloud Assembly の Windows 展開で cloud-init を設定するには、vRealize Automation での cloud-init または cloudbase-init を使用した Windows テンプレートの設定方法を参照してください。

クラウド設定スクリプトの使用に関する詳細については、vRealize Automation Cloud Assembly ブループリントでマシンを自動的に初期化する方法および VMware ブロガーの記事 Customizing Cloud Assembly Deployments with Cloud-Init を参照してください。

vRealize Automation Cloud Assembly ネットワーク プロファイルを追加する方法

vRealize Automation Cloud Assembly 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、ネットワークを内部専用にするのではなくインターネットに接続する必要があるなどです。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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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ブループリン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、ブループリントのタグと照合され、ブループリン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 プロファイルに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ブループリン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、ブループリン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。

- [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。ドメインは、vSphere の構成中に vCenter Server 管理者によって構成されます。ドメインによって、vCenter Server のローカル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、およびブループリント内の 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、およびブループリント内の 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。ブループリン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。ブループリントの展開時に、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。ブループリントの展開時は、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。タグの照合がブループリントの展開で使用可能などのネットワークに影響を与えるか。

  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。

  ブループリントを展開すると、ブループリントのネットワーク 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 プロファイルに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ブループリントの展開時に適用されます。

ネットワーク ポリシー

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。

ネットワーク タイプの詳細については、vRealize Automation Cloud Assembly でのネットワークおよびネットワーク プロファイルの使用を参照してください。

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ブループリントは、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。

  ブループリントのネットワーク タイプが `outbound` または `private` であると、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、ブループリン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作成するオンデマンド 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。

  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。

- [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。

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 プロファイルのロード バランサのいずれかのタグが、ブループリントのロード バランサ コンポーネントで使用されているタグと一致する場合、ロード バランサは展開時に考慮されます。ブループリントが展開されると、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 プロファイルおよびブループリント デザインでのロード バランサ設定の使用および vRealize Automation Cloud Assembly ブループリントのネットワーク、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

ブループリントが展開されると、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。

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。ブループリン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、ブループリント `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、ブループリントで使用することはできません。詳細については、セキュリティ リソースおよび vRealize Automation Cloud Assembly ブループリントのネットワーク、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、ブループリント、およびタグに関する詳細情報

ネットワーク プロファイルの詳細については、ヘルプのこのセクションにある他のトピックおよび WordPress の使用事例：ネットワーク プロファイルの追加を参照してください。

ネットワークの詳細については、ネットワーク リソースを参照してください。

ブループリントのサンプル 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 Cloud Assembly ブループリントのネットワーク、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方法の詳細については、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 Cloud Assembly 内のネットワークおよびネットワーク プロファイルでの IP アドレスの使用

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 Cloud Assembly ブループリン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。

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。詳細については、vRealize Automation Cloud Assembly 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。

vRealize Automation Cloud Assembly ネットワークでの IPv4 および IPv6 のサポート

vRealize Automation Cloud Assembly 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 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での使用で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 アドレス管理のサポートに加え、外部 IP アドレス管理プロバイダを使用して、ブループリントおよび展開内のネットワークの 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。

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、サードパーティの 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 アドレス管理統合ポイントの構成 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、プロバイダ固有の外部 IP アドレス管理統合の使用事例を参照してください。

vRealize Automation Cloud Assembly でのネットワークおよびネットワーク プロファイルの使用

vRealize Automation Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly 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 タイプ

ブループリン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
`existing`	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。
`public`	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。
`private`	オンデマンド ネットワーク タイプ。 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。
`outbound`	オンデマンド ネットワーク タイプ。 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。
`routed`	オンデマンド ネットワーク タイプ。 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。 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。 `routed` ネットワークは、`Cloud.NSX.Network` ネットワーク コンポーネントのブループリント仕様でのみ使用できます。

ネットワーク コンポーネント データを含む、設定済みのブループリントの例については、vRealize Automation Cloud Assembly ブループリントのネットワーク、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ブループリントを展開すると、以下の動作が予想されます。

表 4-1. ネットワーク シナリオ

ネットワークのタイプまたはシナリオ	クラウド ゾ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 ゾ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	ブループリン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。 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。 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。	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。 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。 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。
既存のネットワーク	ブループリントのネットワーク 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。ブループリントのネットワーク 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。 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。 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。 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。	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。 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。 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。 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。
パブリック ネットワーク	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。 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。 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。 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。	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。 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。 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。
プライベート ネットワーク	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。	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。

表 4-1. ネットワーク シナリオ （続き）

ネットワークのタイプまたはシナリオ	クラウド ゾ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 ゾ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に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。 WordPress の使用事例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。 WordPress の使用事例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。 WordPress の使用事例を参照してください。

vRealize Automation Cloud Assembly のネットワーク プロファイルおよびブループリント デザインでのロード バランサ設定の使用

NSX-V または NSX-T ネットワーク プロファイルの設定に基づいて、vRealize Automation Cloud Assembly ブループリントにロード バランサを設定できます。

ロード バランサのオプションは、ブループリントのネットワーク プロファイルの設定と、NSX-T または NSX-V ソース アプリケーションの設定によって異なります。

NSX-T ネットワークとロード バランサのオプション

NSX-T は、Tier-1 論理ルーターごとにロード バランサ サービスを 1 つサポートしています。ロード バランサのオプションは、ブループリント内のロード バランサ コンポーネント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。

- 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送信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

  オンデマンド ネットワーク VIP は、ソース外部ネットワークの VIP に関連付けられている必要があります。

  ロード バランサ コンピューティングがオンデマンド プライベート 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、VIP が外部ネットワーク上にあれば有効です。

- 既存のネットワーク

  ロード バランサは、既存のネットワークに接続されている場合、そのネットワークの 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 サービスがない場合にのみ、新しい小さなロード バランサ サービスが作成されます。ロード バランサ サービスがすでにある場合は、そのサービスに新しい仮想サーバが接続されます。既存のネットワークが 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。

- 新しいセキュリティ グループがあるオンデマンド ネットワーク

  マシンが既存のネットワークに接続され、セキュリティ グループで隔離が確立されているため、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。

オンデマンド ネットワーク用に作成されたロード バランサ サービスは、使用されなくなると破棄されます。ロード バランサが破棄されると、モニタ、プール、アプリケーション プロファイル、および VIP が破棄されます。外部ネットワーク用に作成されたロード バランサ サービスは破棄されません。

ブループリントに NSX-T ロード バランサ設定を指定するには、ブループリント コンポーネント パネルに用意されているクラウドに依存しないロード バランサ コンポーネントを使用できます。

NSX-T ネットワーク展開でロード バランサを使用する方法の詳細については、VMware ブログの vRA Cloud Assembly Load Balancer with NSX-T Deep Dive を参照してください。

NSX-V ネットワークとロード バランサのオプション

新しいネットワーク サブネットまたはセキュリティ グループを使用して隔離されたネットワークを作成したかどうかに基づいて、ロード バランサを 2 アームまたは 1 アームとして設定できます。

ブループリントに NSX-V ロード バランサ設定を指定するには、ブループリント コンポーネント パネルの NSX に用意されているロード バランサ コンポーネントを使用できます。

2 アームのロード バランサを作成するためのワークフローは次のとおりです。

1 サービス エッジを作成します。

2 サービス エッジのアップリンク インターフェイスをパブリック ネットワークに接続します。

3 ダウンリンク インターフェイスを隔離された（送信）ネットワークに接続します。

4 ネットワーク プロファイルの固定 IP アドレス範囲からロード バランサの固定 IP アドレスを割り当てます。

5 ロード バランサを構成します。

6 ファイアウォールを構成します。

7 デフォルト ゲートウェイを構成します。

1 アームのロード バランサを作成するためのワークフローは次のとおりです。

1 サービス エッジを作成します。

2 予約から選択したネットワークをアップリンクとして接続します。

3 ネットワーク プロファイルの固定 IP アドレス範囲からロード バランサの固定 IP アドレスを割り当てます。

4 ロード バランサを構成します。

5 ファイアウォールを構成します。

6 デフォルト ゲートウェイを構成します。

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 Cloud Assembly 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、ストレージ リソース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

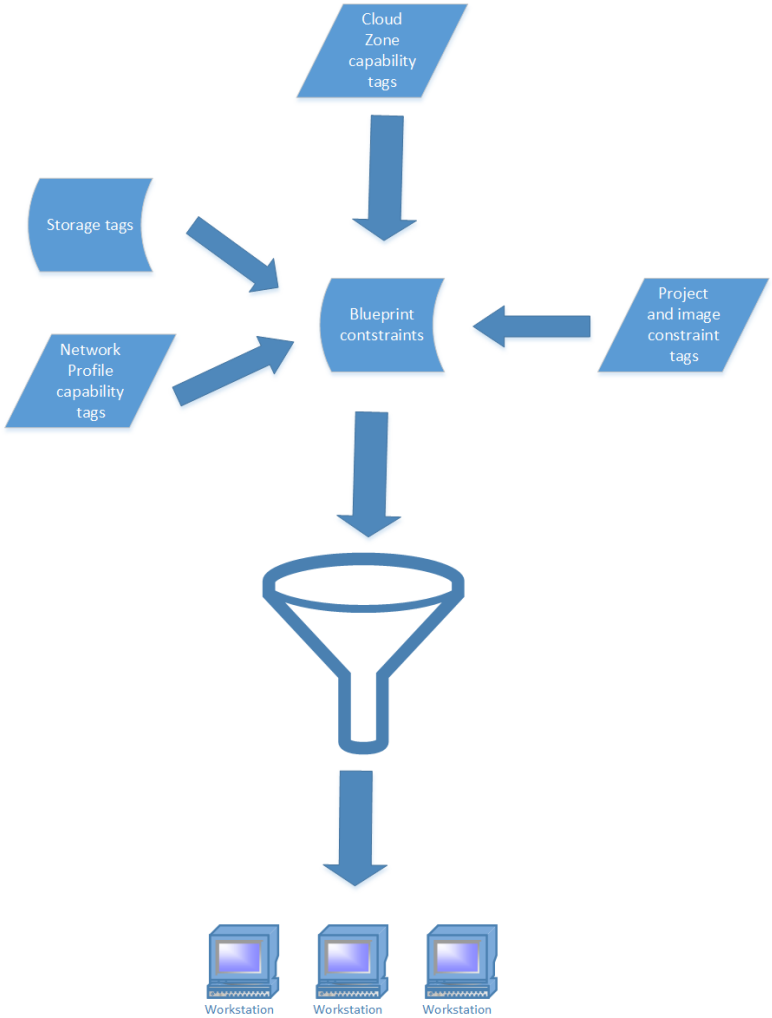
タグは、構造的には `name:value` ペアの規則に準拠する必要がありますが、それを除けばほぼ自由に形成できます。vRealize Automation Cloud Assembly 全体で、いずれのタグも同じように表示され、タグの機能はコンテキストによって決まります。

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

vRealize Automation Cloud Assembly 内のタグの主な機能は、機能と制約を使用して展開を設定することです。クラウド ゾーンに配置された機能タグ、ネットワーク プロファイル、ストレージ プロファイル、および個別のインフラストラクチャ リソースでは、展開に必要な機能を定義します。クラウド管理者は、プロジェクトに制約タグを配置して、そのプロジェクトを制御できるようになります。この制約タグは、ブループリン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 はブループリン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、展開設定を定義します。このタグベースの機能と制約機能は、vRealize Automation Cloud Assembly で展開設定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。

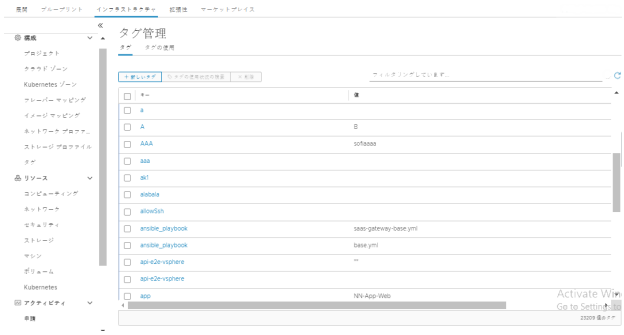
また、[タグの管理] 画面およびリソース設定画面には、タグ名でアイテムを検索できる検索機能があります。タグ名で検索して特定する機能を使用するには、これらのアイテムに論理的で人間が判読可能なタグを使用することが重要です。

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、タグ付けストラテジは組織のニーズ、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 ストラテジ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プランを設計して実装し、該当するすべてのユーザーにこのプランを通達します。プランは、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お勧めします。

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、一致するブループリント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、ブループリント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含むブループリントを設計できます。

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な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用に配置ロジックを定義できます。これらのタグは、配置をハード コーディングするための強力で簡潔なオプションです。

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] ページを使用して、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。タグ戦略の全体と同様に、機能タグは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 ゾーンおよびブループリントの制約と機能タグを照合します。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ブループリン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、Wordpress の例の「クラウドゾーンの追加]」トピックでは、OurCo-AWS-US-East および OurCo AWS-US-West ゾーンに dev タグと test タグを作成しました。これは、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。これらの機能タグは、適切な制約タグと組み合わせて使用することで、目的の環境に展開することができます。

vRealize Automation Cloud Assembly での制約タグの使用

適切な展開を生成できるようにリソース、クラウド ゾーン、およびプロファイルに定義されている機能を一致させるには、ブループリントをはじめ vRealize Automation Cloud Assembly 内のさまざまなコンポーネントに制約タグを追加します。

vRealize Automation Cloud Assembly には、制約タグを適用できる主要な領域が 2 つあります。1 つは、プロジェクトとイメージの設定側にあります。もう 1 つは、ブループリントの展開側にあります。どちらの領域でも、適用した制約がブループリントにマージされて、一連の展開要件となります。

プロジェクトでの制約タグの仕組み

クラウド管理者は、Cloud Assembly を設定するときに、プロジェクトおよびイメージ マップに制約タグを適用できます。このように、クラウド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のために申請されたすべてのブループリントに適用されます。

プロジェクトのタグがブループリントの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、一方で開発者がブループリントに `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。

