



OpenShift Container Platform 4.5

AWS へのインストール

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

OpenShift Container Platform 4.5 AWS へのインストール

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

Enter your first name here. Enter your surname here.

Enter your organisation's name here. Enter your organisational division here.

Enter your email address here.

法律上の通知

Copyright © 2022 | You need to change the HOLDER entity in the en-US/Installing_on_AWS.ent file |.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

概要

本書では、Amazon Web Services に OpenShift Container Platform 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	6
1.1. AWS アカウントの設定	6
1.1.1. Route 53 の設定	6
1.1.2. AWS アカウントの制限	6
1.1.3. 必要な AWS パーミッション	9
1.1.4. IAM ユーザーの作成	16
1.1.5. サポートされている AWS リージョン	17
1.1.6. 次のステップ	18
1.2. クラスターの AWS へのクイックインストール	18
1.2.1. 前提条件	18
1.2.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	19
1.2.3. 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	19
1.2.4. インストールプログラムの取得	21
1.2.5. クラスターのデプロイ	21
1.2.6. バイナリーのダウンロードによる CLI のインストール	23
1.2.6.1. Linux への CLI のインストール	23
1.2.6.2. Windows での CLI のインストール	24
1.2.6.3. macOS への CLI のインストール	24
1.2.7. クラスターへのログイン	25
1.2.8. 次のステップ	25
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	26
1.3.1. 前提条件	26
1.3.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	26
1.3.3. 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	27
1.3.4. インストールプログラムの取得	28
1.3.5. インストール設定ファイルの作成	29
1.3.5.1. インストール設定パラメーター	30
1.3.5.1.1. 必須設定パラメーター	31
1.3.5.1.2. ネットワーク設定パラメーター	32
1.3.5.1.3. オプションの設定パラメーター	34
1.3.5.1.4. オプションの AWS 設定パラメーター	37
1.3.5.2. AWS のカスタマイズされた install-config.yaml ファイルのサンプル	39
1.3.6. クラスターのデプロイ	41
1.3.7. バイナリーのダウンロードによる CLI のインストール	42
1.3.7.1. Linux への CLI のインストール	42
1.3.7.2. Windows での CLI のインストール	43
1.3.7.3. macOS への CLI のインストール	43
1.3.8. クラスターへのログイン	44
1.3.9. 次のステップ	44
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	44
1.4.1. 前提条件	45
1.4.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	45
1.4.3. 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	46
1.4.4. インストールプログラムの取得	47
1.4.5. インストール設定ファイルの作成	48
1.4.5.1. インストール設定パラメーター	49
1.4.5.1.1. 必須設定パラメーター	50
1.4.5.1.2. ネットワーク設定パラメーター	51
1.4.5.1.3. オプションの設定パラメーター	53
1.4.5.1.4. オプションの AWS 設定パラメーター	56
1.4.5.2. ネットワーク設定パラメーター	58

1.4.5.3. AWS のカスタマイズされた install-config.yaml ファイルのサンプル	59
1.4.6. 高度なネットワーク設定パラメーターの変更	61
1.4.7. Cluster Network Operator (CNO) の設定	63
1.4.7.1. OpenShift SDN デフォルト CNI ネットワークプロバイダーの設定パラメーター	64
1.4.7.2. OVN-Kubernetes デフォルト CNI ネットワークプロバイダーの設定パラメーター	64
1.4.7.3. Cluster Network Operator の設定例	65
1.4.8. クラスターのデプロイ	65
1.4.9. バイナリーのダウンロードによる CLI のインストール	67
1.4.9.1. Linux への CLI のインストール	67
1.4.9.2. Windows での CLI のインストール	67
1.4.9.3. macOS への CLI のインストール	68
1.4.10. クラスターへのログイン	68
1.4.11. 次のステップ	69
1.5. AWS のクラスターの既存 VPC へのインストール	69
1.5.1. 前提条件	69
1.5.2. カスタム VPC の使用について	69
1.5.2.1. VPC を使用するための要件	70
1.5.2.2. VPC 検証	72
1.5.2.3. パーMISSIONの区分	73
1.5.2.4. クラスター間の分離	73
1.5.3. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	73
1.5.4. 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	74
1.5.5. インストールプログラムの取得	75
1.5.6. インストール設定ファイルの作成	76
1.5.6.1. インストール設定パラメーター	77
1.5.6.1.1. 必須設定パラメーター	78
1.5.6.1.2. ネットワーク設定パラメーター	79
1.5.6.1.3. オプションの設定パラメーター	81
1.5.6.1.4. オプションの AWS 設定パラメーター	84
1.5.6.2. AWS のカスタマイズされた install-config.yaml ファイルのサンプル	86
1.5.6.3. インストール時のクラスター全体のプロキシの設定	88
1.5.7. クラスターのデプロイ	89
1.5.8. バイナリーのダウンロードによる CLI のインストール	91
1.5.8.1. Linux への CLI のインストール	91
1.5.8.2. Windows での CLI のインストール	91
1.5.8.3. macOS への CLI のインストール	92
1.5.9. クラスターへのログイン	92
1.5.10. 次のステップ	93
1.6. プライベートクラスターの AWS へのインストール	93
1.6.1. 前提条件	93
1.6.2. プライベートクラスター	94
1.6.2.1. AWS のプライベートクラスター	94
1.6.2.1.1. 制限事項	94
1.6.3. カスタム VPC の使用について	95
1.6.3.1. VPC を使用するための要件	95
1.6.3.2. VPC 検証	97
1.6.3.3. パーMISSIONの区分	98
1.6.3.4. クラスター間の分離	98
1.6.4. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	98
1.6.5. 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	99
1.6.6. インストールプログラムの取得	100
1.6.7. インストール設定ファイルの手動作成	101
1.6.7.1. インストール設定パラメーター	102

1.6.7.1.1. 必須設定パラメーター	102
1.6.7.1.2. ネットワーク設定パラメーター	104
1.6.7.1.3. オプションの設定パラメーター	106
1.6.7.1.4. オプションの AWS 設定パラメーター	109
1.6.7.2. AWS のカスタマイズされた install-config.yaml ファイルのサンプル	111
1.6.7.3. インストール時のクラスター全体のプロキシの設定	113
1.6.8. クラスターのデプロイ	115
1.6.9. バイナリーのダウンロードによる CLI のインストール	116
1.6.9.1. Linux への CLI のインストール	116
1.6.9.2. Windows での CLI のインストール	117
1.6.9.3. macOS への CLI のインストール	117
1.6.10. クラスターへのログイン	118
1.6.11. 次のステップ	118
1.7. CLOUDFORMATION テンプレートの使用による、AWS でのユーザーによってプロビジョニングされたインフラストラクチャーへのクラスターのインストール	118
1.7.1. 前提条件	119
1.7.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	119
1.7.3. 必要な AWS インフラストラクチャーコンポーネント	120
1.7.3.1. クラスターマシン	120
1.7.3.2. 証明書署名要求の管理	122
1.7.3.3. 他のインフラストラクチャーコンポーネント	122
1.7.3.4. 必要な AWS パーミッション	128
1.7.4. インストールプログラムの取得	135
1.7.5. 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	136
1.7.6. AWS のインストール設定ファイルの作成	137
1.7.6.1. インストール設定ファイルの作成	137
1.7.6.2. インストール時のクラスター全体のプロキシの設定	139
1.7.6.3. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	140
1.7.7. インフラストラクチャー名の抽出	142
1.7.8. AWS での VPC の作成	143
1.7.8.1. VPC の CloudFormation テンプレート	145
1.7.9. 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	150
1.7.9.1. 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	153
1.7.10. AWS でのセキュリティグループおよびロールの作成	161
1.7.10.1. セキュリティオブジェクトの CloudFormation テンプレート	163
1.7.11. AWS インフラストラクチャーの RHCOS AMI	172
1.7.12. AWS でのブートストラップノードの作成	173
1.7.12.1. ブートストラップマシンの CloudFormation テンプレート	177
1.7.13. AWS でのコントロールプレーンの作成	181
1.7.13.1. コントロールプレーンマシンの CloudFormation テンプレート	186
1.7.14. 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 AWS でのブートストラップノードの初期化	193
1.7.14.1. AWS でのワーカーノードの作成	194
1.7.14.1.1. ワーカーマシンの CloudFormation テンプレート	197
1.7.15. バイナリーのダウンロードによる CLI のインストール	200
1.7.15.1. Linux への CLI のインストール	200
1.7.15.2. Windows での CLI のインストール	201
1.7.15.3. macOS への CLI のインストール	201
1.7.16. クラスターへのログイン	202
1.7.17. マシンの証明書署名要求の承認	202
1.7.18. Operator の初期設定	205
1.7.18.1. イメージレジストリストレージの設定	205
1.7.18.1.1. 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を使用した AWS のレジスト	

