OpenShift Container Platform 4.1

インストール

OpenShift Container Platform 4.1 クラスターのインストール

Last Updated: 2020-05-15

OpenShift Container Platform 4.1 インストール

OpenShift Container Platform 4.1 クラスターのインストール

法律上の通知

概要

本書では、サポートされているすべてのインフラストラクチャーのタイプに OpenShift Container Platform 4.1 クラスターをインストールし、アンインストールする方法について説明します。

目次

第1章 AWS へのインストール

1.1. AWS アカウントの設定

OpenShift Container Platform をインストールする前に、Amazon Web Services (AWS) アカウントを設定する必要があります。

1.1.1. Route53 の設定

OpenShift Container Platform をインストールするには、使用する Amazon Web Services (AWS) アカウントに、Route53 サービスの専用のパブリックホストゾーンが必要になります。このゾーンはドメインに対する権威を持っている必要があります。 Route53 サービスは、クラスターへの外部接続のためのクラスターの DNS 解決および名前検索を提供します。

手順

1. ドメイン、またはサブドメイン、およびレジストラーを特定します。既存のドメインおよびレジストラーを移行するか、または AWS または他のソースから新規のものを取得できます。

注記

AWS で新規ドメインを購入する場合、関連する DNS の変更が伝播するのに時間がかかります。AWS 経由でドメインを購入する方法についての詳細は、AWS ドキュメントの「 Registering Domain Names Using Amazon Route 53 」を参照してください。

2. 既存のドメインおよびレジストラーを使用している場合、その DNS を AWS に移行します。AWS ドキュメントの「Making Amazon Route 53 the DNS Service for an Existing Domain 」を参照してください。

3. ドメインまたはサブドメインのパブリックホストゾーンを作成します。AWS ドキュメントの「Creating a Public Hosted Zone」を参照してください。
openshiftcorp.com などのルートドメインや、 **clusters.openshiftcorp.com** などのサブドメインを使用します。

4. ホストゾーンレコードから新規の権威ネームサーバーを抽出します。AWS ドキュメントの「Getting the Name Servers for a Public Hosted Zone」を参照してください。

5. ドメインが使用する AWS Route53 ネームサーバーのレジストラーレコードを更新します。たとえば、別のアカウントを使ってドメインを Route53 サービスに登録している場合は、AWS ドキュメントの「Adding or Changing Name Servers or Glue Records」のトピックを参照してください。

6. サブドメインを使用する場合は、所属する会社の手順に従ってその委任レコードを親ドメインに追加します。

1.1.2. AWS アカウントの制限

OpenShift Container Platform クラスターは数多くの Amazon Web Services (AWS) コンポーネントを使用し、デフォルトのサービス制限は、OpenShift Container Platform クラスターをインストールする機能に影響を与えます。特定のクラスター設定を使用し、クラスターを特定の AWS リージョンにデプロイするか、またはアカウントを使って複数のクラスターを実行する場合、AWS アカウントの追加リソースを要求することが必要になる場合があります。

以下の表は、OpenShift Container Platform クラスターのインストールおよび実行機能に影響を与える可能性のある AWS コンポーネントの制限を要約しています。

コンポーネント	デフォルトで利用できるクラスターの数	デフォルトの AWS の制限	説明
インスタンスの制限	変動あり。	変動あり。	デフォルトで、各クラスターは以下のインスタンスを作成します。 ● 1 つのブートストラップマシン。これはインストール後に削除されます。 ● 3 つのマスターノード ● 3 つのワーカーノード これらのインスタンスタイプの数は、新規アカウントのデフォルト制限内の値です。追加のワーカーノードをデプロイし、自動スケーリングを有効にし、大規模なワークロードをデプロイするか、または異なるインスタンスタイプを使用するには、アカウントの制限を見直し、クラスターが必要なマシンをデプロイできることを確認します。 ほとんどのリージョンでは、ブートストラップおよびワーカーマシンは **m4.large** マシンを使用し、マスターマシンは **m4.xlarge** インスタンスを使用します。これらのインスタンスタイプをサポートしないすべてのリージョンを含む一部のリージョンでは、**m5.large** および **m5.xlarge** インスタンスが代わりに使用されます。
Elastic IP (EIP)	0 - 1	アカウントごとに 5 つの EIP	クラスターを高可用性設定でプロビジョニングするために、インストールプログラムはそれぞれのリージョン内のアベイラビリティーゾーンにパブリックおよびプライベートのサブネットを作成します。各プライベートサブネットには NAT ゲートウェイが必要であり、各 NAT ゲートウェイには別個の Elastic IP が必要です。AWS リージョンマップを確認して、各リージョンにあるアベイラビリティーゾーンの数を判別します。デフォルトの高可用性を利用するには、少なくとも 3 つのアベイラビリティーゾーンがあるリージョンにクラスターをインストールします。アベイラビリティーゾーンが 6 つ以上あるリージョンにクラスターをインストールするには、EIP 制限を引き上げる必要があります。  **重要** **us-east-1** リージョンを使用するには、アカウントの EIP 制限を引き上げる必要があります。

コンポーネント	デフォルトで利用できるクラスターの数	デフォルトの AWS の制限	説明
Virtual Private Cloud (VPC)	5	リージョンごとに 5 つの VPC	各クラスターは独自の VPC を作成します。
Elastic Load Balancing (ELB/NLB)	3	リージョンごとに 20	デフォルトで、各クラスターは、マスター API サーバーの内部および外部のネットワークロードバランサーおよびルーターの単一の Classic Elastic Load Balancer を作成します。追加の Kubernetes LoadBalancer Service オブジェクトをデプロイすると、追加のロードバランサーが作成されます。
NAT ゲートウェイ	5	アベイラビリティゾーンごとに 5 つ	クラスターは各アベイラビリティーゾーンに 1 つの NAT ゲートウェイをデプロイします。
Elastic Network Interface (ENI)	12 以上	リージョンごとに 350	デフォルトのインストールは 21 の ENI を作成し、リージョンの各アベイラビリティーゾーンに 1 つの ENI を作成します。たとえば、**us-east-1** リージョンには 6 つのアベイラビリティーゾーンが含まれるため、そのゾーンにデプロイされるクラスターは 27 の ENI を使用します。AWS リージョンマップを確認して、各リージョンにあるアベイラビリティーゾーンの数を判別します。 追加の ENI が、クラスターの使用およびデプロイされたワークロード別に作成される追加のマシンおよび Elastic Load Balancer について作成されます。
VPC ゲートウェイ	20	アカウントごとに 20	AWS アカウントは、S3 へのアクセスのために VPC ゲートウェイを使用します。各クラスターは、S3 アクセス用の単一の VPC ゲートウェイを作成します。
S3 バケット	99	アカウントごとに 100 バケット	インストールプロセスでは 1 つの一時的なバケットを作成し、各クラスターのレジストリーコンポーネントがバケットを作成するため、AWS アカウントごとに 99 の OpenShift Container Platform クラスターのみを作成できます。
セキュリティーグループ	250	アカウントごとに 2,500	各クラスターは、10 の個別のセキュリティーグループを作成します。

1.1.3. 必要な AWS パーミッション

AdministratorAccess ポリシーを、作成する IAM ユーザーに割り当てる場合、そのユーザーには必要なパーミッションすべてを付与します。OpenShift Container Platform クラスターをデプロイするために、IAM ユーザーに以下のパーミッションが必要になります。

インストールに必要な EC2 パーミッション

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSecurityGroup**
- **ec2:CreateSubnet**
- **ec2:CreateTags**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:CreateVolume**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**

- **ec2:DescribeInstances**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:ReplaceRouteTableAssociation**
- **ec2:DescribeNetworkInterfaces**
- **ec2:ModifyNetworkInterfaceAttribute**

インストールに必要な Elasticloadbalancing パーミッション

- **elasticloadbalancing:AddTags**

- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

インストールに必要な IAM パーミッション

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**

- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**

インストールに必要な Route53 パーミッション

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:CreateHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

インストールに必要な S3 パーミッション

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**

- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

クラスター Operator が必要とする S3 パーミッション

- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:DeleteObject**

クラスターのアンインストールに必要な追加のすべてのパーミッション

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteNetworkInterface**

- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSnapshot**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSubnet**
- **ec2:DeleteVolume**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DeregisterImage**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **elasticloadbalancing:DescribeTargetGroups**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **route53:DeleteHostedZone**
- **tag:GetResources**

1.1.4. IAM ユーザーの作成

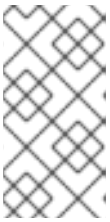
各 Amazon Web Services (AWS) アカウントには、アカウントの作成に使用するメールアドレスに基づく root ユーザーアカウントが含まれます。これは高度な権限が付与されたアカウントであり、初期アカウントにのみ使用し、請求設定また初期のユーザーセットの作成およびアカウントのセキュリティー保護のために使用することが推奨されています。

OpenShift Container Platform をインストールする前に、セカンダリー IAM 管理ユーザーを作成します。AWS ドキュメントの「Creating an IAM User in Your AWS Account 」手順を実行する際に、以下のオプションを設定します。

手順

1. IAM ユーザー名を指定し、**Programmatic access** を選択します。
2. **AdministratorAccess** ポリシーを割り当て、アカウントにクラスターを作成するために十分なパーミッションがあることを確認します。このポリシーはクラスターに対し、各 OpenShift

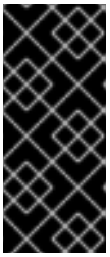
Container Platform コンポーネントに認証情報を付与する機能を提供します。クラスターはコンポーネントに対し、それらが必要とする認証情報のみを付与します。



注記

必要なすべての AWS パーミッションを付与し、これをユーザーに割り当てるポリシーを作成することは可能ですが、これは優先されるオプションではありません。クラスターには追加の認証情報を個別コンポーネントに付与する機能がないため、同じ認証情報がすべてのコンポーネントによって使用されます。

3. オプション: タグを割り当て、メタデータをユーザーに追加します。
4. 指定したユーザー名に **AdministratorAccess** ポリシーが付与されていることを確認します。
5. アクセスキー ID およびシークレットアクセスキーの値を記録します。ローカルマシンをインストールプログラムを実行するように設定する際にこれらの値を使用する必要があります。



重要

クラスターのデプロイ時に、マルチファクター認証デバイスの使用中に生成した一時的なセッショントークンを使用して AWS に対する認証を行うことはできません。クラスターは継続的に現行の AWS 認証情報を使用して、クラスターの有効期間全体にわたって AWS リソースを作成するため、キーをベースとした有効期間の長い認証情報を使用する必要があります。

1.1.5. サポートされている AWS リージョン

OpenShift Container Platform クラスターを以下のリージョンにデプロイできます。

- ap-northeast-1 (Tokyo)
- ap-northeast-2 (Seoul)
- ap-south-1 (Mumbai)
- ap-southeast-1 (Singapore)
- ap-southeast-2 (Sydney)
- ca-central-1 (Central)
- eu-central-1 (Frankfurt)
- eu-west-1 (Ireland)
- eu-west-2 (London)
- eu-west-3 (Paris)
- sa-east-1 (São Paulo)
- us-east-1 (N. Virginia)
- us-east-2 (Ohio)
- us-west-1 (N. California)

- us-west-2 (Oregon)

次のステップ

- OpenShift Container Platform クラスターをインストールします。カスタマイズされたクラスターのインストール、またはデフォルトのオプションでクラスターのクイックインストールを実行できます。

1.2. クラスターの AWS へのクイックインストール

OpenShift Container Platform バージョン 4.1 では、デフォルトの設定オプションを使用してクラスターを Amazon Web Services (AWS) にインストールできます。

前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- AWS アカウントを設定してクラスターをホストします。

重要

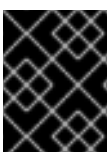
AWS プロファイルがご使用のコンピューターに保存されている場合、マルチファクター認証デバイスを使用中に生成した一時的なセッショントークンを使用することはできません。クラスターは継続的に現行の AWS 認証情報を使用して、クラスターの有効期間全体にわたって AWS リソースを作成するため、キーをベースとした有効期間の長い認証情報を使用する必要があります。適切なキーを生成するには、AWS ドキュメントの「Managing Access Keys for IAM Users」を参照してください。キーは、インストールプログラムの実行時に指定できます。

- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

1.2.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。



重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得
- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

1.2.2. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> (1)
```

(1) ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> (1)

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) ~/**.ssh/id_rsa** などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

1.2.3. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。

重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

1.2.4. クラスターのデプロイ

互換性のあるクラウドに OpenShift Container Platform をインストールできます。

重要

インストールプログラムは、初期インストール時に 1 回だけ実行することができます。

前提条件

- クラスターをホストするクラウドプラットフォームでアカウントを設定します。
- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得します。

手順

1. インストールプログラムを実行します。

```
$ ./openshift-install create cluster --dir=<installation_directory> \ 1
    --log-level info 2
```

1 **<installation_directory>** には、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

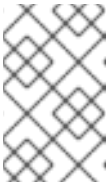
2 異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。

重要

空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

プロンプト時に値を指定します。

a. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。



注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

b. ターゲットに設定するプラットフォームとして AWS を選択します。

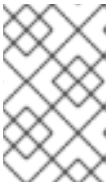
c. Amazon Web Services (AWS) プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。

d. クラスターのデプロイ先とする AWS リージョンを選択します。

e. クラスターに設定した Route53 サービスのベースドメインを選択します。

f. クラスターの記述名を入力します。

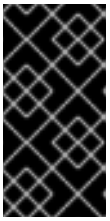
g. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。



注記

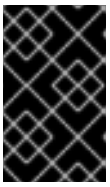
ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。



重要

インストールプログラム、またはインストールプログラムが作成するファイルを削除することはできません。これらはいずれもクラスターを削除するために必要になります。

2. オプション: クラスターのインストールに使用した IAM アカウントから **AdministratorAccess** ポリシーを削除するか、または無効にします。

1.2.5. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。

重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。
2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。

注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

1.2.6. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1

$ oc whoami
system:admin
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

1.3. カスタマイズによる AWS へのクラスターのインストール

OpenShift Container Platform バージョン 4.1 では、インストールプログラムが Amazon Web Services (AWS) にプロビジョニングするインフラストラクチャーにカスタマイズされたクラスターをインストールすることができます。インストールをカスタマイズするには、クラスターをインストールする前に、**install-config.yaml** ファイルで一部のパラメーターを変更します。

前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- AWS アカウントを設定してクラスターをホストします。

重要

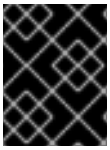
AWS プロファイルがご使用のコンピューターに保存されている場合、マルチファクター認証デバイスを使用中に生成した一時的なセッショントークンを使用することはできません。クラスターは継続的に現行の AWS 認証情報を使用して、クラスターの有効期間全体にわたって AWS リソースを作成するため、有効期間の長い認証情報を使用する必要があります。適切なキーを生成するには、AWS ドキュメントの「Managing Access Keys for IAM Users」を参照してください。キーは、インストールプログラムの実行時に指定できます。

- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

1.3.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。



重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得
- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

1.3.2. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> 1
```

1 ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 ~/**.ssh/id_rsa** などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

1.3.3. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。

2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。

重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

1.3.4. インストール設定ファイルの作成

互換性のあるクラウド上で OpenShift Container Platform のインストールをカスタマイズできます。

前提条件

- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。

手順

1. **install-config.yaml** ファイルを作成します。

   a. 次のコマンドを実行します。

```
$ ./openshift-install create install-config --dir=<installation_directory> 1
```

1 **<installation_directory>** には、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

重要

空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

b. プロンプト時に、クラウドの設定の詳細情報を指定します。

i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

ii. ターゲットに設定するプラットフォームとして **AWS** を選択します。

iii. Amazon Web Services (AWS) プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。

iv. クラスターのデプロイ先とする AWS リージョンを選択します。

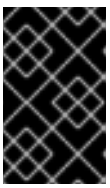
v. クラスターに設定した Route53 サービスのベースドメインを選択します。

vi. クラスターの記述名を入力します。

vii. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。

2. **install-config.yaml** ファイルを変更します。利用可能なパラメーターについての詳細については、「インストール設定パラメーター」セクションおよび Go ドキュメントを参照できます。

3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするために使用できるようにします。



重要

install-config.yaml ファイルはインストールプロセス時に使用されます。このファイルを再利用する必要がある場合は、この段階でこれをバックアップしてください。

1.3.4.1. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、Amazon Web Services (AWS) アカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要

なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。

注記

インストール後には、これらのパラメーターを変更することはできません。

表1.1 必須パラメーター

パラメーター	説明	値
baseDomain	クラウドプロバイダーのベースドメイン。この値は、OpenShift Container Platform クラスターコンポーネントへのルートを作成するために使用されます。クラスターの完全な DNS 名は、**baseDomain** と **<metadata.name>.<baseDomain>** 形式を使用する **metadata.name** パラメーターの値の組み合わせです。	**example.com** などの完全修飾ドメインまたはサブドメイン名。
controlPlane.platform	コントロールプレーンマシンをホストするためのクラウドプロバイダー。このパラメーターの値は **compute.platform** パラメーターの値に一致する必要があります。	**aws**
compute.platform	ワーカーマシンをホストするためのクラウドプロバイダー。このパラメーターの値は **controlPlane.platform** パラメーターの値に一致する必要があります。	**aws**
metadata.name	クラスターの名前。	**dev** などの大文字または小文字を含む文字列。
platform.aws.region	クラスターをデプロイするリージョン。	**us-east-1** などの有効な AWS リージョン。

パラメーター	説明	値
pullSecret	Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレット。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

表1.2 オプションの AWS プラットフォームパラメーター

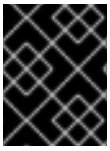
パラメーター	説明	値
sshKey	クラスターマシンにアクセスするために使用する SSH キー。  **注記** インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。	**ssh-agent** プロセスに追加した、有効なローカルのパブリック SSH キー。
compute.hyperthreading	コンピュートマシンで同時スレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。  **重要** 同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。	**Enabled** または **Disabled**

パラメーター	説明	値
compute.platform.aws.rootVolume.iops	ルートボリュームに予約される 1 秒あたりの入出力操作 (IOPS)。	整数 (例: **4000**)。
compute.platform.aws.rootVolume.size	ルートボリュームのサイズ (GiB)。	整数 (例: **500**)。
compute.platform.aws.rootVolume.type	ルートボリュームのインスタンスタイプ。	有効な AWS EBS インスタンスタイプ (例: **io1**)。
compute.platform.aws.type	コンピュートマシンの EC2 インスタンスタイプ。	有効な AWS インスタンスタイプ (例: **c5.9xlarge**)。
compute.platform.aws.zones	インストールプログラムがコンピュート MachinePool のマシンを作成するアベイラビリティーゾーン。	**YAML** シーケンスの us-east-1c などの有効な AWS アベイラビリティーゾーンの一覧。
compute.aws.region	インストールプログラムがコンピュートリソースを作成する AWS リージョン。	有効な AWS リージョン (例: **us-east-1**)。
compute.replicas	プロビジョニングするコンピュートマシン（ワーカーマシンとしても知られる）の数。	**2** 以上の正の整数。デフォルト値は **3** です。
controlPlane.hyperthreading	コントロールプレーンマシンで同時スレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。  **重要** 同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。	**Enabled** または **Disabled**
controlPlane.platform.aws.type	コントロールプレーンマシンの EC2 インスタンスタイプ。	有効な AWS インスタンスタイプ (例: **c5.9xlarge**)。
controlPlane.platform.aws.zones	インストールプログラムがコントロールプレーン MachinePool のマシンを作成するアベイラビリティーゾーン。	**YAML** シーケンスの us-east-1c などの有効な AWS アベイラビリティーゾーンの一覧。

パラメーター	説明	値
controlPlane.aws.region	インストールプログラムがコントロールプレーンのリソースを作成する AWS リージョン。	有効な AWS リージョン (例: **us-east-1**)。
controlPlane.replicas	プロビジョニングするコントロールプレーンマシンの数。	**3** 以上の正の整数。デフォルト値は **3** です。
platform.aws.userTags	インストールプログラムが、作成するすべてのリソースに対するタグとして追加するキーと値のマップ。	**<key>: <value>** 形式のキー値ペアなどの有効な YAML マップ。AWS タグについての詳細は、AWS ドキュメントの「Tagging Your Amazon EC2 Resources」を参照してください。

1.3.4.2. AWS のカスタマイズされた install-config.yaml ファイルのサンプル

install-config.yaml ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。



重要

このサンプルの YAML ファイルは参照用にのみ提供されます。インストールプログラムを使用して **install-config.yaml** ファイルを取得し、これを変更する必要があります。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1
      type: m5.xlarge (5)
  replicas: 3
compute: (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (8)
      type: c5.4xlarge
```

```
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 9
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 10
    userTags:
      adminContact: jdoe
      costCenter: 7536
pullSecret: '{"auths": ...}' 11
sshKey: ssh-ed25519 AAAA... 12
```

1 9 10 11 必須。インストールプログラムはこの値の入力を求めるプロンプトを出します。

2 6 これらのパラメーターおよび値を指定しない場合、インストールプログラムはデフォルトの値を指定します。

3 7 **controlPlane** セクションは単一マッピングですが、コンピュートセクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

4 5 同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。同時マルチスレッドを無効にする場合は、これをすべてのクラスターマシンで無効にする必要があります。

重要

同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。同時マルチスレッドを無効にする場合は、マシンに対して **m4.2xlarge** または **m5.2xlarge** などの大規模なインスタンスタイプを使用します。

8 大規模なクラスターの場合などに etcd の高速のストレージを設定するには、ストレージタイプを **io1** として設定し、**iops** を **2000** に設定します。

12 クラスター内のマシンにアクセスするために使用する **sshKey** 値をオプションで指定できます。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

1.3.5. クラスターのデプロイ

互換性のあるクラウドに OpenShift Container Platform をインストールできます。

重要

インストールプログラムは、初期インストール時に 1 回だけ実行することができます。

前提条件

- クラスターをホストするクラウドプラットフォームでアカウントを設定します。
- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得します。

手順

1. インストールプログラムを実行します。

```
$ ./openshift-install create cluster --dir=<installation_directory> \ 1
    --log-level info 2
```

1. **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
2. **<installation_directory>** については、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

   異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。

重要

空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

プロンプト時に値を指定します。

a. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

b. ターゲットに設定するプラットフォームとして AWS を選択します。

c. Amazon Web Services (AWS) プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。

d. クラスターのデプロイ先とする AWS リージョンを選択します。

e. クラスターに設定した Route53 サービスのベースドメインを選択します。

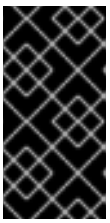
f. クラスターの記述名を入力します。

g. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。

注記

ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

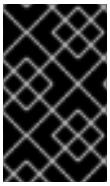
重要

インストールプログラム、またはインストールプログラムが作成するファイルを削除することはできません。これらはいずれもクラスターを削除するために必要になります。

2. オプション: クラスターのインストールに使用した IAM アカウントから **AdministratorAccess** ポリシーを削除するか、または無効にします。

1.3.6. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。



重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。
2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。

注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

1.3.7. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)

$ oc whoami
system:admin
```

(1) **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

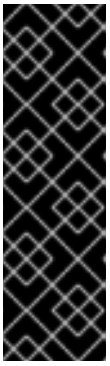
1.4. ネットワークのカスタマイズによる AWS へのクラスターのインストール

OpenShift Container Platform バージョン 4.1 では、カスタマイズされた設定オプションでクラスターを Amazon Web Services (AWS) にインストールできます。ネットワーク設定をカスタマイズすることにより、クラスターは環境内の既存の IP アドレスの割り当てと共存でき、既存の MTU および VXLAN 設定と統合できます。

大半のネットワーク設定パラメーターはインストール時に設定する必要があり、実行中のクラスターで変更できるのは **kubeProxy** 設定パラメーターのみになります。

前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- AWS アカウントを設定してクラスターをホストします。



重要

AWS プロファイルがご使用のコンピューターに保存されている場合、マルチファクター認証デバイスを使用中に生成した一時的なセッショントークンを使用することはできません。クラスターは継続的に現行の AWS 認証情報を使用して、クラスターの有効期間全体にわたって AWS リソースを作成するため、キーをベースとした有効期間の長い認証情報を使用する必要があります。適切なキーを生成するには、AWS ドキュメントの「Managing Access Keys for IAM Users」を参照してください。キーは、インストールプログラムの実行時に指定できます。

- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

1.4.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。

重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得
- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

1.4.2. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> (1)
```

(1) ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

   このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> (1)

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

(1) ~/**.ssh/id_rsa** などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

1.4.3. インストールプログラムの取得

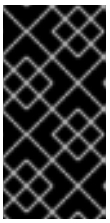
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。



重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

1.4.4. インストール設定ファイルの作成

互換性のあるクラウド上で OpenShift Container Platform のインストールをカスタマイズできます。

前提条件

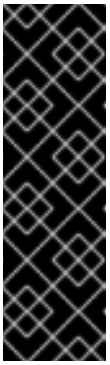
- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。

手順

1. **install-config.yaml** ファイルを作成します。
   a. 次のコマンドを実行します。

```
$ ./openshift-install create install-config --dir=<installation_directory> 1
```

   1 **<installation_directory>** には、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。



重要

空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

   b. プロンプト時に、クラウドの設定の詳細情報を指定します。
      i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

      ii. ターゲットに設定するプラットフォームとして **AWS** を選択します。
      iii. Amazon Web Services (AWS) プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。
      iv. クラスターのデプロイ先とする AWS リージョンを選択します。
      v. クラスターに設定した Route53 サービスのベースドメインを選択します。
      vi. クラスターの記述名を入力します。
      vii. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。

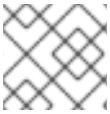
2. **install-config.yaml** ファイルを変更します。利用可能なパラメーターについての詳細については、「インストール設定パラメーター」セクションおよび Go ドキュメントを参照できます。
3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするために使用できるようにします。

重要

install-config.yaml ファイルはインストールプロセス時に使用されます。このファイルを再利用する必要がある場合は、この段階でこれをバックアップしてください。

1.4.4.1. インストール設定パラメーター

OpenShift Container Platform クラスターをデプロイする前に、Amazon Web Services (AWS) アカウントを記述し、クラスターのプラットフォームをオプションでカスタマイズするためにパラメーターの値を指定します。**install-config.yaml** インストール設定ファイルを作成する際に、コマンドラインで必要なパラメーターの値を指定します。クラスターをカスタマイズする場合、**install-config.yaml** ファイルを変更して、プラットフォームについての詳細情報を指定できます。



注記

インストール後には、これらのパラメーターを変更することはできません。

表1.3 必須パラメーター

パラメーター	説明	値
baseDomain	クラウドプロバイダーのベースドメイン。この値は、OpenShift Container Platform クラスターコンポーネントへのルートを作成するために使用されます。クラスターの完全な DNS 名は、**baseDomain** と **<metadata.name>.<baseDomain>** 形式を使用する **metadata.name** パラメーターの値の組み合わせです。	**example.com** などの完全修飾ドメインまたはサブドメイン名。
controlPlane.platform	コントロールプレーンマシンをホストするためのクラウドプロバイダー。このパラメーターの値は **compute.platform** パラメーターの値に一致する必要があります。	**aws**
compute.platform	ワーカーマシンをホストするためのクラウドプロバイダー。このパラメーターの値は **controlPlane.platform** パラメーターの値に一致する必要があります。	**aws**
metadata.name	クラスターの名前。	**dev** などの大文字または小文字を含む文字列。

パラメーター	説明	値
platform.aws.region	クラスターをデプロイするリージョン。	**us-east-1** などの有効な AWS リージョン。
pullSecret	Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレット。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

表1.4 オプションの AWS プラットフォームパラメーター

パラメーター	説明	値
sshKey	クラスターマシンにアクセスするために使用する SSH キー。 **注記** インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。	**ssh-agent** プロセスに追加した、有効なローカルのパブリック SSH キー。

パラメーター	説明	値
compute.hyperthreading	コンピュートマシンで同時スレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。  **重要** 同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。	**Enabled** または **Disabled**
compute.platform.aws.rootVolume.iops	ルートボリュームに予約される 1 秒あたりの入出力操作 (IOPS)。	整数 (例: **4000**)。
compute.platform.aws.rootVolume.size	ルートボリュームのサイズ (GiB)。	整数 (例: **500**)。
compute.platform.aws.rootVolume.type	ルートボリュームのインスタンスタイプ。	有効な AWS EBS インスタンスタイプ (例: **io1**)。
compute.platform.aws.type	コンピュートマシンの EC2 インスタンスタイプ。	有効な AWS インスタンスタイプ (例: **c5.9xlarge**)。
compute.platform.aws.zones	インストールプログラムがコンピュート MachinePool のマシンを作成するアベイラビリティーゾーン。	**YAML** シーケンスの us-east-1c などの有効な AWS アベイラビリティーゾーンの一覧。
compute.aws.region	インストールプログラムがコンピュートリソースを作成する AWS リージョン。	有効な AWS リージョン (例: **us-east-1**)。
compute.replicas	プロビジョニングするコンピュートマシン（ワーカーマシンとしても知られる）の数。	**2** 以上の正の整数。デフォルト値は **3** です。

パラメーター	説明	値
controlPlane.hyperthreading	コントロールプレーンマシンで同時スレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。  **重要** 同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。	**Enabled** または **Disabled**
controlPlane.platform.aws.type	コントロールプレーンマシンの EC2 インスタンスタイプ。	有効な AWS インスタンスタイプ (例: **c5.9xlarge**)。
controlPlane.platform.aws.zones	インストールプログラムがコントロールプレーン MachinePool のマシンを作成するアベイラビリティーゾーン。	**YAML** シーケンスの us-east-1c などの有効な AWS アベイラビリティーゾーンの一覧。
controlPlane.aws.region	インストールプログラムがコントロールプレーンのリソースを作成する AWS リージョン。	有効な AWS リージョン (例: **us-east-1**)。
controlPlane.replicas	プロビジョニングするコントロールプレーンマシンの数。	**3** 以上の正の整数。デフォルト値は **3** です。
platform.aws.userTags	インストールプログラムが、作成するすべてのリソースに対するタグとして追加するキーと値のマップ。	**<key>: <value>** 形式のキー値ペアなどの有効な YAML マップ。AWS タグについての詳細は、AWS ドキュメントの「Tagging Your Amazon EC2 Resources」を参照してください。

1.4.4.2. ネットワーク設定パラメーター

クラスターのネットワーク設定パラメーターは **install-config.yaml** 設定ファイルで変更できます。以下の表では、これらのパラメーターについて説明しています。

注記

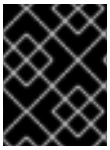
インストール後には、これらのパラメーターを変更することはできません。

表1.5 必要なネットワークパラメーター

パラメーター	説明	値
networking.networkType	デプロイするネットワークプラグイン。**OpenShiftSDN** のみが OpenShift Container Platform 4.1 でサポートされているプラグインです。	**OpenShiftSDN**
networking.clusterNetwork.cidr	Pod IP アドレスの割り当てに使用する IP アドレスのブロック。**OpenShiftSDN** ネットワークプラグインは複数のクラスターネットワークをサポートします。複数のクラスターネットワークのアドレスブロックには重複が許可されません。予想されるワークロードに適したサイズのアドレスプールを選択してください。	CIDR 形式の IP アドレスの割り当て。デフォルト値は **10.128.0.0/14** です。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネットプレフィックスの長さ。たとえば、**hostPrefix** が **23** に設定される場合、各ノードに指定の **cidr** から **/23** サブネットが割り当てられます (510 (2^(32 - 23) - 2) Pod IP アドレスが許可されます)。	サブネットプレフィックス。デフォルト値は **23** です。
networking.serviceNetwork	サービスの IP アドレスのブロック。**OpenShiftSDN** は 1 つの **serviceNetwork** ブロックのみを許可します。このアドレスブロックは他のネットワークブロックと重複できません。	CIDR 形式の IP アドレスの割り当て。デフォルト値は **172.30.0.0/16** です。
networking.machineCIDR	クラスターのインストール中に OpenShift Container Platform インストールプログラムによって使用される IP アドレスのブロック。このアドレスブロックは他のネットワークブロックと重複できません。	CIDR 形式の IP アドレスの割り当て。デフォルト値は **10.0.0.0/16** です。

1.4.4.3. AWS のカスタマイズされた install-config.yaml ファイルのサンプル

install-config.yaml ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。



重要

このサンプルの YAML ファイルは参照用にのみ提供されます。インストールプログラムを使用して **install-config.yaml** ファイルを取得し、これを変更する必要があります。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
```

```
        iops: 4000
        size: 500
        type: io1
      type: m5.xlarge (5)
  replicas: 3
compute: (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (8)
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster (9)
networking:
networking: (10)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineCIDR: 10.0.0.0/16
  networkType: OpenShiftSDN
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 (11)
    userTags:
      adminContact: jdoe
      costCenter: 7536
pullSecret: '{"auths": ...}' (12)
sshKey: ssh-ed25519 AAAA... (13)
```

(1) (9) (11) (12) 必須。インストールプログラムはこの値の入力を求めるプロンプトを出します。

(2) (6) (10) これらのパラメーターおよび値を指定しない場合、インストールプログラムはデフォルトの値を指定します。

(3) (7) **controlPlane** セクションは単一マッピングですが、コンピュートセクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

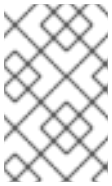
(4) (5) 同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。同時マルチスレッドを無効にする場合は、これをすべてのクラスターマシンで無効にする必要があります。

重要

同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。同時マルチスレッドを無効にする場合は、マシンに対して **m4.2xlarge** または **m5.2xlarge** などの大規模なインスタンスタイプを使用します。

8 大規模なクラスターの場合などに etcd の高速のストレージを設定するには、ストレージタイプを **io1** として設定し、**iops** を **2000** に設定します。

13 クラスター内のマシンにアクセスするために使用する **sshKey** 値をオプションで指定できます。



注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

1.4.5. 高度なネットワーク設定パラメーターの変更

高度なネットワーク設定パラメーターは、クラスターのインストール前にのみ変更することができます。高度な設定のカスタマイズにより、クラスターを既存のネットワーク環境に統合させることができます。 これを実行するには、MTU または VXLAN ポートを指定し、kube-proxy 設定のカスタマイズを許可し、**openshiftSDNConfig** パラメーターに異なる **mode** を指定します。

重要

OpenShift Container Platform マニフェストファイルの直接の変更はサポートされていません。

前提条件

- **install-config.yaml** ファイルを生成し、これに対する変更を完了します。

手順

1. 以下のコマンドを使用してマニフェストを作成します。

```
$ ./openshift-install create manifests --dir=<installation_directory> 1
```

1 **<installation_directory>** については、クラスターの **install-config.yaml** ファイルが含まれるディレクトリーの名前を指定します。

2. **cluster-network-03-config.yml** という名前のファイルを **<installation_directory>/manifests/** ディレクトリーに作成します。

```
$ touch <installation_directory>/manifests/cluster-network-03-config.yml 1
```

1 **<installation_directory>** については、クラスターの **manifests/** ディレクトリーが含まれるディレクトリー名を指定します。

ファイルの作成後は、以下のように 3 つのネットワーク設定ファイルが **manifests/** ディレクトリーに置かれます。

```
$ ls <installation_directory>/manifests/cluster-network-*
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-network-03-config.yml
```

3. エディターで **cluster-network-03-config.yml** ファイルを開き、必要な Operator 設定を記述する CR を入力します。

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec: 1
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork:
  - 172.30.0.0/16
  defaultNetwork:
    type: OpenShiftSDN
    openshiftSDNConfig:
      mode: NetworkPolicy
      mtu: 1450
      vxlanPort: 4789
```

1 **spec** フィールドのパラメーターは例です。CR にネットワーク Operator の設定を指定します。

ネットワーク Operator は CR にパラメーターのデフォルト値を提供するため、**Network.operator.openshift.io** CR には変更が必要なパラメーターのみを指定する必要があります。

4. **cluster-network-03-config.yml** ファイルを保存し、テキストエディターを終了します。

5. オプション: **manifests/cluster-network-03-config.yml** ファイルをバックアップします。インストールプログラムは、クラスターの作成時に **manifests/** ディレクトリーを削除します。

1.4.6. クラスターネットワーク Operator のカスタムリソース (CR、Custom Resource)

Network.operator.openshift.io カスタムリソース (CR) のクラスターネットワーク設定は、Cluster Network Operator (CNO) の設定内容を保存します。

以下の CR は、CNO のデフォルト設定を表示し、設定可能なパラメーターと有効なパラメーターの値の両方について説明しています。

Cluster Network Operator CR

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
```

```
spec:
  clusterNetwork: (1)
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork: (2)
  - 172.30.0.0/16
  defaultNetwork:
    type: OpenShiftSDN (3)
    openshiftSDNConfig: (4)
      mode: NetworkPolicy (5)
      mtu: 1450 (6)
      vxlanPort: 4789 (7)
  kubeProxyConfig: (8)
    iptablesSyncPeriod: 30s (9)
    proxyArguments:
      iptables-min-sync-period: (10)
      - 30s
```

(1) (2) (3) **install-config.yaml** ファイルに指定されます。

(4) OpenShift Container Platform SDN 設定の一部を上書きする必要がある場合にのみ指定します。

(5) **OpenShiftSDN** の分離モードを設定します。許可される値は **Multitenant**、**Subnet**、または **NetworkPolicy** です。デフォルト値は **NetworkPolicy** です。

(6) VXLAN オーバーレイネットワークの MTU。この値は通常は自動的に設定されますが、クラスターにあるノードすべてが同じ MTU を使用しない場合、これを最小のノード MTU 値よりも 50 小さくする必要があります。

(7) すべての VXLAN パケットに使用するポート。デフォルト値は **4789** です。別の VXLAN ネットワークの一部である既存ノードと共に仮想化環境で実行している場合は、これを変更する必要がある可能性があります。たとえば、OpenShift SDN オーバーレイを VMware NSX-T 上で実行する場合は、両方の SDN が同じデフォルトの VXLAN ポート番号を使用するため、VXLAN の別のポートを選択する必要があります。

Amazon Web Services (AWS) では、VXLAN にポート **9000** とポート **9999** 間の代替ポートを選択できます。

(8) このオブジェクトのパラメーターは、**kube-proxy** 設定を指定します。パラメーターの値を指定しない場合 、ネットワーク Operator は表示されるデフォルトのパラメーター値を適用します。

(9) **iptables** ルールの更新期間。デフォルト値は **30s** です。有効なサフィックスには、**s**、**m**、および **h**などが含まれ、これらについては、Go time package ドキュメントで説明されています。

(10) **iptables** ルールを更新する前の最小期間。このパラメーターにより、更新の頻度が高くなり過ぎないようにできます。有効なサフィックスには、**s**、**m**、および **h** が含まれ、これらについては、Go time package で説明されています。

1.4.7. クラスターのデプロイ

互換性のあるクラウドに OpenShift Container Platform をインストールできます。

重要

インストールプログラムは、初期インストール時に1回だけ実行することができます。

前提条件

- クラスターをホストするクラウドプラットフォームでアカウントを設定します。
- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得します。

手順

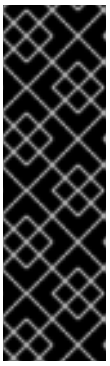
1. インストールプログラムを実行します。

```
$ ./openshift-install create cluster --dir=<installation_directory> \ 1
    --log-level info 2
```

1 **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。

2 **<installation_directory>** については、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。



重要

空のディレクトリーを指定します。ブートストラップX.509証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

プロンプト時に値を指定します。

a. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

b. ターゲットに設定するプラットフォームとして AWS を選択します。

c. Amazon Web Services (AWS) プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。

d. クラスターのデプロイ先とする AWS リージョンを選択します。

e. クラスターに設定した Route53 サービスのベースドメインを選択します。

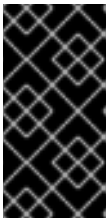
f. クラスターの記述名を入力します。

g. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。

注記

ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

クラスターのデプロイメントが完了すると、Web コンソールへのリンクや **kubeadmin** ユーザーの認証情報を含む、クラスターにアクセスするための指示がターミナルに表示されます。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

重要

インストールプログラム、またはインストールプログラムが作成するファイルを削除することはできません。これらはいずれもクラスターを削除するために必要になります。

2. オプション: クラスターのインストールに使用した IAM アカウントから **AdministratorAccess** ポリシーを削除するか、または無効にします。

1.4.8. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。

重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。

2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。

注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

1.4.9. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)

$ oc whoami
system:admin
```

(1) **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

1.5. AWS でのクラスターのアンインストール

Amazon Web Services (AWS) にデプロイしたクラスターは削除することができます。

1.5.1. AWS からのクラスターの削除

Amazon Web Services (AWS) にインストールしたクラスターを削除することができます。

前提条件

- クラスターをデプロイするために使用したインストールプログラムのコピーがあること。
- クラスター作成時にインストールプログラムが生成したファイルがあること。

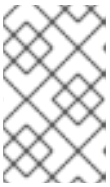
手順

1. クラスターをインストールするために使用したコンピューターから、以下のコマンドを実行します。