ブループリントでの制約タグの仕組み

ブループリントでは、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、ブループリント YAML コードで `tag` ラベルを使用します。ブループリン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 ...
```

ブループリントの操作方法の詳細については、WordPress の使用例：ブループリントの作成および拡張を参照してください。

プロジェクトおよびブループリントでの強い制約および弱い制約の仕組み

プロジェクトとブループリントのどちらでも、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、使用可能である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、各制約が強いか弱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。

注： プロジェクトおよびブループリントでは、`failOnConstraintMergeConflict` フラグによって制約の動作が変更されます。このフラグが true に設定されている場合、プロジェクト制約とブループリント制約の間に競合があると、申請は失敗します。フラグがないか false に設定されている場合、ブループリント制約よりもプロジェクト制約が優先されます。

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 4-2. 標準タグ

説明	タグ
組織	org: `orgID`
プロジェクト	project: `projectID`
申請者	requester: `username`
展開	deployment: `deploymentID`
ブループリント参照（該当する場合）	blueprint: `blueprintID`
ブループリントのコンポーネント名	blueprintResourceName: `CloudMachine_1`
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）	constraints: `key:value:soft`
クラウド アカウント	cloudAccount: `accountID`
ゾーンまたはプロファイル（該当する場合）	zone: `zoneID`、networkProfile: `profileID`、storageProfile: `profileID`

vRealize Automation Cloud Assembly によるタグの処理方法

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、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、操作の概要と、一連の機能タグおよび制約タグの処理の順番を示します。

- 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ハード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、一致するソフト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、ブループリントに示されているハード制約およびソフト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、WordPress の使用事例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 の制約があるブループリントからは、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。

クラウド ゾーンまたはネットワーク プロファイルに対するタグの追加については、WordPress の使用事例：クラウド ゾーンの追加 WordPress の使用事例：ネットワーク プロファイルの追加を参照してください。

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、WordPress の使用事例：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 Cloud Assembly 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 Cloud Assembly 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 Cloud Assembly ブループリントが展開される場所を制御できます。

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、リソースに直接適用するタグを選択し、vRealize Automation Cloud Assembly 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。

ネットワーク リソース

vRealize Automation Cloud Assembly では、クラウド管理者は、プロジェクトのクラウド アカウントと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情報とセキュリティ情報が検出され、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。

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 Cloud Assembly 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。また、新しい IP アドレス範囲を定義できるほか、使用可能なネットワーク内で既存の 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、適切なネットワークをブループリントのネットワーク 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 プロファイルとも照合できます。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 タグがブループリントの他のコンポーネントと一致するためには、まずブループリントが 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 アドレス範囲を選択して、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。

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外部 IP アドレス管理統合を使用している場合は、IP アドレス管理の IP アドレス範囲を追加できます。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 Cloud Assembly で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 IP アドレス管理プロバイダの値を使用する に記述されています。

IP アドレス

組織で使用される定義済みの IP アドレスのステータス（使用可能または割り当て済みなど）を表示します。

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表示します。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。

セキュリティ リソース

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情報とセキュリティ情報が検出され、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できるようになります。

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

vRealize Automation Cloud Assembly で作成されたオンデマンド セキュリティ グループと、データ収集によって公開される NSX-T や Amazon Web Services などのソース アプリケーション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。また、作成したオンデマンド セキュリティ グループを編集することもできます。クラウド管理者は、セキュリティ グループを開いて編集できます。ブループリントの作成者は、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、ブループリント展開のネットワーク ルールやセキュリティのその他の側面を制御できます。

セキュリティ グループ コンポーネントのブループリント コードでファイアウォール ルールを使用する場合、

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 グループは、vRealize Automation Cloud Assembly によってデータ収集されます。既存のセキュリティ グループが表示され、[発生元] 列で `Discovered` と分類されます。vRealize Automation Cloud Assembly でブループリントまたはネットワーク プロファイル内に作成するオンデマンド セキュリティ グループは、[発生元] 列で `Managed by Cloud Assembly` として表示および分類されます。組織内で使用可能なセキュリティ グループのみが表示されます。

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。

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、ブループリントで使用できるようにすることができます。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、ブループリントで使用することはできません。ブループリントの作成者は、ブループリント内の `Cloud.SecurityGroup` コンポーネントを使用して、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。ブループリントでは、`Cloud.SecurityGroup` セキュリティ グループ コンポーネントをマシン NIC に適用して、ネットワーク ルールなど、ブループリントの展開のセキュリティのさまざまな側面を制御する必要があります。

管理者が既存のセキュリティ グループにタグを追加する方法、またはソース アプリケーションのセキュリティ グループからタグを収集する方法については、vRealize Automation Cloud Assembly 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。

ブループリント作成者がブループリント セキュリティ コンポーネントでタグを使用する方法の例については、vRealize Automation Cloud Assembly ブループリントのネットワーク、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。

ブループリント コード サンプルについては、vRealize Automation Cloud Assembly ブループリントのネットワーク、セキュリティ、およびロード バランサの例の「マシン NIC に制約タグが適用されている既存のセキュリティ グループ」の例を参照してください。

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 Cloud Assembly 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 プロファイルおよびブループリントでセキュリティ グループを定義する方法

セキュリティ グループの機能は、次のいずれかの方法で利用できます。

- 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。ブループリント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、ブループリントにセキュリティ グループ コンポーネントを追加する必要はありません。

- ブループリント内のマシン コンポーネント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 コンポーネントをブループリントにドラッグ アンド ドロップし、セキュリティ グループ コンポーネントをマシン NIC にバインドすることができます。その際には、ブループリント内の既存のセキュリティ グループ コンポーネントと、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。

- NSX-T アプリケーションで指定された NSX-T タグ

- ネットワーク コンポーネントがブループリントのマシン NIC に接続されているときに、ブループリント内のネットワーク コンポーネント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。

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 Cloud Assembly 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。

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 Cloud Assembly ストレージ プロファイルの詳細 を参照してください。

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マシン リストに表示されます。

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。

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。

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。

ボリューム リソース

vRealize Automation Cloud Assembly 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。

vRealize Automation Cloud Assembly のリソースの詳細

vRealize Automation Cloud Assembly では、コスト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 vRealize Automation Cloud Assembly でのデータ収集の仕組み

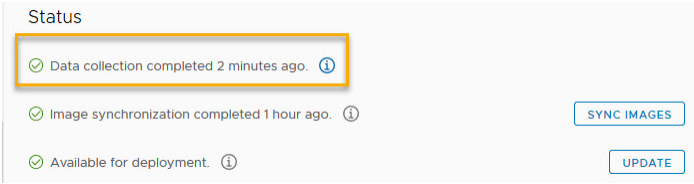
初期データ収集の後、リソース データ収集は 10 分ごとに自動的に実行されます。データ収集間隔は設定できません。また、データ収集を手動で開始することはできません。

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。

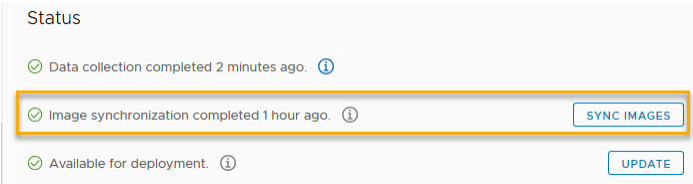


イメージ データ収集

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 Cloud Assembly 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 Cloud Assembly 展開のコストとは

プロジェクト管理者とクラウド管理者は、多くの場合、コストの管理を担当します。vRealize Automation Cloud Assembly コスト管理を使用すると、個々の展開に与える金銭面の影響を把握しやすくなるので、リソースを管理するのに役立ちます。

コスト計算を表示するには、vRealize Operations のコストが vRealize Automation と連携するように設定して有効にする必要があります。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。

展開のコストが時系列で展開カードに表示されます。コスト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。

同じようなビジネス上の理由で、プロジェクト全体の総コストを把握しておく必要がある場合があります。プロジェクト レベルのコストは、プロジェクト チームが実施したすべての展開のコストを集計したものです。プロジェクト コストの全体を表示するには、[インフラストラクチャ] - [プロジェクト] - [プロジェクトの選択] - [コスト] の順に選択します。結果には、合計コストと展開ごとのコストの内訳が表示されます。

注： プロジェクト コストには、プライベート クラウド ワークロードのコストのみが含まれます。パブリック クラウドに属する展開がプロジェクトに含まれている場合、その展開のコストはプロジェクト コストに含まれません。

コストの計算方法

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コストは、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。コスト 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 時間ごとにコストを再計算します。

また、vROPs エンドポイント ページの [インフラストラクチャ] - [統合] - [vROPs エンドポイント] - [] で、コスト 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コスト 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コストが再計算されます。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。

算出されたレートをカスタマイズする方法

クラウド管理者は、一定期間ベンチマーク値を使用した後、実際のコストが割り引かれていることに気づくことがあります。ビジネス プラクティスを的確に反映したものになるように、コストの計算に使用されるレートを調整できます。

調整を行った場合、次にサービスが計算を実行するときに、計算の変更が反映されます。サーバは、24 時間ごとに値を再計算します。

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

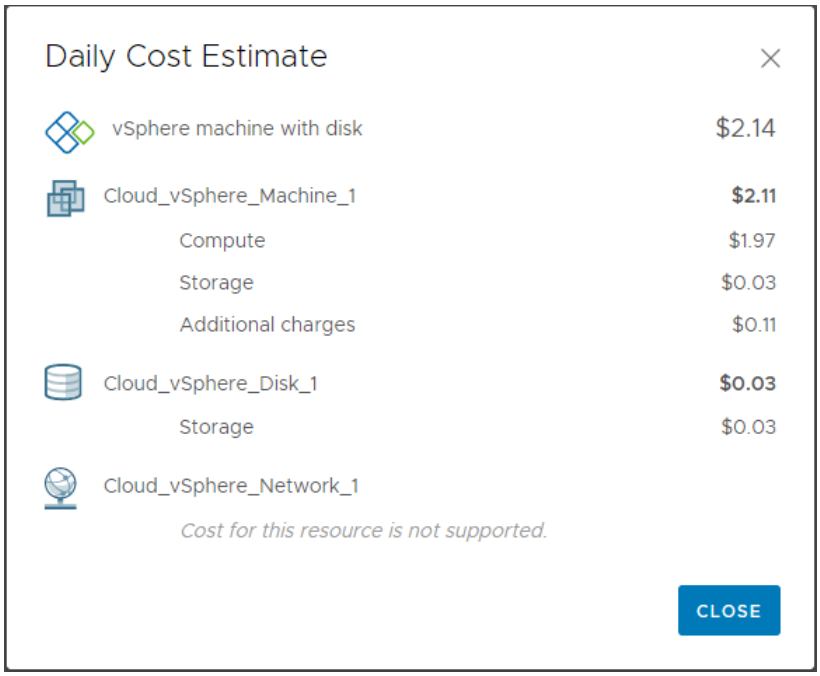
vRealize Automation Cloud Assembly では、以下のブループリント コンポーネント タイプに対するベンチマーク コスト情報を提供しています。

表 4-3. 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。

展開のコストを見積もる方法

カタログ アイテムを展開する前に、展開のコスト見積もりとして、初期コストを使用できます。



展開の初期コストは、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コスト見積もりです。カタログ アイテムの展開後に、今日までの月間コストが初期コストの集計として、[展開] 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。初期コストの計算は、vSphere マシン、vSphere ディスク、Cloud Assembly カタログ 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。

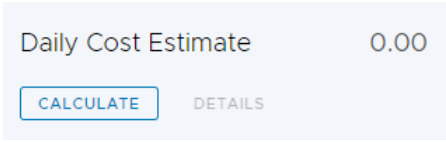
注： パブリック クラウド リソースや、非 vSphere マシンまたはディスクなどのプライベート クラウド リソースでは、初期コストの計算はサポートされていません。

前提条件

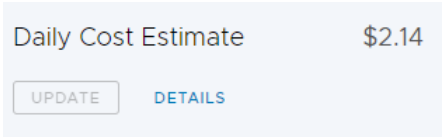