リースストレージの設定	206
1.7.18.1.2. 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	207
1.7.19. ブートストラップリソースの削除	208
1.7.20. Ingress DNS レコードの作成	208
1.7.21. 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	211
1.7.22. 次のステップ	211
1.8. 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ネットワークが制限された環境での AWS へのクラスターのインストール	212
1.8.1. 前提条件	212
1.8.2. ネットワークが制限された環境でのインストールについて	213
1.8.2.1. その他の制限	213
1.8.3. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス	213
1.8.4. 必要な AWS インフラストラクチャーコンポーネント	214
1.8.4.1. クラスターマシン	214
1.8.4.2. 証明書署名要求の管理	216
1.8.4.3. 他のインフラストラクチャーコンポーネント	216
1.8.4.4. 必要な AWS パーミッション	222
1.8.5. 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	229
1.8.6. AWS のインストール設定ファイルの作成	231
1.8.6.1. インストール設定ファイルの作成	231
1.8.6.2. インストール時のクラスター全体のプロキシの設定	233
1.8.6.3. Kubernetes マニフェストおよび Ignition 設定ファイルの作成	235
1.8.7. インフラストラクチャー名の抽出	237
1.8.8. AWS での VPC の作成	237
1.8.8.1. VPC の CloudFormation テンプレート	239
1.8.9. 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	244
1.8.9.1. 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	248
1.8.10. AWS でのセキュリティグループおよびロールの作成	255
1.8.10.1. セキュリティオブジェクトの CloudFormation テンプレート	257
1.8.11. AWS インフラストラクチャーの RHCOS AMI	266
1.8.12. AWS でのブートストラップノードの作成	267
1.8.12.1. ブートストラップマシンの CloudFormation テンプレート	271
1.8.13. AWS でのコントロールプレーンの作成	275
1.8.13.1. コントロールプレーンマシンの CloudFormation テンプレート	280
1.8.14. 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 AWS でのブートストラップノードの初期化	288
1.8.14.1. AWS でのワーカーノードの作成	288
1.8.14.1.1. ワーカーマシンの CloudFormation テンプレート	292
1.8.15. クラスターへのログイン	294
1.8.16. マシンの証明書署名要求の承認	295
1.8.17. Operator の初期設定	297
1.8.17.1. イメージレジストリーストレージの設定	298
1.8.17.1.1. 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を使用した AWS のレジストリーストレージの設定	298
1.8.17.1.2. 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	299
1.8.18. ブートストラップリソースの削除	300
1.8.19. Ingress DNS レコードの作成	301
1.8.20. 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	303
1.8.21. 次のステップ	304
1.9. AWS でのクラスターのアンインストール	304
1.9.1. 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するクラスターの削除	305

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. Route 53 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、Route 53 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。Route 53 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 Route 53 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 Route 53 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している場合は、その委任レコードを親ドメインに追加します。これにより、サブドメインの Amazon Route 53 の責任が付与されます。親ドメインの DNS プロバイダーによって要約された委任手順に従います。高次元の手順の例については、AWS ドキュメントの [Creating a subdomain that uses Amazon Route 53 as the DNS service without migrating the parent domain](#) を参照してください。

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。	<p>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。</p> <ul style="list-style-type: none"> ● 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 <p>これらのインスタンスタイプのは、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。</p> <p>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。</p>

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
Elastic IP (EIP)	0 - 1	アカウントごとに 5 つの EIP	<p>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ゾーンの数を確認します。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。</p> <div>  <div> <p>重要</p> <p>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。</p> </div> </div>
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 Service オブジェクトをタイプ LoadBalancer を指定してデプロイすると、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ゾーンに 1 つの NAT ゲートウェイをデプロイします。

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
Elastic Network Interface (ENI)	12 以上	リージョンごとに 350	<p>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ゾーンの数を判別します。</p> <p>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。</p>
VPC ゲートウェイ	20	アカウントごとに 20	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、Amazon Web Services (AWS) で作成する 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 クラスターのすべてのコンポーネント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。

例1.1 インストールに必要な EC2 パーミッション

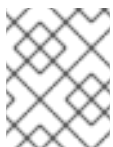
- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**

- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteSnapshot**
- **ec2>DeleteTags**
- **ec2:DeregisterImage**
- **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:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**

- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例1.2 インストール時のネットワークリソースの作成に必要なパーミッション

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



注記

既存の VPC を使用する場合、アカウントではネットワークリソースの作成にこれらのパーミッションを必要としません。

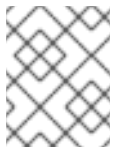
例1.3 インストールに必要な Elastic Load Balancing のパーミッション

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

例1.4 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**



注記

AWS アカウントに Elastic Load Balancer (ELB) を作成していない場合、IAM ユーザーには **iam:CreateServiceLinkedRole** パーミッションも必要です。

例1.5 インストールに必要な Route 53 パーミッション

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53>DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**
- **route53:ListHostedZonesByName**

- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例1.6 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:GetBucketAcl**
- **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**

例1.7 クラスター Operator が必要とする S3 パーミッション

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**

- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

例1.8 ベースクラスターリソースの削除に必要なパーミッション

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**
- **tag:GetResources**

例1.9 ネットワークリソースの削除に必要なパーミッション

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**

- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**



注記

既存の VPC を使用する場合、アカウントではネットワークリソースの削除にこれらのパーミッションを必要としません。

例1.10 マニフェストの作成に必要な追加の IAM および S3 パーミッション

- **iam:CreateAccessKey**
- **iam:CreateUser**
- **iam>DeleteAccessKey**
- **iam>DeleteUser**
- **iam>DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam>ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam>ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

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-north-1** (Stockholm)

- **eu-west-1** (Ireland)
- **eu-west-2** (London)
- **eu-west-3** (Paris)
- **me-south-1** (Bahrain)
- **sa-east-1** (São Paulo)
- **us-east-1** (N. Virginia)
- **us-east-2** (Ohio)
- **us-west-1** (N. California)
- **us-west-2** (Oregon)

1.1.6. 次のステップ

- OpenShift Container Platform クラスターをインストールします。
 - 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のデフォルトオプションを使用した [クラスターのクイックインストール](#)
 - 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へのクラウドのカスタマイズを使用した [クラスターのインストール](#)
 - 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へのネットワークのカスタマイズを使用した [クラスターのインストール](#)
 - CloudFormation テンプレートの使用による、AWS でのユーザーによってプロビジョニングされたインフラストラクチャーへの [クラスターのインストール](#)

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 4.5 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。

1.2.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。
- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

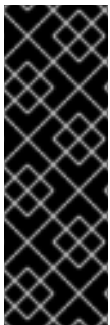
1.2.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.2.3. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
-f <path>/<file_name> ❶
```

- ❶ `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

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

- ❶ `~/.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ルプログラムに指定します。

1.2.4. 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、500 MB のローカルディスク領域が必要です。

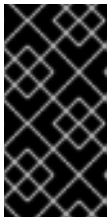
手順

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。



重要

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除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イメージを提供する Quay.io など、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。

1.2.5. クラスターのデプロイ

互換性のあるクラウドプラットフォームに 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の **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

前提条件

- クラスタ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>** の場合、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がある実稼働用の OpenShift Container Platform クラスタでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

- 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 Route 53 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

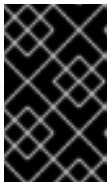
ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。



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。

1.2.6. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。



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 4.5 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、インストールします。

1.2.6.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。
PATH を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.2.6.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.2.6.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。

2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATHを確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.2.7. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

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
```

- 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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.2.8. 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、[リモートの健全性レポートをオプトアウト](#) することができます。

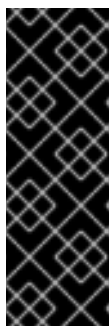
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

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 4.5 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** ファイルでパラメーターを変更します。

1.3.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。
- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

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

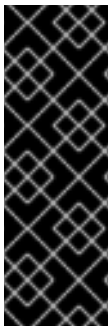
OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリーが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。

- 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に [Quay.io](https://quay.io) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.3.3. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
-f <path>/<file_name> ①
```

- ① `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。

1.3.4. 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。

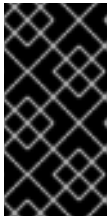
前提条件

- 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、500 MB のローカルディスク領域が必要です。

手順

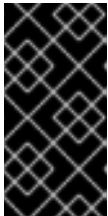
1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページにアクセスします。Red Hat アカウントがある場合は、認証情報を使ってログインします。アカウントがない場合はこれを作成します。

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。



重要

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除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イメージを提供する Quay.io など、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。

1.3.5. インストール設定ファイルの作成

Amazon Web Services (AWS) での 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します。

手順

1. `install-config.yaml` ファイルを作成します。

- a. 以下のコマンドを実行します。

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

- 1 `<installation_directory>` の場合、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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

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。

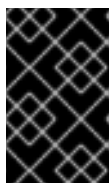
v. クラスターに設定した Route 53 サービスのベースドメインを選択します。

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を参照してください。

3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするの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。

1.3.5.1. 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、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

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.3.5.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.1 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、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。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} 。 {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。

パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト
pullSecret	https://cloud.redhat.com/openshift/install/pull-secret からプルシークレットを取得し、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージのダウンロードを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.3.5.1.2. ネットワーク設定パラメーター


既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.2 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	Pod の IP アドレスブロック。 デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞長。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。 hostPrefix 値の 23 は、510 ($2^{(32-23)} - 2$) Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。
networking.serviceNetwork	サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


パラメーター	説明	値
networking.machineNetwork.cidr	networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。</p> </div> </div>



1.3.5.1.3. オプションの設定パラメーター

オプションのインストール設定パラメーターは、以下の表で説明されています。

表1.3 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュータノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータマシンで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。</p> <div>  <p>重要</p> <p>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ていることを確認します。</p> </div>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (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	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>コントロールプレーンマシンで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。</p> <div>  <p>重要</p> <p>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ていることを確認します。</p> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
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。	サポートされる値は 3 のみです (これはデフォルト値です)。
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
publish	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	Internal または External 。プライベートクラスターをデプロイするには、 publish を Internal に設定します。これはインターネットからアクセスできません。デフォルト値は External です。
sshKey	<p>クラスターマシンへのアクセスを認証するための SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の OpenShift Container Platform クラスターでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	たとえば、 sshKey: ssh-ed25519 AAAA.. です。

1.3.5.1.4. オプションの AWS 設定パラメーター

オプションの AWS 設定パラメーターは、以下の表で説明されています。

表1.4 オプションの AWS パラメーター

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)。

パラメーター	説明	値
<code>compute.platform.aws.rootVolume.size</code>	ルートボリュームのサイズ (GiB)。	整数 (例: 500)。
<code>compute.platform.aws.rootVolume.type</code>	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)。
<code>compute.platform.aws.type</code>	コンピュータマシンの 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)。
<code>compute.platform.aws.zones</code>	インストールプログラムがコンピュータマシンプールのマシンを作成するアベイラビリティゾーン。独自の VPC を指定する場合は、そのアベイラビリティゾーンにサブネットを指定する必要があります。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>compute.aws.region</code>	インストールプログラムがコンピュータ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)。
<code>controlPlane.platform.aws.type</code>	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)。
<code>controlPlane.platform.aws.zones</code>	インストールプログラムがコントロールプレーンマシンプールのマシンを作成するアベイラビリティゾーン。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>controlPlane.aws.region</code>	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)。
<code>platform.aws.userTags</code>	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。

パラメーター	説明	値
platform.aws.subnets	インストールプログラムによる VPC の作成を許可する代わりに VPC を指定する場合は、使用するクラスターのサブネットを指定します。サブネットは、指定する同じ machineNetwork[].cidr 範囲の一部である必要があります。標準クラスターの場合は、各アベイラビリティゾーンのパブリックおよびプライベートサブネットを指定します。プライベートクラスターについては、各アベイラビリティゾーンのプライベートサブネットを指定します。	有効なサブネット ID。

1.3.5.2. 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 ❶
controlPlane: ❷
  hyperthreading: Enabled ❸ ❹
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1
      type: m5.xlarge ❺
    replicas: 3
compute: ❻
- hyperthreading: Enabled ❼
  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
  machineNetwork:
  - cidr: 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
  fips: false ⑪
  sshKey: ssh-ed25519 AAAA... ⑫
  pullSecret: '{"auths": ...}' ⑬