```
$ ./openshift-install destroy cluster \
--dir=<installation_directory> --log-level=info 1 2
```

1 **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

2 異なる詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。



注記

クラスターのクラスター定義ファイルが含まれるディレクトリーを指定する必要があります。クラスターを削除するには、インストールプログラムでこのディレクトリーにある **metadata.json** ファイルが必要になります。

2. オプション: **<installation_directory>** ディレクトリーおよび OpenShift Container Platform インストールプログラムを削除します。

第2章 ユーザーによってプロビジョニングされた AWS へのインストール

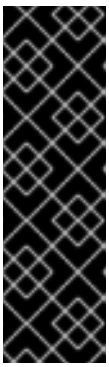
2.1. CLOUDFORMATION テンプレートを使用したクラスターの AWS へのインストール

OpenShift Container Platform version 4.1 では、独自に提供するインフラストラクチャーを使用して、クラスターを Amazon Web Services (AWS) にインストールできます。

このインフラストラクチャーを作成する 1 つの方法として、提供される CloudFormation テンプレートを使用できます。テンプレートを変更してインフラストラクチャーをカスタマイズしたり、それらに含まれる情報を使用し、所属する会社のポリシーに基づいて AWS オブジェクトを作成したりできます。

前提条件

- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- AWS アカウントを設定してクラスターをホストします。



重要

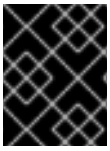
AWS プロファイルがご使用のコンピューターに保存されている場合、マルチファクター認証デバイスを使用中に生成した一時的なセッショントークンを使用することはできません。クラスターは継続的に現行の AWS 認証情報を使用して、クラスターの有効期間全体にわたって AWS リソースを作成するため、キーをベースとした有効期間の長い認証情報を使用する必要があります。適切なキーを生成するには、AWS ドキュメントの「Managing Access Keys for IAM Users」を参照してください。キーは、インストールプログラムの実行時に指定できます。

- AWS CLI をダウンロードし、これをコンピューターにインストールします。AWS ドキュメントの「Install the AWS CLI Using the Bundled Installer (Linux, macOS, or Unix) 」を参照してください。
- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

2.1.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。



重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得
- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

2.1.2. 必要な AWS インフラストラクチャーコンポーネント

OpenShift Container Platform を Amazon Web Services (AWS) のユーザーによってプロビジョニングされるインフラストラクチャーにインストールするには、マシンとサポートするインフラストラクチャーの両方を手動で作成する必要があります。

各種プラットフォームの統合テストの詳細については、「 OpenShift Container Platform 4.x Tested Integrations」のページを参照してください。

提供される CloudFormation テンプレートを使用してこのインフラストラクチャーを作成でき、コンポーネントを手動で作成するか、またはクラスターの要件を満たす既存のインフラストラクチャーを再利用できます。コンポーネントの相互関係についての詳細は、CloudFormation テンプレートを参照してください。

2.1.2.1. クラスターマシン

以下のマシンには **AWS::EC2::Instance** オブジェクトが必要になります。

- ブートストラップマシン。このマシンはインストール時に必要ですが、クラスターのデプロイ後に除去することができます。
- 3 つ以上のコントロールプレーンマシン。コントロールプレーンマシンは MachineSet によって制御されません。
- コンピュートマシン。インストール時に 2 つ以上のコンピュートマシン (ワーカーマシンとしても知られる) を作成する必要があります。これらのマシンは MachineSet によって制御されません。

クラスターのマシンには、以下のインスタンスタイプを使用できます。

マシンの有効なインスタンスタイプ

m4 インスタンスが **eu-west-3** などのリージョンで利用可能ではない場合、**m5** タイプを代わりに使用します。

インスタンスタイプ	ブートストラップ	コントロールプレーン	コンピュート
i3.large	x		
m4.large または **m5.large**			x
m4.xlarge または **m5.xlarge**		x	x
m4.2xlarge		x	x
m4.4xlarge		x	x
m4.8xlarge		x	x
m4.10xlarge		x	x
m4.16xlarge		x	x
c4.large			x
c4.xlarge			x
c4.2xlarge		x	x
c4.4xlarge		x	x
c4.8xlarge		x	x
r4.large			x
r4.xlarge		x	x
r4.2xlarge		x	x
r4.4xlarge		x	x
r4.8xlarge		x	x
r4.16xlarge		x	x

これらのインスタンスタイプの仕様に対応する他のインスタンスタイプを使用できる場合もあります。

2.1.2.2. 証明書署名要求の管理

ユーザーがプロビジョニングするインフラストラクチャーを使用する場合、クラスターの自動マシン管

理へのアクセスは制限されるため、インストール後にクラスターの証明書署名要求 (CSR) のメカニズムを提供する必要があります。**kube-controller-manager** は kubelet クライアント CSR のみを承認します。**machine-approver** は、kubelet 認証情報を使用して要求される提供証明書の有効性を保証できません。適切なマシンがこの要求を発行したかどうかを確認できないためです。kubelet 提供証明書の要求の有効性を検証し、それらを承認する方法を判別し、実装する必要があります。

2.1.2.3. 他のインフラストラクチャーコンポーネント

- 1 つの VPC
- DNS エントリー
- ロードバランサー（classic または network）およびリスナー
- 1 つの Route53 ゾーン
- セキュリティーグループ
- IAM ロール
- S3 バケット

必要な VPC コンポーネント

お使いのマシンとの通信を可能にする適切な VPC およびサブネットを指定する必要があります。

コンポーネント	AWS タイプ	説明
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	使用するクラスターのパブリック VPC を指定する必要があります。VPC には、各サブネットのルートテーブルを参照するエンドポイントが必要です。
パブリックサブネット	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	VPC には 1 から 3 のアベイラビリティーゾーンのパブリックサブネットが必要であり、それらを適切な Ingress ルールに関連付ける必要があります。
インターネットゲートウェイ	• **AWS::EC2::InternetGateway** • **AWS::EC2::VPCGatewayAttachment** • **AWS::EC2::RouteTable** • **AWS::EC2::Route** • **PublicSubnetRouteTableAssociation** • **AWS::EC2::NatGateway** • **AWS::EC2::EIP**	VPC に割り当てられたパブリックルートを持つパブリックインターネットゲートウェイが必要です。それぞれのパブリックサブネットにもルートに割り当てられ、NAT ゲートウェイおよび EIP アドレスがなければなりません。

<table>
<tr><th>コンポーネント</th><th>AWS タイプ</th><th colspan="2">説明</th></tr>
<tr><td>ネットワークアクセス制御</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">VPC が以下のポートにアクセスできるようにする必要があります。</td></tr>
<tr><td></td><td></td><td>ポート</td><td>理由</td></tr>
<tr><td></td><td></td><td>80</td><td>インバウンド HTTP トラフィック</td></tr>
<tr><td></td><td></td><td>443</td><td>インバウンド HTTPS トラフィック</td></tr>
<tr><td></td><td></td><td>22</td><td>インバウンド SSH トラフィック</td></tr>
<tr><td></td><td></td><td>1024 - 65535</td><td>インバウンド一時 (ephemeral) トラフィック</td></tr>
<tr><td></td><td></td><td>0 - 65535</td><td>アウトバウンド一時 (ephemeral) トラフィック</td></tr>
<tr><td>プライベートサブネット</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">VPC にはプライベートサブネットを使用できます。提供される CloudFormation テンプレートは 1 から 3 アベイラビリティーゾーンのプライベートサブネットを作成できます。プライベートサブネットを使用できる場合は、それらの適切なルートおよびテーブルを指定する必要があります。</td></tr>
</table>

必要な DNS および負荷分散コンポーネント

DNS およびロードバランサー設定では、パブリックホストゾーンを使用する必要があり、クラスターのインフラストラクチャーをプロビジョニングする場合にインストールプログラムが使用するものと同様のプライベートホストゾーンを使用できます。ロードバランサーに解決する DNS エントリーを作成する必要があります。**api.<cluster_name>.<domain>** のエントリーは外部ロードバランサーを参照し、**api-int.<cluster_name>.<domain>** のエントリーは内部ロードバランサーを参照する必要があります。

またクラスターには、Kubernetes API とその拡張に必要なポート 6443、および新規マシンの Ignition 設定ファイルに必要なポート 22623 のロードバランサーおよびリスナーが必要です。ターゲットはマスターノードになります。ポート 6443 はクラスター外のクライアントとクラスター内のノードからもアクセスできる必要があります。ポート 22623 はクラスター内のノードからアクセスできる必要があります。

コンポーネント	AWS タイプ	説明
DNS	**AWS::Route53::HostedZone**	内部 DNS のホストゾーン。
etcd レコードセット	**AWS::Route53::RecordSet**	コントロールプレーンマシンの etcd の登録レコード。
パブリックロードバランサー	**AWS::ElasticLoadBalancingV2::LoadBalancer**	パブリックサブネットのロードバランサー。
外部 API サーバーレコード	**AWS::Route53::RecordSetGroup**	外部 API サーバーのエイリアスレコード。
外部リスナー	**AWS::ElasticLoadBalancingV2::Listener**	外部ロードバランサー用のポート 6443 のリスナー。
外部ターゲットグループ	**AWS::ElasticLoadBalancingV2::TargetGroup**	外部ロードバランサーのターゲットグループ。
プライベートロードバランサー	**AWS::ElasticLoadBalancingV2::LoadBalancer**	プライベートサブネットのロードバランサー。
内部 API サーバーレコード	**AWS::Route53::RecordSetGroup**	内部 API サーバーのエイリアスレコード。
内部リスナー	**AWS::ElasticLoadBalancingV2::Listener**	内部ロードバランサー用のポート 22623 のリスナー。
内部ターゲットグループ	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部ロードバランサーのターゲットグループ。

コンポーネント	AWS タイプ	説明
内部リスナー	**AWS::ElasticLoadBalancingV2::Listener**	内部ロードバランサーのポート 6443 のリスナー。
内部ターゲットグループ	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部ロードバランサーのターゲットグループ。

セキュリティーグループ

コントロールプレーンおよびワーカーマシンには、以下のポートへのアクセスが必要です。

グループ	タイプ	IP プロトコル	ポート範囲
MasterSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
		tcp	**6443**
		tcp	**22623**
WorkerSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**
BootstrapSecurityGroup	**AWS::EC2::SecurityGroup**	**tcp**	**22**
		tcp	**19531**

コントロールプレーンの Ingress

コントロールプレーンマシンには、以下の Ingress グループが必要です。それぞれの Ingress グループは **AWS::EC2::SecurityGroupIngress** リソースになります。

Ingress グループ	説明	IP プロトコル	ポート範囲
MasterIngressEtcd	etcd	**tcp**	**2379- 2380**
MasterIngressVxlan	Vxlan パケット	**udp**	**4789**

Ingress グループ	説明	IP プロトコル	ポート範囲
MasterIngressWorkerVxlan	Vxlan パケット	**udp**	**4789**
MasterIngressInternal	内部クラスター通信および Kubernetes プロキシーメトリクス	**tcp**	**9000** - **9999**
MasterIngressWorkerInternal	内部クラスター通信	**tcp**	**9000** - **9999**
MasterIngressKube	Kubernetes kubelet、スケジューラーおよびコントローラーマネージャー	**tcp**	**10250** - **10259**
MasterIngressWorkerKube	Kubernetes kubelet、スケジューラーおよびコントローラーマネージャー	**tcp**	**10250** - **10259**
MasterIngressIngressServices	Kubernetes Ingress サービス	**tcp**	**30000** - **32767**
MasterIngressWorkerIngressServices	Kubernetes Ingress サービス	**tcp**	**30000** - **32767**

ワーカーの Ingress

ワーカーマシンには、以下の Ingress グループが必要です。それぞれの Ingress グループは **AWS::EC2::SecurityGroupIngress** リソースになります。

Ingress グループ	説明	IP プロトコル	ポート範囲
WorkerIngressVxlan	Vxlan パケット	**udp**	**4789**
WorkerIngressWorkerVxlan	Vxlan パケット	**udp**	**4789**
WorkerIngressInternal	内部クラスター通信	**tcp**	**9000** - **9999**
WorkerIngressWorkerInternal	内部クラスター通信	**tcp**	**9000** - **9999**
WorkerIngressKube	Kubernetes kubelet、スケジューラーおよびコントローラーマネージャー	**tcp**	**10250**

Ingress グループ	説明	IP プロトコル	ポート範囲
WorkerIngress WorkerKube	Kubernetes kubelet、スケジューラーおよびコントローラーマネージャー	**tcp**	**10250**
WorkerIngress IngressServices	Kubernetes Ingress サービス	**tcp**	**30000 - 32767**
WorkerIngress WorkerIngress Services	Kubernetes Ingress サービス	**tcp**	**30000 - 32767**

ロールおよびインスタンスプロファイル

マシンには、AWS でのパーミッションを付与する必要があります。提供される CloudFormation テンプレートはマシンに対し、以下の **AWS::IAM::Role** オブジェクトについてのパーミッションを付与し、それぞれのロールセットに **AWS::IAM::InstanceProfile** を指定します。テンプレートを使用しない場合、マシンには以下の広範囲のパーミッションまたは個別のパーミッションを付与することができます。

ロール	結果	アクション	リソース
マスター	**Allow**	**ec2:***	*
	Allow	**elasticloadbalancing:***	*
	Allow	**iam:PassRole**	*
	Allow	**s3:GetObject**	*
ワーカー	**Allow**	**ec2:Describe***	*
ブートストラップ	**Allow**	**ec2:Describe***	*
	Allow	**ec2:AttachVolume**	*
	Allow	**ec2:DetachVolume**	*

2.1.2.4. 必要な AWS パーミッション

AdministratorAccess ポリシーを、作成する IAM ユーザーに割り当てる場合、そのユーザーには必要なパーミッションすべてを付与します。OpenShift Container Platform クラスターをデプロイするために、IAM ユーザーに以下のパーミッションが必要になります。

インストールに必要な EC2 パーミッション

- **ec2:AllocateAddress**

- ec2:AssociateAddress
- ec2:AssociateDhcpOptions
- ec2:AssociateRouteTable
- ec2:AttachInternetGateway
- ec2:AuthorizeSecurityGroupEgress
- ec2:AuthorizeSecurityGroupIngress
- ec2:CopyImage
- ec2:CreateDhcpOptions
- ec2:CreateInternetGateway
- ec2:CreateNatGateway
- ec2:CreateRoute
- ec2:CreateRouteTable
- ec2:CreateSecurityGroup
- ec2:CreateSubnet
- ec2:CreateTags
- ec2:CreateVpc
- ec2:CreateVpcEndpoint
- ec2:CreateVolume
- ec2:DescribeAccountAttributes
- ec2:DescribeAddresses
- ec2:DescribeAvailabilityZones
- ec2:DescribeDhcpOptions
- ec2:DescribeImages
- ec2:DescribeInstanceAttribute
- ec2:DescribeInstanceCreditSpecifications
- ec2:DescribeInstances
- ec2:DescribeInternetGateways
- ec2:DescribeKeyPairs
- ec2:DescribeNatGateways

- **ec2:DescribeNetworkAcls**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:ReplaceRouteTableAssociation**
- **ec2:DescribeNetworkInterfaces**
- **ec2:ModifyNetworkInterfaceAttribute**

インストールに必要な Elasticloadbalancing パーミッション

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**

- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**

インストールに必要な IAM パーミッション

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**

- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagRole**

インストールに必要な Route53 パーミッション

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:CreateHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

インストールに必要な S3 パーミッション

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketReplication**
- **s3:GetBucketRequestPayment**

- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

クラスター Operator が必要とする S3 パーミッション

- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:DeleteObject**

クラスターのアンインストールに必要な追加のすべてのパーミッション

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSnapshot**
- **ec2:DeleteSecurityGroup**

- **ec2:DeleteSubnet**
- **ec2:DeleteVolume**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DeregisterImage**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **elasticloadbalancing:DescribeTargetGroups**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **route53:DeleteHostedZone**
- **tag:GetResources**

2.1.3. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。

重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

2.1.4. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> 1
```

1 ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> (1)

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

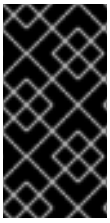
(1) ~/.ssh/id_rsa などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

2.1.5. AWS のインストール設定ファイルの作成

ユーザーによってプロビジョニングされるインフラストラクチャー を使用して OpenShift Container Platform を Amazon Web Services (AWS) にインストールするには、インストールプログラムがクラスターをデプロイするために必要なファイルを生成し、クラスターが使用するマシンのみを作成するようにそれらのファイルを変更する必要があります。**install_config.yaml** ファイル、 Kubernetes マニフェスト、および Ignition 設定ファイルを生成し、カスタマイズします。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターのインストールを完了し、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

前提条件

- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。

手順

1. **install-config.yaml** ファイルを取得します。
   a. 次のコマンドを実行します。

```
$ ./openshift-install create install-config --dir=<installation_directory> (1)
```

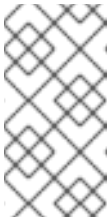
(1) **<installation_directory>** には、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

重要

空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

b. プロンプト時に、クラウドの設定の詳細情報を指定します。

i. オプション: クラスターマシンにアクセスするために使用する SSH キーを選択します。



注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

ii. ターゲットに設定するプラットフォームとして AWS を選択します。

iii. AWS プロファイルをコンピューターに保存していない場合、インストールプログラムを実行するように設定したユーザーの AWS アクセスキー ID およびシークレットアクセスキーを入力します。

iv. クラスターのデプロイ先とする AWS リージョンを選択します。

v. クラスターに設定した Route53 サービスのベースドメインを選択します。

vi. クラスターの記述名を入力します。

vii. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレットを貼り付けます。

2. 以下の **compute** スタンザに示されるように、**install-config.yaml** ファイルを編集し、コンピュート、またはワーカー、レプリカの数を **0** に設定します。

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0
```

3. オプション: **install-config.yaml** ファイルをバックアップします。

重要

install-config.yaml ファイルは次の手順で使用されます。このファイルを再利用する必要がある場合は、バックアップしてください。

4. コントロールプレーンの Kubernetes マニフェストファイルを削除します。これらのファイルを削除することで、クラスターがコントロールプレーンマシンを自動的に生成するのを防ぐことができます。

   a. クラスターの Kubernetes マニフェストを生成します。

   ```
   $ ./openshift-install create manifests --dir=<installation_directory> 1

   WARNING There are no compute nodes specified. The cluster will not fully initialize
   without compute nodes.
   INFO Consuming "Install Config" from target directory
   ```

   1 **<installation_directory>** については、同じインストールディレクトリーを指定します。

   インストールプロセスの後の部分で独自のコンピュートマシンを作成するため、この警告を無視しても問題がありません。

   b. コントロールプレーンのマシンを定義するファイルを削除します。

   ```
   $ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
   ```

5. ワーカーマシンを定義する Kubernetes マニフェストファイルを削除します。

   ```
   $ rm -f openshift/99_openshift-cluster-api_worker-machineset-*
   ```

   ワーカーマシンは独自に作成し、管理するため、これらのマシンを初期化する必要はありません。

6. Ignition 設定ファイルを取得します。

   ```
   $ ./openshift-install create ignition-configs --dir=<installation_directory> 1
   ```

   1 **<installation_directory>** については、同じインストールディレクトリーを指定します。

   以下のファイルはディレクトリーに生成されます。

   ```
   .
   ├── auth
   │   ├── kubeadmin-password
   │   └── kubeconfig
   ├── bootstrap.ign
   ├── master.ign
   ├── metadata.json
   └── worker.ign
   ```

2.1.6. インフラストラクチャー名の抽出

Ignition 設定には、Amazon Web Services (AWS) タグでクラスターを一意に識別するために使用できる一意のクラスター ID が含まれます。提供される CloudFormation テンプレートにはこのタグの参照が含まれるため、これを抽出する必要があります。

前提条件

- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。
- クラスターの Ignition 設定ファイルを生成します。
- **jq** パッケージのインストール。

手順

- Ignition 設定ファイルメタデータからインフラストラクチャー名を抽出するには、以下のコマンドを実行します。

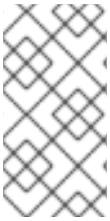
```
$ jq -r .infraID /<installation_directory>/metadata.json 1
openshift-vw9j6 2
```

1. **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。
2. このコマンドの出力はクラスター名とランダムな文字列です。

提供される CloudFormation テンプレートを設定するには、このコマンドの出力が必要になります。 これは他の AWS タグでも使用できます。

2.1.7. AWS での VPC の作成

OpenShift Container Platform クラスターで使用する VPC を Amazon Web Services (AWS) で作成する必要があります。VPN およびルートテーブルを含む、各種要件を満たすように VPC をカスタマイズできます。VPC を作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。



注記

提供される CloudFormation テンプレートを使用して AWS インフラストラクチャーを使用しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。

手順

1. テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
```

```
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC の CIDR ブロック。
2. **x.x.x.x/16-24** 形式で CIDR ブロックを指定します。
3. VPC をデプロイするアベイラビリティーゾーンの数。
4. **1** から **3** の間の整数を指定します。
5. 各アベイラビリティーゾーン内の各サブネットのサイズ。
6. **5** から **13** の間の整数を指定します。 ここで、**5** は **/27** であり、 **13** は **/19** です。

2. このトピックの「**VPC の CloudFormation テンプレート**」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要な VPC について記述しています。
3. テンプレートを起動します。

重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1. **<name>** は **cluster-vpc** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。
2. **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。
3. **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

4. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

StackStatus が **CREATE_COMPLETE** を表示した後に、出力には以下のパラメーターの値が表示されます。これらのパラメーターの値をクラスターを作成するために実行する他の CloudFormation テンプレートに指定する必要があります。

VpcId	VPC の ID。
PublicSubnetIds	新規パブリックサブネットの ID。
PrivateSubnetIds	新規プライベートサブネットの ID。

2.1.7.1. VPC の CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要な VPC をデプロイすることができます。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4]
[0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 =
/19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
```

```
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
```

```
  Type: "AWS::EC2::Route"
  DependsOn: GatewayToInternet
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PublicSubnet2
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
  Condition: DoAz3
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 0
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
```

```
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT
  PrivateSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable2:
    Type: "AWS::EC2::RouteTable"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PrivateSubnet2
      RouteTableId: !Ref PrivateRouteTable2
  NAT2:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Condition: DoAz2
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP2
        - AllocationId
      SubnetId: !Ref PublicSubnet2
  EIP2:
    Type: "AWS::EC2::EIP"
    Condition: DoAz2
    Properties:
      Domain: vpc
  Route2:
    Type: "AWS::EC2::Route"
    Condition: DoAz2
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT2
  PrivateSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
```

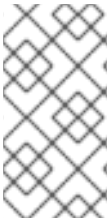
```
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
      - Effect: Allow
        Principal: '*'
        Action:
        - '*'
        Resource:
        - '*'
    RouteTableIds:
    - !Ref PublicRouteTable
    - !Ref PrivateRouteTable
    - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
```

```
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
```

2.1.8. AWS でのネットワークおよび負荷分散コンポーネントの作成

OpenShift Container Platform クラスターで使用するネットワークおよび負荷分散 (classic または network) を Amazon Web Services (AWS) で設定する必要があります。これらのコンポーネントを作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。 これにより、ホストゾーンおよびサブネットのタグも作成されます。

単一 VPC 内でテンプレートを複数回実行することができます。



注記

提供される CloudFormation テンプレートを使用して AWS インフラストラクチャーを使用しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。

手順

1. クラスターの **install-config.yaml** ファイルに指定した Route53 ゾーンのホストゾーン ID を取得します。この ID は、AWS コンソールから、または以下のコマンドを実行して取得できます。

重要

単一行にコマンドを入力してください。

```
$ aws route53 list-hosted-zones-by-name |
    jq --arg name "<route53_domain>." \ 1
    -r '.HostedZones | .[] | select(.Name=="\($name)") | .Id'
```

1 **<route53_domain>** について、クラスターの **install-config.yaml** ファイルを生成した時に作成した Route53 ベースドメインを指定します。

2. テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

1 ホスト名などに使用するクラスターを表す短いクラスターの名前。

2 クラスターの **install-config.yaml** ファイルを生成した時に使用したクラスター名を指定します。

3 クラスターの Ingition 設定ファイルでエンコードされるクラスターインフラストラクチャーの名前。

4 形式が **<cluster-name>-<random-string>** の Ignition 設定ファイルから抽出したインフラストラクチャー名を指定します。

5 ターゲットの登録に使用する Route53 パブリックトゾーン ID。

6 **Z21IXYZABCZ2A4** に類する形式の Route53 パブリックゾーン ID を指定します。この値は AWS コンソールから取得できます。

7 ターゲットの登録に使用する Route53 ゾーン。

8 クラスターの **install-config.yaml** ファイルを生成した時に使用した Route53 ベースドメインを指定します。AWS コンソールに表示される末尾のピリド (.) は含めないでください。

9 VPC 用に作成したパブリックサブネット。

10 VPC の CloudFormation テンプレートの出力から **PublicSubnetIds** 値を指定します。

11 VPC 用に作成したプライベートサブネット。

12 VPC の CloudFormation テンプレートの出力から **PrivateSubnetIds** 値を指定します。

13 クラスター用に作成した VPC。

14 VPC の CloudFormation テンプレートの出力から **VpcId** 値を指定します。

3. このトピックの「ネットワークおよびロードバランサーの **CloudFormation** テンプレート」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要なネットワークおよび負荷分散オブジェクトについて記述しています。

4. テンプレートを起動します。

重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM
```

1 **<name>** は **cluster-dns** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。

2 **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。

3 **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

5. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

StackStatus が **CREATE_COMPLETE** を表示した後に、出力には以下のパラメーターの値が表示されます。これらのパラメーターの値をクラスターを作成するために実行する他の CloudFormation テンプレートに指定する必要があります。

PrivateHostedZoneId	プライベート DNS のホストゾーン ID。
ExternalApiLoadBalancerName	外部 API ロードバランサーのフルネーム。
InternalApiLoadBalancerName	内部 API ロードバランサーのフルネーム。
ApiServerDnsName	API サーバーの完全ホスト名。
RegisterNlbIpTargetsLambda	これらのロードバランサーの登録/登録解除に役立つ Lambda ARN。
ExternalApiTargetGroupArn	外部 API ターゲットグループの ARN。
InternalApiTargetGroupArn	内部 API ターゲットグループの ARN。
InternalServiceTargetGroupArn	内部サービスターゲットグループの ARN。

2.1.8.1. ネットワークおよびロードバランサーの CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要なネットワークオブジェクトおよびロードバランサーをデプロイすることができます。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
  ClusterName:
```

```
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
    Description: The Route53 zone to register the targets with, such as example.com. Omit the trailing
period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - ClusterName
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
    - Label:
        default: "DNS"
      Parameters:
      - HostedZoneName
      - HostedZoneId
    ParameterLabels:
      ClusterName:
```

```
      default: "Cluster Name"
    InfrastructureName:
      default: "Infrastructure Name"
    VpcId:
      default: "VPC ID"
    PublicSubnets:
      default: "Public Subnets"
    PrivateSubnets:
      default: "Private Subnets"
    HostedZoneName:
      default: "Public Hosted Zone Name"
    HostedZoneId:
      default: "Public Hosted Zone ID"

Resources:
  ExtApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
      IpAddressType: ipv4
      Subnets: !Ref PublicSubnets
      Type: network

  IntApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "int"]]
      Scheme: internal
      IpAddressType: ipv4
      Subnets: !Ref PrivateSubnets
      Type: network

  IntDns:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: "Managed by CloudFormation"
      Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
      HostedZoneTags:
      - Key: Name
        Value: !Join ["-", [!Ref InfrastructureName, "int"]]
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "owned"
      VPCs:
      - VPCId: !Ref VpcId
        VPCRegion: !Ref "AWS::Region"

  ExternalApiServerRecord:
    Type: AWS::Route53::RecordSetGroup
    Properties:
      Comment: Alias record for the API server
      HostedZoneId: !Ref HostedZoneId
      RecordSets:
      - Name:
          !Join [
            ".",
```

```
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref IntDns
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName
    - Name:
        !Join [
          ".",
          ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
    LoadBalancerArn:
      Ref: ExtApiElb
    Port: 6443
    Protocol: TCP

ExternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60
```

```
InternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
    Port: 6443
    Protocol: TCP

InternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalServiceInternalListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalServiceTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
    Port: 22623
    Protocol: TCP

InternalServiceTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    Port: 22623
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

RegisterTargetLambdaIamRole:
  Type: AWS::IAM::Role
  Properties:
    RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
```

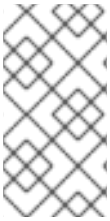
```
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalApiTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalServiceTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref ExternalApiTargetGroup

    RegisterNlbIpTargets:
      Type: "AWS::Lambda::Function"
      Properties:
        Handler: "index.handler"
        Role:
          Fn::GetAtt:
          - "RegisterTargetLambdaIamRole"
          - "Arn"
        Code:
          ZipFile: |
            import json
            import boto3
            import cfnresponse
            def handler(event, context):
              elb = boto3.client('elbv2')
              if event['RequestType'] == 'Delete':
                elb.deregister_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetIp']}])
              elif event['RequestType'] == 'Create':
                elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=[{'Id':
event['ResourceProperties']['TargetIp']}])
              responseData = {}
              cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
```

```
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
      Runtime: "python3.7"
      Timeout: 120

  RegisterSubnetTagsLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",
                "ec2:CreateTags"
              ]
            Resource: "arn:aws:ec2:*:*:subnet/*"
          - Effect: "Allow"
            Action:
              [
                "ec2:DescribeSubnets",
                "ec2:DescribeTags"
              ]
            Resource: "*"

  RegisterSubnetTags:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterSubnetTagsLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            ec2_client = boto3.client('ec2')
            if event['RequestType'] == 'Delete':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
```

```
event['ResourceProperties']['InfrastructureName']}]);
            elif event['RequestType'] == 'Create':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
      Runtime: "python3.7"
      Timeout: 120

  RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PublicSubnets

  RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PrivateSubnets

Outputs:
  PrivateHostedZoneId:
    Description: Hosted zone ID for the private DNS, which is required for private records.
    Value: !Ref IntDns
  ExternalApiLoadBalancerName:
    Description: Full name of the External API load balancer created.
    Value: !GetAtt ExtApiElb.LoadBalancerFullName
  InternalApiLoadBalancerName:
    Description: Full name of the Internal API load balancer created.
    Value: !GetAtt IntApiElb.LoadBalancerFullName
  ApiServerDnsName:
    Description: Full hostname of the API server, which is required for the Ignition config files.
    Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]
  RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn
  ExternalApiTargetGroupArn:
    Description: ARN of External API target group.
    Value: !Ref ExternalApiTargetGroup
  InternalApiTargetGroupArn:
    Description: ARN of Internal API target group.
    Value: !Ref InternalApiTargetGroup
  InternalServiceTargetGroupArn:
    Description: ARN of internal service target group.
    Value: !Ref InternalServiceTargetGroup
```

2.1.9. AWS でのセキュリティーグループおよびロールの作成

OpenShift Container Platform クラスターで使用するセキュリティーグループおよびロールを Amazon Web Services (AWS) で作成する必要があります。これらのコンポーネントを作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。



注記

提供される CloudFormation テンプレートを使用して AWS インフラストラクチャーを使用しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。

手順

1. テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

1. クラスターの Ingition 設定ファイルでエンコードされるクラスターインフラストラクチャーの名前。
2. 形式が **<cluster-name>-<random-string>** の Ignition 設定ファイルから抽出したインフラストラクチャー名を指定します。
3. VPC の CIDR ブロック。
4. **x.x.x.x/16-24** の形式で定義した VPC に使用した CIDR ブロックパラメーターを指定します。
5. VPC 用に作成したプライベートサブネット。