vRealize Automation Cloud Assembly の初期コストを表示するには、vRealize Operations 統合エンドポイントで、コスト計算を有効にし、通貨が事前設定されている必要があります。

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。



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。



3 （オプション）[毎日のコスト見積もり] ウィンドウにコスト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

毎日のコスト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。

展開が高コストと思われる理由を判断する方法

プロジェクト管理者とクラウド管理者は、コスト管理の際に展開コストの高さに気付くことがあります。

展開のコストは、そのリソースごとに計算されます。1 つの展開に含まれるリソースを合計するとコストが増大し、ビジネス ニーズから見て展開のコストが高くなることがあります。そのため、プロジェクトに 1 つでも高コストの展開があると、プロジェクトの総コストも増加します。

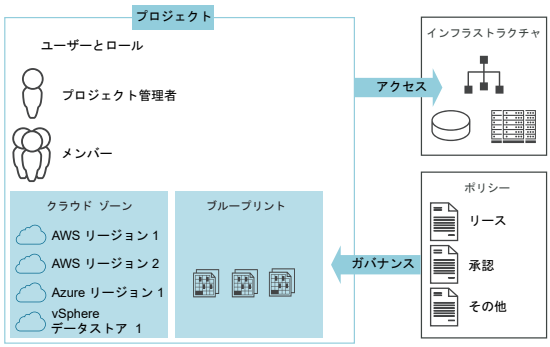
自分のビジネス ニーズに応じて、展開が高コストであるかどうか判断してください。展開のコストの内訳を表示するには、展開の詳細を開き、[コスト] をクリックします。展開のコストを変更するには、そのブループリントのリソース構成を編集してから再展開する必要があります。

vRealize Automation Cloud Assembly プロジェクトの追加と管理

5

プロジェクトでは、vRealize Automation Cloud Assembly のブループリントへのアクセス権とブループリントの展開先が制御されます。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ブループリン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。ブループリントを作成および展開するユーザーは、いずれも 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ブループリン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。

ブループリン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、ブループリン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がブループリントをサポートする方法の例については、WordPress の使用事例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。開発チームがブループリントを作成および展開するときに、プロジェクトに変更を行う場合があります。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構築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あ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構築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 Cloud Assembly 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 Cloud Assembly 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あ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] - [設定] - [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 プロジェクト メンバーによる展開を所有者のみが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。

   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。

   クラウド ゾーンには、ユーザーによって展開されたブループリントをサポートするリソースが含まれている必要があります。

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 [設定のテスト] 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、リソース設定を修正します。

次のステップ

ブループリン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、ブループリントとリソースの間のコネクタです。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 ブループリントと異なる場合、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]。ブループリントを機能タグと一致する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ブループリントは、PCI 準拠のネットワーク上でプロビジョニングされる必要があります。`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ブループリン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、ブループリン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。

プロジェクトの制約タグの優先順位はブループリントの制約タグよりも高く、展開時にオーバーライドされる点は重要です。この状況が発生しないブループリントを使用している場合は、ブループリントで `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、ブループリントには `loc:mumbai` があるときに、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、レポート作成への使用、拡張性アクションおよびワークフローのトリガと設定、およびブループリント 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。

ブループリントには、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。クラウド管理者もブループリント開発者も、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、プロジェクトがブループリント 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、ブループリン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を行うのに役立ちます。

ブループリントを作成する場合は、まずそのブループリントをプロジェクトに関連付けます。展開時に、ブループリン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。

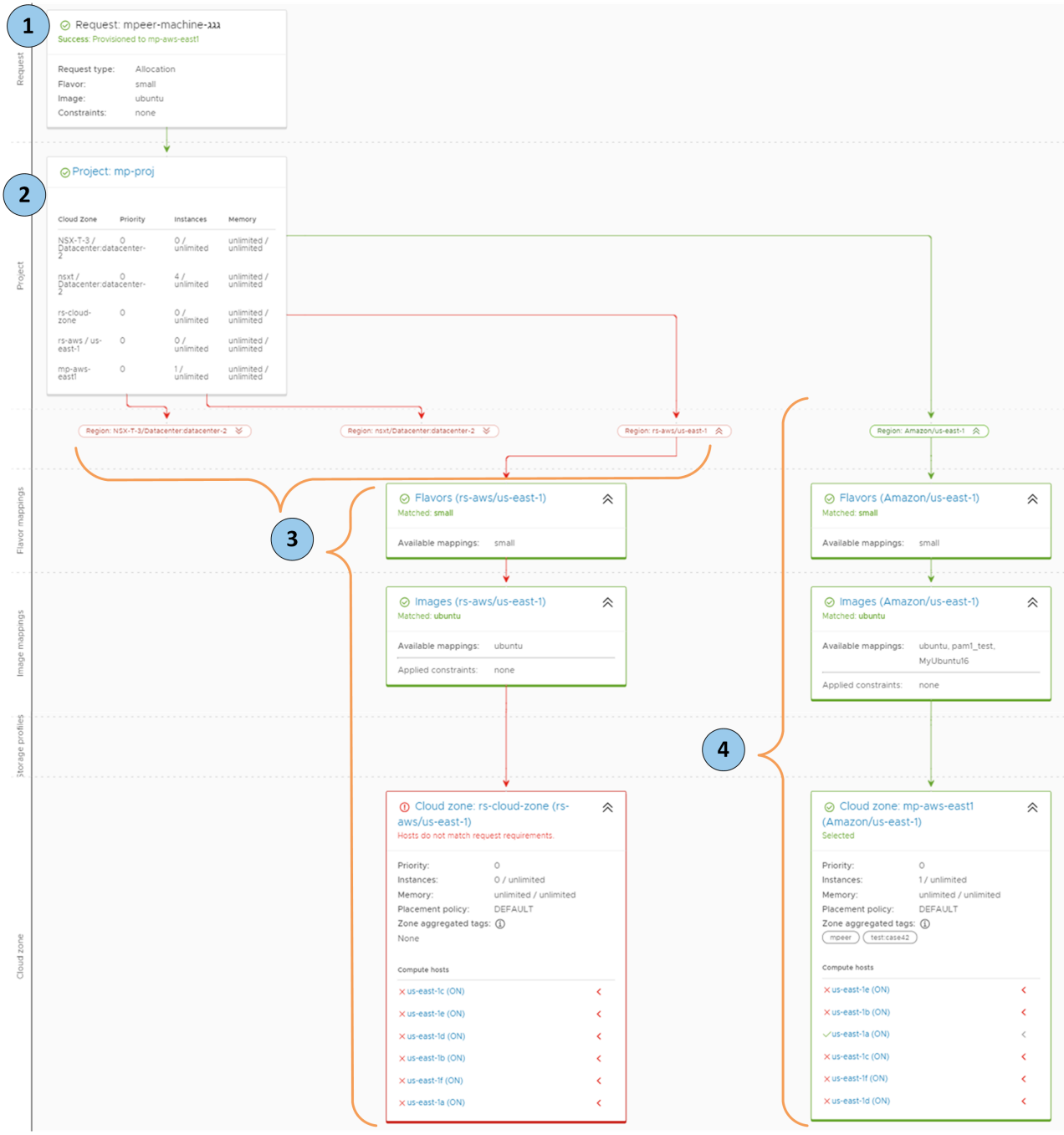
次のワークフローは、このプロセスを示しています。

1 ブループリン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、制約タグなどのブループリン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ので、ブループリントはこのクラウド ゾーンのアカウント リージョンに展開されます。



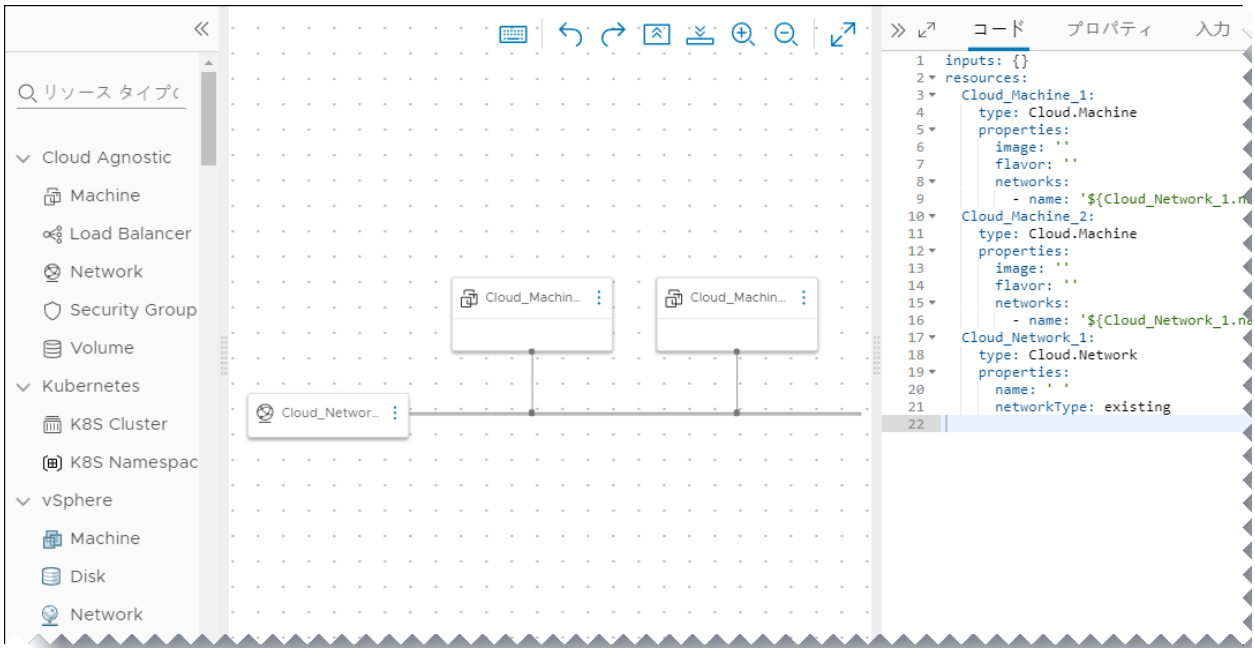
vRealize Automation Cloud Assembly 展開の設計

6

展開の基礎はブループリントです。ブループリントとは、クラウド リソースに導入されるマシン、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。

ブループリント開発者は、特定のクラウド ベンダーを対象としたブループリントを設計したり、ブループリントがクラウドに依存しないようにしたりすることができます。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てください。

vRealize Automation Cloud Assembly ブループリントの作成は、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 キャンバスでコンポーネントを追加し、接続して、開始することができます。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。



この章には、次のトピックが含まれています。

- ブループリントを作成する前に
- ブループリントを作成する方法

- シンプルな vRealize Automation Cloud Assembly ブループリントをゼロから作成する方法
- シンプルな vRealize Automation Cloud Assembly ブループリントを拡張する方法
- vRealize Automation Cloud Assembly ブループリントの別バージョンを保存する方法
- 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
- vRealize Automation リソースのプロパティについて
- ブループリント コードの例
- vRealize Automation Cloud Assembly マーケットプレイスの使用方法
- 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

ブループリントを作成する前に

vRealize Automation Cloud Assembly ブループリン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構築

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。

- 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

ブループリントを作成する方法

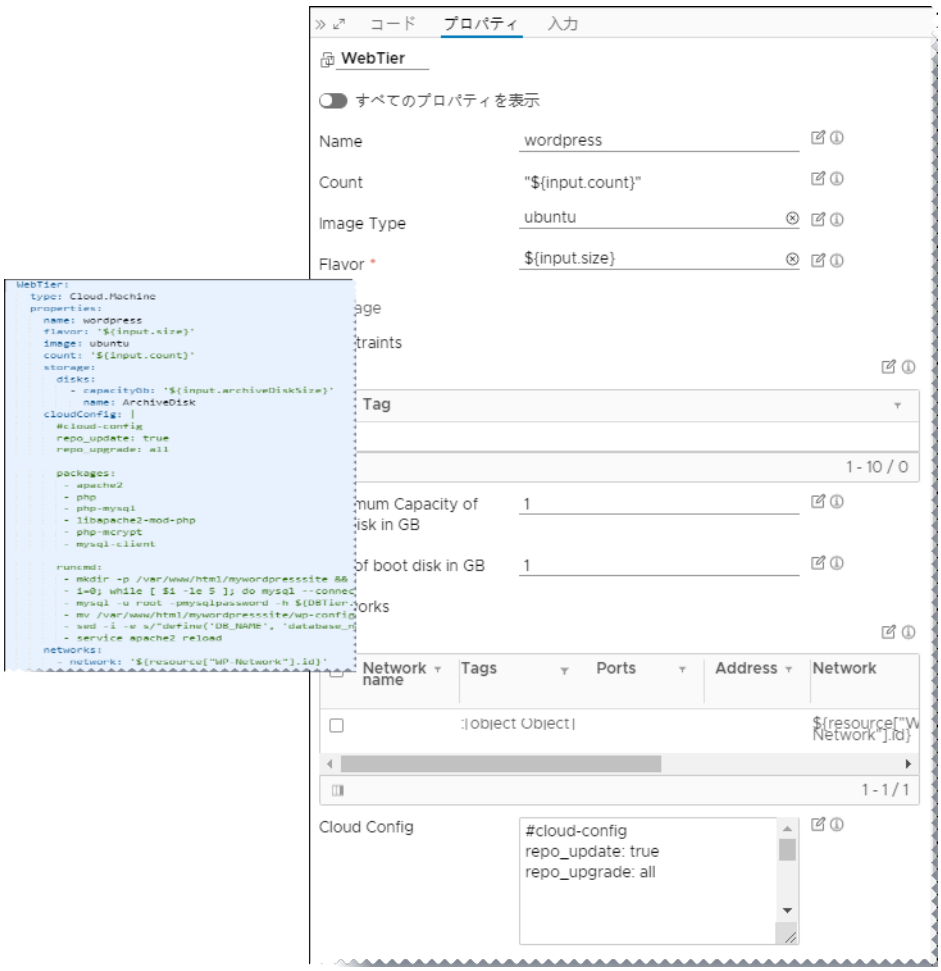
vRealize Automation Cloud Assembly は、ブループリントを作成してコードとして保存できるため、ブループリントを簡単に設計して再利用できます。

ブループリントは、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 ブループリントのデザイン画面

ブループリントを最初から作成するには、[ブループリント] に移動し、[新規] をクリックします。コンポーネント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。



また、ブループリントから別のブループリントにコードをコピー アンド ペーストできます。

ブループリントのクローン作成

ブループリントのクローンを作成するには、[ブループリント] 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。ブループリン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、または新しいアプリケーションのスタータ 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、完成したブループリン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ブループリント 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、ブループリント コードを変更することもできます。

注： 共有ブループリント コードを検証するには、ブループリントのデザイン画面で 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。

シンプルな vRealize Automation Cloud Assembly ブループリントをゼロから作成する方法

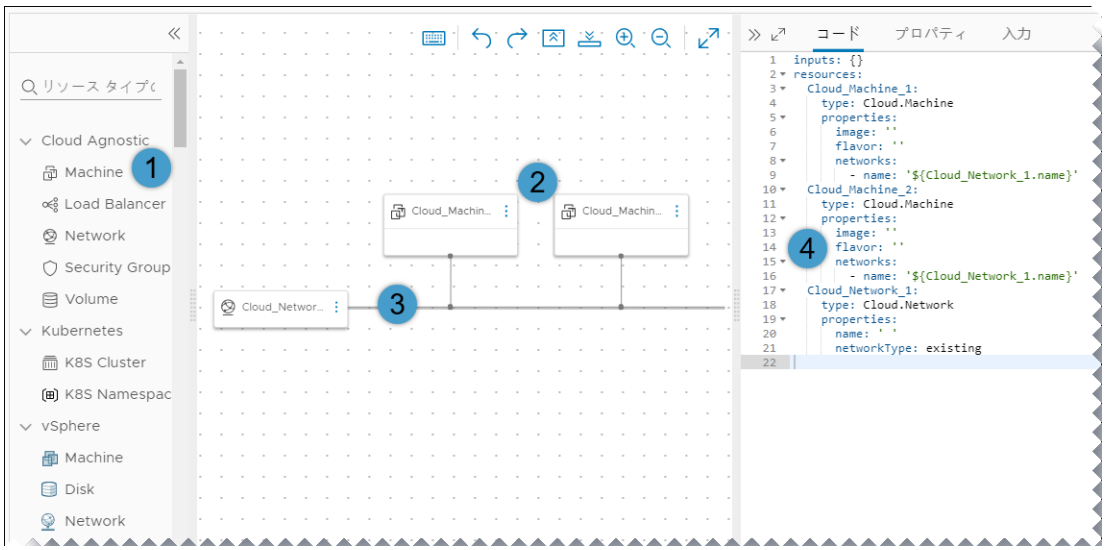
デザイン画面を使用して、プロビジョニングするマシンまたはアプリケーションの vRealize Automation Cloud Assembly ブループリント仕様を作成します。

1 コンポーネントを特定します。

2 コンポーネントをキャンバスにドラッグします。

3 コンポーネントを接続します。

4 ブループリント コードを編集して、コンポーネントを構成します。



デザイン画面では、ブループリント名の変更、バージョンの変更、またはブループリン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 コンポーネントを選択してブループリントに追加する方法

vRealize Automation Cloud Assembly コンポーネントはブループリントのビルディング ブロックです。デザイン画面では、クラウドに依存しないコンポーネントやクラウド ベンダー固有のコンポーネントを使用できます。

コンポーネントは、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コンポーネント

クラウドに依存しないコンポーネントを任意のクラウド 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、ブループリントを 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コンポーネントを使用します。

クラウド ベンダーのコンポーネント

Amazon Web Services、Microsoft Azure、および VMware vSphere の各コンポーネントは、一致する AWS、Azure、または vSphere のクラウド ゾーンにのみ展開できます。

特定のベンダーのクラウド固有のコンポーネントを含むブループリントに、クラウドに依存しないコンポーネントを追加できます。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。

構成管理コンポーネント

構成管理コンポーネントは、統合されたアプリケーションに依存します。たとえば、Puppet コンポーネントでは、他のコンポーネントの構成を監視し、適用することができます。

vRealize Automation Cloud Assembly でブループリント リソースを接続する方法

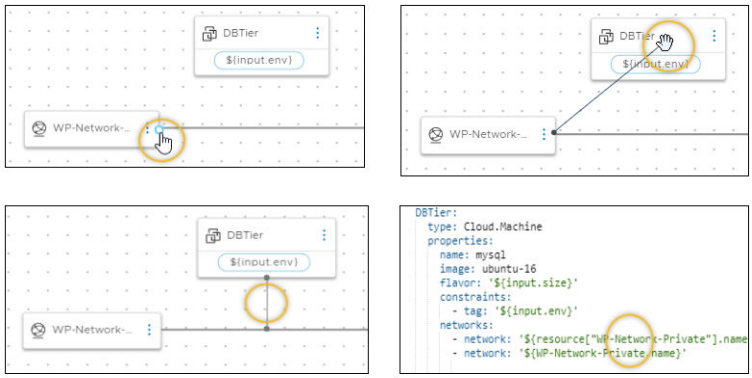
グラフィカル デザイン キャンバスを使用して、vRealize Automation Cloud Assembly ブループリント リソースを接続します。

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。

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。

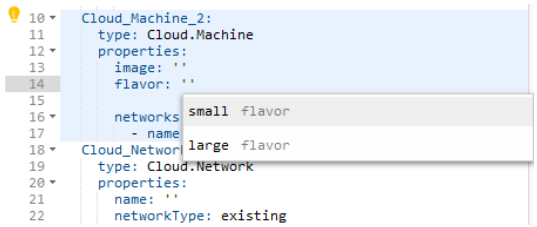
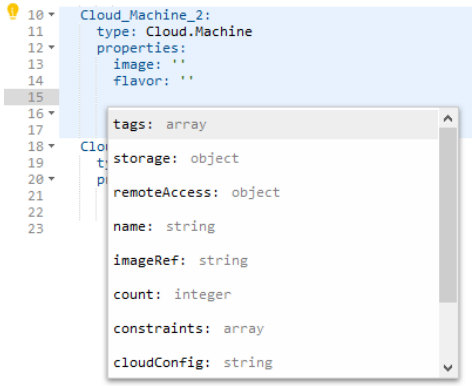
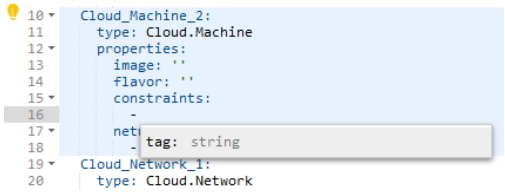


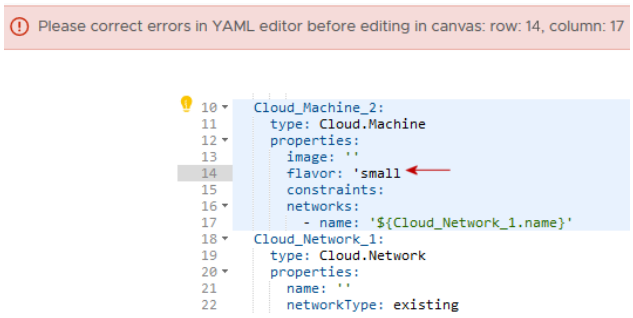
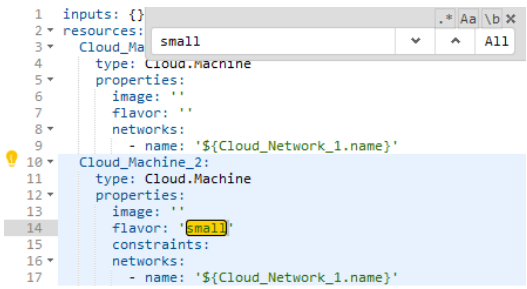
リソース間の実線は、リソースが同じ場所に配置される必要があることを示しています。キャンバス上で接続を追加できる場合でも、配置の制約タグの競合があった場合、展開は失敗します。たとえば、テスト用の us-west-1 クラウド ゾーンに強い制約を受けているリソースを、本番用の us-east-1 クラウド ゾーンと接続すると、展開は失敗します。

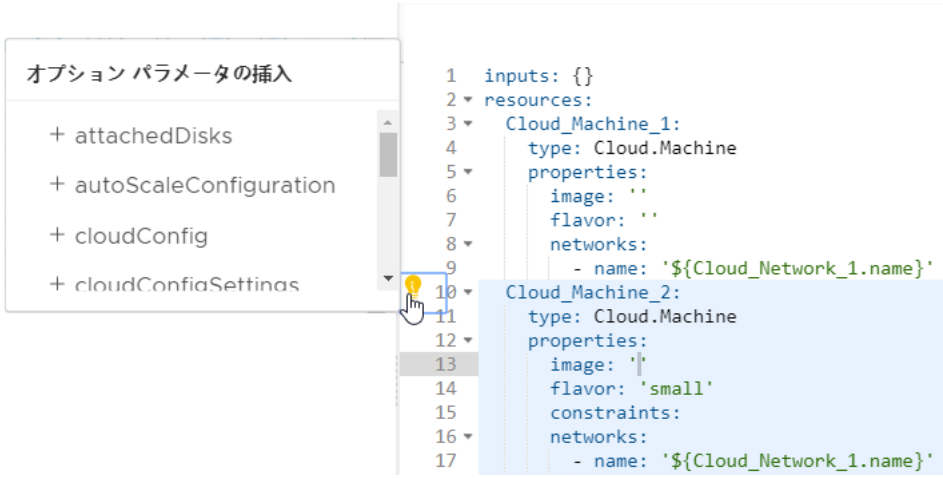
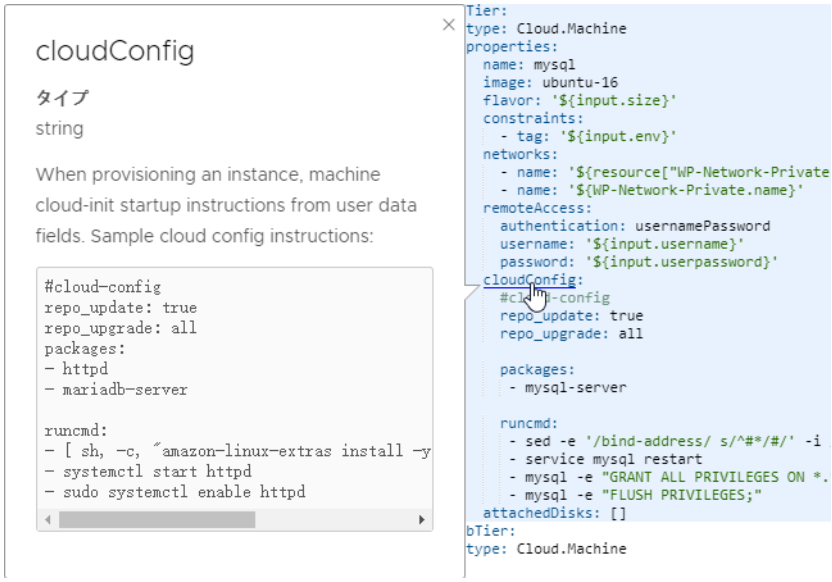
有効な vRealize Automation Cloud Assembly ブループリント コードを作成する方法

キャンバスで vRealize Automation Cloud Assembly コンポーネント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	
使用できるプロパティ	
子プロパティ	

エディタのヒント	例
構文エラー	
Ctrl + F キーによる検索	

エディタのヒント	例
オプション パラメータ	
スキーマ ヘルプ	すべてのカスタム プロパティで、VMware {code} サイトの統合されたリソース スキーマを参照することもできます。 

シンプルな vRealize Automation Cloud Assembly ブループリントを拡張する方法

シンプルなブループリントを次のレベルに進めることができる vRealize Automation Cloud Assembly ブループリント コードの高度な手法があります。

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。

ユーザー入力による vRealize Automation Cloud Assembly ブループリントのカスタマイズ方法

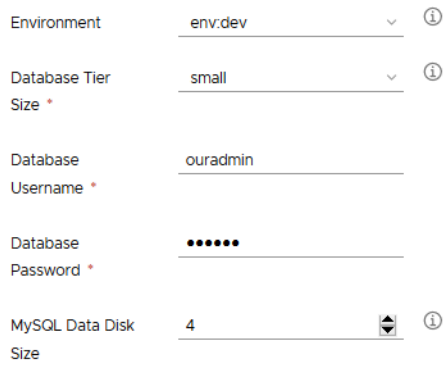
ブループリント開発者は、入力パラメータを使用して、ユーザーが申請時にカスタム選択を行えるようにできます。

ユーザーが入力するようにすると、わずかな違いしかない複数のブループリントを保存する必要がなくなります。また、カタログ ユーザーは、展開を更新すると必ず新規入力を要求されるため、ユーザーの入力によって Day 2 操作のブループリントを準備することもできます。

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。

Day 2 の変更をサポートするように入力を設計するときは、リソースを削除して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を確認するには、VMware {code} の vRealize Automation ブループリント スキーマで統合リソース スキーマを参照してください。

次の入力は、1 つのブループリントを MySQL データベース サーバ用に作成し、ユーザーがそのブループリントを別のクラウド 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。



ブループリントの入力パラメータを定義する方法

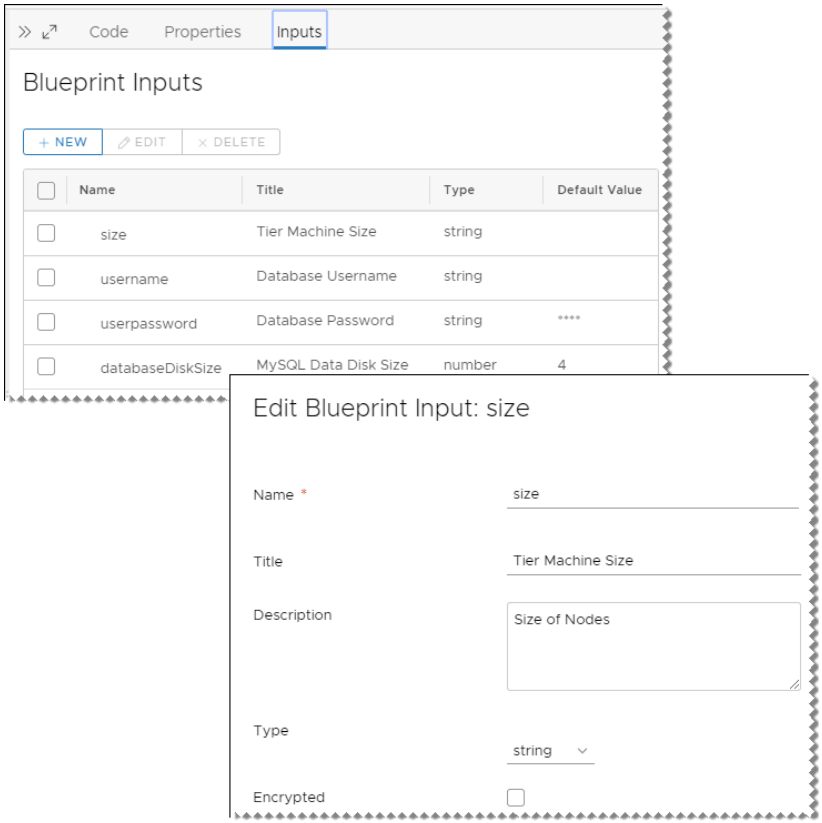
ブループリント コードに `inputs` セクションを追加し、選択可能な値を設定します。

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。



ブループリン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： ブループリント コードでは、入力パラメータを示すため以外に `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。
	`enum:`
	`  - value 1`
	`  - value 2`

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、 `maxLength: 25` 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.
  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
```