```

① ⑨ ⑩ ⑬ 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。一部のクラスターマシンで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。

- 11 FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat
- 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。
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。

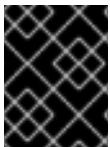


注記

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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

1.3.6. クラスターのデプロイ

互換性のあるクラウドプラットフォームに 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の **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

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
```

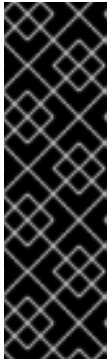
- 1 **<installation_directory>** については、カスタマイズした **./install-config.yaml** ファイルの場所を指定します。
- 2 異なるインストールの詳細情報を表示するには、**info** ではなく、**warn**、**debug**、または **error** を指定します。



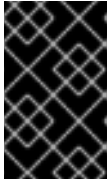
注記

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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。

**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。

1.3.7. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。

**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 4.5 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、インストールします。

1.3.7.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.3.7.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.3.7.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATH を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.3.8. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

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
```

- 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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.3.9. 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、[リモートの健全性レポートをオプトアウト](#) することができます。
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

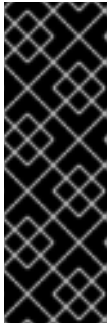
1.4. ネットワークのカスタマイズによる AWS へのクラスターのインストール

OpenShift Container Platform バージョン 4.5 では、カスタマイズされたネットワーク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。

1.4.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。
- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

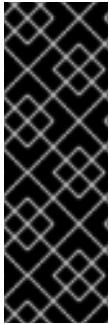
1.4.2. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリーが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.4.3. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
-f <path>/<file_name> 1
```

- 1 `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。

1.4.4. 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

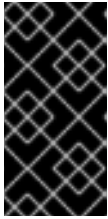
- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、500 MB のローカルディスク領域が必要です。

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。

**重要**

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除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イメージを提供する Quay.io など、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。

1.4.5. インストール設定ファイルの作成

Amazon Web Services (AWS) での 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します。

手順

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

- a. 以下のコマンドを実行します。

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

- 1** **<installation_directory>** の場合、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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

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 Route 53 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を参照してください。

3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするの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。

1.4.5.1. 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、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

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.4.5.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.5 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、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。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} 。 {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。

パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト
pullSecret	https://cloud.redhat.com/openshift/install/pull-secret からプルシークレットを取得し、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージのダウンロードを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.4.5.1.2. ネットワーク設定パラメーター


既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.6 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	Pod の IP アドレスブロック。 デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞長。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。 hostPrefix 値の 23 は、510 ($2^{(32-23)} - 2$) Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。
networking.serviceNetwork	サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


パラメーター	説明	値
networking.machineNetwork.cidr	networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。</p> </div> </div>



1.4.5.1.3. オプションの設定パラメーター

オプションのインストール設定パラメーターは、以下の表で説明されています。

表1.7 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータマシンで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。</p> <div>  <div> <p>重要</p> <p>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ていることを確認します。</p> </div> </div>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (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	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>コントロールプレーンマシンで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。</p> <div>  <div> <p>重要</p> <p>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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
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。	サポートされる値は 3 のみです (これはデフォルト値です)。
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <div> <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
publish	Kubernetes API、OpenShift ルートなどのクラスタのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	Internal または External 。プライベートクラスタをデプロイするには、 publish を Internal に設定します。これはインターネットからアクセスできません。デフォルト値は External です。
sshKey	<p>クラスタマシンへのアクセスを認証するための SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の OpenShift Container Platform クラスタでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	たとえば、 sshKey: ssh-ed25519 AAAA.. です。

1.4.5.1.4. オプションの AWS 設定パラメーター

オプションの AWS 設定パラメーターは、以下の表で説明されています。

表1.8 オプションの AWS パラメーター

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)。

パラメーター	説明	値
<code>compute.platform.aws.rootVolume.size</code>	ルートボリュームのサイズ (GiB)。	整数 (例: 500)。
<code>compute.platform.aws.rootVolume.type</code>	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)。
<code>compute.platform.aws.type</code>	コンピュータマシンの 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)。
<code>compute.platform.aws.zones</code>	インストールプログラムがコンピュータマシンプールのマシンを作成するアベイラビリティゾーン。独自の VPC を指定する場合は、そのアベイラビリティゾーンにサブネットを指定する必要があります。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>compute.aws.region</code>	インストールプログラムがコンピュータ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)。
<code>controlPlane.platform.aws.type</code>	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)。
<code>controlPlane.platform.aws.zones</code>	インストールプログラムがコントロールプレーンマシンプールのマシンを作成するアベイラビリティゾーン。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>controlPlane.aws.region</code>	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)。
<code>platform.aws.userTags</code>	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。

パラメーター	説明	値
platform.aws.subnets	インストールプログラムによる VPC の作成を許可する代わりに VPC を指定する場合は、使用するクラスターのサブネットを指定します。サブネットは、指定する同じ machineNetwork[].cidr 範囲の一部である必要があります。標準クラスターの場合は、各アベイラビリティゾーンのパブリックおよびプライベートサブネットを指定します。プライベートクラスターについては、各アベイラビリティゾーンのプライベートサブネットを指定します。	有効なサブネット ID。



重要

Open Virtual Networking (OVN) Kubernetes ネットワークプラグインは、テクノロジープレビュー機能です。テクノロジープレビュー機能は Red Hat の実稼働環境でのサービスレベルアグリーメント (SLA) ではサポートされていないため、Red Hat では実稼働環境での使用を推奨していません。Red Hat は実稼働環境でこれらを使用することを推奨していません。これらの機能は、近々発表予定の製品機能をリリースに先駆けてご提供することにより、お客様は機能性をテストし、開発プロセス中にフィードバックをお寄せいただくことができます。

OVN テクノロジープレビュー機能のサポート範囲についての詳細は、<https://access.redhat.com/articles/4380121> を参照してください。

1.4.5.2. 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

インストール後は、**install-config.yaml** ファイルでこれらのパラメーターを変更することはできません。

表1.9 必要なネットワークパラメーター

パラメーター	説明	値
--------	----	---

パラメーター	説明	値
networking.net workType	デプロイするデフォルトの Container Network Interface (CNI) ネットワークプロバイダープラグイン。 OpenShiftSDN プラグインのみが OpenShift Container Platform 4.5 でサポートされているプラグインです。 OVNKubernetes プラグインは、OpenShift Container Platform 4.5 でテクノロジープレビューとしてご利用いただけます。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clus terNetwork[].cid r	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.clus terNetwork[].ho stPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞の長さ。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)。	サブネット接頭辞。デフォルト値は 23 です。
networking.serv iceNetwork[]	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.mac hineNetwork[].ci dr	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ノードに割り当てられる 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。複数の CIDR 範囲を指定できます。	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。

1.4.5.3. 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: 10
    clusterNetwork:
      - cidr: 10.128.0.0/14
        hostPrefix: 23
    machineNetwork:
      - cidr: 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
    fips: false 12
    sshKey: ssh-ed25519 AAAA... 13
    pullSecret: '{"auths": ...}' 14

```

1 9 11 14 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す

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。一部のクラスターマシンで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** FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。
- 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。
- 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。

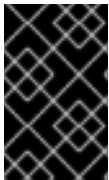


注記

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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

1.4.6. 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

インストールプログラムで作成される OpenShift Container Platform マニフェストファイルの変更はサポートされていません。以下の手順のように、作成するマニフェストファイルを適用することがサポートされています。

前提条件

- **install-config.yaml** ファイルを作成し、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> ❶
```

- ❶ **<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 ❶
```

- ❶ **<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、以下のようにいくつか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-*
```

出力例