6 VPC の CloudFormation テンプレートの出力から **PrivateSubnetIds** 値を指定します。

7 クラスター用に作成した VPC。

8 VPC の CloudFormation テンプレートの出力から **VpcId** 値を指定します。

2. このトピックの「セキュリティーオブジェクトの **CloudFormation** テンプレート」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要なセキュリティーグループおよびロールについて記述しています。

3. テンプレートを起動します。

重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM
```

1 **<name>** は **cluster-secs** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。

2 **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。

3 **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

4. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

StackStatus が **CREATE_COMPLETE** を表示した後に、出力には以下のパラメーターの値が表示されます。これらのパラメーターの値をクラスターを作成するために実行する他の CloudFormation テンプレートに指定する必要があります。

MasterSecurityGroupId	マスターセキュリティーグループ ID
WorkerSecurityGroupId	ワーカーセキュリティーグループ ID
MasterInstanceProfile	マスター IAM インスタンスプロファイル

WorkerInstanceProfile	ワーカー IAM インスタンスプロファイル

2.1.9.1. セキュリティーオブジェクトの CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要なセキュリティーオブジェクトをデプロイすることができます。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4]
[0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - VpcCidr
      - PrivateSubnets
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
```

```
      VpcCidr:
        default: "VPC CIDR"
      PrivateSubnets:
        default: "Private Subnets"

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Master Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        ToPort: 6443
        FromPort: 6443
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22623
        ToPort: 22623
        CidrIp: !Ref VpcCidr
      VpcId: !Ref VpcId

  WorkerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Worker Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
      VpcId: !Ref VpcId

  MasterIngressEtcd:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: etcd
      FromPort: 2379
      ToPort: 2380
      IpProtocol: tcp

  MasterIngressVxlan:
    Type: AWS::EC2::SecurityGroupIngress
```

```
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressWorkerInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes kubelet, scheduler and controller manager
    FromPort: 10250
    ToPort: 10259
    IpProtocol: tcp

MasterIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes kubelet, scheduler and controller manager
    FromPort: 10250
    ToPort: 10259
```

```
    IpProtocol: tcp

MasterIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

MasterIngressWorkerIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

WorkerIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

WorkerIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressWorkerInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes secure kubelet port
    FromPort: 10250
    ToPort: 10250
    IpProtocol: tcp

WorkerIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal Kubernetes communication
    FromPort: 10250
    ToPort: 10250
    IpProtocol: tcp

WorkerIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressWorkerIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

MasterIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
```

```
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:*"
            Resource: "*"
          - Effect: "Allow"
            Action: "elasticloadbalancing:*"
            Resource: "*"
          - Effect: "Allow"
            Action: "iam:PassRole"
            Resource: "*"
          - Effect: "Allow"
            Action: "s3:GetObject"
            Resource: "*"

  MasterInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "MasterIamRole"

  WorkerIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"

  WorkerInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "WorkerIamRole"

Outputs:
  MasterSecurityGroupId:
    Description: Master Security Group ID
```

```
    Value: !GetAtt MasterSecurityGroup.GroupId

  WorkerSecurityGroupId:
    Description: Worker Security Group ID
    Value: !GetAtt WorkerSecurityGroup.GroupId

  MasterInstanceProfile:
    Description: Master IAM Instance Profile
    Value: !Ref MasterInstanceProfile

  WorkerInstanceProfile:
    Description: Worker IAM Instance Profile
    Value: !Ref WorkerInstanceProfile
```

2.1.10. AWS インフラストラクチャーの RHCOS AMI

OpenShift Container Platform ノードについて、Amazon Web Services (AWS) ゾーンの有効な Red Hat Enterprise Linux CoreOS (RHCOS) AMI を使用する必要があります。

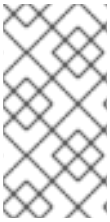
表2.1 RHCOS AMI

AWS ゾーン	AWS AMI
ap-northeast-1	**ami-0c63b39219b8123e5**
ap-northeast-2	**ami-073cba0913d2250a4**
ap-south-1	**ami-0270be11430101040**
ap-southeast-1	**ami-06eb9d35ede4f08a3**
ap-southeast-2	**ami-0d980796ce258b5d5**
ca-central-1	**ami-0f907257d1686e3f7**
eu-central-1	**ami-02fdd627029c0055b**
eu-west-1	**ami-0d4839574724ed3fa**
eu-west-2	**ami-053073b95aa285347**
eu-west-3	**ami-09deb5deb6567bcd5**
sa-east-1	**ami-068a2000546e1889d**
us-east-1	**ami-046fe691f52a953f9**
us-east-2	**ami-0649fd5d42859bdfc**
us-west-1	**ami-0c1d2b5606111ac8c**

AWS ゾーン	AWS AMI
us-west-2	**ami-00745fcbb14a863ed**

2.1.11. AWS でのブートストラップノードの作成

OpenShift Container Platform クラスターの初期化で使用するブートストラップノードを Amazon Web Services (AWS) で作成する必要があります。このノードを作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。



注記

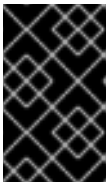
提供される CloudFormation テンプレートを使用してブートストラップノードを作成しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。
- AWS で DNS、ロードバランサー、およびリスナーを作成し、設定します。
- コントロールプレーンおよびコンピュートロールを作成します。

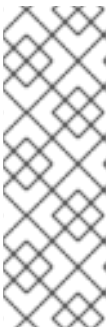
手順

1. **bootstrap.ign** Ignition 設定ファイルをクラスターに送るための場所を指定します。このファイルはインストールディレクトリーに置かれます。これを実行するための 1 つの方法として、クラスターのリージョンに S3 バケットを作成し、Ignition 設定ファイルをこれにアップロードします。



重要

提供される CloudFormation テンプレートでは、クラスターの Ignition 設定ファイルは S3 バケットから送られることを前提としています。このファイルを別の場所から送ることを選択する場合は、テンプレートを変更する必要があります。



注記

ブートストラップ Ignition 設定ファイルには、X.509 キーのようなシークレットが含まれません。以下の手順では、S3 バケットの基本的なセキュリティーを提供します。追加のセキュリティーを提供するには、OpenShift IAM ユーザーなどの特定のユーザーのみがバケットに含まれるオブジェクトにアクセスできるように S3 バケットポリシーを有効にできます。S3 を完全に回避し、ブートストラップマシンが到達できるアドレスからブートストラップ Ignition 設定ファイルを送ることができます。

a. バケットを作成します。

```
$ aws s3 mb s3://<cluster-name>-infra (1)
```

(1) **<cluster-name>-infra** はバケット名です。

b. **bootstrap.ign** Ignition 設定ファイルをバケットにアップロードします。

```
$ aws s3 cp bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign
```

c. ファイルがアップロードされていることを確認します。

```
$ aws s3 ls s3://<cluster-name>-infra/

2019-04-03 16:15:16     314878 bootstrap.ign
```

2. テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
 {
  "ParameterKey": "InfrastructureName", (1)
  "ParameterValue": "mycluster-<random_string>" (2)
 },
 {
  "ParameterKey": "RhcosAmi", (3)
  "ParameterValue": "ami-<random_string>" (4)
 },
 {
  "ParameterKey": "AllowedBootstrapSshCidr", (5)
  "ParameterValue": "0.0.0.0/0" (6)
 },
 {
  "ParameterKey": "PublicSubnet", (7)
  "ParameterValue": "subnet-<random_string>" (8)
 },
 {
  "ParameterKey": "MasterSecurityGroupId", (9)
  "ParameterValue": "sg-<random_string>" (10)
 },
 {
  "ParameterKey": "VpcId", (11)
  "ParameterValue": "vpc-<random_string>" (12)
 },
 {
  "ParameterKey": "BootstrapIgnitionLocation", (13)
  "ParameterValue": "s3://<bucket_name>/bootstrap.ign" (14)
 },
 {
  "ParameterKey": "AutoRegisterELB", (15)
  "ParameterValue": "yes" (16)
 },
 {
```

```
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 23
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]
```

1. クラスターの Ingition 設定ファイルでエンコードされるクラスターインフラストラクチャーの名前。
2. 形式が **<cluster-name>-<random-string>** の Ignition 設定ファイルから抽出したインフラストラクチャー名を指定します。
3. ブートストラップノードに使用する最新の Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 有効な **AWS::EC2::Image::Id** 値を指定します。
5. ブートストラップノードへの SSH アクセスを許可する CIDR ブロック。
6. **x.x.x.x/16-24** 形式で CIDR ブロックを指定します。
7. ブートストラップを起動するために VPC に関連付けられるパブリックサブネット。
8. VPC の CloudFormation テンプレートの出力から **PublicSubnetIds** 値を指定します。
9. マスターセキュリティーグループ ID (一時ルールの登録用)
10. セキュリティーグループおよびロールの CloudFormation テンプレートから **MasterSecurityGroupId** 値を指定します。
11. 作成されたリソースが属する VPC。
12. VPC の CloudFormation テンプレートの出力から **VpcId** 値を指定します。
13. ブートストラップの Ignition 設定ファイルをフェッチする場所。
14. **s3://<bucket_name>/bootstrap.ign** の形式で S3 バケットおよびファイル名を指定します。
15. ネットワークロードバランサー (NLB) を登録するかどうか。

16 **yes** または **no** を指定します。**yes** を指定する場合、Lambda Amazon Resource Name (ARN) の値を指定する必要があります。

17 NLB IP ターゲット登録 lambda グループの ARN。

18 DNS および負荷分散の CloudFormation テンプレートの出力から **RegisterNlbIpTargetsLambda** 値を指定します。

19 外部 API ロードバランサーのターゲットグループの ARN。

20 DNS および負荷分散の CloudFormation テンプレートの出力から **ExternalApiTargetGroupArn** 値を指定します。

21 内部 API ロードバランサーのターゲットグループの ARN。

22 DNS および負荷分散の CloudFormation テンプレートの出力から **InternalApiTargetGroupArn** 値を指定します。

23 内部サービスバランサーのターゲットグループの ARN。

24 DNS および負荷分散の CloudFormation テンプレートの出力から **InternalServiceTargetGroupArn** 値を指定します。

3. このトピックの「ブートストラップマシンの **CloudFormation** テンプレート」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要なブートストラップマシンについて記述しています。

4. テンプレートを起動します。

重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM
```

1 **<name>** は **cluster-bootstrap** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。

2 **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。

3 **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

5. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

StackStatus が **CREATE_COMPLETE** を表示した後に、出力には以下のパラメーターの値が表示されます。これらのパラメーターの値をクラスターを作成するために実行する他の CloudFormation テンプレートに指定する必要があります。

BootstrapInstanceId	ブートストラップインスタンス ID。
BootstrapPublicIp	ブートストラップノードのパブリック IP アドレス。
BootstrapPrivateIp	ブートストラップノードのプライベート IP アドレス。

2.1.11.1. ブートストラップマシンの CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要なブートストラップマシンをデプロイできます。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4]
[0-9]|25[0-5])(\/([0-9]|1[0-9]|2[0-9]|3[0-2]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
```

```
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - PublicSubnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      AllowedBootstrapSshCidr:
        default: "Allowed SSH Source"
      PublicSubnet:
        default: "Public Subnet"
```

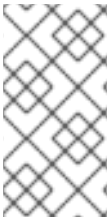
```
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Bootstrap Ignition Source"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  BootstrapIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:AttachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:DetachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "s3:GetObject"
            Resource: "*"

  BootstrapInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Path: "/"
      Roles:
      - Ref: "BootstrapIamRole"

  BootstrapSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Bootstrap Security Group
      SecurityGroupIngress:
```

```
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref AllowedBootstrapSshCidr
      - IpProtocol: tcp
        ToPort: 19531
        FromPort: 19531
        CidrIp: 0.0.0.0/0
      VpcId: !Ref VpcId

  BootstrapInstance:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      IamInstanceProfile: !Ref BootstrapInstanceProfile
      InstanceType: "i3.large"
      NetworkInterfaces:
      - AssociatePublicIpAddress: "true"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "BootstrapSecurityGroup"
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "PublicSubnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"replace":{"source":"${S3Loc}","verification":{}}},"timeouts":
{},"version":"2.1.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}'
        - {
          S3Loc: !Ref BootstrapIgnitionLocation
        }

  RegisterBootstrapApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

  RegisterBootstrapInternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

  RegisterBootstrapInternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

Outputs:
```

```
BootstrapInstanceId:
  Description: Bootstrap Instance ID.
  Value: !Ref BootstrapInstance

BootstrapPublicIp:
  Description: The bootstrap node public IP address.
  Value: !GetAtt BootstrapInstance.PublicIp

BootstrapPrivateIp:
  Description: The bootstrap node private IP address.
  Value: !GetAtt BootstrapInstance.PrivateIp
```

2.1.12. AWS でのコントロールプレーンの作成

クラスターで使用するコントロールプレーンマシンを Amazon Web Services (AWS) で作成する必要があります。これらのノードを作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。



注記

提供される CloudFormation テンプレートを使用してコントロールプレーンノードを作成しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。
- AWS で DNS、ロードバランサー、およびリスナーを作成し、設定します。
- コントロールプレーンおよびコンピュートロールを作成します。
- ブートストラップマシンを作成します。

手順

1. テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
```

```
    "ParameterValue": "yes" (6)
  },
  {
    "ParameterKey": "PrivateHostedZoneId", (7)
    "ParameterValue": "<random_string>" (8)
  },
  {
    "ParameterKey": "PrivateHostedZoneName", (9)
    "ParameterValue": "mycluster.example.com" (10)
  },
  {
    "ParameterKey": "Master0Subnet", (11)
    "ParameterValue": "subnet-<random_string>" (12)
  },
  {
    "ParameterKey": "Master1Subnet", (13)
    "ParameterValue": "subnet-<random_string>" (14)
  },
  {
    "ParameterKey": "Master2Subnet", (15)
    "ParameterValue": "subnet-<random_string>" (16)
  },
  {
    "ParameterKey": "MasterSecurityGroupId", (17)
    "ParameterValue": "sg-<random_string>" (18)
  },
  {
    "ParameterKey": "IgnitionLocation", (19)
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
(20)
  },
  {
    "ParameterKey": "CertificateAuthorities", (21)
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" (22)
  },
  {
    "ParameterKey": "MasterInstanceProfileName", (23)
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" (24)
  },
  {
    "ParameterKey": "MasterInstanceType", (25)
    "ParameterValue": "m4.xlarge" (26)
  },
  {
    "ParameterKey": "AutoRegisterELB", (27)
    "ParameterValue": "yes" (28)
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", (29)
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" (30)
  },
  {
```

```
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]
```

1. クラスターの Ingition 設定ファイルでエンコードされるクラスターインフラストラクチャーの名前。
2. 形式が **<cluster-name>-<random-string>** の Ignition 設定ファイルから抽出したインフラストラクチャー名を指定します。
3. コントロールプレーンマシンに使用する最新の Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. **AWS::EC2::Image::Id** 値を指定します。
5. DNS etcd 登録を実行するかどうか。
6. **yes** または **no** を指定します。**yes** を指定する場合、ホストゾーンの情報を指定する必要があります。
7. etcd ターゲットの登録に使用する Route53 プライベートゾーン ID。
8. DNS および負荷分散の CloudFormation テンプレートの出力から **PrivateHostedZoneId** 値を指定します。
9. ターゲットの登録に使用する Route53 ゾーン。
10. **<cluster_name>.<domain_name>** を指定します。ここで、**<domain_name>** はクラスターの **install-config.yaml** ファイルの生成時に使用した Route53 ベースドメインです。AWS コンソールに表示される末尾のピリド (.) は含めないでください。
11. 13. 15. コントロールプレーンマシンの起動に使用するサブネット (プライベートが望ましい)。
12. 14. 16. DNS および負荷分散の CloudFormation テンプレートの出力から **PrivateSubnets** 値のサブネットを指定します。
17. マスターノードに関連付けるマスターセキュリティーグループ ID。
18. セキュリティーグループおよびロールの CloudFormation テンプレートから **MasterSecurityGroupId** 値を指定します。
19. コントロールプレーンの Ignition 設定ファイルをフェッチする場所。

20 生成される Ignition 設定ファイルの場所を指定します (**https://api-int.<cluster_name>.<domain_name>:22623/config/master**)。

21 使用する base64 でエンコードされた認証局の文字列。

22 インストールディレクトリーにある **master.ign** ファイルから値を指定します。この値は、**data:text/plain;charset=utf-8;base64,ABC...xYz==** 形式の長い文字列です。

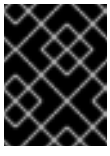
23 マスターロールに関連付ける IAM プロファイル。

24 セキュリティーグループおよびロールの CloudFormation テンプレートの出力から **MasterInstanceProfile** パラメーターの値を指定します。

25 コントロールプレーンマシンに使用する AWS インスタンスのタイプ。

26 許可される値:

- **m4.xlarge**
- **m4.2xlarge**
- **m4.4xlarge**
- **m4.8xlarge**
- **m4.10xlarge**
- **m4.16xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**



重要

m4 インスタンスタイプが **eu-west-3** などのリージョンで利用可能ではない場合、**m5.xlarge** などのように **m5** タイプを代わりに使用します。

27 ネットワークロードバランサー (NLB) を登録するかどうか。

28 **yes** または **no** を指定します。**yes** を指定する場合、Lambda Amazon Resource Name (ARN) の値を指定する必要があります。

29 NLB IP ターゲット登録 lambda グループの ARN。

30 DNS および負荷分散の CloudFormation テンプレートの出力から **RegisterNlbIpTargetsLambda** 値を指定します。

31 外部 API ロードバランサーのターゲットグループの ARN。

32 DNS および負荷分散の CloudFormation テンプレートの出力から **ExternalApiTargetGroupArn** 値を指定します。

33 内部 API ロードバランサーのターゲットグループの ARN。

34 DNS および負荷分散の CloudFormation テンプレートの出力から **InternalApiTargetGroupArn** 値を指定します。

35 内部サービスバランサーのターゲットグループの ARN。

36 DNS および負荷分散の CloudFormation テンプレートの出力から **InternalServiceTargetGroupArn** 値を指定します。

2. このトピックの「コントロールプレーンマシンの **CloudFormation** テンプレート」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要なコントロールプレーンのマシンについて記述しています。

3. **m5** インスタンスタイプを **MasterInstanceType** の値として指定している場合、そのインスタンスタイプを CloudFormation テンプレートの **MasterInstanceType.AllowedValues** パラメーターに追加します。

4. テンプレートを起動します。

重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1 **<name>** は **cluster-control-plane** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。

2 **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。

3 **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

5. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

2.1.12.1. コントロールプレーンマシンの CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要なコントロールプレーンマシンをデプロイすることができます。

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke DNS etcd registration, which requires Hosted Zone
information?
    Type: String
  PrivateHostedZoneId:
    Description: The Route53 private zone ID to register the etcd targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  PrivateHostedZoneName:
    Description: The Route53 zone to register the targets with, such as cluster.example.com. Omit the
trailing period.
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  MasterInstanceProfileName:
    Description: IAM profile to associate with master nodes.
```

```
    Type: String
  MasterInstanceType:
    Default: m4.xlarge
    Type: String
    AllowedValues:
    - "m4.xlarge"
    - "m4.2xlarge"
    - "m4.4xlarge"
    - "m4.8xlarge"
    - "m4.10xlarge"
    - "m4.16xlarge"
    - "c4.2xlarge"
    - "c4.4xlarge"
    - "c4.8xlarge"
    - "r4.xlarge"
    - "r4.2xlarge"
    - "r4.4xlarge"
    - "r4.8xlarge"
    - "r4.16xlarge"
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - MasterInstanceType
      - RhcosAmi
      - IgnitionLocation
```

```
    - CertificateAuthorities
    - MasterSecurityGroupId
    - MasterInstanceProfileName
  - Label:
      default: "Network Configuration"
    Parameters:
    - VpcId
    - AllowedBootstrapSshCidr
    - Master0Subnet
    - Master1Subnet
    - Master2Subnet
  - Label:
      default: "DNS"
    Parameters:
    - AutoRegisterDNS
    - PrivateHostedZoneName
    - PrivateHostedZoneId
  - Label:
      default: "Load Balancer Automation"
    Parameters:
    - AutoRegisterELB
    - RegisterNlbIpTargetsLambdaArn
    - ExternalApiTargetGroupArn
    - InternalApiTargetGroupArn
    - InternalServiceTargetGroupArn
  ParameterLabels:
    InfrastructureName:
      default: "Infrastructure Name"
    VpcId:
      default: "VPC ID"
    Master0Subnet:
      default: "Master-0 Subnet"
    Master1Subnet:
      default: "Master-1 Subnet"
    Master2Subnet:
      default: "Master-2 Subnet"
    MasterInstanceType:
      default: "Master Instance Type"
    MasterInstanceProfileName:
      default: "Master Instance Profile Name"
    RhcosAmi:
      default: "Red Hat Enterprise Linux CoreOS AMI ID"
    BootstrapIgnitionLocation:
      default: "Master Ignition Source"
    CertificateAuthorities:
      default: "Ignition CA String"
    MasterSecurityGroupId:
      default: "Master Security Group ID"
    AutoRegisterDNS:
      default: "Use Provided DNS Automation"
    AutoRegisterELB:
      default: "Use Provided ELB Automation"
    PrivateHostedZoneName:
      default: "Private Hosted Zone Name"
    PrivateHostedZoneId:
      default: "Private Hosted Zone ID"
```

```
Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]
  DoDns: !Equals ["yes", !Ref AutoRegisterDNS]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master0Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"append":[{"source":"${SOURCE}","verification":{}}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}","verification":{}}]}},"timeouts":
{},"version":"2.2.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster0:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
```

```
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  Master1:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master1Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"append":[{"source":"${SOURCE}","verification":{}}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}","verification":{}}]}},"timeouts":
{},"version":"2.2.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster1:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
```

```
      TargetIp: !GetAtt Master1.PrivateIp

  Master2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"append":[{"source":"${SOURCE}","verification":{}}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}","verification":{}}]}},"timeouts":
{},"version":"2.2.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster2:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp
```

```
  EtcdSrvRecords:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["_etcd-server-ssl._tcp", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]],
      ]
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]],
      ]
      - !Join [
        " ",
        ["0 10 2380", !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]],
      ]
      TTL: 60
      Type: SRV

  Etcd0Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-0", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master0.PrivateIp
      TTL: 60
      Type: A

  Etcd1Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-1", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master1.PrivateIp
      TTL: 60
      Type: A

  Etcd2Record:
    Condition: DoDns
    Type: AWS::Route53::RecordSet
    Properties:
      HostedZoneId: !Ref PrivateHostedZoneId
      Name: !Join [".", ["etcd-2", !Ref PrivateHostedZoneName]]
      ResourceRecords:
      - !GetAtt Master2.PrivateIp
      TTL: 60
      Type: A

Outputs:
```

```
PrivateIPs:
  Description: The control-plane node private IP addresses.
  Value:
    !Join [
      ",",
      [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
    ]
```

2.1.13. ユーザーによってプロビジョニングされるインフラストラクチャーでの AWS でのブートストラップノードの初期化

Amazon Web Services (AWS) ですべての必要なインフラストラクチャーを作成した後に、クラスターをインストールできます。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。
- AWS で DNS、ロードバランサー、およびリスナーを作成し、設定します。
- コントロールプレーンおよびコンピュートロールを作成します。
- ブートストラップマシンを作成します。
- コントロールプレーンマシンを作成します。
- ワーカーマシンを手動で管理する予定の場合には、ワーカーマシンを作成します。

手順

1. インストールプログラムが含まれるディレクトリーに切り替え、以下のコマンドを実行します。

```
$ ./openshift-install wait-for bootstrap-complete --dir=<installation_directory> \ 1
    --log-level info 2
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

2 異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。

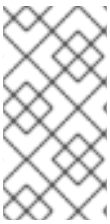
コマンドが **FATAL** 警告を出さずに終了する場合、実稼働用のコントロールプレーンは初期化されています。