vRealize Automation Cloud Assembly でコンポーネントの展開順序を設定する方法

vRealize Automation Cloud Assembly ブループリントを展開するときに、あるコンポーネントの前に別のコンポーネントを展開することが必要な場合があります。

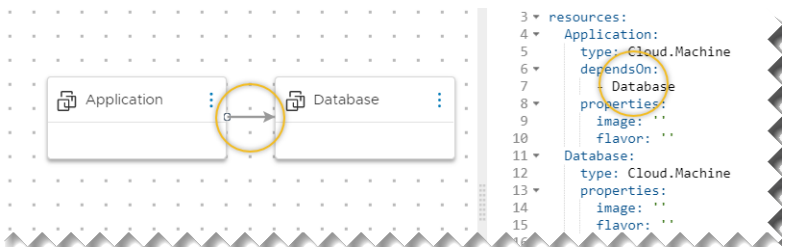
明示的な依存関係を作成する方法

あるコンポーネントの前に別のコンポーネント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 アクションの構築順を設定します。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：依存コンポーネントから始まり最初に展開するコンポーネントで終わる接続を描画します。
- コード エディタ オプション：依存コンポーネントに `dependsOn` プロパティを追加し、最初に展開するコンポーネントを特定します。

  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

コンポーネントのプロパティに、別のコンポーネント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コンポーネントの展開を待つことによって構築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。

- 依存コンポーネントを編集して、最初に配置されている必要があるコンポーネント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。

式を使用して vRealize Automation Cloud Assembly ブループリント 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 のブループリント 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。未編集のブループリン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:
        - name: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - name: '${resource.Cloud_Network_1.name}'
```

```
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      name: Default
      networkType: existing
```

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 ${resource.apitier.networks[0].address}
  apitier:
    type: Cloud.Machine
    properties:
      networks:
        - name: '${resource.Cloud_Network_1.name}'
```

ブループリントの完了

```
inputs:
  environment:
    type: string
    enum:
      - AWS
      - vSphere
      - Azure
      - VMC
      - GCP
    default: vSphere
  sshKey:
    type: string
    maxLength: 500
  envsize:
    type: string
    enum:
      - Small
      - Large
```

```
resources:
  frontend:
    type: Cloud.Machine
    properties:
      folderName: '${input.environment == "VMC" ? "Workload" : ""}'
      image: ubuntu
      flavor: medium
      count: '${input.envsize == "Small" ? 1 : 2}'
      remoteAccess:
        authentication: publicPrivateKey
        sshKey: '${input.sshKey}'
      cloudConfig: |
        packages:
          - nginx
        runcmd:
          - echo ${resource.apitier.networks[0].address}
      constraints:
        - tag: '${"env:" + to_lower(input.environment)}'
      networks:
        - name: '${resource.Cloud_Network_1.name}'
  apitier:
    type: Cloud.Machine
    properties:
      folderName: '${input.environment == "VMC" ? "Workload" : ""}'
      image: ubuntu
      flavor: small
      cloudConfig: |
        #cloud-config
        runcmd:
          - export apikey=${base64_encode(input.username:input.password)}
          - curl -i -H 'Accept:application/json' -H 'Authorization:Basic :$apikey' http://
example.com
      remoteAccess:
        authentication: publicPrivateKey
        sshKey: '${input.sshKey}'
      constraints:
        - tag: '${"env:" + to_lower(input.environment)}'
      networks:
        - name: '${resource.Cloud_Network_1.name}'
  Cloud_Network_1:
    type: Cloud.Network
    properties:
      name: Default
      networkType: existing
      constraints:
        - tag: '${"env:" + to_lower(input.environment)}'
```

vRealize Automation Cloud Assembly ブループリント式の構文

vRealize Automation Cloud Assembly ブループリント式の構文は、式のすべての機能を公開します。

この構文は、式を使用して vRealize Automation Cloud Assembly ブループリント 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

表 6-1. 関数 （続き）

関数	説明
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タ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
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。

表 6-1. 関数 （続き）

関数	説明
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 ブループリン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、vSphere のカスタマイズ仕様を使用するか、コマンドを直接実行します。

ブループリント コード内のプロパティは、vSphere のカスタマイズ仕様を名前で参照できます。または、cloudConfig セクションをブループリントに追加して、特定のコマンドを組み込むこともできます。

組み込みコマンドとカスタマイズ仕様の初期化を組み合わせる場合は、ご注意ください。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。

カスタマイズ仕様が cloudConfig セクションのコマンドにどのように干渉するかを示す例については、固定 IP アドレスを使用して Linux マシンを展開する方法を参照してください。

vRealize Automation Cloud Assembly ブループリントの vSphere カスタマイズ仕様

カスタマイズ仕様を使用すると、vSphere ベースのクラウド ゾーンに展開するときにゲスト OS の設定を適用できます。

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。

ブループリント 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'
```

カスタマイズ仕様と初期化コマンド

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には、ブループリントの cloudConfig セクションに初期化コマンドを組み込むほうがより無難なアプローチです。

ブループリントの cloudConfig セクションの詳細については、vRealize Automation Cloud Assembly ブループリントの設定コマンドを参照してください。

vRealize Automation Cloud Assembly ブループリントの設定コマンド

cloudConfig セクションを vRealize Automation Cloud Assembly ブループリント 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。

- Linux：初期化コマンドはオープン スタンダードの 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。

注： 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。

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

vRealize Automation Cloud Assembly では、インフラストラクチャの構成時に、マシン イメージに初期化コマンドを事前に追加することもできます。ソース イメージを参照するすべてのブループリントは、同じ初期化を取得します。

重要： イメージ マップとブループリン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 Cloud Assembly でのイメージ マッピングの詳細を参照してください。

次の cloudConfig セクションの例は、Linux ベースの MySQL サーバ用の WordPress の使用事例：基本的なブループリントの作成 ブループリント コードから取られています。

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。

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。

```
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
```

固定 IP アドレスを使用して Linux マシンを展開する方法

vSphere に展開する場合に固定 IP アドレスを使用するには、vRealize Automation Cloud Assembly が vSphere カスタマイズ仕様を生成する必要があります。これは、cloud-init コマンドに干渉する可能性があります。

問題

- vRealize Automation Cloud Assembly ブループリントには、vSphere 仮想マシンに固定 IP アドレスを適用するための `assignment: static` が含まれています。
- このブループリントには、cloudConfig セクションも含まれています。このセクションには、cloud-init を使用して実行される初期化コマンドが含まれています。
- 仮想マシンに固定 IP アドレスを付与するために、vRealize Automation Cloud Assembly は、適用する vSphere カスタマイズ仕様を動的に生成します。
- カスタマイズ仕様が適用されるときには、最後の操作によって仮想マシンが再起動されます。
- カスタマイズ仕様では、cloud-init コマンドが実行中であることは認識されないため、再起動によってコマンドの実行が中断されます。
- cloud-init コマンドは、最初の起動時にのみ実行され、中断されたときに自動的にはリカバリされません。
- その結果、仮想マシンは部分的にのみ構成されたままになります。

回避策

cloud-init が指定の時間に無効になるように設定されたマシン テンプレートを作成します。次に、テンプレートに基づいてマシンを展開し、カスタマイズ仕様と再起動が cloud-init の前に実行されるようにします。

手順の例 - Ubuntu 18.04

次の手順は、Ubuntu 18.04 に適用されます。その他の Linux バージョンまたはオファリングについては、調整を加えて、ここに示されているアイデアを適合させることが必要になる場合があります。

1 仮想マシンを作成し、必要なバージョンのアップデートとパッケージを使用して最新の状態にします。

Ubuntu 18.04 と異なり、他の Linux オファリングでは cloud-init が事前にインストールされていない場合があることに注意してください。

2 データソースを OVF に設定して cloud-init を再設定します。

```
sudo dpkg-reconfigure cloud-init
```

3 次のファイルを編集します。

```
/etc/cloud/cloud.cfg
```

a 次の行を追加して、従来のゲスト OS のカスタマイズ (GOSC) を有効にします。

```
disable_vmware_customization: true
```

b ネットワーク構成が有効になっていることを確認します。無効化設定がある場合は、これを削除するか、コメント アウトします。

```
network:
  # config: disabled
```

または、次のディレクトリにあるすべての構成ファイルを調べます。

```
/etc/cloud/cloud.cfg.d/*
```

`network: {config: disabled}` 設定が含まれているすべてのファイルを削除します。

4 次のファイルを編集します。

```
/usr/lib/tmpfiles.d/tmp.conf
```

- 一時ディレクトリが消去されないように、設定をコメント アウトします。

```
# D /tmp 1777 root root -
```

5 次のファイルを編集します。

```
/lib/systemd/system/open-vmtools.service
```

- `[Unit]` セクションの下に次の行を追加して、open-vmtools が dbus.service の後に起動するように設定します。