```
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: ❶
  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
```

- ❶ **spec** パラメーターのパラメーターは例です。CR に Cluster Network Operator の設定を指定します。

CNO は CR にパラメーターのデフォルト値を提供するため、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。

1.4.7. Cluster Network Operator (CNO) の設定

クラスターネットワークの設定は、Cluster Network Operator (CNO) 設定の一部として指定され、**cluster** という名前の CR オブジェクトに保存されます。CR は **operator.openshift.io** API グループの **Network** API のパラメーターを指定します。

defaultNetwork パラメーターのパラメーター値を CNO CR に設定することにより、OpenShift Container Platform クラスターのクラスターネットワーク設定を指定できます。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: ❶
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  serviceNetwork: ❷
    - 172.30.0.0/16
  defaultNetwork: ❸
  ...
  kubeProxyConfig: ❹
    iptablesSyncPeriod: 30s ❺
    proxyArguments:
      iptables-min-sync-period: ❻
      - 0s
```

- ❶ ❷ **install-config.yaml** ファイルに指定されます。
- ❸ クラスターネットワークのデフォルトの Container Network Interface (CNI) ネットワークプロバイダーを設定します。
- ❹ 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、クラスターネットワーク 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。OVN-Kubernetes デフォルト CNI ネットワークプロバイダーを使用している場合、**kube-proxy** 設定は機能しません。
- ❺ **iptables** ルールの更新期間。デフォルト値は **30s** です。有効な接尾辞には、**s**、**m**、および **h** などが含まれ、これらについては、[Go Package time](#) ドキュメントで説明されています。



注記

OpenShift Container Platform 4.3 以降で強化されたパフォーマンスの向上により、**iptablesSyncPeriod** パラメーターを調整する必要はなくなりました。

- 6 **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。有効な接尾辞には、**s**、**m**、および **h** などが含まれ、これらについては、[Go](#)

1.4.7.1. OpenShift SDN デフォルト CNI ネットワークプロバイダーの設定パラメーター

以下の YAML オブジェクトは、OpenShift SDN デフォルト Container Network Interface (CNI) ネットワークプロバイダーの設定パラメーターについて説明しています。

```
defaultNetwork:
  type: OpenShiftSDN ①
  openshiftSDNConfig: ②
    mode: NetworkPolicy ③
    mtu: 1450 ④
    vxlanPort: 4789 ⑤
```

- ① **install-config.yaml** ファイルに指定されます。
- ② OpenShift 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。
- ③ OpenShift SDN のネットワーク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。
- ④ VXLAN オーバーレイネットワークの最大転送単位 (MTU)。これは、プライマリーネットワークインターフェイスの MTU に基づいて自動的に検出されます。通常、検出された MTU を上書きする必要はありません。

自動検出した値が予想される値ではない場合は、ノード上のプライマリーネットワークインターフェイスの MTU が正しいことを確認します。このオプションを使用して、ノード上のプライマリーネットワークインターフェイスの MTU 値を変更することはできません。

クラスターで異なるノードに異なる MTU 値が必要な場合、この値をクラスター内の最小の MTU 値よりも **50** 小さく設定する必要があります。たとえば、クラスター内の一部のノードでは MTU が **9001** であり、MTU が **1500** のクラスターもある場合には、この値を **1450** に設定する必要があります。

- ⑤ 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。

1.4.7.2. OVN-Kubernetes デフォルト CNI ネットワークプロバイダーの設定パラメーター

以下の YAML オブジェクトは OVN-Kubernetes デフォルト CNI ネットワークプロバイダーの設定パラメーターについて説明しています。

```
defaultNetwork:
  type: OVNKubernetes ①
  ovnKubernetesConfig: ②
```

```
mtu: 1400 3
genevePort: 6081 4
```

- 1 **install-config.yaml** ファイルに指定されます。
- 2 OVN-Kubernetes 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。
- 3 Geneve (Generic Network Virtualization Encapsulation) オーバーレイネットワークの MTU (maximum transmission unit)。これは、プライマリーネットワークインターフェースの MTU に基づいて自動的に検出されます。通常、検出された MTU を上書きする必要はありません。

自動検出した値が予想される値ではない場合は、ノード上のプライマリーネットワークインターフェースの MTU が正しいことを確認します。このオプションを使用して、ノード上のプライマリーネットワークインターフェースの MTU 値を変更することはできません。

クラスターで異なるノードに異なる MTU 値が必要な場合、この値をクラスター内の最小の MTU 値よりも **100** 小さく設定する必要があります。たとえば、クラスター内の一部のノードでは MTU が **9001** であり、MTU が **1500** のクラスターもある場合には、この値を **1400** に設定する必要があります。
- 4 Geneve オーバーレイネットワークの UDP ポート。

1.4.7.3. Cluster Network Operator の設定例

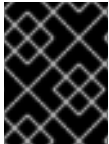
以下の例のように、CNO の完全な CR オブジェクトが表示されます。

Cluster Network Operator のサンプル CR

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  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
  kubeProxyConfig:
    iptablesSyncPeriod: 30s
    proxyArguments:
      iptables-min-sync-period:
        - 0s
```

1.4.8. クラスターのデプロイ

互換性のあるクラウドプラットフォームに 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の **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

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 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。



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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。



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。

1.4.9. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。



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 4.5 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、インストールします。

1.4.9.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。
PATH を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.4.9.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。

5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.4.9.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATH を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.4.10. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

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
```


- 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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.4.11. 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、[リモートの健全性レポートをオプトアウト](#) することができます。
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

1.5. AWS のクラスターの既存 VPC へのインストール

OpenShift Container Platform バージョン 4.5 では、クラスターを Amazon Web Services (AWS) の既存の Amazon Virtual Private Cloud (VPC) にインストールできます。インストールプログラムは、カスタマイズ可能な残りの必要なインフラストラクチャーをプロビジョニングします。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** ファイルでパラメーターを変更します。

1.5.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。
- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

1.5.2. カスタム VPC の使用について

OpenShift Container Platform 4.5 では、Amazon Web Services (AWS) の既存 Amazon Virtual Private Cloud (VPC) の既存のサブネットにクラスターをデプロイできます。OpenShift Container Platform を既存の AWS VPC にデプロイすると、新規アカウントの制限を回避したり、会社のガイドラインによる運用上の制約をより容易に遵守することが可能になる場合があります。VPC を作成するために必要なインフラストラクチャーの作成パーミッションを取得できない場合は、このインストールオプションを使用します。

インストールプログラムは既存のサブネットにある他のコンポーネントを把握できないため、ユーザーの代わりにサブネットの CIDR を選択することはできません。クラスターをインストールするサブネットのネットワークを独自に設定する必要があります。

1.5.2.1. VPC を使用するための要件

インストールプログラムは、以下のコンポーネントを作成しなくなりました。

- インターネットゲートウェイ
- NAT ゲートウェイ
- サブネット
- ルートテーブル
- VPC
- VPC DHCP オプション
- VPC エンドポイント

カスタム VPC を使用する場合は、そのカスタム VPC と使用するインストールプログラムおよびクラスターのサブネットを適切に設定する必要があります。インストールプログラムは、使用するクラスターのネットワーク範囲を細分化できず、サブネットのルートテーブルを設定するか、または DHCP などの VPC オプションを設定します。これは、クラスターのインストール前に設定する必要があります。

VPC は以下の特性を満たす必要があります。

- VPC の CIDR ブロックには、クラスターマシンの IP アドレスプールである **Networking.MachineCIDR** 範囲が含まれている必要があります。
- VPC は **kubernetes.io/cluster/.*: owned** タグを使用できません。
- VPC で **enableDnsSupport** および **enableDnsHostnames** 属性を有効にし、クラスターが VPC に割り当てられている Route 53 ゾーンを使用してクラスターの内部 DNS レコードを解決できるようにする必要があります。AWS ドキュメントの [DNS Support in Your VPC](#) を参照してください。

パブリックアクセスでクラスターを使用する場合、クラスターが使用する各アベイラビリティゾーンのパブリックおよびプライベートサブネットを作成する必要があります。

インストールプログラムは **kubernetes.io/cluster/.*: shared** タグを追加するようにサブネットを変更するため、サブネットでは1つ以上の空のタグスロットが利用可能である必要があります。AWS ドキュメントで現在の [タグ制限](#) を確認し、インストールプログラムでタグを指定する各サブネットに追加できるようにします。

非接続環境で作業している場合、EC2 および ELB エンドポイントのパブリック IP アドレスに到達することはできません。これを解決するには、VPC エンドポイントを作成し、これをクラスターが使用するサブネットに割り当てる必要があります。エンドポイントの名前は以下のように指定する必要があります。

す。

- `ec2.<region>.amazonaws.com`
- `elasticloadbalancing.<region>.amazonaws.com`
- `s3.<region>.amazonaws.com`

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	<ul style="list-style-type: none">● 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 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を使用して、S3 でホストされているレジストリーとの通信を強化します。	
パブリックサブネット	<ul style="list-style-type: none">● AWS::EC2::Subnet● AWS::EC2::SubnetNetworkAclAssociation	VPC には1から3のアベイラビリティ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	<ul style="list-style-type: none">● AWS::EC2::InternetGateway● AWS::EC2::VPCGatewayAttachment● AWS::EC2::RouteTable● AWS::EC2::Route● AWS::EC2::SubnetRouteTableAssociation● 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。提供されるテンプレートでは、各パブリックサブネットに EIP アドレスと NAT ゲートウェイがあります。これらの NAT ゲートウェイは、プライベートサブネットインスタンスなどのクラスターリソースがインターネットに到達できるようにするもので、一部のネットワークが制限された環境またはプロキシのシナリオでは必要ありません。	
ネットワークアクセス制御	<ul style="list-style-type: none">● AWS::EC2::NetworkAcl● AWS::EC2::NetworkAclEntry	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。	
		ポート	理由
		80	インバウンド HTTP トラフィック

コンポーネント	AWS タイプ	説明	
		443	インバウンド HTTPS トラフィック
		22	インバウンド SSH トラフィック
		1024 - 65535	インバウンド一時 (ephemeral) トラフィック
		0 - 65535	アウトバウンド一時 (ephemeral) トラフィック
プライベートサブネット	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	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。	