2.1.13.1. AWS でのワーカーノードの作成

クラスターで使用するワーカーノードを Amazon Web Services (AWS) で作成できます。これらのノードを作成するための最も簡単な方法として、提供される CloudFormation テンプレートを変更することができます。

重要

CloudFormation テンプレートは、1 つのワーカーマシンを表すスタックを作成します。それぞれのワーカーマシンにスタックを作成する必要があります。



注記

提供される CloudFormation テンプレートを使用してワーカーノードを作成しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

前提条件

- AWS アカウントを設定します。
- クラスターの Ignition 設定ファイルを生成します。
- AWS で VPC および関連するサブネットを作成し、設定します。
- AWS で DNS、ロードバランサー、およびリスナーを作成し、設定します。
- コントロールプレーンおよびコンピュートロールを作成します。
- ブートストラップマシンを作成します。
- コントロールプレーンマシンを作成します。

手順

1. CloudFormation テンプレートが必要とするパラメーター値が含まれる JSON ファイルを作成します。

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
10
```

```
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
    "ParameterValue": "m4.large" 16
  }
]
```

1. クラスターの Ingition 設定ファイルでエンコードされるクラスターインフラストラクチャーの名前。
2. 形式が **<cluster-name>-<random-string>** の Ignition 設定ファイルから抽出したインフラストラクチャー名を指定します。
3. ワーカーノードに使用する最新の Red Hat Enterprise Linux CoreOS（RHCOS）AMI。
4. **AWS::EC2::Image::Id** 値を指定します。
5. ワーカーノードを起動するサブネット (プライベートが望ましい)。
6. DNS および負荷分散の CloudFormation テンプレートの出力から **PrivateSubnets** 値のサブネットを指定します。
7. ワーカーノードに関連付けるワーカーセキュリティーグループ ID。
8. セキュリティーグループおよびロールの CloudFormation テンプレートの出力から **WorkerSecurityGroupId** 値を指定します。
9. ブートストラップの Ignition 設定ファイルをフェッチする場所。
10. 生成される Ignition 設定の場所を指定します。 **https://api-int.<cluster_name>.<domain_name>:22623/config/worker**
11. 使用する base64 でエンコードされた認証局の文字列。
12. インストールディレクトリーにある **worker.ign** ファイルから値を指定します。この値は、**data:text/plain;charset=utf-8;base64,ABC...xYz==** 形式の長い文字列です。
13. ワーカーロールに関連付ける IAM プロファイル。
14. セキュリティーグループおよびロールの CloudFormation テンプレートの出力から **WokerInstanceProfile** パラメーターの値を指定します。
15. コントロールプレーンマシンに使用する AWS インスタンスのタイプ。
16. 許可される値:
    - **m4.large**

- **m4.xlarge**
- **m4.2xlarge**
- **m4.4xlarge**
- **m4.8xlarge**
- **m4.10xlarge**
- **m4.16xlarge**
- **c4.large**
- **c4.xlarge**
- **c4.2xlarge**
- **c4.4xlarge**
- **c4.8xlarge**
- **r4.large**
- **r4.xlarge**
- **r4.2xlarge**
- **r4.4xlarge**
- **r4.8xlarge**
- **r4.16xlarge**

重要

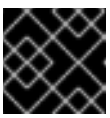
m4 インスタンスが **eu-west-3** などのリージョンで利用可能ではない場合、**m5** タイプを代わりに使用します。

2. このトピックの「ワーカーマシンの **CloudFormation** テンプレート」セクションからテンプレートをコピーし、これをコンピューター上に YAML ファイルとして保存します。このテンプレートは、クラスターに必要なネットワークオブジェクトおよびロードバランサーについて記述しています。

3. **m5** インスタンスタイプを **WorkerInstanceType** の値として指定している場合、そのインスタンスタイプを CloudFormation テンプレートの **WorkerInstanceType.AllowedValues** パラメーターに追加します。

4. ワーカースタックを作成します。

   a. テンプレートを起動します。



重要

単一行にコマンドを入力してください。

```
$ aws cloudformation create-stack --stack-name <name> (1)
    --template-body file://<template>.yaml \ (2)
    --parameters file://<parameters>.json (3)
```

(1) **<name>** は **cluster-workers** などの CloudFormation スタックの名前です。クラスターを削除する場合に、このスタックの名前が必要になります。

(2) **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスまたは名前です。

(3) **<parameters>** は、CloudFormation パラメーター JSON ファイルへの相対パスまたは名前です。

b. テンプレートのコンポーネントが存在することを確認します。

```
$ aws cloudformation describe-stacks --stack-name <name>
```

5. クラスターに作成するワーカーマシンが十分な数に達するまでワーカースタックの作成を継続します。

重要

2 つ以上のワーカーマシンを作成する必要があるため、この CloudFormation テンプレートを使用する 2 つ以上のスタックを作成する必要があります。

2.1.13.1.1. ワーカーマシンの CloudFormation テンプレート

以下の CloudFormation テンプレートを使用し、OpenShift Container Platform クラスターに必要なワーカーマシンをデプロイすることができます。

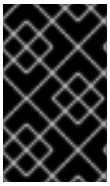
```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
```

```
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  WorkerInstanceType:
    Default: m4.large
    Type: String
    AllowedValues:
    - "m4.large"
    - "m4.xlarge"
    - "m4.2xlarge"
    - "m4.4xlarge"
    - "m4.8xlarge"
    - "m4.10xlarge"
    - "m4.16xlarge"
    - "c4.large"
    - "c4.xlarge"
    - "c4.2xlarge"
    - "c4.4xlarge"
    - "c4.8xlarge"
    - "r4.large"
    - "r4.xlarge"
    - "r4.2xlarge"
    - "r4.4xlarge"
    - "r4.8xlarge"
    - "r4.16xlarge"

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - WorkerInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - WorkerSecurityGroupId
      - WorkerInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - Subnet
    ParameterLabels:
      Subnet:
        default: "Subnet"
      InfrastructureName:
```

```
        default: "Infrastructure Name"
      WorkerInstanceType:
        default: "Worker Instance Type"
      WorkerInstanceProfileName:
        default: "Worker Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      IgnitionLocation:
        default: "Worker Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      WorkerSecurityGroupId:
        default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "WorkerSecurityGroupId"
        SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"append":[{"source":"${SOURCE}","verification":{}}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}","verification":{}}]}},"timeouts":
{},"version":"2.2.0"},"networkd":{},"passwd":{},"storage":{},"systemd":{}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
```

2.1.14. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。



重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。
2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。

注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

2.1.15. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1

$ oc whoami
system:admin
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

2.1.16. マシンの CSR の承認

マシンをクラスターに追加する際に、追加したそれぞれのマシンについて 2 つの保留状態の証明書署名要求 (CSR) が生成されます。これらの CSR が承認されていることを確認するか、または必要な場合はそれらを承認してください。

前提条件

- マシンをクラスターに追加していること。

手順

1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes

NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.13.4+b626c2fe1
master-1  Ready     master  63m  v1.13.4+b626c2fe1
master-2  Ready     master  64m  v1.13.4+b626c2fe1
worker-0  NotReady  worker  76s  v1.13.4+b626c2fe1
worker-1  NotReady  worker  70s  v1.13.4+b626c2fe1
```

   出力には作成したすべてのマシンが一覧表示されます。

2. 保留中の証明書署名要求 (CSR) を確認し、クラスターに追加したそれぞれのマシンのクライアントおよびサーバー要求に **Pending** または **Approved** ステータスが表示されていることを確認します。

```
$ oc get csr

NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending (1)
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending (2)
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

   (1) クライアント要求の CSR。

   (2) サーバー要求の CSR。

   この例では、2 つのマシンがクラスターに参加しています。この一覧にはさらに多くの承認された CSR が表示される可能性があります。

3. 追加したマシンの保留中の CSR すべてが **Pending** ステータスになった後に CSR が承認されない場合には、クラスターマシンの CSR を承認します。

注記

CSR のローテーションは自動的に実行されるため、クラスターにマシンを追加後 1 時間以内に CSR を承認してください。1 時間以内に承認しない場合には、証明書のローテーションが行われ、各ノードに 3 つ以上の証明書が存在するようになります。これらの証明書すべてを承認する必要があります。最初の CSR の承認後、後続のノードクライアント CSR はクラスターの **kube-controller-manger** によって自動的に承認されます。kubelet 提供証明書の要求を自動的に承認する方法を実装する必要があります。

- それらを個別に承認するには、それぞれの有効な CSR について以下のコマンドを実行します。

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

2.1.17. Operator の初期設定

コントロールプレーンの初期化後に、一部の Operator を利用可能にするためにそれらをすぐに設定する必要があります。

前提条件

- コントロールプレーンが初期化されていること。

手順

1. クラスターコンポーネントがオンラインになることを確認します。

```
$ watch -n5 oc get clusteroperators

NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.1.0     True        False         False      69s
cloud-credential                           4.1.0     True        False         False      12m
cluster-autoscaler                         4.1.0     True        False         False      11m
console                                    4.1.0     True        False         False      46s
dns                                        4.1.0     True        False         False      11m
image-registry                             4.1.0     False       True          False      5m26s
ingress                                    4.1.0     True        False         False      5m36s
kube-apiserver                             4.1.0     True        False         False      8m53s
kube-controller-manager                    4.1.0     True        False         False      7m24s
kube-scheduler                             4.1.0     True        False         False      12m
machine-api                                4.1.0     True        False         False      12m
machine-config                             4.1.0     True        False         False      7m36s
marketplace                                4.1.0     True        False         False      7m54m
monitoring                                 4.1.0     True        False         False      7h54s
network                                    4.1.0     True        False         False      5m9s
node-tuning                                4.1.0     True        False         False      11m
openshift-apiserver                        4.1.0     True        False         False      11m
openshift-controller-manager               4.1.0     True        False         False      5m943s
openshift-samples                          4.1.0     True        False         False      3m55s
operator-lifecycle-manager                 4.1.0     True        False         False      11m
operator-lifecycle-manager-catalog         4.1.0     True        False         False      11m
service-ca                                 4.1.0     True        False         False      11m
```

```
service-catalog-apiserver           4.1.0     True        False         False      5m26s
service-catalog-controller-manager  4.1.0     True        False         False      5m25s
storage                             4.1.0     True        False         False      5m30s
```

2. 利用不可の Operator を設定します。

2.1.17.1. イメージレジストリーストレージの設定

image-registry Operator が利用できない場合、そのストレージを設定する必要があります。実稼働クラスターに必要な PersistentVolume の設定方法と、実稼働用ではないクラスターにのみ使用できる空のディレクトリーをストレージの場所として設定する方法が表示されます。

2.1.17.1.1. ユーザーによってプロビジョニングされるインフラストラクチャーで AWS のレジストリーストレージを設定する

インストール時に、S3 バケットを作成するにはクラウド認証情報を使用でき、レジストリー Operator がストレージを自動的に設定します。

レジストリー Operator が S3 バケットを作成できず、ストレージを自動的に設定する場合、以下の手順により S3 バケットを作成し、ストレージを設定することができます。

前提条件

- ユーザーによってプロビジョニングされるインフラストラクチャーでの AWS 上のクラスター

手順

レジストリー Operator が S3 バケットを作成できず、ストレージを自動的に設定する場合は、以下の手順を使用してください。

1. バケットライフサイクルポリシーを設定し、1 日以上経過している未完了のマルチパートアップロードを中止します。
2. **configs.imageregistry.operator.openshift.io/cluster** にストレージ設定を入力します。

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster

storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

警告

AWS でレジストリーイメージのセキュリティーを保護するには、S3 バケットに対してパブリックアクセスのブロックを実行します。

2.1.17.1.2. 実稼働以外のクラスターでのイメージレジストリーのストレージの設定

イメージレジストリー Operator のストレージを設定する必要があります。実稼働用以外のクラスターの場合、イメージレジストリーは空のディレクトリーに設定することができます。これを実行する場合、レジストリーを再起動するとすべてのイメージが失われます。

手順

- イメージレジストリーストレージを空のディレクトリーに設定するには、以下を実行します。

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 実稼働用以外のクラスターにのみこのオプションを設定します。

イメージレジストリー Operator がそのコンポーネントを初期化する前にこのコマンドを実行する場合、**oc patch** コマンドは以下のエラーを出して失敗します。

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

数分待機した後に、このコマンドを再び実行します。

2.1.18. ユーザーによってプロビジョニングされるインフラストラクチャーでの AWS インストールの実行

Amazon Web Service (AWS) のユーザーによってプロビジョニングされるインフラストラクチャーで OpenShift Container Platform のインストールを開始した後は、ブートストラップノードを削除し、インストールが完了するまで待機します。

前提条件

- OpenShift Container Platform クラスターのブートストラップノードを、ユーザーによってプロビジョニングされた AWS インフラストラクチャーにデプロイします。
- **oc** CLI をインストールし、ログインします。

手順

1. ブートストラップリソースを削除します。CloudFormation テンプレートを使用した場合は、そのスタックを削除します。

```
$ aws cloudformation delete-stack --stack-name <name> 1
```

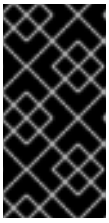
1 **<name>** は、ブートストラップスタックの名前です。

2. クラスターのインストールを完了します。

```
$ ./openshift-install --dir=<installation_directory> wait-for install-complete 1
```

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

第3章 ベアメタルへのインストール

3.1. クラスターのベアメタルへのインストール

OpenShift Container Platform version 4.1 では、クラスターをプロビジョニングするベアメタルのインフラストラクチャーにインストールできます。

重要

以下の手順に従って仮想化環境またはクラウド環境にクラスターをデプロイすることができますが、ベアメタルプラットフォーム以外の場合は追加の考慮事項に注意してください。このような環境で OpenShift Container Platform クラスターのインストールを試行する前に、「guidelines for deploying OpenShift Container Platform on non-tested platforms」にある情報を確認してください。

前提条件

- クラスターの永続ストレージをプロビジョニングします。プライベートイメージレジストリーをデプロイするには、ストレージで ReadWriteMany アクセスモードを指定する必要があります。
- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

3.1.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。

重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得

- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

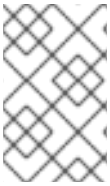
3.1.2. ユーザーによってプロビジョニングされるインフラストラクチャーを使用する場合のクラスターのマシン要件

ユーザーによってプロビジョニングされるインフラストラクチャーを含むクラスターの場合、必要なマシンすべてをデプロイする必要があります。

3.1.2.1. 必要なマシン

最小の OpenShift Container Platform クラスターでは以下のホストが必要です。

- 1 つのブートストラップマシン
- 3 つのコントロールプレーン、またはマスター、マシン
- 少なくとも 2 つのコンピュートマシン (ワーカーマシンとしても知られる)。



注記

クラスターでは、ブートストラップマシンが OpenShift Container Platform クラスターを 3 つのコントロールプレーンマシンにデプロイする必要があります。クラスターのインストール後にブートストラップマシンを削除できます。

重要

クラスターの高可用性を維持するには、これらのクラスターマシンについて別個の物理ホストを使用します。

ブートストラップおよびコントロールプレーンマシンでは、Red Hat Enterprise Linux CoreOS (RHCOS) をオペレーティングシステムとして使用する必要があります。

RHCOS は Red Hat Enterprise Linux 8 をベースとしており、そのハードウェア認定および要件が継承されることに注意してください。「Red Hat Enterprise Linux テクノロジーの機能と制限 」を参照してください。

3.1.2.2. ネットワーク接続の要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定ファイルをフェッチする必要があります。初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーが必要になります。

3.1.2.3. 最小リソース要件

それぞれのクラスターマシンは、以下の最小要件を満たしている必要があります。

マシン	Operating System	vCPU	RAM	ストレージ
ブートストラップ	RHCOS	4	16 GB	120 GB

マシン	Operating System	vCPU	RAM	ストレージ
コントロールプレーン	RHCOS	4	16 GB	120 GB
コンピュート	RHCOS または RHEL 7.6	2	8 GB	120 GB

3.1.2.4. 証明書署名要求の管理

ユーザーがプロビジョニングするインフラストラクチャーを使用する場合、クラスターの自動マシン管理へのアクセスは制限されるため、インストール後にクラスターの証明書署名要求 (CSR) のメカニズムを提供する必要があります。**kube-controller-manager** は kubelet クライアント CSR のみを承認します。**machine-approver** は、kubelet 認証情報を使用して要求される提供証明書の有効性を保証できません。適切なマシンがこの要求を発行したかどうかを確認できないためです。kubelet 提供証明書の要求の有効性を検証し、それらを承認する方法を判別し、実装する必要があります。

3.1.3. ユーザーによってプロビジョニングされるインフラストラクチャーの作成

ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform クラスターをデプロイする前に、基礎となるインフラストラクチャーを作成する必要があります。

前提条件

- クラスターでサポートするインフラストラクチャーを作成する前に、「OpenShift Container Platform 4.x Tested Integrations」ページを参照してください。

手順

1. DHCP を設定します。
2. 必要なロードバランサーをプロビジョニングします。
3. マシンのポートを設定します。
4. DNS を設定します。
5. ネットワーク接続を確認します。

3.1.3.1. ユーザーによってプロビジョニングされるインフラストラクチャのネットワーク要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定をフェッチする必要があります。

初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーが必要になります。

クラスターのマシンを長期間管理するために DHCP サーバーを使用することが推奨されています。DHCP サーバーが永続 IP アドレスおよびホスト名をクラスターマシンに提供するように設定されていることを確認します。

Kubernetes API サーバーはクラスターマシンのノード名を解決できる必要があります。API サーバーお

よびワーカーノードが異なるゾーンに置かれている場合、デフォルトの DNS 検索ゾーンを、API サーバーでノード名を解決できるように設定することができます。もう 1 つの実行可能な方法として、ノードオブジェクトとすべての DNS 要求の両方において、ホストを完全修飾ドメイン名で常に参照することができます。

マシン間のネットワーク接続を、クラスターのコンポーネントが通信できるように設定する必要があります。すべてのマシンではクラスターの他のすべてのマシンのホスト名を解決できる必要があります。

表3.1 すべてのマシンに対応するすべてのマシン

プロトコル	ポート	説明
ICMP	該当なし	ネットワーク到達性のテスト
TCP	**9000**-**9999**	ホストレベルのサービス。 ポート **9100**-**9101** のノードエクスポーター、ポート **9099** の Cluster Version Operator が含まれます。
	10250-**10259**	Kubernetes が予約するデフォルトポート
	10256	openshift-sdn
UDP	**4789**	VXLAN および GENEVE
	6081	VXLAN および GENEVE
	9000-**9999**	ポート **9100**-**9101** のノードエクスポーターを含む、ホストレベルのサービス。
TCP/UDP	**30000**-**32767**	Kubernetes NodePort

表3.2 コントロールプレーンへのすべてのマシン

プロトコル	ポート	説明
TCP	**2379**-**2380**	etcd サーバー、ピア、およびメトリクスポート
	6443	Kubernetes API

ネットワークトポロジー要件
クラスター用にプロビジョニングするインフラストラクチャーは、ネットワークトポロジーの以下の要件を満たす必要があります。

重要

OpenShift Container Platform では、すべてのノードが、プラットフォームコンテナーのイメージをプルし、Telemetry データを Red Hat に提供するためにインターネットへの直接のアクセスが必要です。

ロードバランサー

OpenShift Container Platform をインストールする前に、2 つの Layer 4 ロードバランサーをプロビジョニングする必要があります。API には 1 つのロードバランサーが必要で、デフォルトの Ingress コントローラーには、Ingress をアプリケーションに提供する 2 番目のロードバランサーが必要です。

ポート	マシン	内部	外部	説明
6443	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。	x	x	Kubernetes API サーバー
22623	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。	x		マシン設定サーバー
443	デフォルトで Ingress ルーター Pod、コンピュート、またはワーカーを実行するマシン。	x	x	HTTPS トラフィック
80	デフォルトで Ingress ルーター Pod、コンピュート、またはワーカーを実行するマシン。	x	x	HTTP トラフィック

注記

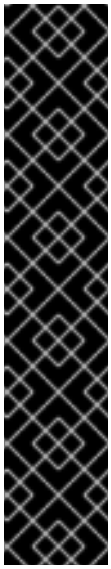
Ingress ルーターの作業用の設定が OpenShift Container Platform クラスターに必要です。コントロールプレーンの初期化後に Ingress ルーターを設定する必要があります。

3.1.3.2. ユーザーによってプロビジョニングされるインフラストラクチャーの DNS 要件

以下の DNS レコードは、ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform クラスターに必要です。各レコードで、 **<cluster_name>** はクラスター名で、**<base_domain>** は、**install-config.yaml** ファイルに指定するクラスターのベースドメインです。

表3.3 必要な DNS レコード

Component	レコード	説明
Kubernetes API	**api.<cluster_name>.<base_domain>**	この DNS レコードは、コントロールプレーンマシンのロードバランサーを参照する必要があります。このレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。

Component	レコード	説明
	api-int.<cluster_name>.<base_domain>	この DNS レコードは、コントロールプレーンマシンのロードバランサーを参照する必要があります。このレコードは、クラスター内のすべてのノードで解決できる必要があります。  **重要** API サーバーは、Kubernetes に記録されるホスト名でワーカーノードを解決できる必要があります。これがノード名を解決できない場合、プロキシーされる API 呼び出しが失敗し、Pod からログを取得できなくなる可能性があります。
Routes	***.apps.<cluster_name>.<base_domain>**	Ingress ルーター Pod を実行するマシンをターゲットにするロードバランサーを参照するワイルドカード DNS レコードです。このレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。
etcd	**etcd-<index>.<cluster_name>.<base_domain>**	OpenShift Container Platform では、各 etcd インスタンスの DNS レコードがインスタンスをホストするコントロールプレーンマシンを参照する必要があります。etcd インスタンスは **<index>** 値によって差別化されます。この値は **0** で始まり、**n-1** で終了します。ここで、**n** はクラスターのコントロールプレーンマシンの数です。DNS レコードはコントロールプレーンマシンのユニキャスト IPV4 アドレスに解決し、レコードはクラスター内のすべてのノードで解決可能である必要があります。

Component	レコード	説明
	_etcd-server-ssl._tcp.<cluster_name>.<base_domain>	それぞれのコントロールプレーンマシンについて、OpenShift Container Platform では、そのマシンに優先度 **0**、重み **10** およびポート **2380** の etcd サーバーの SRV DNS レコードも必要になります。3 つのコントロールプレーンマシンを使用するクラスターには以下のレコードが必要です。 `# _service._proto.name. TTL class SRV priority weight port target. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-0.<cluster_name>.<base_domain>. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-1.<cluster_name>.<base_domain>. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-2.<cluster_name>.<base_domain>.`

3.1.4. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> 1
```

1 ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

   このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 ~/**.ssh/id_rsa** などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

3.1.5. インストールプログラムの取得

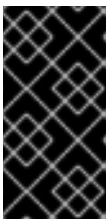
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。



重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

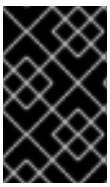
3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

3.1.6. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。

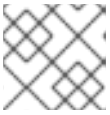


重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。
2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。



注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

3.1.7. インストール設定ファイルの手動作成

ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform のインストールでは、インストール設定ファイルを手動で生成する必要があります。

前提条件

- OpenShift Container Platform インストーラープログラムおよびクラスターのアクセストークンを取得します。

手順

1. 必要なインストールアセットを保存するためのインストールディレクトリーを作成します。

```
$ mkdir <installation_directory>
```

重要

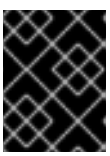
ディレクトリーを作成する必要があります。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

2. 以下の **install-config.yaml** ファイルテンプレートをカスタマイズし、これを **<installation_directory>** に保存します。