```
After=dbus.service
```

6 cloud-init を無効にする、新しい空のファイルを作成します。

```
sudo touch /etc/cloud/cloud-init.disabled
```

7 re_init.sh スクリプトを作成します。このスクリプトでは、カスタマイズ仕様のために一時停止する cron ジョブの遅延の後、cloud-init を再度有効にし、初期化します。

```
sudo rm -rf /etc/cloud/cloud-init.disabled
sudo cloud-init init
sleep 20
sudo cloud-init modules --mode config
sleep 20
sudo cloud-init modules --mode final
```

8 スクリプトの実行権限を追加します。

```
sudo chmod +x re_init.sh
```

9 起動時に 90 秒間のスリープ後に実行される cron ジョブを作成します。`crontab -e` と入力し、次のように入力します。

```
@reboot ( sleep 90 ; sh /script_path/delay_init.sh )
```

カスタマイズ仕様と再起動が完了するまでに時間がかかる場合は、90 秒以上を適用できます。

10 テンプレートをクリーンアップする cleaner.sh スクリプトを作成します。`cloudadmin` を、オペレーティングシステムのインストール時に設定した独自のユーザーに置き換えます。

このサンプル スクリプトは、Ubuntu に固有のものです。その他の Linux オファリング用のスクリプトを作成するには、強調表示されている必須のセクションを必ず含めてください。

```
#!/bin/bash

# Add usernames to add to /etc/sudoers for passwordless sudo
users=("ubuntu" "cloudadmin")

for user in "${users[@]}"
do
cat /etc/sudoers | grep ^$user
RC=$?
if [ $RC != 0 ]; then
bash -c "echo \"$user ALL=(ALL) NOPASSWD:ALL\" >> /etc/sudoers"
fi
done

#grab Ubuntu Codename
codename="$(lsb_release -c | awk {'print $2}')"


#Stop services for cleanup
service rsyslog stop

#clear audit logs
if [ -f /var/log/audit/audit.log ]; then
cat /dev/null > /var/log/audit/audit.log
fi
if [ -f /var/log/wtmp ]; then
cat /dev/null > /var/log/wtmp
fi
```

```
if [ -f /var/log/lastlog ]; then
cat /dev/null > /var/log/lastlog
fi

#cleanup persistent udev rules
if [ -f /etc/udev/rules.d/70-persistent-net.rules ]; then
rm /etc/udev/rules.d/70-persistent-net.rules
fi

#cleanup /tmp directories
rm -rf /tmp/*
rm -rf /var/tmp/*

#cleanup current ssh keys
#rm -f /etc/ssh/ssh_host_*

#cat /dev/null > /etc/hostname

#cleanup apt
apt-get clean

#Clean Machine ID

truncate -s 0 /etc/machine-id
rm /var/lib/dbus/machine-id
ln -s /etc/machine-id /var/lib/dbus/machine-id

#Clean Cloud-init
cloud-init clean --logs --seed

#cleanup shell history
history -w
history -c
```

11 テンプレートのクリーニング スクリプトの実行権限を追加します。

```
sudo chmod +x cleaner.sh
```

12 Ubuntu 18.04 では、クリーンアップ スクリプトを実行するのに root 権限が必要です。次のファイルを編集します。

```
/etc/ssh/sshd_config
```

a root ユーザーに切り替えることができることを確認します。

```
PermitRootLogin yes
```

b root のパスワードを設定します。

```
sudo passwd root
```

13 クリーンアップ スクリプトを実行します。

```
sudo ./script_path/cleaner.sh
```

14 （オプション）セキュリティのために、手順 12 の逆の操作を行って、これ以降の root でのログインを防止します。

15　仮想マシンをシャットダウンし、vSphere を使用してテンプレートに変換します。

テンプレートの更新

cron ジョブは、テンプレートを更新するたびに実行されます。更新が遅延（90 秒など）よりも長くかかる場合は、`/etc/cloud/cloug-init.disabled` ファイルを再度追加し、クリーンアップ スクリプトを再実行してから、テンプレートをシャットダウンする必要があります。そうしないと、最初の起動時に cloud-init が無効にならず、カスタマイズ仕様が再起動した結果、cloud-init コマンドが中断されます。

トラブルシューティング

vSphere のカスタマイズ仕様によって cloud-init の完了が妨げられている可能性がある場合は、カスタマイズ仕様を一時的に無効にして、cloud-init が想定どおりに終了するかどうかを判断します。カスタマイズ仕様を一時的に無効にするには、`customizeGuestOS: false` プロパティを使用します。

```
properties:
  image: ubuntu
  cpuCount: 1
  totalMemoryMB: 8192
  customizeGuestOS: false
```

vRealize Automation Cloud Assembly の展開が初期化を待機するようにする方法

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 でコンポーネント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、ブループリント内に cloudConfig コマンドが必要です。
- アイデア次第で用途が広がります。cloudConfig セクションには、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、ブループリントベースの 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、ブループリントに `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。

vRealize Automation Cloud Assembly ブループリントでリモート アクセスを有効にする方法

vRealize Automation Cloud Assembly によって展開されるマシンにリモート アクセスするには、展開する前にそのマシンのブループリン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、ブループリントに 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 ブループリン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` プロパティをブループリントに追加します。

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、マシンをブループリン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` プロパティをブループリントに追加します。

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+dstIcowRRkyobbfpA1mj9tslf
qGxvU66PX9IeZax5hZvNWFgjw6ag+ZlzndOLhVdVoW49f274/mIRild7UUW...
    username: testuser
```

3 vRealize Automation Cloud Assembly で、マシンをブループリン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 ペア名をブループリン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

ブループリン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リモート 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` プロパティをブループリントに追加します。

   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、マシンをブループリン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 ブループリントの別バージョンを保存する方法

ブループリントの開発時、正常に機能しているブループリントに変更を加えることにはリスクが伴うため、その前に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。

ブループリント バージョンのキャプチャ方法

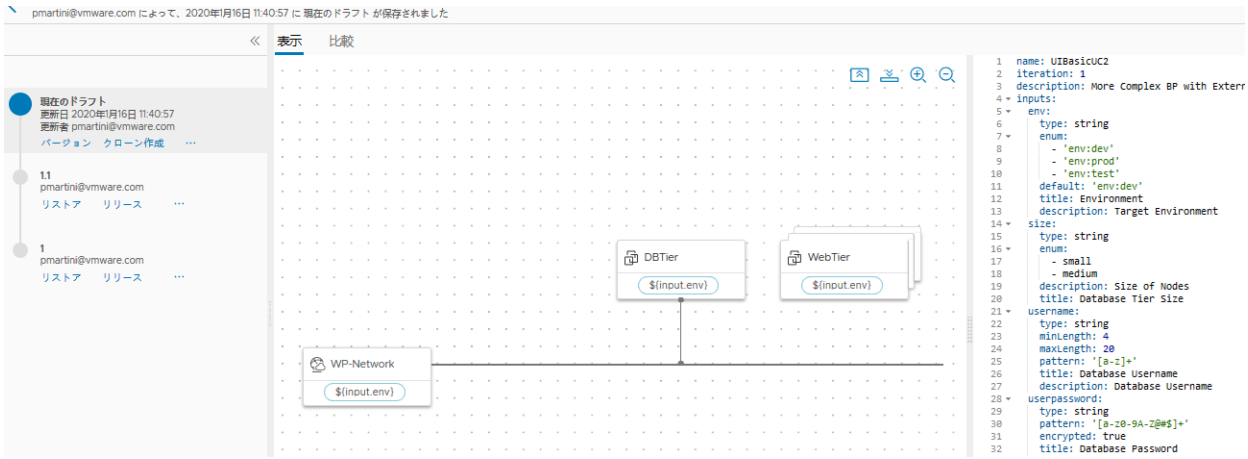
デザイン画面で、[バージョン] をクリックし、名前を入力します。

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。

以前のバージョンをリストアする方法

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 のユーザーにバージョンをリリースする方法

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、[リリース] をクリックします。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。

1 つのブループリントについて複数のバージョンをリリースすると、vRealize Automation Service Broker では最新のブループリントが使用されます。

ブループリントのバージョンを比較する方法

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。

[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

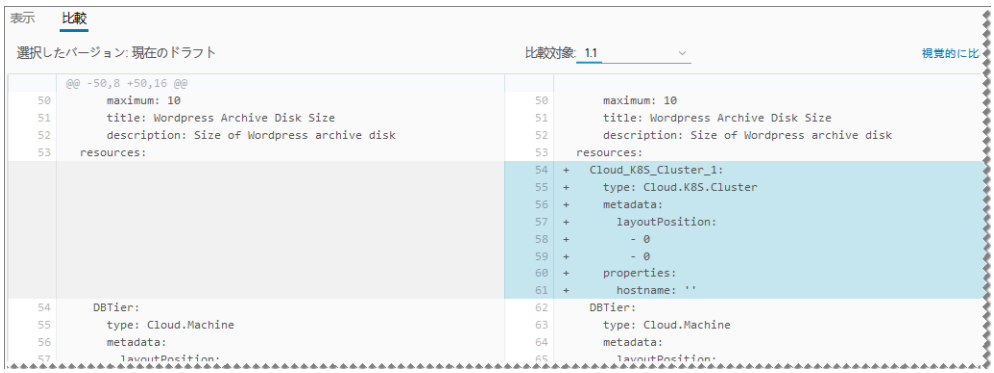
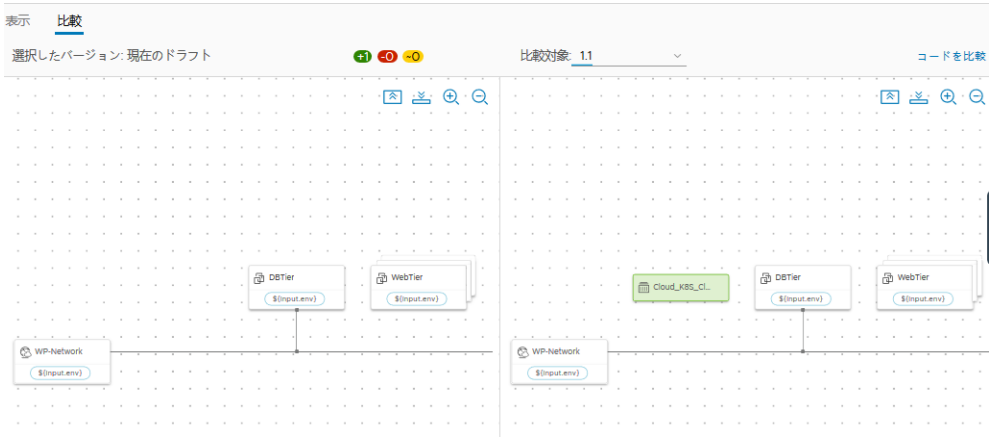


図 6-2. 視覚的なトポロジの相違点



ブループリントのクローン作成方法

バージョンの保存とは異なりますが、デザイン画面で [アクション] - [クローン作成] の順に選択すると、現在のブループリントのコピーが作成され、別の開発に使用できます。

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設定します。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。

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高いと番号付けがランダムに見える場合がありますが、一意性は維持されます。

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ブループリン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。

a 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。

b 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、構築時にヒント 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。

5 プロジェクトに関連付けられているブループリントを展開して、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。

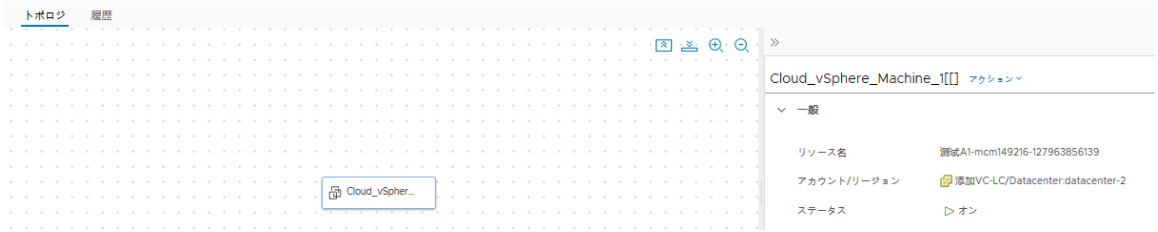
a [ブループリント] タブをクリックし、プロジェクトに関連付けられているブループリントをクリックします。

b ブループリントを展開します。

[展開] タブが開き、進行中の展開が示されます。

c　展開が完了したら、展開名をクリックします。

d　[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。



6　命名規則を確認するためにテスト ブループリン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 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。ブループリント 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、ブループリントで使用できるリソース オブジェクトを一覧表示します。

- VMware {code} の vRealize Automation Resource Type Schema

ブループリント コードの例

vRealize Automation Cloud Assembly のブループリント 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無限です。

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 ブループリント内の vSphere マシン例

以下は、vRealize Automation Cloud Assembly ブループリント内の vSphere リソースの定義の基本的な例です。

リソース	ブループリント例

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	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  name: demo-machine`
	`  cpuCount: 1`
	`  totalMemoryMB: 1024`
	`  image: ubuntu`

データストア リソースを含む vSphere マシン	`resources:`
	`  demo-vsphere-disk-001:`
	`  type: Cloud.vSphere.Disk`
	`  properties:`
	`  name: DISK_001`
	`  type: 'HDD'`
	`  capacityGb: 10`
	`  dataStore: 'datastore-01'`
	`  provisioningType: thick`

ディスクが接続された vSphere マシン	`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}'`

リンク クローン イメージの vSphere マシン(スラッシュとスナップショット名を付加)	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  imageRef: 'demo-machine/snapshot-01'`
	`  cpuCount: 1`
	`  totalMemoryMB: 1024`

リソース	ブループリント例

vCenter 内の特定のフォルダ内の vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  name: demo-machine`
	`  cpuCount: 2`
	`  totalMemoryMB: 1024`
	`  imageRef: ubuntu`
	`  resourceGroupName: 'myFolder'`

複数の NIC を含む vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.Machine`
	`  properties:`
	`  image: ubuntu`
	`  flavor: small`
	`  networks:`
	`  - name: '${network-01.name}'`
	`  deviceIndex: 0`
	`  - name: '${network-02.name}'`
	`  deviceIndex: 1`
	`  network-01:`
	`  type: Cloud.vSphere.Network`
	`  properties:`
	`  name: network-01`
	`  network-02:`
	`  type: Cloud.vSphere.Network`
	`  properties:`
	`  name: network-02`

スナップショット イメージからの vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  imageRef: 'demo-machine/snapshot-01'`
	`  cpuCount: 1`
	`  totalMemoryMB: 1024`

vCenter にタグが接続された vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.Machine`
	`  properties:`
	`  flavor: small`
	`  image: ubuntu`
	`  tags:`
	`  - key: env`
	`  value: demo`

カスタマイズ仕様を含む vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  name: demo-machine`
	`  image: ubuntu`
	`  flavor: small`
	`  customizationSpec: Linux`

リソース	ブループリント例

vSphere ネットワーク コンポーネントと固定 IP アドレスを持つ vSphere マシン	`resources:`
	`  demo-network:`
	`  type: Cloud.vSphere.Network`
	`  properties:`
	`  name: demo-network`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  image: ubuntu`
	`  flavor: small`
	`  networks:`
	`  - name: demo-network`
	`  assignment: static`