1.5.2.2. VPC 検証

指定するサブネットが適切であることを確認するには、インストールプログラムが以下のデータを確認します。

- 指定したサブネットすべてが存在します。
- プライベートサブネットを指定します。
- サブネットの CIDR は指定されたマシン CIDR に属します。
- 各アベイラビリティーゾーンのサブネットを指定します。それぞれのアベイラビリティーゾーンには、複数のパブリックおよびプライベートサブネットがありません。プライベートクラスターを使用する場合、各アベイラビリティーゾーンのプライベートサブネットのみを指定します。それ以外の場合は、各アベイラビリティーゾーンのパブリックサブネットおよびプライベートサブネットを指定します。
- 各プライベートサブネットアベイラビリティーゾーンのパブリックサブネットを指定します。マシンは、プライベートサブネットを指定しないアベイラビリティーゾーンにはプロビジョニングされません。

既存の VPC を使用するクラスターを破棄しても、VPC は削除されません。VPC から OpenShift Container Platform クラスターを削除する場合、**kubernetes.io/cluster/.*: shared** タグは、それが使用したサブネットから削除されます。

1.5.2.3. パーミッションの区分

OpenShift Container Platform 4.3 以降、クラスターのデプロイに、インストールプログラムがプロビジョニングするインフラストラクチャークラスターに必要なすべてのパーミッションを必要としなくなりました。この変更は、ある会社で個人がクラウドで他とは異なるリソースを作成できるようにパーミッションが区分された状態に類似するものです。たとえば、インスタンス、バケット、ロードバランサーなどのアプリケーション固有のアイテムを作成することはできますが、VPC、サブネット、または Ingress ルールなどのネットワーク関連のコンポーネントは作成できない可能性があります。

クラスターの作成時に使用する AWS の認証情報には、VPC、およびサブネット、ルーティングテーブル、インターネットゲートウェイ、NAT、VPN などの VPC 内のコアとなるネットワークコンポーネントの作成に必要なネットワークのパーミッションは必要ありません。ELB、セキュリティグループ、S3 バケットおよびノードなどの、クラスター内でマシンに必要なアプリケーションリソースを作成するパーミッションは依然として必要になります。

1.5.2.4. クラスター間の分離

OpenShift Container Platform を既存のネットワークにデプロイする場合、クラスターサービスの分離の規模は以下の方法で縮小されます。

- 複数の OpenShift Container Platform クラスターを同じ VPC にインストールできます。
- ICMP Ingress はネットワーク全体から許可されます。
- TCP 22 Ingress (SSH) はネットワーク全体に対して許可されます。
- コントロールプレーンの TCP 6443 Ingress (Kubernetes API) はネットワーク全体に対して許可されます。
- コントロールプレーンの TCP 22623 Ingress (MCS) はネットワーク全体に対して許可されます。

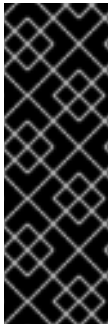
1.5.3. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリーが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.5.4. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
-f <path>/<file_name> 1
```

- 1 **~/.ssh/id_rsa** などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が **~/.ssh** ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。

1.5.5. インストールプログラムの取得

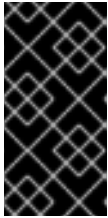
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

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、500 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。

**重要**

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除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イメージを提供する Quay.io など、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。

1.5.6. インストール設定ファイルの作成

Amazon Web Services (AWS) での 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します。

手順

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

- a. 以下のコマンドを実行します。

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

- 1** **<installation_directory>** の場合、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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

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。

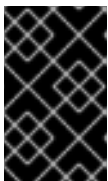
v. クラスターに設定した Route 53 サービスのベースドメインを選択します。

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を参照してください。

3. **install-config.yaml** ファイルをバックアップし、複数のクラスターをインストールするの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。

1.5.6.1. 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、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

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。



重要

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.5.6.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.10 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、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。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} 。 {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。

パラメーター	説明	値
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト
pullSecret	https://cloud.redhat.com/openshift/install/pull-secret からプルシークレットを取得し、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージのダウンロードを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.5.6.1.2. ネットワーク設定パラメーター


既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.11 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>

パラメーター	説明	値
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。
networking.clusterNetwork	Pod の IP アドレスブロック。 デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞長。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。 hostPrefix 値の 23 は、510 ($2^{(32-23)} - 2$) Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。
networking.serviceNetwork	サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 <pre>networking: serviceNetwork: - 172.30.0.0/16</pre>
networking.machineNetwork	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 <pre>networking: machineNetwork: - cidr: 10.0.0.0/16</pre>


パラメーター	説明	値
networking.machineNetwork.cidr	networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。	<p>CIDR 表記の IP ネットワークブロック。</p> <p>例: 10.0.0.0/16</p> <div>  <div> <p>注記</p> <p>優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。</p> </div> </div>

1.5.6.1.3. オプションの設定パラメーター

オプションのインストール設定パラメーターは、以下の表で説明されています。

表1.12 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータマシンで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。</p> <div>  <div> <p>重要</p> <p>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ていることを確認します。</p> </div> </div>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (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	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>コントロールプレーンマシンで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。</p> <div>  <div> <p>重要</p> <p>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ていることを確認します。</p> </div> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
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。	サポートされる値は 3 のみです (これはデフォルト値です)。
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <div> <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
publish	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	Internal または External 。プライベートクラスターをデプロイするには、 publish を Internal に設定します。これはインターネットからアクセスできません。デフォルト値は External です。
sshKey	<p>クラスターマシンへのアクセスを認証するための SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の OpenShift Container Platform クラスターでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	たとえば、 sshKey: ssh-ed25519 AAAA.. です。

1.5.6.1.4. オプションの AWS 設定パラメーター

オプションの AWS 設定パラメーターは、以下の表で説明されています。

表1.13 オプションの AWS パラメーター

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)。

パラメーター	説明	値
<code>compute.platform.aws.rootVolume.size</code>	ルートボリュームのサイズ (GiB)。	整数 (例: 500)。
<code>compute.platform.aws.rootVolume.type</code>	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)。
<code>compute.platform.aws.type</code>	コンピュータマシンの 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)。
<code>compute.platform.aws.zones</code>	インストールプログラムがコンピュータマシンプールのマシンを作成するアベイラビリティゾーン。独自の VPC を指定する場合は、そのアベイラビリティゾーンにサブネットを指定する必要があります。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>compute.aws.region</code>	インストールプログラムがコンピュータ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)。
<code>controlPlane.platform.aws.type</code>	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)。
<code>controlPlane.platform.aws.zones</code>	インストールプログラムがコントロールプレーンマシンプールのマシンを作成するアベイラビリティゾーン。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>controlPlane.aws.region</code>	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)。
<code>platform.aws.userTags</code>	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。

パラメーター	説明	値
platform.aws.subnets	インストールプログラムによる VPC の作成を許可する代わりに VPC を指定する場合は、使用するクラスターのサブネットを指定します。サブネットは、指定する同じ machineNetwork[].cidr 範囲の一部である必要があります。標準クラスターの場合は、各アベイラビリティゾーンのパブリックおよびプライベートサブネットを指定します。プライベートクラスターについては、各アベイラビリティゾーンのプライベートサブネットを指定します。	有効なサブネット ID。

1.5.6.2. 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 ❶
controlPlane: ❷
  hyperthreading: Enabled ❸ ❹
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1
      type: m5.xlarge ❺
    replicas: 3
compute: ❻
  - hyperthreading: Enabled ❼
    name: worker
    platform:
      aws:
        rootVolume:

```

```

    iops: 2000
    size: 500
    type: io1 ❸
    type: c5.4xlarge
    zones:
    - us-west-2c
  replicas: 3
  metadata:
    name: test-cluster ❹
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 10.0.0.0/16
    networkType: OpenShiftSDN
    serviceNetwork:
    - 172.30.0.0/16
  platform:
    aws:
      region: us-west-2 ❺
      userTags:
        adminContact: jdoe
        costCenter: 7536
      subnets: ❻
      - subnet-1
      - subnet-2
      - subnet-3
    fips: false ❼
    sshKey: ssh-ed25519 AAAA... ❽
    pullSecret: '{"auths": ...}' ❾

```

❶ ❹ ❺ ❾ 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。

❷ ❻ 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。

❸ ❼ **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。

❹ ❺ 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。一部のクラスターマシンで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。
- 11 独自の VPC を指定する場合は、クラスターが使用する各アベイラビリティゾーンのサブネットを指定します。
- 12 FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。
- 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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

1.5.6.3. インストール時のクラスター全体のプロキシの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシを使用することができます。プロキシ設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルが必要です。
- クラスターがアクセスする必要があるサイトを確認し、プロキシをバイパスする必要があるかどうかを判別します。デフォルトで、すべてのクラスター egress トラフィック (クラスターをホストするクラウドについてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。**Proxy** オブジェクトの **spec.noProxy** フィールドにサイトを追加し、必要に応じてプロキシをバイパスします。



注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Google Cloud Platform (GCP)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシ設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
```

```

httpProxy: http://<username>:<pswd>@<ip>:<port> ❶
httpsProxy: http://<username>:<pswd>@<ip>:<port> ❷
noProxy: example.com ❸
additionalTrustBundle: | ❹
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
...

```

- ❶ クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpProxy** 値を指定することはできません。
- ❷ クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。このフィールドが指定されていない場合、HTTP および HTTPS 接続の両方に **httpProxy** が使用されます。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpsProxy** 値を指定することはできません。
- ❸ プロキシを除外するための宛先ドメイン名、ドメイン、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシをバイパスします。
- ❹ 指定されている場合、インストールプログラムは HTTPS 接続のプロキシに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** 設定マップを作成し、この設定マップは **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、MITM CA 証明書を指定する必要があります。



注記

インストールプログラムは、プロキシの **readinessEndpoints** フィールドをサポートしません。

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシ設定を使用する **cluster** という名前のクラスター全体のプロキシを作成します。プロキシ設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。

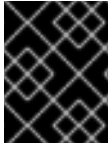


注記

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシを作成することはできません。

1.5.7. クラスターのデプロイ

互換性のあるクラウドプラットフォームに 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の **create cluster** コマンドは、初期インストール時に 1 回だけ実行できます。

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> \ ❶
--log-level=info ❷
```