注記

この設定ファイル **install-config.yaml** に名前を付ける必要があります。

3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするために使用できるようにします。



重要

install-config.yaml ファイルは、インストールプロセスの次の手順で使用されます。この時点でこれをバックアップする必要があります。

3.1.7.1. ベアメタルのサンプル install-config.yaml ファイル

install-config.yaml ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master (7)
  replicas: 3 (8)
metadata:
  name: test (9)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (10)
    hostPrefix: 23 (11)
  networkType: OpenShiftSDN
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
pullSecret: '{"auths": ...}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
```

(1) クラスターのベースドメイン。すべての DNS レコードはこのベースのサブドメインである必要があり、クラスター名が含まれる必要があります。

(2) (5) **controlPlane** セクションは単一マッピングですが、コンピュートセクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

(3) (6) (7) 同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。同時マルチスレッドを無効にする場合は、これをすべてのクラスターマシンで無効にする必要があります。

重要

同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。

(4) **replicas** パラメーターの値を **0** に設定する必要があります。このパラメーターはクラスターが作成し、管理するワーカーの数を制御します。これは、ユーザーによってプロビジョニングされるインフラストラクチャーを使用する場合にクラスターが実行しない機能です。OpenShift Container Platform のインストールが終了する前に、クラスターが使用するワーカーマシンを手動でデプロイする必要があります。

(8) クラスターに追加するコントロールプレーンマシンの数。クラスターをこの値をクラスターの etcd エンドポイント数として使用するため、値はデプロイするコントロールプレーンマシンの数に一致する必要があります。

9 DNS レコードに指定したクラスター名。

10 Pod IP アドレスの割り当てに使用する IP アドレスのブロック。このブロックは既存の物理ネットワークと重複できません。これらの IP アドレスは Pod ネットワークに使用され、外部ネットワークから Pod にアクセスする必要がある場合には、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

11 それぞれの個別ノードに割り当てるサブネットプレフィックスの長さ。たとえば、**hostPrefix** が **23** に設定され、各ノードに指定の **cidr** から **/23** サブネットが割り当てられます (510 (2^(32 - 23) - 2) Pod IP アドレスが許可されます)。外部ネットワークからのノードへのアクセスを提供する必要がある場合には、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

12 サービス IP アドレスに使用する IP アドレスプール。1 つの IP アドレスプールのみを入力できます。外部ネットワークからサービスにアクセスする必要がある場合、ロードバランサーおよびルーターを、トラフィックを管理するように設定します。

13 プラットフォームを **none**に設定する必要があります。ベアメタルインフラストラクチャー用に追加のプラットフォーム設定変数を指定することはできません。

14 Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレット。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

15 Red Hat Enterprise Linux CoreOS (RHCOS) の **core** ユーザーのデフォルト SSH キーの公開部分。

注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストールプログラムに指定する必要があります。

3.1.8. Ignition 設定ファイルの作成

クラスターマシンは手動で起動する必要があるため、クラスターがマシンを作成するために必要な Ignition 設定ファイルを生成する必要があります。

重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターのインストールを完了し、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

前提条件

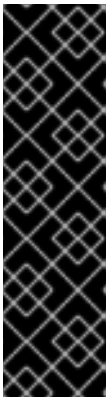
- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。

手順

1. Ignition 設定ファイルを取得します。
   -

```
$ ./openshift-install create ignition-configs --dir=<installation_directory> 1
```

1 **<installation_directory>** については、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。



重要

install-config.yaml ファイルを作成している場合、それが含まれるディレクトリーを指定します。または、空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

以下のファイルはディレクトリーに生成されます。

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

3.1.9. Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされるベアメタルインフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージまたはネットワーク PXE ブートを使用する手順を実行してマシンを作成することができます。

3.1.9.1. ISO イメージを使用した Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされるベアメタルインフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。ISO イメージを使用してマシンを作成することができます。

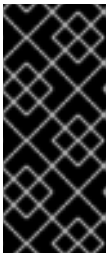
前提条件

- クラスターの Ignition 設定ファイルを取得していること。
- お使いのコンピューターからアクセスでき、作成するマシンがアクセスできる HTTP サーバーへのアクセスがあること。

手順

1. インストールプログラムが作成したコントロールプレーン、コンピュート、およびブートストラップ Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。

2. Red Hat カスタマーポータルの「製品のダウンロード」ページまたは「RRHCOS イメージミラー」ページからオペレーティングシステムのインスタンスをインストールするために優先される方法で必要な RHCOS イメージを取得します。



重要

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。

ISO ファイルおよび BIOS または UEFI ファイルをダウンロードする必要があります。これらのファイルの名前は以下の例のようになります。

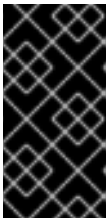
- ISO: **rhcos-<version>-<architecture>-installer.iso**
- 圧縮された metal BIOS: **rhcos-<version>-<architecture>-metal-bios.raw.gz**
- 圧縮された metal UEFI: **rhcos-<version>-<architecture>-metal-uefi.raw.gz**

3. BIOS または UEFI RHCOS イメージファイルのいずれかを HTTP サーバーにアップロードし、その URL をメモします。
4. ISO を使用し、RHCOS インストールを開始します。以下のインストールオプションのいずれかを使用します。
   - ディスクに ISO イメージを書き込み、これを直接起動します。
   - LOM インターフェースで ISO リダイレクトを使用します。
5. インスタンスの起動後に、**TAB** または **E** キーを押してカーネルコマンドラインを編集します。
6. パラメーターをカーネルコマンドラインに追加します。

```
coreos.inst=yes
coreos.inst.install_dev=sda 1
coreos.inst.image_url=<bare_metal_image_URL> 2
coreos.inst.ignition_url=http://example.com/config.ign 3
```

1. インストール先のシステムのブロックデバイスを指定します。
2. サーバーにアップロードした UEFI または BIOS イメージの URL を指定します。
3. このマシンタイプの Ignition 設定ファイルの URL を指定します。

7. Enter を押してインストールを完了します。RHCOS のインストール後に、システムは再起動します。システムの再起動後、指定した Ignition 設定ファイルを適用します。
8. 継続してクラスターのマシンを作成します。



重要

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。一部の Pod はデフォルトでコンピュートマシンにデプロイされるため、クラスターのインストール前に、2 つ以上のコンピュートマシンを作成します。

3.1.9.2. PXE または iPXE ブートによる Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされるベアメタルインフラストラクチャーにクラスターをインストールする前に、それが使用する RHCOS マシンを作成する必要があります。PXE または iPXE ブートを使用してマシンを作成することができます。

前提条件

- クラスターの Ignition 設定ファイルを取得していること。
- 適切な PXE または iPXE インフラストラクチャーを設定します。
- 使用しているコンピューターからアクセス可能な HTTP サーバーへのアクセスがあること。

手順

1. インストールプログラムが作成したマスター、ワーカーおよびブートストラップの Ignition 設定を HTTP サーバーにアップロードします。これらのファイルの URL をメモします。
2. Red Hat カスタマーポータルの「製品のダウンロード」ページまたは「RHCOS イメージミラー」ページから RHCOS ISO イメージ、圧縮されたメタル BIOS、 **kernel** および **initramfs** ファイルを取得します。

重要

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。

ファイル名には、OpenShift Container Platform のバージョン名が含まれます。以下の例のようになります。

- ISO: **rhcos-<version>-<architecture>-installer.iso**
- 圧縮された metal BIOS: **rhcos-<version>-<architecture>-metal-bios.raw.gz**
- **kernel**: **rhcos-<version>-<architecture>-installer-kernel**
- **initramfs**: **rhcos-<version>-<architecture>-installer-initramfs.img**

3. 圧縮された metal BIOS ファイルおよび **kernel** および **initramfs** ファイルを HTTP サーバーにアップロードします。
4. RHCOS のインストール後にマシンがローカルディスクから起動されるようにネットワークブートインフラストラクチャーを設定します。

5. RHCOS イメージに PXE または iPXE インストールを設定します。
ご使用の環境についての以下の例で示されるメニューエントリーのいずれかを変更し、イメージおよび Ignition ファイルが適切にアクセスできることを確認します。

- PXE の場合:

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-<architecture>-installer-kernel (1)
    APPEND ip=dhcp rd.neednet=1 initrd=http://<HTTP_server>/rhcos-<version>-
<architecture>-installer-initramfs.img console=tty0 console=ttyS0 coreos.inst=yes
coreos.inst.install_dev=sda coreos.inst.image_url=http://<HTTP_server>/rhcos-
<version>-<architecture>-metal-bios.raw.gz
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) HTTP サーバーにアップロードした **kernel** ファイルの場所を指定します。

(2) 複数の NIC を使用する場合、**ip** オプションに単一インターフェースを指定します。たとえば、**eno1** という名前の NIC で DHCP を使用するには、 **ip=eno1:dhcp** を設定します。

(3) HTTP サーバーにアップロードした RHCOS ファイルの場所を指定します。**initrd** パラメーター値は **initramfs** ファイルの場所であり、**coreos.inst.image_url** パラメーター値は圧縮された metal BIOS ファイルの場所、および **coreos.inst.ignition_url** パラメーター値はブートストラップ Ignition 設定ファイルの場所になります。

- iPXEの場合:

```
kernel  http://<HTTP_server>/rhcos-<version>-<architecture>-installer-kernel ip=dhcp
rd.neednet=1 initrd=http://<HTTP_server>/rhcos-<version>-<architecture>-installer-
initramfs.img console=tty0 console=ttyS0 coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal-
bios.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1) (2)
initrd http://<HTTP_server>/rhcos-<version>-<architecture>-installer-initramfs.img (3)
boot
```

(1) HTTP サーバーにアップロードした RHCOS ファイルの場所を指定します。**kernel** パラメーター値は **kernel** ファイルの場所であり、**initrd** パラメーター値は **initramfs** ファイルの場所、**coreos.inst.image_url** パラメーター値は圧縮された metal BIOS ファイルの場所、および **coreos.inst.ignition_url** パラメーター値はブートストラップ Ignition 設定ファイルの場所になります。

(2) 複数の NIC を使用する場合、**ip** オプションに単一インターフェースを指定します。たとえば、**eno1** という名前の NIC で DHCP を使用するには、 **ip=eno1:dhcp** を設定します。

(3) HTTP サーバーにアップロードした **initramfs** ファイルの場所を指定します。

6. UEFI を使用する場合、ダウンロードした ISO に含まれている **grub.conf** ファイルを編集して以下のインストールオプションを組み込みます。

```
menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class
gnu --class os {
 linux /images/vmlinuz nomodeset rd.neednet=1 coreos.inst=yes coreos.inst.install_dev=sda
coreos.inst.image_url=http://<HTTP_server>/rhcos-<version>-<architecture>-metal-
uefi.raw.gz coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1)
 initrd http://<HTTP_server>/rhcos-<version>-<architecture>-installer-initramfs.img (2)
}
```

(1) **coreos.inst.image_url** パラメーター値には、HTTP サーバーにアップロードした圧縮された metal UEFI ファイルの場所を指定します。**coreos.inst.ignition_url** には、HTTP サーバーにアップロードしたブートストラップ Ingition 設定ファイルの場所を指定します。

(2) HTTP サーバーにアップロードした **initramfs** ファイルの場所を指定します。

7. 継続してクラスターのマシンを作成します。

重要

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。一部の Pod はデフォルトでコンピュートマシンにデプロイされるため、クラスターのインストール前に、2 つ以上のコンピュートマシンを作成します。

3.1.10. クラスターの作成

OpenShift Container Platform クラスターを作成するには、インストールプログラムで生成した Ignition 設定ファイルを使用してプロビジョニングしたマシンで、ブートストラッププロセスが完了するのを待機します。

前提条件

- クラスターに必要なインフラストラクチャーを作成すること。
- インストールプログラムを取得し、クラスターの Ignition 設定ファイルを生成していること。
- クラスターの RHCOS マシンを作成するために Ignition 設定ファイルを使用していること。
- 使用するマシンでインターネットに直接アクセスできること。

手順

1. ブートストラッププロセスをモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for bootstrap-complete \ (1)
    --log-level info (2)

INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.13.4+b626c2fe1 up
INFO Waiting up to 30m0s for the bootstrap-complete event...
```

(1) **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

2 異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。

Kubernetes API サーバーでこれがコントロールプレーンマシンにブートストラップされていることを示すシグナルが出されるとコマンドは成功します。

2. ブートストラッププロセスが完了したら、ブートストラップマシンをロードバランサーから削除します。

重要

この時点で、ブートストラップマシンをロードバランサーから削除する必要があります。さらに、マシン自体を削除し、再フォーマットすることができます。

3.1.11. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1

$ oc whoami
system:admin
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

3.1.12. マシンの CSR の承認

マシンをクラスターに追加する際に、追加したそれぞれのマシンについて 2 つの保留状態の証明書署名要求 (CSR) が生成されます。これらの CSR が承認されていることを確認するか、または必要な場合はそれらを承認してください。

前提条件

- マシンをクラスターに追加していること。

手順

1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes

NAME     STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.13.4+b626c2fe1
master-1  Ready     master  63m  v1.13.4+b626c2fe1
master-2  Ready     master  64m  v1.13.4+b626c2fe1
worker-0  NotReady  worker  76s  v1.13.4+b626c2fe1
worker-1  NotReady  worker  70s  v1.13.4+b626c2fe1
```

出力には作成したすべてのマシンが一覧表示されます。

2. 保留中の証明書署名要求 (CSR) を確認し、クラスターに追加したそれぞれのマシンのクライアントおよびサーバー要求に **Pending** または **Approved** ステータスが表示されていることを確認します。

```
$ oc get csr

NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending (1)
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal                        Pending (2)
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal                        Pending
...
```

(1) クライアント要求の CSR。

(2) サーバー要求の CSR。

この例では、2 つのマシンがクラスターに参加しています。この一覧にはさらに多くの承認された CSR が表示される可能性があります。

3. 追加したマシンの保留中の CSR すべてが **Pending** ステータスになった後に CSR が承認されない場合には、クラスターマシンの CSR を承認します。

注記

CSR のローテーションは自動的に実行されるため、クラスターにマシンを追加後 1 時間以内に CSR を承認してください。1 時間以内に承認しない場合には、証明書のローテーションが行われ、各ノードに 3 つ以上の証明書が存在するようになります。これらの証明書すべてを承認する必要があります。最初の CSR の承認後、後続のノードクライアント CSR はクラスターの **kube-controller-manger** によって自動的に承認されます。kubelet 提供証明書の要求を自動的に承認する方法を実装する必要があります。

- それらを個別に承認するには、それぞれの有効な CSR について以下のコマンドを実行します。

```
$ oc adm certificate approve <csr_name> (1)
```

1 **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

3.1.13. Operator の初期設定

コントロールプレーンの初期化後に、一部の Operator を利用可能にするためにそれらをすぐに設定する必要があります。

前提条件

- コントロールプレーンが初期化されていること。

手順

1. クラスターコンポーネントがオンラインになることを確認します。

```
$ watch -n5 oc get clusteroperators

NAME                                 VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                       4.1.0     True        False         False      69s
cloud-credential                     4.1.0     True        False         False      12m
cluster-autoscaler                   4.1.0     True        False         False      11m
console                              4.1.0     True        False         False      46s
dns                                  4.1.0     True        False         False      11m
image-registry                       4.1.0     False       True          False      5m26s
ingress                              4.1.0     True        False         False      5m36s
kube-apiserver                       4.1.0     True        False         False      8m53s
kube-controller-manager              4.1.0     True        False         False      7m24s
kube-scheduler                       4.1.0     True        False         False      12m
machine-api                          4.1.0     True        False         False      12m
machine-config                       4.1.0     True        False         False      7m36s
marketplace                          4.1.0     True        False         False      7m54m
monitoring                           4.1.0     True        False         False      7h54s
network                              4.1.0     True        False         False      5m9s
node-tuning                          4.1.0     True        False         False      11m
openshift-apiserver                  4.1.0     True        False         False      11m
openshift-controller-manager         4.1.0     True        False         False      5m943s
openshift-samples                    4.1.0     True        False         False      3m55s
operator-lifecycle-manager           4.1.0     True        False         False      11m
operator-lifecycle-manager-catalog   4.1.0     True        False         False      11m
service-ca                           4.1.0     True        False         False      11m
service-catalog-apiserver            4.1.0     True        False         False      5m26s
service-catalog-controller-manager   4.1.0     True        False         False      5m25s
storage                              4.1.0     True        False         False      5m30s
```

2. 利用不可の Operator を設定します。

3.1.13.1. イメージレジストリーストレージの設定

image-registry Operator が利用できない場合、そのストレージを設定する必要があります。実稼働クラスターに必要な PersistentVolume の設定方法と、実稼働用ではないクラスターにのみ使用できる空のディレクトリーをストレージの場所として設定する方法が表示されます。

3.1.13.1.1. ベアメタルの場合のレジストリーストレージの設定

クラスター管理者は、インストール後にレジストリーをストレージを使用できるように設定する必要があります。

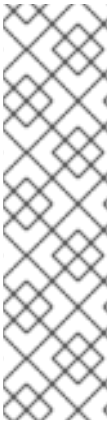
前提条件

- クラスター管理者のパーミッション。
- ベアメタル上のクラスター。
- **ReadWriteMany** アクセスモードのプロビジョニングされた永続ボリューム (PV)(例: **NFS**)。
- 容量は「100Gi」以上である。

手順

1. レジストリーをストレージを使用できるように設定するには、**configs.imageregistry/cluster** リソースの **spec.storage.pvc** を変更します。
2. レジストリー Pod がないことを確認します。

```
$ oc get pod -n openshift-image-registry
```



注記

ストレージタイプが **emptyDIR** の場合、レプリカ数が **1** を超えることはありません。ストレージタイプが **NFS** で、レジストリー Pod を **replica>1** を設定してスケールアップする必要がある場合、**no_wdelay** マウントオプションを有効にする必要があります。以下は例になります。

```
# cat /etc/exports
/mnt/data *(rw,sync,no_wdelay,no_root_squash,insecure,fsid=0)
sh-4.3# exportfs -rv
exporting *:/mnt/data
```

1. レジストリー設定を確認します。

```
$ oc edit configs.imageregistry.operator.openshift.io

storage:
  pvc:
    claim:
```

claim フィールドを空のままにし、 **image-registry-storage** PVC の自動作成を可能にします。

2. **clusteroperator** ステータスを確認します。

```
$ oc get clusteroperator image-registry
```

3.1.13.1.2. 実稼働以外のクラスターでのイメージレジストリーのストレージの設定

イメージレジストリー Operator のストレージを設定する必要があります。実稼働用以外のクラスターの場合、イメージレジストリーは空のディレクトリーに設定することができます。これを実行する場合、レジストリーを再起動するとすべてのイメージが失われます。

手順

- イメージレジストリーストレージを空のディレクトリーに設定するには、以下を実行します。

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"storage":{"emptyDir":{}}}}'
```

警告

実稼働用以外のクラスターにのみこのオプションを設定します。

イメージレジストリー Operator がそのコンポーネントを初期化する前にこのコマンドを実行する場合、**oc patch** コマンドは以下のエラーを出して失敗します。

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

数分待機した後に、このコマンドを再び実行します。

3.1.14. ユーザーによってプロビジョニングされるインフラストラクチャーでのインストールの完了

Operator 設定の完了後に、提供するインフラストラクチャーでのクラスターのインストールを終了できます。

前提条件

- コントロールプレーンが初期化されていること。
- Operator の初期設定を完了していること。

手順

1. すべてのクラスターコンポーネントがオンラインであることを確認します。

```
$ watch -n5 oc get clusteroperators

NAME                      VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication            4.1.0     True        False         False      10m
cloud-credential          4.1.0     True        False         False      22m
cluster-autoscaler        4.1.0     True        False         False      21m
console                   4.1.0     True        False         False      10m
dns                       4.1.0     True        False         False      21m
```

```
image-registry                 4.1.0     True        False         False      16m
ingress                        4.1.0     True        False         False      16m
kube-apiserver                 4.1.0     True        False         False      19m
kube-controller-manager        4.1.0     True        False         False      18m
kube-scheduler                 4.1.0     True        False         False      22m
machine-api                    4.1.0     True        False         False      22m
machine-config                 4.1.0     True        False         False      18m
marketplace                    4.1.0     True        False         False      18m
monitoring                     4.1.0     True        False         False      18m
network                        4.1.0     True        False         False      16m
node-tuning                    4.1.0     True        False         False      21m
openshift-apiserver            4.1.0     True        False         False      21m
openshift-controller-manager   4.1.0     True        False         False      17m
openshift-samples              4.1.0     True        False         False      14m
operator-lifecycle-manager     4.1.0     True        False         False      21m
operator-lifecycle-manager-catalog   4.1.0     True        False         False      21m
service-ca                     4.1.0     True        False         False      21m
service-catalog-apiserver      4.1.0     True        False         False      16m
service-catalog-controller-manager   4.1.0     True        False         False      16m
storage                        4.1.0     True        False         False      16m
```

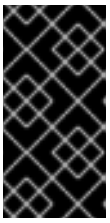
すべてのクラスター Operator が **AVAILABLE** の場合、インストールを完了することができます。

2. クラスターの完了をモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for install-complete 1
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

Cluster Version Operator が Kubernetes API サーバーから OpenShift Container Platform クラスターのデプロイを終了するとコマンドは成功します。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

3. Kubernetes API サーバーが Pod と通信していることを確認します。

   a. すべての Pod の一覧を表示するには、以下のコマンドを使用します。

```
$ oc get pods --all-namespaces

NAMESPACE                       NAME                                         READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running   1          9m
openshift-apiserver             apiserver-67b9g                              1/1     Running     0
3m
```

```
openshift-apiserver            apiserver-ljcmx                         1/1     Running     0
1m
openshift-apiserver            apiserver-z25h4                         1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running     0          5m
...
```

b. 以下のコマンドを使用して、直前のコマンドの出力に一覧表示される Pod のログを表示します。

```
$ oc logs <pod_name> -n <namespace> 1
```

1 直前のコマンドの出力にあるように、Pod 名および namespace を指定します。

Pod のログが表示される場合、Kubernetes API サーバーはクラスターマシンと通信できます。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

第4章 VSPHERE へのインストール

4.1. クラスターの VSPHERE へのインストール

OpenShift Container Platform バージョン 4.1 では、プロビジョニングする VMware vSphere インフラストラクチャーにクラスターをインストールできます。

前提条件

- クラスターの永続ストレージをプロビジョニングします。プライベートイメージレジストリーをデプロイするには、ストレージで ReadWriteMany アクセスモードを指定する必要があります。
- OpenShift Container Platform のインストールおよび更新 プロセスについての詳細を確認します。
- ファイアウォールを使用する場合、Red Hat Insights にアクセスできるように設定 する必要があります。

4.1.1. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.1 では、Telemetry はクラスターの健全性および更新の正常な実行についてのメトリクスを提供するコンポーネントです。Red Hat がご使用のクラスターが有効なサブスクリプションを使用しているかどうかを判別できるかどうかを含め、サブスクリプション管理を実行できるようにするには、Telemetry サービスを使用し、Red Hat OpenShift Cluster Manager ページにアクセスできる必要があります。

非接続のサブスクリプション管理はオプションとして選択できないため、データを Red Hat に戻すことを選択しないことと、サブスクリプションを有効にすることは両立できません。非接続のサブスクリプション管理のサポートは OpenShift Container Platform の今後のリリースで追加される可能性があります。

重要

マシンには、クラスターをインストールするためにインターネットへの直接のアクセスが必要です。

インターネットへのアクセスは以下を実行するために必要です。

- Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページにアクセスして、インストールプログラムをダウンロードします。
- クラスターのインストールに必要なパッケージを取得するための Quay.io へのアクセス
- クラスターの更新を実行するために必要なパッケージの取得
- サブスクリプション管理を実行するための Red Hat の SaaS (サービスとしてのソフトウェア) ページへのアクセス

4.1.2. VMware vSphere インフラストラクチャーの要件

OpenShift Container Platform クラスターを、VMware vSphere のバージョン 6.5 または 6.7U2 以降のインスタンスにインストールする必要があります。

VMware では、vSphere バージョン 6.7 U2 以降を OpenShift Container Platform クラスターで使用することを推奨しています。 vSphere 6.7U2 には以下が含まれます。

- VMware NSX-T のサポート
- インツリー (in-tree) VCP を使用する vSAN、VMFS および NFS のサポート

vSphere 6.5 とハードウェアのバージョン 13 の使用がサポートされていますが、OpenShift Container Platform クラスターは以下の制限を受けます。

- NSX-T SDN はサポートされません。
- OpenShift Container Platform がサポートする別の SDN またはストレージプロバイダーを使用する必要があります。

vSphere バージョン 6.5 インスタンスを使用している場合は、OpenShift Container Platform をインストールする前に 6.7U2 にアップグレードすることを検討してください。