リモート アクセスでの vSphere マシン	`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}`

ドキュメント化された vRealize Automation Cloud Assembly ブループリントの例

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 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 blueprint is enhanced with comments to explain its
# parameters.
#
# Try cloning it and experimenting with its YAML code. If you're new to
# YAML, visit yaml.org for general information.
#
# The blueprint deploys a minimum of 3 virtual machines and runs scripts
# to install packages.
#
# ****************************************************************************
#
# ----------------------------------------------------------------------------
# Blueprints need a descriptive name and version if
# source controlled in git.
# ----------------------------------------------------------------------------
name: WordPress Blueprint with Comments
formatVersion: 1
version: 1
#
# ----------------------------------------------------------------------------
# Inputs create user selections that appear at deployment time. Inputs
# can set placement decisions and configurations, and are referenced
# later, by the resources section.
# ----------------------------------------------------------------------------
inputs:
#
# ----------------------------------------------------------------------------
# Choose a cloud endpoint. 'Title' is the visible
# option text (oneOf allows for the friendly title). 'Const' is the
# tag that identifies the endpoint, which was set up earlier, under the
# Cloud Assembly Infrastructure tab.
# ----------------------------------------------------------------------------
  platform:
    type: string
    title: Deploy to
    oneOf:
      - title: AWS
        const: aws
      - title: Azure
        const: azure
      - title: vSphere
        const: vsphere
    default: vsphere
#
# ----------------------------------------------------------------------------
# Choose the operating system. Note that the Cloud Assembly
```

```
# Infrastructure must also have an AWS, Azure, and vSphere Ubuntu image
# mapped. In this case, enum sets the option that you see, meaning there's
# no friendly title feature this time. Also, only Ubuntu is available
# here, but having this input stubbed in lets you add more operating
# systems later.
# ------------------------------------------------------------------------
  osimage:
    type: string
    title: Operating System
    description: Which OS to use
    enum:
      - Ubuntu
#
# ------------------------------------------------------------------------
# Set the number of machines in the database cluster. Small and large
# correspond to 1 or 2  machines, respectively, which you see later,
# down in the resources section.
# ------------------------------------------------------------------------
  dbenvsize:
    type: string
    title: Database cluster size
    enum:
      - Small
      - Large
#
# ------------------------------------------------------------------------
# Dynamically tag the machines that will be created. The
# 'array' of objects means you can create as many key-value pairs as
# needed. To see how array input looks when it's collected,
# open the blueprint and click TEST.
# ------------------------------------------------------------------------
  Mtags:
    type: array
    title: Tags
    description: Tags to apply to machines
    items:
      type: object
      properties:
        key:
          type: string
          title: Key
        value:
          type: string
          title: Value
#
# ------------------------------------------------------------------------
# Create machine credentials. These credentials are needed in
# remote access configuration later, in the resources section.
# ------------------------------------------------------------------------
  username:
    type: string
    minLength: 4
    maxLength: 20
    pattern: '[a-z]+'
    title: Database Username
```

```
    description: Database Username
  userpassword:
    type: string
    pattern: '[a-z0-9A-Z@#$]+'
    encrypted: true
    title: Database Password
    description: Database Password
#
# ---------------------------------------------------------------------
# Set the database storage disk size.
# ---------------------------------------------------------------------
  databaseDiskSize:
    type: number
    default: 4
    maximum: 10
    title: MySQL Data Disk Size
    description: Size of database disk
#
# ---------------------------------------------------------------------
# Set the number of machines in the web cluster. Small, medium, and large
# correspond to 2, 3, and 4 machines, respectively, which you see later,
# in the WebTier part of the resources section.
# ---------------------------------------------------------------------
  clusterSize:
    type: string
    enum:
      - small
      - medium
      - large
    title: Wordpress Cluster Size
    description: Wordpress Cluster Size
#
# ---------------------------------------------------------------------
# Set the archive storage disk size.
# ---------------------------------------------------------------------
  archiveDiskSize:
    type: number
    default: 4
    maximum: 10
    title: Wordpress Archive Disk Size
    description: Size of Wordpress archive disk
#
# ---------------------------------------------------------------------
# The resources section configures the deployment of machines, disks,
# networks, and other objects. In several places, the code pulls from
# the preceding interactive user inputs.
# ---------------------------------------------------------------------
resources:
#
# ---------------------------------------------------------------------
# Create the database server. Choose a cloud agnostic machine 'type' so
# that it can deploy to AWS, Azure, or vSphere. Then enter its property
# settings.
# ---------------------------------------------------------------------
  DBTier:
```

```
    type: Cloud.Machine
    properties:
#
# ----------------------------------------------------------------------
# Descriptive name for the virtual machine. Does not become the hostname
# upon deployment.
# ----------------------------------------------------------------------
      name: mysql
#
# ----------------------------------------------------------------------
# Hard-coded operating system image to use. To pull from user input above,
# enter the following instead.
# image: '${input.osimage}'
# ----------------------------------------------------------------------
      image: Ubuntu
#
# ----------------------------------------------------------------------
# Hard-coded capacity to use. Note that the Cloud Assembly
# Infrastructure must also have AWS, Azure, and vSphere flavors
# such as small, medium, and large mapped.
# ----------------------------------------------------------------------
      flavor: small
#
# ----------------------------------------------------------------------
# Tag the database machine to deploy to the cloud vendor chosen from the
# user input. Tags are case-sensitive, so 'to_lower' forces the tag to
# lowercase to ensure a match with a site's tagging convention. It's
# important if platform input were to contain any upper case characters.
# ----------------------------------------------------------------------
      constraints:
        - tag: '${"env:" + to_lower(input.platform)}'
#
# ----------------------------------------------------------------------
# Also tag the database machine with any free-form tags that were created
# during user input.
# ----------------------------------------------------------------------
      tags: '${input.Mtags}'
#
# ----------------------------------------------------------------------
# Set the database cluster size by referencing the dbenvsize user
# input. Small is one machine, and large defaults to two.
# ----------------------------------------------------------------------
      count: '${input.dbenvsize == "Small" ? 1 : 2}'
#
# ----------------------------------------------------------------------
# Add a variable to connect the machine to a network resource based on
# a property binding to another resource. In this case, it's the
# 'WP_Network' network that gets defined further below.
# ----------------------------------------------------------------------
      networks:
        - name: '${resource.WP_Network.name}'
          network: '${resource.WP_Network.id}'
#
# ----------------------------------------------------------------------
# Enable remote access to the database server. Reference the credentials
```

```
# from the user input.
# -----------------------------------------------------------------------
      remoteAccess:
        authentication: usernamePassword
        username: '${input.username}'
        password: '${input.userpassword}'
#
# -----------------------------------------------------------------------
# You are free to add custom properties, which might be used to initiate
# an 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 extensiblity subscription, for example.
# -----------------------------------------------------------------------
      ABC-Company-ID: 9393
#
# -----------------------------------------------------------------------
# Tag the web server to deploy to the cloud vendor chosen from the
# user input. Tags are case-sensitive, so 'to_lower' forces the tag to
# lowercase to ensure a match with your site's tagging convention. It's
# important if platform input were to contain any upper case characters.
# -----------------------------------------------------------------------
      constraints:
        - tag: '${"env:" + to_lower(input.platform)}'
#
# -----------------------------------------------------------------------
# Add a variable to connect the machine to a network resource based on
# a property binding to another resource. In this case, it's the
# 'WP_Network' network that gets defined further below.
# -----------------------------------------------------------------------
      networks:
        - name: '${resource.WP_Network.name}'
          network: '${resource.WP_Network.id}'
#
# -----------------------------------------------------------------------
# Run OS commands or scripts to further configure the web server,
# with operations such as setting a hostname, generating SSH private keys,
# or installing packages.
# -----------------------------------------------------------------------
      cloudConfig: |
        #cloud-config
        repo_update: true
        repo_upgrade: all
        packages:
          - apache2
          - php
```

```
          - php-mysql
          - libapache2-mod-php
          - php-mcrypt
          - mysql-client
        runcmd:
          - mkdir -p /var/www/html/mywordpresssite && cd /var/www/html && wget https://
wordpress.org/latest.tar.gz && tar -xzf /var/www/html/latest.tar.gz -C /var/www/html/
mywordpresssite --strip-components 1
          - i=0; while [ $i -le 5 ]; do mysql --connect-timeout=3 -h $
{DBTier.networks[0].address} -u root -pmysqlpassword -e "SHOW STATUS;" && break || sleep 15;
i=$((i+1)); done
          - mysql -u root -pmysqlpassword -h ${resource.DBTier.networks[0].address} -e
"create database wordpress_blog;"
          - mv /var/www/html/mywordpresssite/wp-config-sample.php /var/www/html/
mywordpresssite/wp-config.php
          - sed -i -e s/"define('DB_NAME', 'database_name_here');"/"define('DB_NAME',
'wordpress_blog');"/ /var/www/html/mywordpresssite/wp-config.php && sed -i -e
s/"define('DB_USER', 'username_here');"/"define('DB_USER', 'root');"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define('DB_PASSWORD',
'password_here');"/"define('DB_PASSWORD', 'mysqlpassword');"/ /var/www/html/
mywordpresssite/wp-config.php && sed -i -e s/"define('DB_HOST',
'localhost');"/"define('DB_HOST', '${resource.DBTier.networks[0].address}');"/ /var/www/html/
mywordpresssite/wp-config.php
          - service apache2 reload
#
# ---------------------------------------------------------------------
# Create the network that the database and web servers connect to.
# Choose a cloud agnostic network 'type' so that it can deploy to AWS,
# Azure, or vSphere. Then enter its property settings.
# ---------------------------------------------------------------------
  WP_Network:
    type: Cloud.Network
    properties:
#
# ---------------------------------------------------------------------
# Descriptive name for the network. Does not become the network name
# upon deployment.
# ---------------------------------------------------------------------
      name: WP_Network
#
# ---------------------------------------------------------------------
# Set the networkType to an existing network. You could also use a
# constraint tag to target a specific, like-tagged network.
# The other network types are private or public.
# ---------------------------------------------------------------------
      networkType: existing
#
# *********************************************************************
#
# VMware hopes that you found this commented blueprint useful. Note that
# you can also access an API to create blueprints, or query for input
# schema that you intend to request. See the following Swagger
# documentation.
```

```
#
# www.mgmt.cloud.vmware.com/blueprint/api/swagger/swagger-ui.html
#
# ***********************************************************************
```

vRealize Automation Cloud Assembly ブループリントのネットワーク、セキュリティ、およびロード バランサの例

以下のブループリント コードの例では、基本的なネットワーク、セキュリティ、およびロードバランサの構成について説明します。

詳細なネットワークとセキュリティの実装シナリオの例については、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）

すべてのブループリント スキーマ オプションの詳細については、vRealize Automation Resource Type Schema を参照してください。

ネットワーク タイプの詳細については、vRealize Automation Cloud Assembly でのネットワークおよびネットワーク プロファイルの使用を参照してください。

リソース シナリオ	ブループリント コードの例

複数の NIC を含む vSphere マシン	`resources:`
	`  demo-machine:`
	`  type: Cloud.vSphere.Machine`
	`  properties:`
	`  image: ubuntu`
	`  flavor: small`
	`  networks:`
	`  - name: '${network-01.id}'`
	`  deviceIndex: 0`
	`  - name: '${network-02.id}'`
	`  deviceIndex: 1`
	`  Cloud_vSphere_Network_1:`
	`  type: Cloud.vSphere.Network`
	`  properties:`
	`  networkType: existing`
	`  name: network-01`
	`  Cloud_vSphere_Network_2:`
	`  type: Cloud.NSX.Network`
	`  properties:`
	`  networkType: existing`
	`  name: network-02`

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	`resources:`
	`  wf_proxy:`
	`  type: Cloud.Machine`
	`  properties:`
	`  image: ubuntu 16.04`
	`  flavor: small`
	`  constraints:`
	`  - tag: 'platform:vsphere'`
	`  networks:`
	`  - name: '${resource.wf_net.id}'`
	`  assignPublicIpAddress: false`

NSX ネットワーク コンポーネント 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	`Cloud_NSX_Network_1:`
	`  type: Cloud.NSX.Network`
	`  properties:`
	`  networkType: routed`

NSX 論理スイッチに対する送信ネットワークのタグ付け
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。
	`Cloud_NSX_Network_1:`
	`  type: Cloud.NSX.Network`
	`  properties:`
	`  networkType: outbound`
	`  tags:`
	`  - key: app`
	`  value: opencart`

リソース シナリオ	ブループリント 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、`Cloud.SecurityGroup` コンポーネントの `securityGroupType` プロパティに *existing* を入力します。オンデマンドのセキュリティ グループを作成するには、`Cloud.SecurityGroup` コンポーネントの `securityGroupType` プロパティに *new* を入力します。
タグを `Cloud.SecurityGroup` コンポーネントに割り当てて、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 グループはブループリントで使用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	ブループリント 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	<pre>inputs: {}
	resources:
	  mp-existing:
	    type: Cloud.Network
	    properties:
	      name: mp-existing
	      networkType: existing
	  mp-wordpress:
	    type: Cloud.vSphere.Machine
	    properties:
	      name: wordpress
	      count: 2
	      flavor: small
	      image: tiny
	      customizationSpec: Linux
	      networks:
	        - network: '${resource["mp-
	private"].id}'
	  mp-private:
	    type: Cloud.NSX.Network
	    properties:
	      name: mp-private
	      networkType: private
	      constraints:
	        - tag: nsxt
	  mp-wordpress-lb:
	    type: Cloud.LoadBalancer
	    properties:
	      name: wordpress-lb
	      internetFacing: false
	      network: '${resource.mp-existing.id}'
	      instances: '${resource["mp-
	wordpress"].id}'
	      routes:
	        - protocol: HTTP
	          port: '80'
	          instanceProtocol: HTTP
	          instancePort: '80'
	          healthCheckConfiguration:
	            protocol: HTTP
	            port: '80'
	            urlPath: /index.pl
	            intervalSeconds: 60
	            timeoutSeconds: 30
	            unhealthyThreshold: 5
	            healthyThreshold: 2</pre>

ロードバランサを使用する既存のネットワーク	<pre>formatVersion: 1
	inputs:
	  count:
	    type: integer
	    default: 1
	resources:
	  ubuntu-vm:
	    type: Cloud.Machine
	    properties:
	      name: ubuntu
	      flavor: small
	      image: tiny
	      count: '${input.count}'
	      networks:</pre>

リソース シナリオ	ブループリント 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
```

ユーザー名とパスワードでアクセスできる Puppet 対応のブループリント

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ブループリントに 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、ブループリン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 構成管理のブループリントの例と vCenter Puppet 設定ブループリン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 目的のブループリン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	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。 **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。	
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 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 構成管理のブループリントの例

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ブループリントの構成には、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。	<pre>inputs:
	  username:
	    type: string
	    title: Username
	    default: puppet
	  password:
	    type: string
	    title: Password
	    encrypted: true
	    default: VMware@123
	resources:
	  Webserver:
	    type: Cloud.AWS.EC2.Instance
	    properties:
	      flavor: small
	      image: centos
	      cloudConfig: |
	        #cloud-config
	        ssh_pwauth: yes
	        chpasswd:
	          list: |
	            ${input.username}:${input.password}
	          expire: false
	        users:
	          - default
	          - name: ${input.username}
	            lock_passwd: false
	            sudo: ['ALL=(ALL) NOPASSWD:ALL']
	            groups: [wheel, sudo, admin]
	            shell: '/bin/bash'
	            ssh-authorized-keys:
	              - ssh-rsa
	AAAAB3NzaC1yc2EAAAADAQABAAABAQDytVL+Q6/+vGbmkXoRpX
	dmettem@dmettem-m01.vmware.com
	        runcmd:
	          - echo "Defaults:${input.username} !requiretty"
	>> /etc/sudoers.d/${input.username}
	  Puppet_Agent:
	    type: Cloud.Puppet
	    properties:
	      provider: PEOnAWS
	      environment: production
	      role: 'role::linux_webserver'
	      host: '${Webserver.*}'
	      osType: linux
	      username: '${input.username}'
	      password: '${input.password}'
	      useSudo: true</pre>

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。	<pre>inputs:
	  username:
	    type: string
	    title: Username
	    default: puppet
	  password:
	    type: string
	    title: Password
	    encrypted: true
	    default: VMware@123</pre>

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: PEOnAWS
	  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 設定ブループリントの例

Puppet ベースの構成管理で vCenter コンピューティング リソースを管理するようにブループリントを設定できます。そのためのオプションがいくつかあります。

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 6-2.