❶ **<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。

❷ 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。



注記

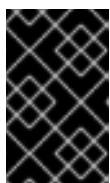
ホストに設定した AWS アカウントにクラスターをデプロイするための十分なパーミッションがない場合、インストールプログラムは停止し、不足しているパーミッションが表示されます。

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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。



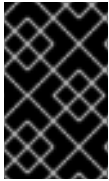
重要

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。

- 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。

1.5.8. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。



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 4.5 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、インストールします。

1.5.8.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

- Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
- インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
- Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
- アーカイブを展開します。

```
$ tar xvfz <file>
```

- oc** バイナリーを、**PATH** にあるディレクトリーに配置します。**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.5.8.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

- Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
- インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。

3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.5.8.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATH を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.5.9. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

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
```

- 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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.5.10. 次のステップ

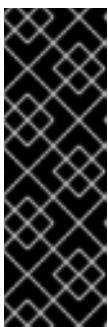
- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

1.6. プライベートクラスターの AWS へのインストール

OpenShift Container Platform バージョン 4.5 では、プライベートクラスターを Amazon Web Services (AWS) の既存の VPC にインストールできます。インストールプログラムは、カスタマイズ可能な残りの必要なインフラストラクチャーをプロビジョニングします。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** ファイルでパラメーターを変更します。

1.6.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。

- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

1.6.2. プライベートクラスター

お使いの環境で外部のインターネット接続を必要としない場合には、外部エンドポイントを公開しないプライベート OpenShift Container Platform クラスターをデプロイすることができます。プライベートクラスターは内部ネットワークからのみアクセス可能で、インターネット上では表示されません。

デフォルトで、OpenShift Container Platform はパブリックにアクセス可能な DNS およびエンドポイントを使用できるようにプロビジョニングされます。プライベートクラスターは、クラスターのデプロイ時に DNS、Ingress コントローラー、および API サーバーを private に設定します。つまり、クラスターリソースは内部ネットワークからのみアクセスでき、インターネット上では表示されません。

プライベートクラスターをデプロイするには、要件を満たす既存のネットワークを使用する必要があります。クラスターリソースはネットワーク上の他のクラスター間で共有される可能性があります。

さらに、プロビジョニングするクラウドの API サービスにアクセスできるマシンから、プロビジョニングするネットワーク上のホストおよびインストールメディアを取得するために使用するインターネットにプライベートクラスターをデプロイする必要があります。これらのアクセス要件を満たし、所属する会社のガイドラインに準拠したすべてのマシンを使用することができます。たとえば、このマシンには、クラウドネットワーク上の bastion ホスト、または VPN 経由でネットワークにアクセスできるマシンを使用できます。

1.6.2.1. AWS のプライベートクラスター

Amazon Web Services (AWS) でプライベートクラスターを作成するには、クラスターをホストするために既存のプライベート VPC およびサブネットを指定する必要があります。インストールプログラムは、クラスターが必要とする DNS レコードを解決できる必要もあります。インストールプログラムは、プライベートネットワークからのみアクセスできるように Ingress Operator および API サーバーを設定します。

クラスターには、引き続き AWS API にアクセスするためにインターネットへのアクセスが必要になります。

以下のアイテムは、プライベートクラスターのインストール時に必要ではなく、作成されません。

- パブリックサブネット
- パブリック Ingress をサポートするパブリックロードバランサー
- クラスターの **baseDomain** に一致するパブリック Route 53 ゾーン

インストールプログラムは、プライベート Route 53 ゾーンを作成するために指定する **baseDomain** とクラスターに必要なレコードを使用します。クラスターは、Operator がクラスターのパブリックレコードを作成せず、すべてのクラスターマシンが指定するプライベートサブネットに配置されるように設定されます。

1.6.2.1.1. 制限事項

プライベートクラスターにパブリック機能を追加する機能には制限があります。

- Kubernetes API エンドポイントは、追加のアクションを実行せずにインストールする場合はパブリックにすることができません。これらのアクションには、使用中のアベイラビリティゾーンごとに VPC でパブリックサブネットやパブリックのロードバランサーを作成すること

や、6443 のインターネットからのトラフィックを許可するようにコントロールプレーンのセキュリティグループを設定することなどが含まれます。

- パブリックのサービスタイプのロードバランサーを使用する場合には、各アベイラビリティゾーンのパブリックサブネットに **kubernetes.io/cluster/<cluster-infra-id>: shared** のタグを付け、AWS がそれらを使用してパブリックロードバランサーを作成できるようにします。

1.6.3. カスタム VPC の使用について

OpenShift Container Platform 4.5 では、Amazon Web Services (AWS) の既存 Amazon Virtual Private Cloud (VPC) の既存のサブネットにクラスターをデプロイできます。OpenShift Container Platform を既存の AWS VPC にデプロイすると、新規アカウントの制限を回避したり、会社のガイドラインによる運用上の制約をより容易に遵守することが可能になる場合があります。VPC を作成するために必要なインフラストラクチャーの作成パーミッションを取得できない場合は、このインストールオプションを使用します。

インストールプログラムは既存のサブネットにある他のコンポーネントを把握できないため、ユーザーの代わりにサブネットの CIDR を選択することはできません。クラスターをインストールするサブネットのネットワークを独自に設定する必要があります。

1.6.3.1. VPC を使用するための要件

インストールプログラムは、以下のコンポーネントを作成しなくなりました。

- インターネットゲートウェイ
- NAT ゲートウェイ
- サブネット
- ルートテーブル
- VPC
- VPC DHCP オプション
- VPC エンドポイント

カスタム VPC を使用する場合は、そのカスタム VPC と使用するインストールプログラムおよびクラスターのサブネットを適切に設定する必要があります。インストールプログラムは、使用するクラスターのネットワーク範囲を細分化できず、サブネットのルートテーブルを設定するか、または DHCP などの VPC オプションを設定します。これは、クラスターのインストール前に設定する必要があります。

VPC は以下の特性を満たす必要があります。

- VPC の CIDR ブロックには、クラスターマシンの IP アドレスプールである **Networking.MachineCIDR** 範囲が含まれている必要があります。
- VPC は **kubernetes.io/cluster/.*: owned** タグを使用できません。
- VPC で **enableDnsSupport** および **enableDnsHostnames** 属性を有効にし、クラスターが VPC に割り当てられている Route 53 ゾーンを使用してクラスターの内部 DNS レコードを解決できるようにする必要があります。AWS ドキュメントの [DNS Support in Your VPC](#) を参照してください。

パブリックアクセスでクラスターを使用する場合、クラスターが使用する各アベイラビリティゾーンのパブリックおよびプライベートサブネットを作成する必要があります。

インストールプログラムは **kubernetes.io/cluster/.*: shared** タグを追加するようにサブネットを変更するため、サブネットでは1つ以上の空のタグスロットが利用可能である必要があります。AWS ドキュメントで現在の **タグ制限** を確認し、インストールプログラムでタグを指定する各サブネットに追加できるようにします。

非接続環境で作業している場合、EC2 および ELB エンドポイントのパブリック IP アドレスに到達することはできません。これを解決するには、VPC エンドポイントを作成し、これをクラスターが使用するサブネットに割り当てる必要があります。エンドポイントの名前は以下のように指定する必要があります。

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

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	<ul style="list-style-type: none"> • 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 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を使用して、S3 でホストされているレジストリーとの通信を強化します。
パブリックサブネット	<ul style="list-style-type: none"> • AWS::EC2::Subnet • AWS::EC2::SubnetNetworkAclAssociation 	VPC には1から3のアベイラビリティ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	<ul style="list-style-type: none"> • AWS::EC2::InternetGateway • AWS::EC2::VPCGatewayAttachment • AWS::EC2::RouteTable • AWS::EC2::Route • AWS::EC2::SubnetRouteTableAssociation • 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。提供されるテンプレートでは、各パブリックサブネットに EIP アドレスと NAT ゲートウェイがあります。これらの NAT ゲートウェイは、プライベートサブネットインスタンスなどのクラスターリソースがインターネットに到達できるようにするもので、一部のネットワークが制限された環境またはプロキシのシナリオでは必要ありません。

コンポーネント	AWS タイプ	説明	
ネットワークアクセス制御	<ul style="list-style-type: none">● AWS::EC2::NetworkAcl● AWS::EC2::NetworkAclEntry	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。	
		ポート	理由
		80	インバウンド HTTP トラフィック
		443	インバウンド HTTPS トラフィック
		22	インバウンド SSH トラフィック
		1024 - 65535	インバウンド一時 (ephemeral) トラフィック
		0 - 65535	アウトバウンド一時 (ephemeral) トラフィック
プライベートサブネット	<ul style="list-style-type: none">● AWS::EC2::Subnet● AWS::EC2::RouteTable● AWS::EC2::SubnetRouteTableAssociation	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。	

1.6.3.2. VPC 検証

指定するサブネットが適切であることを確認するには、インストールプログラムが以下のデータを確認します。

- 指定したサブネットすべてが存在します。
- プライベートサブネットを指定します。
- サブネットの CIDR は指定されたマシン CIDR に属します。
- 各アベイラビリティゾーンのサブネットを指定します。それぞれのアベイラビリティゾーンには、複数のパブリックおよびプライベートサブネットがありません。プライベートクラスターを使用する場合、各アベイラビリティゾーンのプライベートサブネットのみを指定しま

す。それ以外の場合は、各アベイラビリティゾーンのパブリックサブネットおよびプライベートサブネットを指定します。

- 各プライベートサブネットアベイラビリティゾーンのパブリックサブネットを指定します。マシンは、プライベートサブネットを指定しないアベイラビリティゾーンにはプロビジョニングされません。

既存の VPC を使用するクラスターを破棄しても、VPC は削除されません。VPC から OpenShift Container Platform クラスターを削除する場合、**kubernetes.io/cluster/.*: shared** タグは、それが使用したサブネットから削除されます。

1.6.3.3. パーMISSIONの区分

OpenShift Container Platform 4.3 以降、クラスターのデプロイに、インストールプログラムがプロビジョニングするインフラストラクチャクラスターに必要なすべてのパーミッションを必要としなくなりました。この変更は、ある会社で個人がクラウドで他とは異なるリソースを作成できるようにパーミッションが区分された状態に類似するものです。たとえば、インスタンス、バケット、ロードバランサーなどのアプリケーション固有のアイテムを作成することはできますが、VPC、サブネット、または Ingress ルールなどのネットワーク関連のコンポーネントは作成できない可能性があります。

クラスターの作成時に使用する AWS の認証情報には、VPC、およびサブネット、ルーティングテーブル、インターネットゲートウェイ、NAT、VPN などの VPC 内のコアとなるネットワークコンポーネントの作成に必要なネットワークのパーミッションは必要ありません。ELB、セキュリティグループ、S3 バケットおよびノードなどの、クラスター内でマシンに必要なアプリケーションリソースを作成するパーミッションは依然として必要になります。

1.6.3.4. クラスター間の分離

OpenShift Container Platform を既存のネットワークにデプロイする場合、クラスターサービスの分離の規模は以下の方法で縮小されます。

- 複数の OpenShift Container Platform クラスターを同じ VPC にインストールできます。
- ICMP Ingress はネットワーク全体から許可されます。
- TCP 22 Ingress (SSH) はネットワーク全体に対して許可されます。
- コントロールプレーンの TCP 6443 Ingress (Kubernetes API) はネットワーク全体に対して許可されます。
- コントロールプレーンの TCP 22623 Ingress (MCS) はネットワーク全体に対して許可されます。

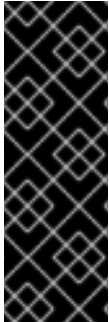
1.6.4. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.6.5. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。



注記

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
  -f <path>/<file_name> ❶
```