重要

OpenShift Container Platform をインストールする前に、ESXi ホストの時間が同期されていることを確認する必要があります。VMware ドキュメントの「Edit Time Configuration for a Host」を参照してください。

4.1.3. ユーザーによってプロビジョニングされるインフラストラクチャーを使用する場合のクラスターのマシン要件

ユーザーによってプロビジョニングされるインフラストラクチャーを含むクラスターの場合、必要なマシンすべてをデプロイする必要があります。

4.1.3.1. 必要なマシン

最小の OpenShift Container Platform クラスターでは以下のホストが必要です。

- 1 つのブートストラップマシン
- 3 つのコントロールプレーン、またはマスター、マシン
- 少なくとも 2 つのコンピュートマシン (ワーカーマシンとしても知られる)。

注記

クラスターでは、ブートストラップマシンが OpenShift Container Platform クラスターを 3 つのコントロールプレーンマシンにデプロイする必要があります。クラスターのインストール後にブートストラップマシンを削除できます。

重要

クラスターの高可用性を維持するには、これらのクラスターマシンについて別個の物理ホストを使用します。

ブートストラップおよびコントロールプレーンマシンでは、Red Hat Enterprise Linux CoreOS (RHCOS) をオペレーティングシステムとして使用する必要があります。

RHCOS は Red Hat Enterprise Linux 8 をベースとしており、そのハードウェア認定および要件が継承されることに注意してください。「Red Hat Enterprise Linux テクノロジーの機能と制限 」を参照してください。

4.1.3.2. ネットワーク接続の要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定ファイルをフェッチする必要があります。初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーが必要になります。

4.1.3.3. 最小リソース要件

それぞれのクラスターマシンは、以下の最小要件を満たしている必要があります。

マシン	Operating System	vCPU	RAM	ストレージ
ブートストラップ	RHCOS	4	16 GB	120 GB
コントロールプレーン	RHCOS	4	16 GB	120 GB
コンピュート	RHCOS または RHEL 7.6	2	8 GB	120 GB

4.1.3.4. 証明書署名要求の管理

ユーザーがプロビジョニングするインフラストラクチャーを使用する場合、クラスターの自動マシン管理へのアクセスは制限されるため、インストール後にクラスターの証明書署名要求 (CSR) のメカニズムを提供する必要があります。**kube-controller-manager** は kubelet クライアント CSR のみを承認します。**machine-approver** は、kubelet 認証情報を使用して要求される提供証明書の有効性を保証できません。適切なマシンがこの要求を発行したかどうかを確認できないためです。kubelet 提供証明書の要求の有効性を検証し、それらを承認する方法を判別し、実装する必要があります。

4.1.4. ユーザーによってプロビジョニングされるインフラストラクチャーの作成

ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform クラスターをデプロイする前に、基礎となるインフラストラクチャーを作成する必要があります。

前提条件

- クラスターでサポートするインフラストラクチャーを作成する前に、「OpenShift Container Platform 4.x Tested Integrations」ページを参照してください。

手順

1. DHCP を設定します。
2. 必要なロードバランサーをプロビジョニングします。
3. マシンのポートを設定します。

4. DNS を設定します。

5. ネットワーク接続を確認します。

4.1.4.1. ユーザーによってプロビジョニングされるインフラストラクチャのネットワーク要件

すべての Red Hat Enterprise Linux CoreOS (RHCOS) マシンでは、起動時に **initramfs** のネットワークがマシン設定サーバーから Ignition 設定をフェッチする必要があります。

初回の起動時に、Ignition 設定ファイルをダウンロードできるようネットワーク接続を確立するために、マシンには DHCP サーバーが必要になります。

クラスターのマシンを長期間管理するために DHCP サーバーを使用することが推奨されています。DHCP サーバーが永続 IP アドレスおよびホスト名をクラスターマシンに提供するように設定されていることを確認します。

Kubernetes API サーバーはクラスターマシンのノード名を解決できる必要があります。API サーバーおよびワーカーノードが異なるゾーンに置かれている場合、デフォルトの DNS 検索ゾーンを、API サーバーでノード名を解決できるように設定することができます。もう 1 つの実行可能な方法として、ノードオブジェクトとすべての DNS 要求の両方において、ホストを完全修飾ドメイン名で常に参照することができます。

マシン間のネットワーク接続を、クラスターのコンポーネントが通信できるように設定する必要があります。すべてのマシンではクラスターの他のすべてのマシンのホスト名を解決できる必要があります。

表4.1 すべてのマシンに対応するすべてのマシン

プロトコル	ポート	説明
ICMP	該当なし	ネットワーク到達性のテスト
TCP	**9000**-**9999**	ホストレベルのサービス。 ポート **9100**-**9101** のノードエクスポーター、ポート **9099** の Cluster Version Operator が含まれます。
	10250-**10259**	Kubernetes が予約するデフォルトポート
	10256	openshift-sdn
UDP	**4789**	VXLAN および GENEVE
	6081	VXLAN および GENEVE
	9000-**9999**	ポート **9100**-**9101** のノードエクスポーターを含む、ホストレベルのサービス。
TCP/UDP	**30000**-**32767**	Kubernetes NodePort

表4.2 コントロールプレーンへのすべてのマシン

プロトコル	ポート	説明
TCP	**2379-2380**	etcd サーバー、ピア、およびメトリクスポート
	6443	Kubernetes API

ネットワークトポロジー要件
クラスター用にプロビジョニングするインフラストラクチャーは、ネットワークトポロジーの以下の要件を満たす必要があります。

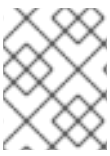
重要

OpenShift Container Platform では、すべてのノードが、プラットフォームコンテナーのイメージをプルし、Telemetry データを Red Hat に提供するためにインターネットへの直接のアクセスが必要です。

ロードバランサー

OpenShift Container Platform をインストールする前に、2 つの Layer 4 ロードバランサーをプロビジョニングする必要があります。API には 1 つのロードバランサーが必要で、デフォルトの Ingress コントローラーには、Ingress をアプリケーションに提供する 2 番目のロードバランサーが必要です。

ポート	マシン	内部	外部	説明
6443	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。	x	x	Kubernetes API サーバー
22623	ブートストラップおよびコントロールプレーン。ブートストラップマシンがクラスターのコントロールプレーンを初期化した後に、ブートストラップマシンをロードバランサーから削除します。	x		マシン設定サーバー
443	デフォルトで Ingress ルーター Pod、コンピュート、またはワーカーを実行するマシン。	x	x	HTTPS トラフィック
80	デフォルトで Ingress ルーター Pod、コンピュート、またはワーカーを実行するマシン。	x	x	HTTP トラフィック



注記

Ingress ルーターの作業用の設定が OpenShift Container Platform クラスターに必要です。コントロールプレーンの初期化後に Ingress ルーターを設定する必要があります。

Ethernet アダプターのハードウェアアドレス要件
クラスターの仮想マシンをプロビジョニングする場合、各仮想マシンに設定されたイーサネットインターフェースは VMware Organizationally Unique Identifier (OUI) 割り当て範囲から MAC アドレスを使用する必要があります。

- **00:05:69:00:00:00** - **00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00** - **00:0c:29:FF:FF:FF**
- **00:1c:14:00:00:00** - **00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00** - **00:50:56:FF:FF:FF**

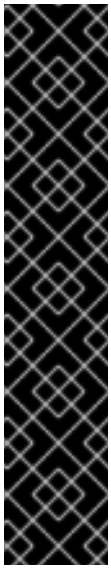
VMware OUI 外の MAC アドレスが使用される場合、クラスターのインストールは成功しません。

4.1.4.2. ユーザーによってプロビジョニングされるインフラストラクチャーの DNS 要件

以下の DNS レコードは、ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform クラスターに必要です。各レコードで、 **<cluster_name>** はクラスター名で、**<base_domain>** は、**install-config.yaml** ファイルに指定するクラスターのベースドメインです。

表4.3 必要な DNS レコード

Component	レコード	説明
Kubernetes API	**api.<cluster_name>.<base_domain>**	この DNS レコードは、コントロールプレーンマシンのロードバランサーを参照する必要があります。このレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。

Component	レコード	説明
	api-int.<cluster_name>.<base_domain>	この DNS レコードは、コントロールプレーンマシンのロードバランサーを参照する必要があります。このレコードは、クラスター内のすべてのノードで解決できる必要があります。  **重要** API サーバーは、Kubernetes に記録されるホスト名でワーカーノードを解決できる必要があります。これがノード名を解決できない場合、プロキシーされる API 呼び出しが失敗し、Pod からログを取得できなくなる可能性があります。
Routes	***.apps.<cluster_name>.<base_domain>**	Ingress ルーター Pod を実行するマシンをターゲットにするロードバランサーを参照するワイルドカード DNS レコードです。このレコードは、クラスター外のクライアントおよびクラスター内のすべてのノードで解決できる必要があります。

Component	レコード	説明
etcd	**etcd-<index>.<cluster_name>.<base_domain>**	OpenShift Container Platform では、各 etcd インスタンスの DNS レコードがインスタンスをホストするコントロールプレーンマシンを参照する必要があります。etcd インスタンスは **<index>** 値によって差別化されます。この値は **0** で始まり、**n-1** で終了します。ここで、**n** はクラスターのコントロールプレーンマシンの数です。DNS レコードはコントロールプレーンマシンのユニキャスト IPV4 アドレスに解決し、レコードはクラスター内のすべてのノードで解決可能である必要があります。
	_etcd-server-ssl._tcp.<cluster_name>.<base_domain>	それぞれのコントロールプレーンマシンについて、OpenShift Container Platform では、そのマシンに優先度 **0**、重み **10** およびポート **2380** の etcd サーバーの SRV DNS レコードも必要になります。3 つのコントロールプレーンマシンを使用するクラスターには以下のレコードが必要です。 `# _service._proto.name. TTL class SRV priority weight port target. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-0.<cluster_name>.<base_domain>. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-1.<cluster_name>.<base_domain>. _etcd-server-ssl._tcp.<cluster_name>.<base_domain> 86400 IN SRV 0 10 2380 etcd-2.<cluster_name>.<base_domain>.`

4.1.5. SSH プライベートキーの生成およびエージェントへの追加

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

このキーを使用して、ユーザー **core** としてマスターノードに対して SSH を実行できます。クラスターをデプロイする際に、キーは **core** ユーザーの ~/**.ssh/authorized_keys** 一覧に追加されます。

注記

AWS キーペアなどのプラットフォームに固有の方法で設定したキーではなく、ローカルキーを使用する必要があります。

手順

1. パスワードなしの認証に設定されている SSH キーがコンピューター上にない場合は、これを作成します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ ssh-keygen -t rsa -b 4096 -N '' \
    -f <path>/<file_name> 1
```

1 ~/**.ssh/id_rsa** などの、SSH キーのパスおよびファイル名を指定します。

   このコマンドを実行すると、指定した場所にパスワードを必要としない SSH キーが生成されます。

2. **ssh-agent** プロセスをバックグラウンドタスクとして開始します。

```
$ eval "$(ssh-agent -s)"

Agent pid 31874
```

3. SSH プライベートキーを **ssh-agent** に追加します。

```
$ ssh-add <path>/<file_name> 1

Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

1 ~/**.ssh/id_rsa** などの、SSH プライベートキーのパスおよびファイル名を指定します。

次のステップ

OpenShift Container Platform をインストールする際に、SSH パブリックキーをインストーラーに指定します。クラスターを独自にプロビジョニングするインフラストラクチャーにインストールする場合は、このキーをクラスターのマシンに指定する必要があります。

4.1.6. インストールプログラムの取得

OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

前提条件

- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、300 MB のローカルディスク領域が必要です。

手順

1. Red Hat OpenShift Cluster Manager サイトの「Infrastructure Provider」ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。
2. 選択するインストールタイプのページに移動し、オペレーティングシステムのインストールプログラムをダウンロードし、ファイルをインストール設定ファイルを保存するディレクトリーに配置します。

重要

インストールプログラムは、クラスターのインストールに使用するコンピューターにいくつかのファイルを作成します。クラスターインストールの完了後は、インストールプログラムおよびインストールプログラムが作成するファイルの両方を保持する必要があります。

3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ tar xvf <installation_program>.tar.gz
```

4. Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから、インストールプルシークレットを **.txt** ファイルとしてダウンロードするか、またはこれをクリップボードにコピーします。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

4.1.7. インストール設定ファイルの手動作成

ユーザーによってプロビジョニングされるインフラストラクチャーを使用する OpenShift Container Platform のインストールでは、インストール設定ファイルを手動で生成する必要があります。

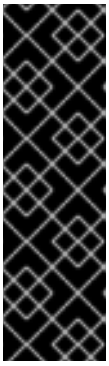
前提条件

- OpenShift Container Platform インストーラープログラムおよびクラスターのアクセストークンを取得します。

手順

1. 必要なインストールアセットを保存するためのインストールディレクトリーを作成します。

```
$ mkdir <installation_directory>
```



重要

ディレクトリーを作成する必要があります。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

2. 以下の **install-config.yaml** ファイルテンプレートをカスタマイズし、これを **<installation_directory>** に保存します。

注記

この設定ファイル **install-config.yaml** に名前を付ける必要があります。

3. **install-config.yaml** ファイルをバックアップし、これを複数のクラスターをインストールするために使用できるようにします。

重要

install-config.yaml ファイルは、インストールプロセスの次の手順で使用されます。この時点でこれをバックアップする必要があります。

4.1.7.1. VMware vSphere のサンプル install-config.yaml ファイル

install-config.yaml ファイルをカスタマイズして、OpenShift Container Platform クラスターのプラットフォームについての詳細を指定するか、または必要なパラメーターの値を変更することができます。

```
apiVersion: v1
baseDomain: example.com (1)
compute:
- hyperthreading: Enabled (2) (3)
  name: worker
  replicas: 0 (4)
controlPlane:
  hyperthreading: Enabled (5) (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
platform:
  vsphere:
    vcenter: your.vcenter.server (9)
    username: username (10)
    password: password (11)
    datacenter: datacenter (12)
    defaultDatastore: datastore (13)
pullSecret: '{"auths": ...}' (14)
sshKey: 'ssh-ed25519 AAAA...' (15)
```

1 クラスターのベースドメイン。すべての DNS レコードはこのベースのサブドメインである必要があり、クラスター名が含まれる必要があります。

2 5 **controlPlane** セクションは単一マッピングですが、コンピュートセクションはマッピングのシーケンスになります。複数の異なるデータ構造の要件を満たすには、 **compute** セクションの最初の行はハイフン **-** で始め、**controlPlane** セクションの最初の行はハイフンで始めることができません。どちらのセクションも、現時点では単一のマシンプールを定義しますが、OpenShift Container Platform の今後のバージョンでは、インストール時の複数のコンピュートプールの定義をサポートする可能性があります。1 つのコントロールプレーンプールのみが使用されます。

3 6 同時マルチスレッドまたは **hyperthreading** を有効/無効にするかどうか。デフォルトでは、同時スレッドはマシンのコアのパフォーマンスを上げるために有効にされます。パラメーター値を **Disabled** に設定するとこれを無効にすることができます。同時マルチスレッドを無効にする場合は、これをすべてのクラスターマシンで無効にする必要があります。

重要

同時スレッドを無効にする場合は、容量計画においてマシンパフォーマンスの大幅な低下が考慮に入れられていることを確認します。同時マルチスレッドを無効にする場合は、マシンで最低でも 8 CPU および 32 GB の RAM を使用する必要があります。

4 **replicas** パラメーターの値を **0** に設定する必要があります。このパラメーターはクラスターが作成し、管理するワーカーの数を制御します。これは、ユーザーによってプロビジョニングされるインフラストラクチャーを使用する場合にクラスターが実行しない機能です。OpenShift Container Platform のインストールが終了する前に、クラスターが使用するワーカーマシンを手動でデプロイする必要があります。

7 クラスターに追加するコントロールプレーンマシンの数。クラスターをこの値をクラスターの etcd エンドポイント数として使用するため、値はデプロイするコントロールプレーンマシンの数に一致する必要があります。

8 DNS レコードに指定したクラスター名。

9 vCenter サーバーの完全修飾ホスト名または IP アドレス。

10 サーバーにアクセスするユーザーの名前。このユーザーには、少なくとも vSphere の動的永続ボリュームのプロビジョニングに必要なロールおよび権限がなければなりません。

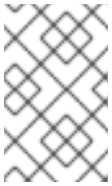
11 vSphere ユーザーに関連付けられたパスワード。

12 vSphere データセンター。

13 使用するデフォルトの vSphere データストア。

14 Red Hat OpenShift Cluster Manager サイトの「Pull Secret」ページから取得したプルシークレット。このプルシークレットを使用し、OpenShift Container Platform コンポーネントのコンテナーイメージを提供する、Quay.io などの組み込まれた各種の認証局によって提供されるサービスで認証できます。

15 Red Hat Enterprise Linux CoreOS (RHCOS) の **core** ユーザーのデフォルト SSH キーの公開部分。

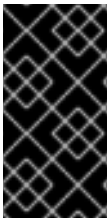


注記

インストールのデバッグまたは障害復旧を実行する必要のある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーをインストーラーに指定する必要があります。

4.1.8. Ignition 設定ファイルの作成

クラスターマシンは手動で起動する必要があるため、クラスターがマシンを作成するために必要な Ignition 設定ファイルを生成する必要があります。



重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターのインストールを完了し、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

前提条件

- OpenShift Container Platform インストールプログラム、およびクラスターのプルシークレットを取得すること。

手順

1. Ignition 設定ファイルを取得します。

```
$ ./openshift-install create ignition-configs --dir=<installation_directory> 1
```

1 **<installation_directory>** については、インストールプログラムが作成するファイルを保存するためにディレクトリー名を指定します。

重要

install-config.yaml ファイルを作成している場合、それが含まれるディレクトリーを指定します。または、空のディレクトリーを指定します。ブートストラップ X.509 証明書などの一部のインストールアセットの有効期限は短く設定されているため、インストールディレクトリーを再利用することができません。別のクラスターインストールの個別のファイルを再利用する必要がある場合は、それらをディレクトリーにコピーすることができます。ただし、インストールアセットのファイル名はリリース間で変更される可能性があります。インストールファイルを以前のバージョンの OpenShift Container Platform からコピーする場合は注意してコピーを行ってください。

以下のファイルはディレクトリーに生成されます。

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
```

```
├── master.ign
├── metadata.json
└── worker.ign
```

4.1.9. vSphere での Red Hat Enterprise Linux CoreOS (RHCOS) マシンの作成

ユーザーによってプロビジョニングされるインフラストラクチャーが含まれるクラスターを VMware vSphere にインストールする前に、それが使用する RHCOS マシンを vSphere ホストに作成する必要があります。

前提条件

- クラスターの Ignition 設定ファイルを取得していること。
- お使いのコンピューターからアクセスでき、作成するマシンがアクセスできる HTTP サーバーへのアクセスがあること。
- vSphere クラスターを作成します。

手順

1. **<installation_directory>/bootstrap.ign** という名前のインストールプログラムが作成したブートストラップ Ignition 設定ファイルを HTTP サーバーにアップロードします。このファイルの URL をメモします。
   ブートストラップ Ignition 設定ファイルはサイズが大きすぎて vApp プロパティーに適さないため、これをホストする必要があります。
2. ブートストラップノードの以下の二次的な Ignition 設定ファイルを、**<installation_directory>/append-bootstrap.ign** としてコンピューターに保存します。

```
{
  "ignition": {
    "config": {
      "append": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "2.1.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

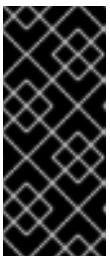
1 ホストしているブートストラップの Ignition 設定ファイルの URL を指定します。

ブートストラップマシンの仮想マシン (VM) を作成する場合に、この Ignition 設定ファイルを使用します。

3. マスター、ワーカー、および二次的なブートストラップ Ignition 設定ファイルを Base64 エンコーディングに変換します。
たとえば、Linux オペレーティングシステムを使用する場合、 **base64** コマンドを使用してファイルをエンコードできます。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
$ base64 -w0 <installation_directory>/append-bootstrap.ign >
<installation_directory>/append-bootstrap.64
```

4. Red Hat カスタマーポータルの「製品のダウンロード」ページまたは「RHCOS イメージミラー」ページから、RHCOS OVA イメージを取得します。



重要

RHCOS イメージは OpenShift Container Platform の各リリースごとに変更されない可能性があります。インストールする OpenShift Container Platform バージョンと等しいか、それ以下のバージョンの内で最も新しいバージョンのイメージをダウンロードする必要があります。利用可能な場合は、OpenShift Container Platform バージョンに一致するイメージのバージョンを使用します。

ファイル名には、**rhcos-<version>-<architecture>-vmware.ova** 形式の OpenShift Container Platform のバージョン番号が含まれます。

5. vSphere クライアントで、仮想マシンを保管するフォルダーをデータセンターに作成します。
   a. **VMs and Templates** ビューをクリックします。
   b. データセンターの名前を右クリックします。
   c. **New Folder → New VM and Template Folder**をクリックします。
   d. 表示されるウィンドウで、フォルダー名を入力します。フォルダー名は、**install-config.yaml** ファイルで指定したクラスター名と一致している必要があります。
6. vSphere クライアントで、OVA イメージのテンプレートを作成します。

注記

以下の手順では、すべてのクラスターマシンに同じテンプレートを使用し、仮想マシンのプロビジョニングの際に該当するマシンタイプの Ignition 設定ファイルの場所を指定します。

   a. **Hosts and Clusters** タブで、クラスターの名前を右クリックし、**Deploy OVF Template** をクリックします。
   b. **Select an OVF** タブで、ダウンロードした RHCOS OVA ファイルの名前を指定します。
   c. **Select a name and folder**タブで、RHCOS などの **Virtual machine name**を設定し、vSphere クラスターの名前をクリックし、直前のステップで作成したフォルダーを選択します。
   d. **Select a compute resource** タブで、vSphere クラスターの名前をクリックします。
   e. **Select storage** タブで、仮想マシンのストレージオプションを設定します。

- Thin Provision を選択します。
- **install-config.yaml** ファイルで指定したデータストアを選択します。

f. Select network タブで、クラスターに設定したネットワークを指定します (ある場合)。

g. すべてのクラスターマシンタイプに同じテンプレートを使用する予定の場合、Customize template タブに値を指定しないでください。

7. テンプレートがデプロイされた後に、マシンの仮想マシンをクラスターにデプロイします。

a. テンプレートの名前を右クリックし、Clone → Clone to Virtual Machineをクリックします。

b. Select a name and folderタブで、仮想マシンの名前を指定します。**control-plane-0** または **compute-1** などのように、マシンタイプを名前に含めることができるかもしれません。

c. Select a name and folderタブで、クラスターに作成したフォルダーの名前を選択します。

d. Select a compute resource タブで、データセンター内のホストの名前を選択します。

e. オプション: Select storage タブで、ストレージオプションをカスタマイズします。

f. Select clone options で、 Customize this virtual machine's hardware を選択します。

g. Customize hardware タブで、 VM Options→ Advanced をクリックします。

- オプション: クラスターのパフォーマンスに問題が生じる場合は、Latency Sensitivity 一覧から High を選択します。
- Edit Configuration をクリックし、Configuration Parameters ウィンドウで Add Configuration Params をクリックします。以下のパラメーター名および値を定義します。
  - **guestinfo.ignition.config.data**: このマシンファイルの base64 でエンコードした Ignition 設定ファイルの内容を貼り付けます。
  - **guestinfo.ignition.config.data.encoding**: **base64** を指定します。
  - **disk.EnableUUID**: **TRUE** を指定します。
- または、仮想マシンの電源を入れる前に vApp プロパティーを使用して追加します。
  - vCenter Server インベントリーから仮想マシンに移動します。
  - Configure タブで Settings を展開し、 vAPP options を選択します。
  - スクロールダウンし、Properties の下で上記の設定を適用します。

h. Customize hardware タブの Virtual Hardware パネルで、必要に応じて指定した値を変更します。RAM、CPU、およびディスクストレージの量がマシンタイプの最小要件を満たすことを確認してください。

i. 設定を完了し、仮想マシンの電源をオンにします。

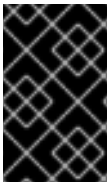
8. 各マシンごとに先の手順に従って、クラスターの残りのマシンを作成します。

重要

この時点でブートストラップおよびコントロールプレーンマシンを作成する必要があります。一部の Pod はデフォルトでコンピュートマシンにデプロイされるため、クラスターのインストール前に、2 つ以上のコンピュートマシンを作成します。