例...	ブループリント YAML のサンプル

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。	<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: ${input.username}
	            lock_passwd: false
	            sudo: ['ALL=(ALL) NOPASSWD:ALL']
	            groups: [wheel, sudo, admin]
	            shell: '/bin/bash'
	            ssh-authorized-keys:
	              - ssh-rsa
	AAAAB3NzaC1yc2EAAAADAQABAAABAQDytVL+Q6+vGbmkXoRpX
	dmettem@dmettem-m01.vmware.com
	        runcmd:
	          - echo "Defaults:${input.username}</pre>

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。	<pre>inputs:
	  username:
	    type: string
	    title: Username
	    default: puppet</pre>

表 6-2. （続き）

例...	ブループリント YAML のサンプル
	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: ${input.username} lock_passwd: false sudo: ['ALL=(ALL) NOPASSWD:ALL'] groups: [wheel, sudo, admin] shell: '/bin/bash' ssh-authorized-keys: - ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDytVL+Q6+vGbmkXoRpX dmettem@dmettem-m01.vmware.com runcmd: - echo "Defaults:${input.username}
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。	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

表 6-2. （続き）

例...	ブループリント YAML のサンプル

	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}'

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 6-3.

例...	ブループリント YAML のサンプル

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。	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}

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では、完成したブループリントとオープンの仮想化イメージが提供され、VMware Solution Exchange 上で管理されています。`cloud assembly` というタグの付いた Solution Exchange ファイルが、vRealize Automation Cloud Assembly の [マーケットプレイス] タブに表示されます。

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ブループリント 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、ブループリントのエンド ユーザー使用許諾契約書 (EULA) に同意します。次に、ブループリントを 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。ブループリントは [ブループリント] 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 ブループリントをマーケットプレイスで特定します。次に、そのブループリン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ブループリントを 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、ブループリント 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ブループリント コードを更新します。

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。

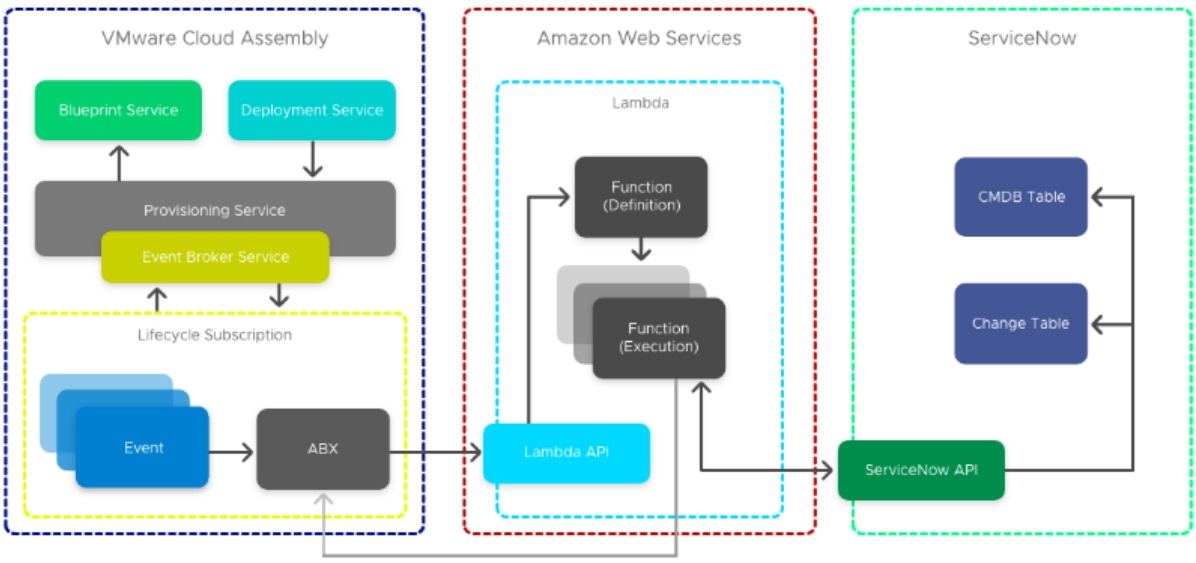
拡張性アクションのサブスクリプション

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。

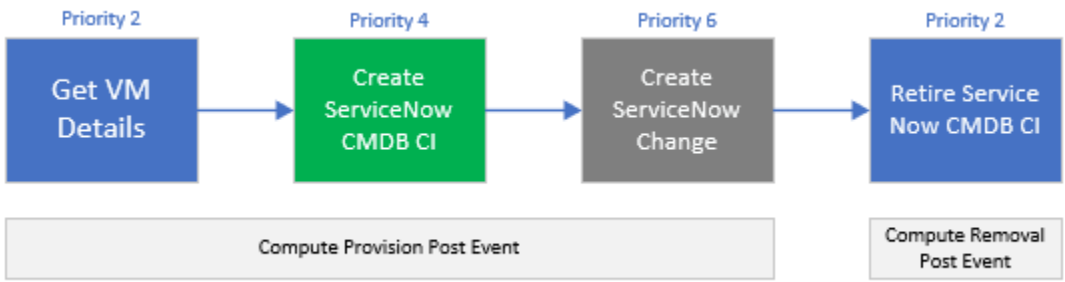


エンタープライズ ユーザーは、コンプライアンスを実現するために、一般的に Cloud Management Platform と IT サービス管理 (ITSM) および構成管理データベース (CMDB) プラットフォームを統合します。この例のとおりに、拡張性アクション スクリプトを使用することにより、vRealize Automation Cloud Assembly を CMDB および ITSM 用の ServiceNow と統合できます。

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ことが必要になる場合があります。

[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、次の条件付きブループリント 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 との統合が必要なブループリントで設定できます。

- Python アプリケーションがインストールされている。

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):
    baseUri = inputs['url']
    casToken = client.get_parameter(Name="casToken",WithDecryption=True)

    url = baseUri + "/iaas/login"
    headers = {"Accept":"application/json","Content-Type":"application/json"}
    payload = {"refreshToken":casToken['Parameter']['Value']}

    results = requests.post(url,json=payload,headers=headers)

    bearer = "Bearer "
    bearer = bearer + results.json()["token"]

    deploymentId = inputs['deploymentId']
    resourceId = inputs['resourceIds'][0]

    print("deploymentId: "+ deploymentId)
    print("resourceId:" + resourceId)

    machineUri = baseUri + "/iaas/machines/" + resourceId
    headers = {"Accept":"application/json","Content-Type":"application/json",
"Authorization":bearer }
    resultMachine = requests.get(machineUri,headers=headers)
    print("machine: " + resultMachine.text)

    print( "serviceNowCPUCount: "+ json.loads(resultMachine.text)["customProperties"]
["cpuCount"] )
    print( "serviceNowMemoryInMB: "+ json.loads(resultMachine.text)["customProperties"]
["memoryInMB"] )
```

```
    #update customProperties
    outputs = {}
    outputs['customProperties'] = inputs['customProperties']
    outputs['customProperties']['serviceNowCPUCount'] = int(json.loads(resultMachine.text)
["customProperties"]["cpuCount"])
    outputs['customProperties']['serviceNowMemoryInMB'] = json.loads(resultMachine.text)
["customProperties"]["memoryInMB"]
    return outputs
```

3 CMDB 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。

(例：resourceNames = ["DB_VM"] and target = world)。

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。

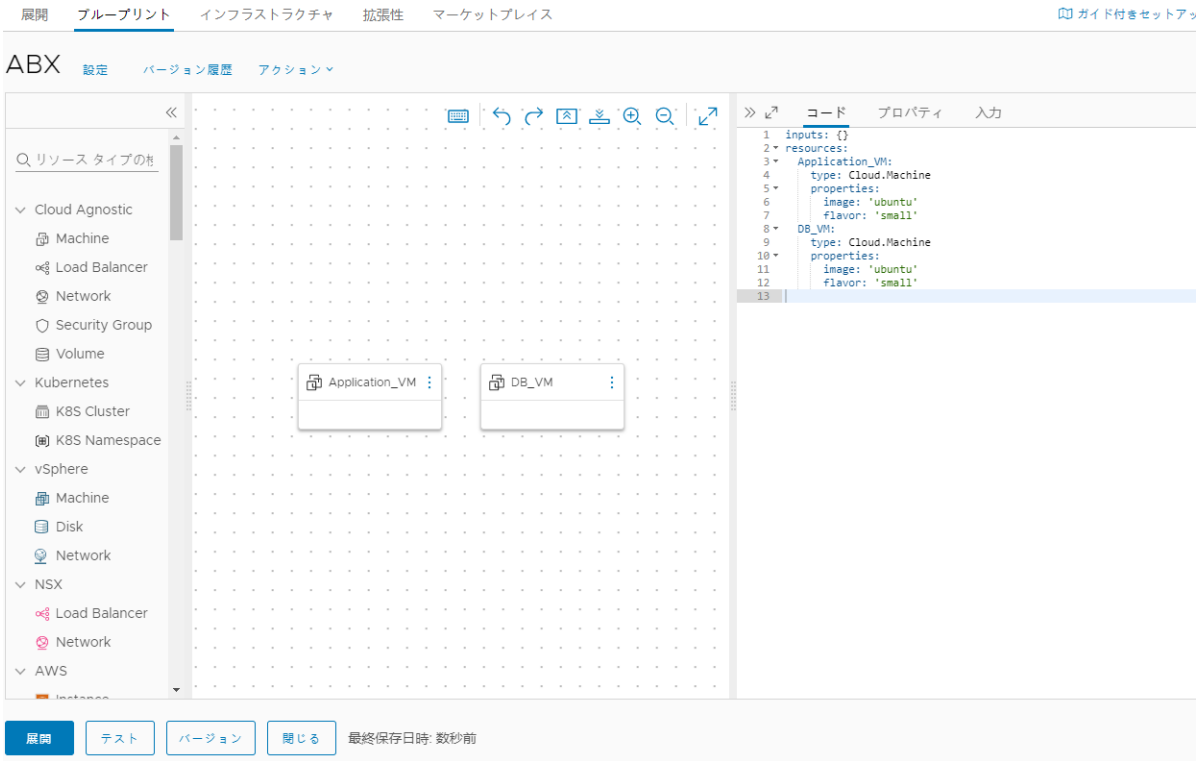
8 [新しいサブスクリプション] をクリックします。

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、サブスクリプション スキーマ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。
実行可能な項目タイプ	実行可能な拡張性アクションのタイプを選択します。
実行可能 ID	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、[作成] をクリックします。

11 2 台の仮想マシン（アプリケーション仮想マシンと DB 仮想マシン）が含まれている [新規ブループリント] を作成します。



12 仮想マシンを展開するには、[展開] をクリックします。

13 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。

14 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、イベント ブローカ サービス (EBS) によって行われる特定のイベントで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。ワークフローと同様に、拡張性アクション スクリプトは、拡張性サブスクリプションに含まれているイベントが発生したときにトリガされます。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。また、これらは vRealize Orchestrator サーバを使用してオンプレミスでホストされているワークフローとは異なり、クラウド上にホストされます。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、軽量な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は、Node.js と Python の両方のランタイム環境をサポートし、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 パッケージの作成または 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： 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。

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。

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される 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、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使用できる、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。

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。

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。

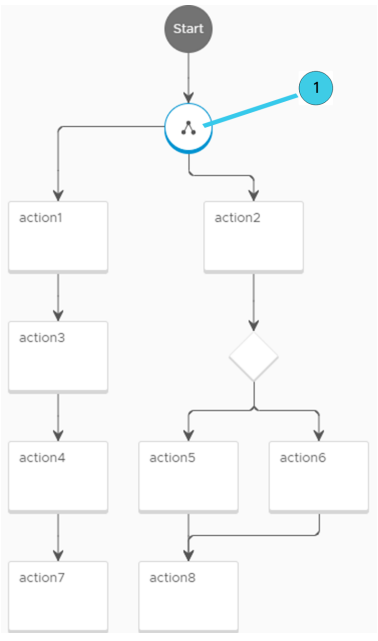
```
version: "1"
flow:
 flow_start:
  next: action1
action1:
  action: <action_name>
  next: action2
action2:
  action: <action_name>
  next: flow_end
```

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>
```



1 フォーク要素

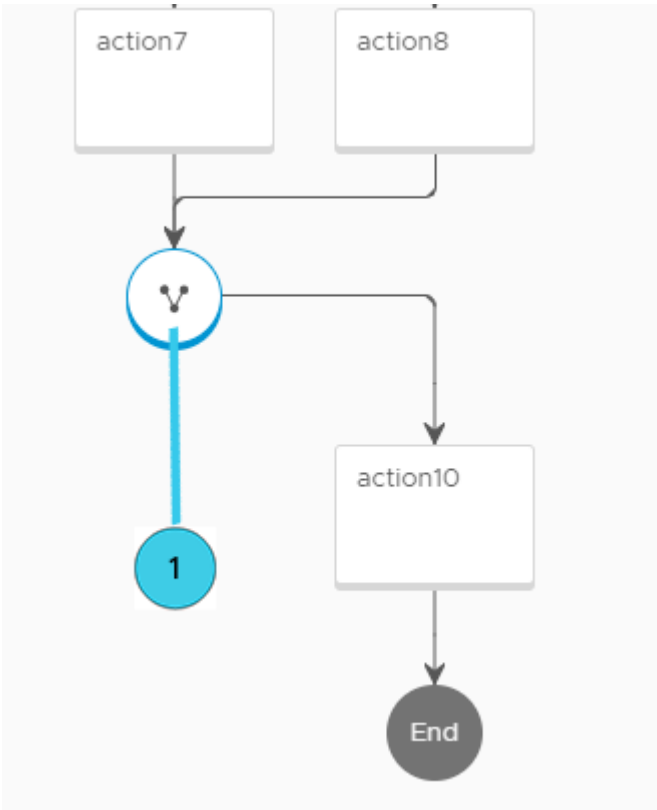
注： アクションを next: アクションとして割り当てると、以前のアクションにループ バックできます。たとえば、アクション フローを終了する next: flow_end の代わりに next: action1 と入力して action1 を再実行し、アクションのシーケンスを再起動します。

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。



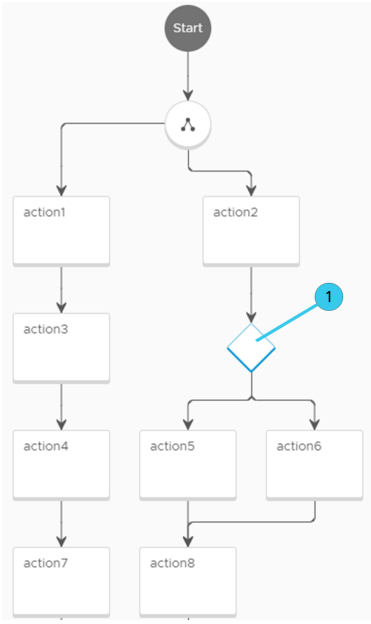
1 結合要素

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

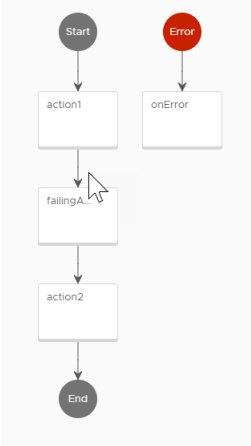
注： アクションを next: アクションとして割り当てると、以前のアクションにループ バックできます。たとえば、アクション フローを終了する next: flow_end の代わりに next: action1 と入力して action1 を再実行し、アクションのシーケンスを再起動します。

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。

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、Active Directory に追加することができます。

サブスクリプション スキーマで、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
実行可能項目	vRealize Orchestrator の実行可能なタイプを選択します。
Runnable ID	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ために、[作成] をクリックします。

5 ブループリントを作成するか、既存のブループリントを展開して、サブスクリプションを割り当て、有効にします。

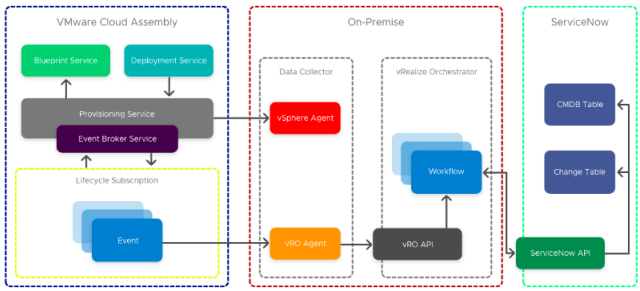
次のステップ

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。この例のとおりに、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 CAS 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 CAS API トークンを保存します。