❶

`~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。

1.6.6. インストールプログラムの取得

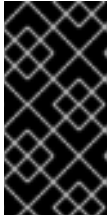
OpenShift Container Platform をインストールする前に、インストールファイルをローカルコンピューターにダウンロードします。

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、500 MB のローカルディスク領域が必要です。

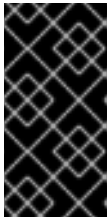
手順

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。



重要

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除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イメージを提供する Quay.io など、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。

1.6.7. インストール設定ファイルの手動作成

内部ネットワークからのみアクセスでき、インターネット上に表示されないプライベート OpenShift Container Platform クラスターのインストールの場合、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、複数のクラスターをインストールするのに使用できるようにします。

**重要**

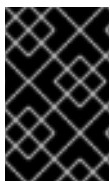
install-config.yaml ファイルは、インストールプロセスの次の手順で使用されます。この時点でこれをバックアップする必要があります。

1.6.7.1. 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、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**

インストール後は、これらのパラメーターを **install-config.yaml** ファイルで変更することはできません。

**重要**

openshift-install コマンドは、パラメーターのフィールド名を検証しません。正しくない名前を指定すると、関連するファイルまたはオブジェクトは作成されず、エラーが報告されません。指定されたパラメーターのフィールド名が正しいことを確認します。

1.6.7.1.1. 必須設定パラメーター

必須のインストール設定パラメーターは、以下の表で説明されています。

表1.14 必須パラメーター

パラメーター	説明	値
apiVersion	install-config.yaml コンテンツの API バージョン。現在のバージョンは v1 です。インストーラーは、古い API バージョンをサポートすることもできます。	文字列
baseDomain	クラウドプロバイダーのベースドメイン。ベースドメインは、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。
metadata	Kubernetes リソース ObjectMeta 。ここからは name パラメーターのみが消費されます。	オブジェクト
metadata.name	クラスターの名前。クラスターの DNS レコードはすべて {{.metadata.name}} . {{.baseDomain}} のサブドメインです。	dev などの小文字、ハイフン (-)、およびピリオド (.) が含まれる文字列。
platform	インストールの実行に使用する特定プラットフォームの設定: aws 、 baremetal 、 azure 、 openstack 、 ovirt 、 vsphere 。 platform.<platform> パラメーターに関する追加情報は、以下の表で特定のプラットフォームについて参照してください。	オブジェクト


パラメーター	説明	値
pullSecret	https://cloud.redhat.com/openshift/install/pull-secret からプルシークレットを取得し、Quay.io などのサービスから OpenShift Container Platform コンポーネントのコンテナイメージのダウンロードを認証します。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.6.7.1.2. ネットワーク設定パラメーター


既存のネットワークインフラストラクチャーの要件に基づいて、インストール設定をカスタマイズできます。たとえば、クラスターネットワークの IP アドレスブロックを拡張するか、デフォルトとは異なる IP アドレスブロックを指定できます。

IPv4 アドレスのみがサポートされます。

表1.15 ネットワークパラメーター

パラメーター	説明	値
networking	クラスターのネットワークの設定。	<p>オブジェクト</p>  <p>注記</p> <p>インストール後に networking オブジェクトで指定したパラメーターを変更することはできません。</p>
networking.networkType	インストールするクラスターネットワークプロバイダー Container Network Interface (CNI) プラグイン。	<p>OpenShiftSDN または OVNKubernetes のいずれか。デフォルト値は OpenShiftSDN です。</p>

パラメーター	説明	値
networking.clusterNetwork	Pod の IP アドレスブロック。 デフォルト値は 10.128.0.0/14 で、ホストの接頭辞は /23 です。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	networking.clusterNetwork を使用する場合に必須です。IP アドレスブロック。 IPv4 ネットワーク	CIDR (Classless Inter-Domain Routing) 表記の IP アドレスブロック。IPv4 ブロックの接頭辞長は 0 から 32 の間になります。
networking.clusterNetwork.hostPrefix	それぞれの個別ノードに割り当てるサブネット接頭辞長。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。 hostPrefix 値の 23 は、510 ($2^{(32 - 23)} - 2$) Pod IP アドレスを提供します。	サブネット接頭辞。 デフォルト値は 23 です。
networking.serviceNetwork	サービスの IP アドレスブロック。デフォルト値は 172.30.0.0/16 です。 OpenShift SDN および OVN-Kubernetes ネットワークプロバイダーは、サービスネットワークの単一 IP アドレスブロックのみをサポートします。	CIDR 形式の IP アドレスブロックを持つ配列。以下に例を示します。 networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	マシンの IP アドレスブロック。 複数の IP アドレスブロックを指定する場合は、ブロックが重複しないようにしてください。	オブジェクトの配列。以下に例を示します。 networking: machineNetwork: - cidr: 10.0.0.0/16

パラメーター	説明	値
networking.machineNetwork.cidr	networking.machineNetwork を使用する場合に必須です。IP アドレスブロック。libvirt 以外のすべてのプラットフォームでは、デフォルト値は 10.0.0.0/16 です。libvirt の場合、デフォルト値は 192.168.126.0/24 です。	CIDR 表記の IP ネットワークブロック。 例: 10.0.0.0/16  注記 優先される NIC が置かれている CIDR に一致する networking.machineNetwork を設定します。



1.6.7.1.3. オプションの設定パラメーター

オプションのインストール設定パラメーターは、以下の表で説明されています。

表1.16 オプションのパラメーター

パラメーター	説明	値
additionalTrustBundle	ノードの信頼済み証明書ストアに追加される PEM でエンコードされた X.509 証明書バンドル。この信頼バンドルは、プロキシが設定される際にも使用できます。	文字列
compute	コンピュートノードを設定するマシンの設定。	machine-pool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
compute.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
compute.hyperthreading	<p>コンピュータマシンで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。</p> <div>  <div> <p>重要</p> <p>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ていることを確認します。</p> </div> </div>	Enabled または Disabled
compute.name	compute を使用する場合に必須です。マシンプールの名前。	worker
compute.platform	compute を使用する場合に必須です。このパラメーターを使用して、ワーカーマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 o virt 、 vsphere 、または {}
compute.replicas	プロビジョニングするコンピュータマシン (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	コントロールプレーンを設定するマシンの設定。	MachinePool オブジェクトの配列。詳細は、以下の Machine-pool の表を参照してください。
controlPlane.architecture	プール内のマシンの命令セットアーキテクチャーを決定します。現時点で異種クラスターはサポートされていないため、すべてのプールが同じアーキテクチャーを指定する必要があります。有効な値は amd64 (デフォルト) です。	文字列

パラメーター	説明	値
controlPlane.hyperthreading	<p>コントロールプレーンマシンで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。</p> <div>  <p>重要</p> <p>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ていることを確認します。</p> </div>	Enabled または Disabled
controlPlane.name	controlPlane を使用する場合に必須です。マシンプールの名前。	master
controlPlane.platform	controlPlane を使用する場合に必須です。このパラメーターを使用して、コントロールプレーンマシンをホストするクラウドプロバイダーを指定します。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 、 azure 、 gcp 、 openstack 、 ovirt 、 vsphere 、または {}
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。	サポートされる値は 3 のみです (これはデフォルト値です)。
fips	<p>FIPS モードを有効または無効にします。デフォルトは false (無効) です。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。</p> <div>  <p>注記</p> <p>Azure File ストレージを使用している場合、FIPS モードを有効にすることはできません。</p> </div>	false または true