4.1.10. OpenShift コマンドラインインターフェースのインストール

oc として知られる OpenShift コマンドラインインターフェース (CLI) をダウンロードし、インストールすることができます。



重要

以前のバージョンの **oc** をインストールしている場合、これを使用して OpenShift Container Platform 4.1 のすべてのコマンドを実行することはできません。新規バージョンの **oc** をダウンロードし、インストールする必要があります。

手順

1. Red Hat OpenShift Cluster Manager サイトの Infrastructure Provider ページから、選択するインストールタイプのページに移動し、**Download Command-line Tools** をクリックします。
2. 表示されているサイトから、オペレーティングシステムの圧縮されたファイルをダウンロードします。

注記

oc は Linux、Windows、または macOS にインストールできます。

3. 圧縮ファイルを展開して、指定のパスにあるディレクトリーに配置します。

4.1.11. クラスターの作成

OpenShift Container Platform クラスターを作成するには、インストールプログラムで生成した Ignition 設定ファイルを使用してプロビジョニングしたマシンで、ブートストラッププロセスが完了するのを待機します。

前提条件

- クラスターに必要なインフラストラクチャーを作成すること。
- インストールプログラムを取得し、クラスターの Ignition 設定ファイルを生成していること。
- クラスターの RHCOS マシンを作成するために Ignition 設定ファイルを使用していること。
- 使用するマシンでインターネットに直接アクセスできること。

手順

1. ブートストラッププロセスをモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for bootstrap-complete \ 1
    --log-level info 2
```

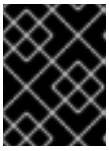
```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.13.4+b626c2fe1 up
INFO Waiting up to 30m0s for the bootstrap-complete event...
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

2 異なるインストールの詳細情報を表示するには、 **info** ではなく、**warn**、**debug**、または **error** を指定します。

Kubernetes API サーバーでこれがコントロールプレーンマシンにブートストラップされていることを示すシグナルが出されるとコマンドは成功します。

2. ブートストラッププロセスが完了したら、ブートストラップマシンをロードバランサーから削除します。



重要

この時点で、ブートストラップマシンをロードバランサーから削除する必要があります。さらに、マシン自体を削除し、再フォーマットすることができます。

4.1.12. クラスターへのログイン

クラスター **kubeconfig** ファイルをエクスポートし、デフォルトシステムユーザーとしてクラスターにログインできます。**kubeconfig** ファイルには、クライアントを正しいクラスターおよび API サーバーに接続するために CLI で使用されるクラスターについての情報が含まれます。このファイルはクラスターに固有のファイルであり、OpenShift Container Platform のインストール時に作成されます。

前提条件

- OpenShift Container Platform クラスターのデプロイ。
- **oc** CLI のインストール。

手順

1. **kubeadmin** 認証情報をエクスポートします。

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1

$ oc whoami
system:admin
```

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

4.1.13. マシンの CSR の承認

マシンをクラスターに追加する際に、追加したそれぞれのマシンについて 2 つの保留状態の証明書署名要求 (CSR) が生成されます。これらの CSR が承認されていることを確認するか、または必要な場合はそれらを承認してください。

前提条件

- マシンをクラスターに追加していること。

手順

1. クラスターがマシンを認識していることを確認します。

```
$ oc get nodes

NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.13.4+b626c2fe1
master-1  Ready     master  63m  v1.13.4+b626c2fe1
master-2  Ready     master  64m  v1.13.4+b626c2fe1
worker-0  NotReady  worker  76s  v1.13.4+b626c2fe1
worker-1  NotReady  worker  70s  v1.13.4+b626c2fe1
```

   出力には作成したすべてのマシンが一覧表示されます。

2. 保留中の証明書署名要求 (CSR) を確認し、クラスターに追加したそれぞれのマシンのクライアントおよびサーバー要求に **Pending** または **Approved** ステータスが表示されていることを確認します。

```
$ oc get csr

NAME        AGE     REQUESTOR                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending (1)
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending (2)
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

   (1) クライアント要求の CSR。

   (2) サーバー要求の CSR。

   この例では、2 つのマシンがクラスターに参加しています。この一覧にはさらに多くの承認された CSR が表示される可能性があります。

3. 追加したマシンの保留中の CSR すべてが **Pending** ステータスになった後に CSR が承認されない場合には、クラスターマシンの CSR を承認します。

注記

CSR のローテーションは自動的に実行されるため、クラスターにマシンを追加後 1 時間以内に CSR を承認してください。1 時間以内に承認しない場合には、証明書のローテーションが行われ、各ノードに 3 つ以上の証明書が存在するようになります。これらの証明書すべてを承認する必要があります。最初の CSR の承認後、後続のノードクライアント CSR はクラスターの **kube-controller-manger** によって自動的に承認されます。kubelet 提供証明書の要求を自動的に承認する方法を実装する必要があります。

- それらを個別に承認するには、それぞれの有効な CSR について以下のコマンドを実行します。

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** は、現行の CSR の一覧からの CSR の名前です。

- すべての保留中の CSR を承認するには、以下のコマンドを実行します。

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

4.1.14. Operator の初期設定

コントロールプレーンの初期化後に、一部の Operator を利用可能にするためにそれらをすぐに設定する必要があります。

前提条件

- コントロールプレーンが初期化されていること。

手順

1. クラスターコンポーネントがオンラインになることを確認します。

```
$ watch -n5 oc get clusteroperators

NAME                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication             4.1.0     True        False         False      69s
cloud-credential           4.1.0     True        False         False      12m
cluster-autoscaler         4.1.0     True        False         False      11m
console                    4.1.0     True        False         False      46s
dns                        4.1.0     True        False         False      11m
image-registry             4.1.0     False       True          False      5m26s
ingress                    4.1.0     True        False         False      5m36s
kube-apiserver             4.1.0     True        False         False      8m53s
kube-controller-manager    4.1.0     True        False         False      7m24s
kube-scheduler             4.1.0     True        False         False      12m
machine-api                4.1.0     True        False         False      12m
machine-config             4.1.0     True        False         False      7m36s
marketplace                4.1.0     True        False         False      7m54m
monitoring                 4.1.0     True        False         False      7h54s
```

```
network                                    4.1.0     True        False         False      5m9s
node-tuning                                4.1.0     True        False         False      11m
openshift-apiserver                        4.1.0     True        False         False      11m
openshift-controller-manager               4.1.0     True        False         False      5m943s
openshift-samples                          4.1.0     True        False         False      3m55s
operator-lifecycle-manager                 4.1.0     True        False         False      11m
operator-lifecycle-manager-catalog         4.1.0     True        False         False      11m
service-ca                                 4.1.0     True        False         False      11m
service-catalog-apiserver                  4.1.0     True        False         False      5m26s
service-catalog-controller-manager         4.1.0     True        False         False      5m25s
storage                                    4.1.0     True        False         False      5m30s
```

2. 利用不可の Operator を設定します。

4.1.14.1. イメージレジストリーストレージの設定

image-registry Operator が利用できない場合、そのストレージを設定する必要があります。実稼働クラスターに必要な PersistentVolume の設定方法と、実稼働用ではないクラスターにのみ使用できる空のディレクトリーをストレージの場所として設定する方法が表示されます。

4.1.14.1.1. VMware vSphere のレジストリーストレージの設定

クラスター管理者は、インストール後にレジストリーをストレージを使用できるように設定する必要があります。

前提条件

- クラスター管理者のパーミッション。
- VMware vSphere 上のクラスター。
- **ReadWriteMany** アクセスモードのプロビジョニングされた永続ボリューム (PV)(例: **NFS**)。

重要

vSphere ボリュームは **ReadWriteMany** アクセスモードをサポートしません。レジストリーストレージを設定するには、**NFS**などの異なるストレージバックエンドを使用する必要があります。

- 容量は「100Gi」以上である。

手順

1. レジストリーをストレージを使用できるように設定するには、**configs.imageregistry/cluster** リソースの **spec.storage.pvc** を変更します。
2. レジストリー Pod がないことを確認します。

```
$ oc get pod -n openshift-image-registry
```

注記

ストレージタイプが **emptyDIR** の場合、レプリカ数が **1** を超えることはありません。ストレージタイプが **NFS** で、レジストリー Pod を **replica>1** を設定してスケールアップする必要がある場合、**no_wdelay** マウントオプションを有効にする必要があります。以下は例になります。

```
# cat /etc/exports
/mnt/data *(rw,sync,no_wdelay,no_root_squash,insecure,fsid=0)
sh-4.3# exportfs -rv
exporting *:/mnt/data
```

1. レジストリー設定を確認します。

```
$ oc edit configs.imageregistry.operator.openshift.io

storage:
  pvc:
    claim:
```

claim フィールドを空のままにし、 **image-registry-storage** PVC の自動作成を可能にします。

2. **clusteroperator** ステータスを確認します。

```
$ oc get clusteroperator image-registry
```

4.1.14.1.2. 実稼働以外のクラスターでのイメージレジストリーのストレージの設定

イメージレジストリー Operator のストレージを設定する必要があります。実稼働用以外のクラスターの場合、イメージレジストリーは空のディレクトリーに設定することができます。これを実行する場合、レジストリーを再起動するとすべてのイメージが失われます。

手順

- イメージレジストリーストレージを空のディレクトリーに設定するには、以下を実行します。

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"storage":{"emptyDir":{}}}}'
```

警告

実稼働用以外のクラスターにのみこのオプションを設定します。

イメージレジストリー Operator がそのコンポーネントを初期化する前にこのコマンドを実行する場合、**oc patch** コマンドは以下のエラーを出して失敗します。

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

数分待機した後に、このコマンドを再び実行します。

4.1.15. ユーザーによってプロビジョニングされるインフラストラクチャーでのインストールの完了

Operator 設定の完了後に、提供するインフラストラクチャーでのクラスターのインストールを終了できます。

前提条件

- コントロールプレーンが初期化されていること。
- Operator の初期設定を完了していること。

手順

1. すべてのクラスターコンポーネントがオンラインであることを確認します。

```
$ watch -n5 oc get clusteroperators

NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.1.0     True        False         False      10m
cloud-credential                           4.1.0     True        False         False      22m
cluster-autoscaler                         4.1.0     True        False         False      21m
console                                    4.1.0     True        False         False      10m
dns                                        4.1.0     True        False         False      21m
image-registry                             4.1.0     True        False         False      16m
ingress                                    4.1.0     True        False         False      16m
kube-apiserver                             4.1.0     True        False         False      19m
kube-controller-manager                    4.1.0     True        False         False      18m
kube-scheduler                             4.1.0     True        False         False      22m
machine-api                                4.1.0     True        False         False      22m
machine-config                             4.1.0     True        False         False      18m
marketplace                                4.1.0     True        False         False      18m
monitoring                                 4.1.0     True        False         False      18m
network                                    4.1.0     True        False         False      16m
node-tuning                                4.1.0     True        False         False      21m
openshift-apiserver                        4.1.0     True        False         False      21m
openshift-controller-manager               4.1.0     True        False         False      17m
openshift-samples                          4.1.0     True        False         False      14m
operator-lifecycle-manager                 4.1.0     True        False         False      21m
operator-lifecycle-manager-catalog         4.1.0     True        False         False      21m
service-ca                                 4.1.0     True        False         False      21m
service-catalog-apiserver                  4.1.0     True        False         False      16m
service-catalog-controller-manager         4.1.0     True        False         False      16m
storage                                    4.1.0     True        False         False      16m
```

すべてのクラスター Operator が **AVAILABLE** の場合、インストールを完了することができます。

2. クラスターの完了をモニターします。

```
$ ./openshift-install --dir=<installation_directory> wait-for install-complete 1
INFO Waiting up to 30m0s for the cluster to initialize...
```

-

1 **<installation_directory>** については、インストールファイルを保存したディレクトリーへのパスを指定します。

Cluster Version Operator が Kubernetes API サーバーから OpenShift Container Platform クラスターのデプロイを終了するとコマンドは成功します。

重要

インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになる証明書が含まれます。最初の証明書のローテーションが正常に実行されるようにするには、クラスターを動作が低下していない状態で 24 時間実行し続ける必要があります。

3. Kubernetes API サーバーが Pod と通信していることを確認します。

   a. すべての Pod の一覧を表示するには、以下のコマンドを使用します。

```
$ oc get pods --all-namespaces

NAMESPACE                       NAME                                         READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running    1          9m
openshift-apiserver             apiserver-67b9g                               1/1     Running     0
3m
openshift-apiserver             apiserver-ljcmx                               1/1     Running     0
1m
openshift-apiserver             apiserver-z25h4                               1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
Running    0          5m
...
```

   b. 以下のコマンドを使用して、直前のコマンドの出力に一覧表示される Pod のログを表示します。

```
$ oc logs <pod_name> -n <namespace>
```

1 直前のコマンドの出力にあるように、Pod 名および namespace を指定します。

   Pod のログが表示される場合、Kubernetes API サーバーはクラスターマシンと通信できます。

次のステップ

- クラスターをカスタマイズします。
- 必要な場合は、Telemetry をオプトアウトすることができます。

第5章 インストールログの収集

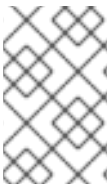
OpenShift Container Platform のインストールした場合のトラブルシューティングのために、ブートストラップおよびコントロールプレーン、またはマスター、マシンからログを収集できます。

前提条件

- OpenShift Container Platform クラスターのインストールを試みたが、インストールに失敗している。
- SSH キーをインストールプログラムに指定しており、そのキーは実行中の **ssh-agent** プロセスにある。

5.1. 失敗したインストールのログの収集

SSH キーをインストールプログラムに指定している場合、失敗したインストールについてのデータを収集することができます。



注記

実行中のクラスターからログを収集する場合とは異なるコマンドを使用して失敗したインストールについてのログを収集します。実行中のクラスターからログを収集する必要がある場合は、**oc adm must-gather** コマンドを使用します。

前提条件

- OpenShift Container Platform のインストールがブートストラッププロセスの終了前に失敗している。ブートストラップノードは実行中であり、SSH でアクセスできる必要がある。
- **ssh-agent** プロセスはコンピューター上でアクティブであり、**ssh-agent** プロセスとインストールプログラムの両方に同じ SSH キーを提供している。
- 独自にプロビジョニングしたインフラストラクチャーにクラスターのインストールを試行した場合には、コントロールプレーン、またはマスター、マシンの完全修飾ドメイン名があること。

手順

1. ブートストラップおよびコントロールプレーンマシンからインストールログを収集するために必要なコマンドを生成します。
   - インストーラーでプロビジョニングされるインフラストラクチャーを使用している場合は、以下のコマンドを実行します。

     ```
     $ ./openshift-install gather bootstrap --dir=<directory> 1
     ```

     1 **installation_directory** は、インストールプログラムが作成する OpenShift Container Platform 定義ファイルを保存しているディレクトリーです。

     インストーラーでプロビジョニングされるインフラストラクチャーの場合、インストールプログラムは、ホスト名または IP アドレスを指定しなくてもよいようにクラスターについての情報を保存します。

- 独自にプロビジョニングしたインフラストラクチャーを使用している場合は、以下のコマンドを実行します。

```
$ ./openshift-install gather bootstrap --dir=<directory> \ 1
  --bootstrap <bootstrap_address> \ 2
  --master "<master_address> <master_address> <master_address>" 3
```

1. **installation_directory** は、インストールプログラムが作成する OpenShift Container Platform 定義ファイルを保存しているディレクトリーです。
2. **<bootstrap_address>** は、クラスターのブートストラップマシンの完全修飾ドメイン名または IP アドレスです。
3. **<master_address>** は、クラスター内のコントロールプレーン、またはマスター、マシンの完全修飾ドメイン名または IP アドレスです。クラスター内のすべてのコントロールプレーンマシンが含まれるスペースで区切られた一覧を指定します。

コマンド出力は以下の例のようになります。

```
INFO Use the following commands to gather logs from the cluster
INFO ssh -A core@<bootstrap_address> '/usr/local/bin/installer-gather.sh <master_address>
<master_address> <master_address>'
INFO scp core@<bootstrap_address>:~/log-bundle.tar.gz .
```

表示される両方のコマンドを使用し、ログを収集し、ダウンロードします。

2. ブートストラップおよびマスターマシンからログを収集します。

```
$ ssh -A core@<bootstrap_address> '/usr/local/bin/installer-gather.sh <master_address>
<master_address> <master_address>'
```

SSH をブートストラップマシンに対して実行し、クラスター内のブートストラップおよびコントロールプレーンマシンからできる限り多くのデータを収集することを目的に設計された収集ツールを実行します。 その後に収集されたファイルすべてを圧縮します。

注記

通常、コマンド出力にエラーが表示されることがあります。コマンド出力に、圧縮されたログファイル **log-bundle.tar.gz** のダウンロード方法についての説明が表示される場合、コマンドは問題なく実行されたことを示します。

3. ログが含まれる圧縮ファイルをダウンロードします。

```
$ scp core@<bootstrap_address>:~/log-bundle.tar.gz . 1
```

1. **<bootstrap_address>** は、ブートストラップマシンの完全修飾ドメイン名または IP アドレスです。

ログファイルをダウンロードするためのコマンドは、収集コマンドの出力の末尾に含まれます。

インストールの失敗についての Red Hat サポートケースを作成する場合は、圧縮したログをケースに含めるようにしてください。

5.2. ホストへの SSH アクセスによるログの手動収集

must-gather または自動化された収集方法が機能しない場合にログを手動で収集します。

前提条件

- ホストへの SSH アクセスがあること。

手順

1. 以下を実行し、**journalctl** コマンドを使用してブートストラップホストから **bootkube.service** サービスログを収集します。

```
$ journalctl -b -f -u bootkube.service
```

2. Podman ログを使用して、ブートストラップホストのコンテナーログを収集します。これは、ホストからすべてのコンテナーログを取得するためにループで表示されます。

```
$ for pod in $(sudo podman ps -a -q); do sudo podman logs $pod; done
```

3. または、以下を実行し、**tail** コマンドを使用してホストのコンテナーログを収集します。

```
# tail -f /var/lib/containers/storage/overlay-containers/*/userdata/ctr.log
```

4. 以下を実行し、**journalctl** コマンドを使用して **kubelet.service** および **crio.service** サービスログをマスターホストから収集します。

```
$ journalctl -b -f -u kubelet.service -u crio.service
```

5. 以下を実行し、**tail** コマンドを使用してマスターおよびワーカーホストコンテナーログを収集します。

```
$ sudo tail -f /var/log/containers/*
```

5.3. ホストへの SSH アクセスを使用しないログの手動収集

must-gather または自動化された収集方法が機能しない場合にログを手動で収集します。

ノードへの SSH アクセスがない場合は、システムジャーナルにアクセスし、ホストで生じていることを調査できます。

前提条件

- OpenShift Container Platform のインストールが完了している。
- API サービスが機能している。
- システム管理者権限がある。

手順

1. 以下を実行し、**/var/log** の下にある **journald** ユニットログにアクセスします。

```
$ oc adm node-logs --role=master -u kubelet
```

2. 以下を実行し、**/var/log** の下にあるホストファイルのパスにアクセスします。

```
$ oc adm node-logs --role=master --path=openshift-apiserver
```

第6章 インストール設定

6.1. 利用可能なクラスターのカスタマイズ

OpenShift Container Platform クラスターのデプロイ後は、大半のクラスター設定およびカスタマイズが終了していることになります。数多くの設定リソースが利用可能です。

イメージレジストリー、ネットワーク設定、イメージビルドの動作およびアイデンティティープロバイダーなどのクラスターの主要な機能を設定するために設定リソースを変更します。

これらのリソースを使用して制御する設定の現行の記述については、**oc explain** コマンド (例: **oc explain builds --api-version=config.openshift.io/v1**) を使用します。

6.1.1. クラスター設定リソース

すべてのクラスター設定リソースはグローバルにスコープが設定され (namescope が使用されない)、**cluster**という名前が付けられます。

リソース名	説明
apiserver.config.openshift.io	証明書および認証局などの api-server 設定を指定します。
authentication.config.openshift.io	クラスターのアイデンティティープロバイダーおよび認証設定を制御します。
build.config.openshift.io	クラスターのすべてのビルドについてのデフォルトおよび有効にされている設定を制御します。
console.config.openshift.io	ログアウト動作を含む Web コンソールインターフェースの動作を設定します。
featuregate.config.openshift.io	FeatureGates を有効にして、テクノロジープレビュー機能を使用できるようにします。
image.config.openshift.io	特定の イメージレジストリー が処理される方法を設定します (allowed、disallowed、insecure、CA details)。
ingress.config.openshift.io	ルートのデフォルトドメインなどのルーティングに関連する設定の詳細。
oauth.config.openshift.io	内部 OAuth サーバーフローに関連するアイデンティティープロバイダーおよび他の動作を設定します。
project.config.openshift.io	プロジェクトテンプレートを含む、プロジェクトの作成方法 を設定します。
proxy.config.openshift.io	外部ネットワークアクセスを必要とするコンポーネントで使用されるプロキシーを定義します。注: すべてのコンポーネントがこの値を使用する訳ではありません。

リソース名	説明
scheduler.config.openshift.io	ポリシーやデフォルトノードセレクターなどのスケジューラーの動作を設定します。

6.1.2. Operator 設定リソース

これらの設定リソースは、**cluster**という名前のクラススコープのインスタンスです。これは、特定の Operator によって所有される特定コンポーネントの動作を制御します。

リソース名	説明
console.operator.openshift.io	ブランドのカスタマイズなどのコンソールの外観の制御
config.imageregistry.operator.openshift.io	パブリックルーティング、プロキシー設定、リソース設定、レプリカ数およびストレージタイプなどの 内部イメージレジストリー設定を設定します。
config.samples.operator.openshift.io	Samples Operator を設定して、クラスターにインストールされるイメージストリームとテンプレートのサンプルを制御します。

6.1.3. 追加の設定リソース

これらの設定リソースは、特定コンポーネントの単一のインスタンスを表します。リソースの複数インスタンスを作成して、複数インスタンスを要求できる場合があります。他の場合は、特定の namespace の特定のリソースインスタンス名のみが Operator によって消費されます。追加のリソースインスタンスの作成方法や作成するタイミングについての詳細は、コンポーネント固有のドキュメントを参照してください。

リソース名	インスタンス名	Namespace	説明
alertmanager.monitoring.coreos.com	main	openshift-monitoring	alertmanager デプロイメントパラメーターを制御します。
ingresscontroller.operator.openshift.io	default	openshift-ingress-operator	ドメイン、レプリカ数、証明書、およびコントローラーの配置などの Ingress Operator 動作を設定します。

6.1.4. 情報リソース

これらのリソースを使用して、クラスターについての情報を取得します。これらのリソースは直接編集しないでください。

リソース名	インスタンス名	説明
clusterversion.config.openshift.io	version	OpenShift Container Platform 4.1 では、実稼働クラスターの ClusterVersion リソースをカスタマイズすることはできません。その代わりとして、 クラスターの更新プロセスを実行します。
dns.config.openshift.io	cluster	クラスターの DNS 設定を変更することはできません。DNS Operator ステータスを表示できます。
infrastructure.config.openshift.io	cluster	クラスターはそのクラウドプロバイダーとの対話を可能にする設定の詳細。
network.config.openshift.io	cluster	インストール後にクラスターのネットワークを変更することはできません。ネットワークをカスタマイズするには、インストール時にネットワークをカスタマイズするプロセスを実行します。

6.2. ファイアウォールの設定

ファイアウォールを使用する場合、OpenShift Container Platform が Red Hat Insights にアクセスできるように設定する必要があります。このアクセスは、OpenShift Container Platform が更新を受信するために必要な Telemetry サービスにアクセスするために必要です。

6.2.1. OpenShift Container Platform のファイアウォールの設定

OpenShift Container Platform をインストールする前に、ファイアウォールを Red Hat Insights にアクセスするように設定する必要があります。

手順

- 発信ネットワークのファイアウォール上で以下のホスト名とポートを許可するようにファイアウォールを設定します。

```
cert-api.access.redhat.com:443
api.access.redhat.com:443
infogw.api.openshift.com:443
```