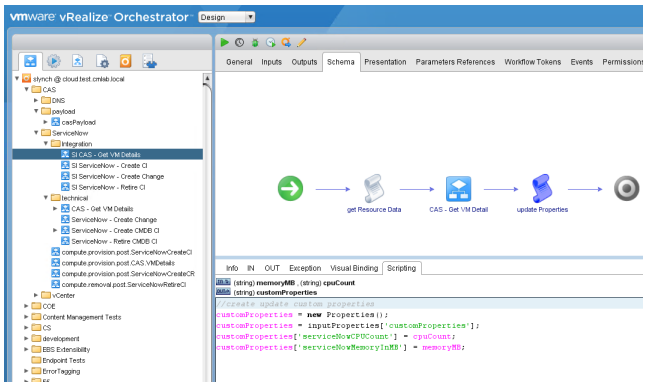
   注： CAS API トークンには有効期限があります。
3. 指定したスクリプト要素を使用して、vRealize Orchestrator にワークフローを作成します。このスクリプトは、REST ホストを参照して特定します。また、トークンのオプション パラメータを使用する REST アクションを標準化します。これは追加の認証ヘッダーとして追加されます。

```
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。

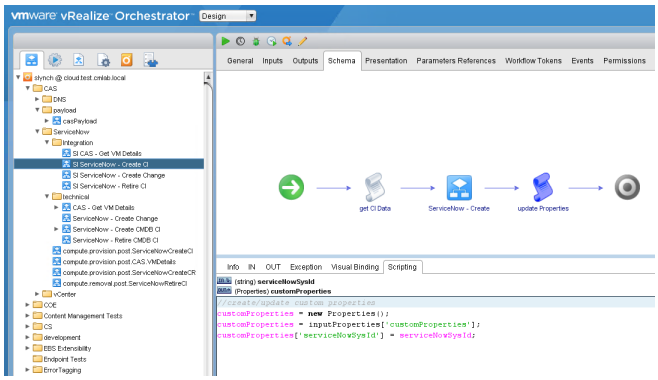


4 ServiceNow CMDB CI の作成スクリプトをワークフローに追加します。この要素は、構成アイテムを使用して ServiceNow REST ホストを特定し、`cmdb_ci_vmware_instance` テーブルの REST 操作を作成します。さらに、POST データのワークフロー入力に基づいてコンテンツ オブジェクトの文字列を作成して、返された `sys_id` を出力します。

```
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 Cloud Assembly には、既存のオンプレミス ワークフローをサブスクリプションにインポートし、リンクするための vRealize Orchestrator の統合が含まれています。これらのワークフローは、vRealize Orchestrator サーバで保持されます。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する必要があります。

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、特定のトピック スキーマに基づきます。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] タブには、サブスクリプションで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] を使用します。

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。	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。	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。 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。	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ブループリン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 内で拡張性や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 6-4. 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、登録した利用者に送るサービスです。
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のイベントをトリガ後に実行できる、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 6-5. 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
`Blueprint.configuration`	いいえ	ブループリントの作成または削除などのブループリント構成イベントが発生したときに発行され、そのようなイベントを外部システムに通知するのに役立つ場合があります。
`Blueprint.version.configuration`	いいえ	新しいブループリン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	`resourcenames` および `hostselections` の割り当て前に発行されるイベント。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で発行されるイベント。
`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るイベント。
`Compute provision`	はい	リソースがハイパーバイザーでプロビジョニングされる前に発生するイベント。 **注：** 割り当てられた IP アドレスは変更できます。
`Compute removal`	はい	リソースが削除される前に発行されるイベント。
`Compute reservation`	はい	予約時に発行されるイベント。 **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。
`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	ブループリントまたはカタログ申請の展開後に発行されます。

表 6-5. 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 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	ブループリン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 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	関連するイベントをログに記録します。
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。
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。
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るイベント。 **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。
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 スキーマ

イベント トピックを追加した後、イベント スキーマが表示されます。このスキーマでは、イベントのペイロード構造、つまり `inputProperties` を定義します。

拡張性イベント ログ

[拡張性イベント] タブには、お使いの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] を使用します。また、一度に 1 つまたは複数のプロパティを使用して、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 をクリックします。

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/アクションがトリガされるようにします。これは、イベントとアクションがいつトリガされるかを制御する場合に便利です。

ヒント: [トピックのイベントをフィルタリング] テキスト ボックスで、Alt + Space キー（Windows の場合）または Option + Space キー（Mac の場合）を使用して、フィルタ オプションを表示します。

前提条件

- クラウド管理者のユーザー ロール

- vRealize Orchestrator ワークフローを使用する場合：
  - 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する場合：
  - 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 [新しい統合] 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。

6 （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設定します。

7 [実行可能な項目] をクリックして、ドロップダウン メニューから [vRO ワークフロー] または [ABX アクション] を選択します。

8 サブスクリプションで実行するワークフローまたは拡張性アクションを選択します。

9 （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。

10 [作成] をクリックしてサブスクリプションを保存します。

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、すべてのサブスクライバに通知されます。

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ブループリン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。

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 Cloud Assembly 展開の管理

7

vRealize Automation Cloud Assembly ブループリント開発者は、[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、ブループリン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 でアクティブな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 でアクティブな展開を監視する方法

vRealize Automation Cloud Assembly ブループリントを展開した後、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。

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もあります。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生じる場合があります。

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。

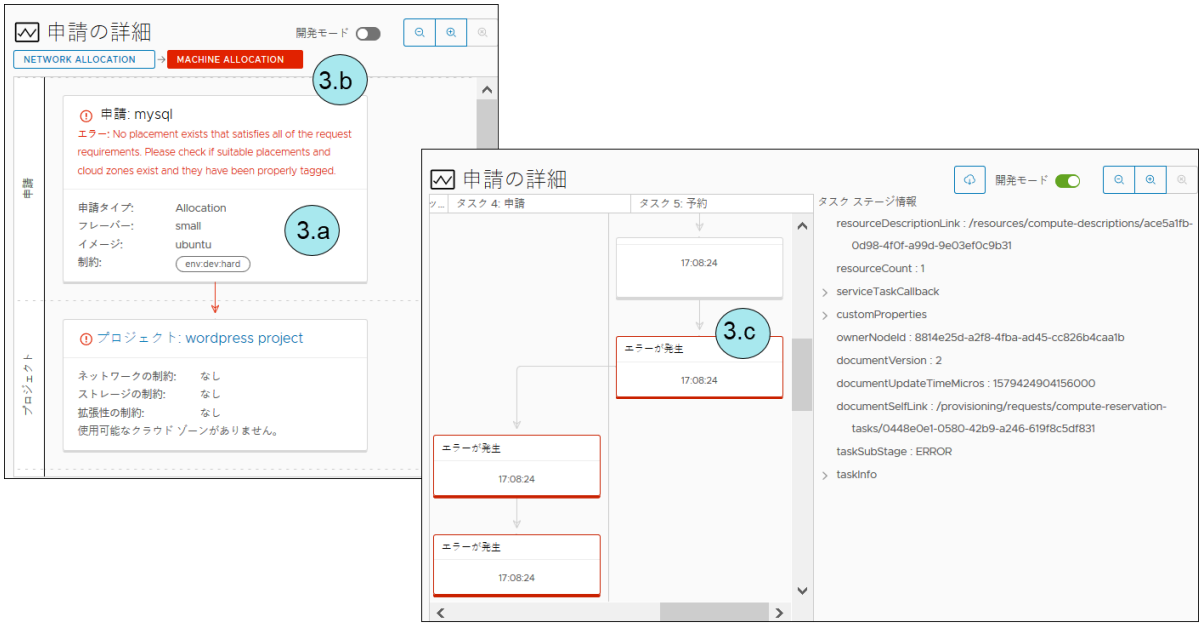
a イベント ツリーを確認して、プロビジョニング プロセスが失敗した場所を調べます。このツリーは、展開を変更する場合に便利ですが、変更は失敗します。

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 のブループリントの場合は、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。



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、ブループリントを再展開します。

エラーは、ブループリン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設定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、ブループリントが展開されると、[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

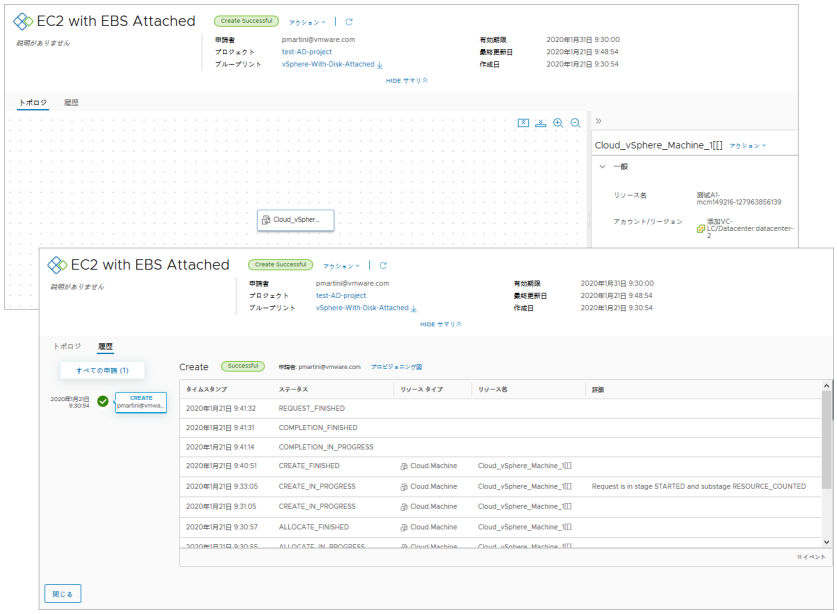
1 [展開] をクリックして展開を見つけます。

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。

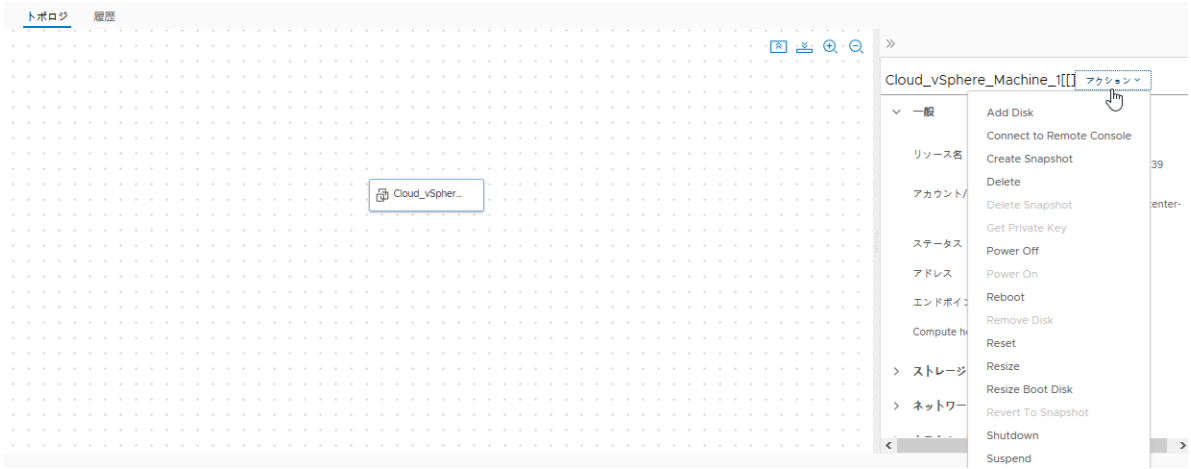
[履歴] タブには、すべてのプロビジョニング イベントと、申請された項目が展開された後に実行するアクションに関連するイベントがすべて表示されます。プロビジョニングのプロセスに問題がある場合は、[履歴] タブのイベントを使用して、障害のトラブルシューティングを行うことができます。

[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。



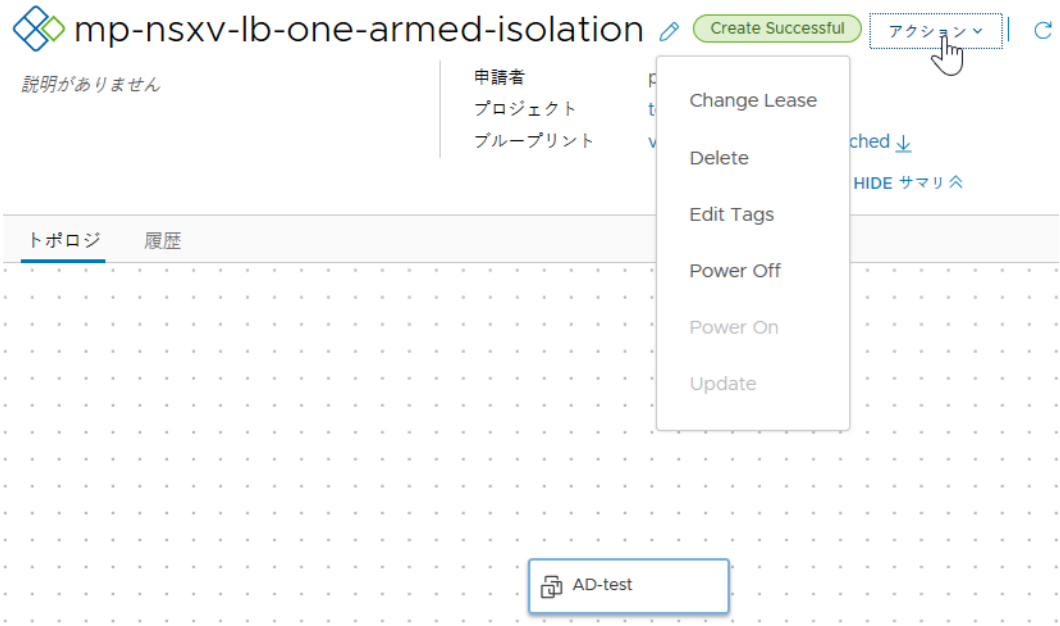
3 現在の設定では展開にコストがかかりすぎると判断し、コンポーネントのサイズを変更する場合は、[トポロジ] 画面でコンポーネントを選択してから、[アクション] - [サイズ変更] の順に選択します。

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

ブループリントを展開した後、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。

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	■ 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	■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere	リースの有効期限の日時を変更します。 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。 リース ポリシーは vRealize Automation Service Broker で設定されます。

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
リモート コンソールへの接続	マシン	■ VMware vSphere	選択したマシンでリモート セッションを開きます。 正常に接続できるように、次の要件を確認します。 ■ 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	■ Google Cloud Platform ■ VMware vSphere	仮想マシンのスナップショットを作成します。 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。
削除	展開	■ Amazon Web Service ■ Google Cloud Platform ■ Microsoft Azure ■ VMware vSphere	展開を削除します。 すべてのリソースが削除され、再要求されます。 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。
	マシンとロード バランサ	■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere	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。
スナップショットの削除	マシン	■ 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	■ 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。
パワーオフ	展開	■ 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	■ 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	■ 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	■ 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。

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
再起動	マシン	■ Amazon Web Service ■ VMware vSphere	仮想マシンのゲスト OS を再起動します。 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。
再構成	ロード バランサ	■ Amazon Web Service ■ Microsoft Azure ■ Google Cloud Platform ■ VMware vSphere	ロード バランサ プロトコル、ポート、健全性の設定、およびメンバー プールの設定を変更します。
ディスクの削除	マシン	■ 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	■ 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	■ 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	■ 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	■ 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	■ 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	■ Google Cloud Platform ■ VMware vSphere	マシンの以前のスナップショットに戻ります。 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。
Puppet タスクの実行	管理対象リソース	■ Puppet Enterprise	環境内のマシンで選択したタスクを実行します。 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。
シャットダウン	マシン	■ 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	■ 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	■ Amazon Web Service ■ Microsoft Azure ■ VMware vSphere	入力パラメータに基づいて展開を変更します。
タグの更新	マシンとディスク	■ 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。