パラメーター	説明	値
imageContentSources	release-image コンテンツのソースおよびリポジトリ。	オブジェクトの配列。この表の以下の行で説明されているように、 source およびオプションで mirrors が含まれます。
imageContentSources.source	imageContentSources を使用する場合に必須です。ユーザーが参照するリポジトリを指定します (例: イメージプル仕様)。	文字列
imageContentSources.mirrors	同じイメージが含まれる可能性のあるリポジトリを1つ以上指定します。	文字列の配列。
publish	Kubernetes API、OpenShift ルートなどのクラスターのユーザーに表示されるエンドポイントをパブリッシュまたは公開する方法。	Internal または External 。プライベートクラスターをデプロイするには、 publish を Internal に設定します。これはインターネットからアクセスできません。デフォルト値は External です。
sshKey	<p>クラスターマシンへのアクセスを認証するための SSH キー。</p> <div>  <div> <p>注記</p> <p>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がある実稼働用の OpenShift Container Platform クラスターでは、ssh-agent プロセスが使用する SSH キーを指定します。</p> </div> </div>	たとえば、 sshKey: ssh-ed25519 AAAA.. です。

1.6.7.1.4. オプションの AWS 設定パラメーター

オプションの AWS 設定パラメーターは、以下の表で説明されています。

表1.17 オプションの AWS パラメーター

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)。

パラメーター	説明	値
<code>compute.platform.aws.rootVolume.size</code>	ルートボリュームのサイズ (GiB)。	整数 (例: 500)。
<code>compute.platform.aws.rootVolume.type</code>	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)。
<code>compute.platform.aws.type</code>	コンピュータマシンの 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)。
<code>compute.platform.aws.zones</code>	インストールプログラムがコンピュータマシンプールのマシンを作成するアベイラビリティゾーン。独自の VPC を指定する場合は、そのアベイラビリティゾーンにサブネットを指定する必要があります。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>compute.aws.region</code>	インストールプログラムがコンピュータ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)。
<code>controlPlane.platform.aws.type</code>	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)。
<code>controlPlane.platform.aws.zones</code>	インストールプログラムがコントロールプレーンマシンプールのマシンを作成するアベイラビリティゾーン。	YAML シーケンス の us-east-1c などの有効な AWS アベイラビリティゾーンの一覧。
<code>controlPlane.aws.region</code>	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)。
<code>platform.aws.userTags</code>	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。

パラメーター	説明	値
platform.aws.subnets	インストールプログラムによる VPC の作成を許可する代わりに VPC を指定する場合は、使用するクラスターのサブネットを指定します。サブネットは、指定する同じ machineNetwork[].cidr 範囲の一部である必要があります。標準クラスターの場合は、各アベイラビリティゾーンのパブリックおよびプライベートサブネットを指定します。プライベートクラスターについては、各アベイラビリティゾーンのプライベートサブネットを指定します。	有効なサブネット ID。

1.6.7.2. 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에만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 ❶
controlPlane: ❷
  hyperthreading: Enabled ❸ ❹
  name: master
  platform:
    aws:
      zones:
        - us-west-2a
        - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1
      type: m5.xlarge ❺
    replicas: 3
compute: ❻
  - hyperthreading: Enabled ❼
    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
    machineNetwork:
    - cidr: 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
      subnets: 11
      - subnet-1
      - subnet-2
      - subnet-3
    fips: false 12
    sshKey: ssh-ed25519 AAAA... 13
    publish: Internal 14
    pullSecret: '{"auths": ...}' 15

```

1 9 10 15 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す。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。一部のクラスターマシンで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。
- 11 独自の VPC を指定する場合は、クラスターが使用する各アベイラビリティゾーンのサブネットを指定します。
- 12 FIPS モードを有効または無効にするかどうか。デフォルトでは、FIPS モードは有効にされません。FIPS モードが有効にされている場合、OpenShift Container Platform が実行される Red Hat Enterprise Linux CoreOS (RHCOS) マシンがデフォルトの Kubernetes 暗号スイートをバイパスし、代わりに RHCOS で提供される暗号モジュールを使用します。
- 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がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

- 14 クラスターのユーザーに表示されるエンドポイントをパブリッシュする方法。プライベートクラスターをデプロイするには、**publish** を **Internal** に設定します。これはインターネットからアクセスできません。デフォルト値は **External** です。

1.6.7.3. インストール時のクラスター全体のプロキシの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシを使用することができます。プロキシ設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルが必要です。
- クラスターがアクセスする必要があるサイトを確認し、プロキシをバイパスする必要があるかどうかを判別します。デフォルトで、すべてのクラスター egress トラフィック (クラスターをホストするクラウドについてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。**Proxy** オブジェクトの **spec.noProxy** フィールドにサイトを追加し、必要に応じてプロキシをバイパスします。



注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Google Cloud Platform (GCP)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシ設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: http://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

- 1 クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpProxy** 値を指定することはできません。
- 2 クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。このフィールドが指定されていない場合、HTTP および HTTPS 接続の両方に **httpProxy** が使用されます。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpsProxy** 値を指定することはできません。
- 3 プロキシを除外するための宛先ドメイン名、ドメイン、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシをバイパスします。
- 4 指定されている場合、インストールプログラムは HTTPS 接続のプロキシに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** 設定マップを作成し、この設定マップは **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、MITM CA 証明書を指定する必要があります。

**注記**

インストールプログラムは、プロキシの **readinessEndpoints** フィールドをサポートしません。

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシ設定を使用する **cluster** という名前のクラスター全体のプロキシを作成します。プロキシ設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。

**注記**

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシを作成することはできません。

1.6.8. クラスターのデプロイ

互換性のあるクラウドプラットフォームに 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の **create cluster** コマンドは、初期インストール時に1回だけ実行できます。

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。

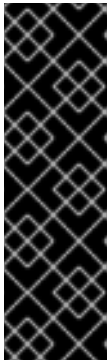
```
$ ./openshift-install create cluster --dir=<installation_directory> \ ❶
--log-level=info ❷
```

- ❶ **<installation_directory>** については、以下を指定します。
- ❷ 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。

**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**

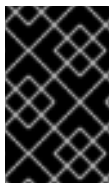
インストールプログラムが生成する Ignition 設定ファイルには、24 時間が経過すると期限切れになり、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。

**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。

1.6.9. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。

**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 4.5 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、インストールします。

1.6.9.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

■


```
$ oc <command>
```

1.6.9.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.6.9.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATH を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.6.10. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

- ❶ **<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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

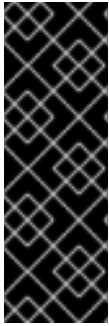
1.6.11. 次のステップ

- [クラスターをカスタマイズ](#) します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

1.7. CLOUDFORMATION テンプレートの使用による、AWS でのユーザーによってプロビジョニングされたインフラストラクチャーへのクラスターのインストール

OpenShift Container Platform バージョン 4.5 では、独自に提供するインフラストラクチャーを使用するクラスターを Amazon Web Services (AWS) にインストールできます。

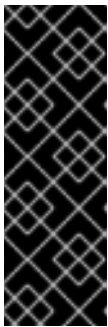
このインフラストラクチャーを作成する1つの方法として、提供される CloudFormation テンプレートを使用できます。テンプレートを変更してインフラストラクチャーをカスタマイズしたり、それらに含まれる情報を使用し、所属する会社のポリシーに基づいて AWS オブジェクトを作成したりできます。

**重要**

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インストールする手順は、例としてのみ提供されます。独自にプロビジョニングするインフラストラクチャーでクラスターをインストールするには、クラウドプロバイダーおよび OpenShift Container Platform のインストールプロセスについて理解する必要があります。これらの手順を実行するか、独自の手順を作成するのに役立つ複数の CloudFormation テンプレートが提供されます。他の方法を使用して必要なリソースを作成することもできます。これらのテンプレートはサンプルとしてのみ提供されます。

1.7.1. 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、クラスターがアクセスを必要とする [サイトを許可するようにファイアウォールを設定](#) する必要があります。

**注記**

プロキシを設定する場合は、このサイト一覧も確認してください。

- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

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

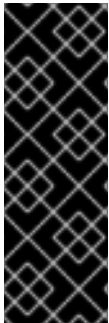
OpenShift Container Platform 4.5 では、クラスターをインストールするためにインターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブス

クリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.7.3. 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 のテスト済みインテグレーション](#) のページを参照してください。

提供される Cloud Formation 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、Cloud Formation テンプレートを参照してください。

1.7.3.1. 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マシンセット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。これらのマシンはマシンセットによって制御されません。

提供される Cloud Formation テンプレートを使用して、クラスターマシンの以下のインスタンスタイプを使用できます。

**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。

表1.18 マシンのインスタンスタイプ

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。

1.7.3.2. 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。

1.7.3.3. 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
- パブリックおよびプライベート Route 53 ゾーン
- セキュリティグループ
- IAM ロール
- S3 バケット

非接続環境で作業している場合、EC2 および ELB エンドポイントのパブリック IP アドレスに到達することはできません。これを解決するには、VPC エンドポイントを作成し、これをクラスターが使用するサブネットに割り当てる必要があります。エンドポイントの名前は以下のように指定する必要があります。

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

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	<ul style="list-style-type: none"> • 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 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を使用して、S3 でホストされているレジストリーとの通信を強化します。

コンポーネント	AWS タイプ	説明	
パブリックサブネット	<ul style="list-style-type: none">● 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	<ul style="list-style-type: none">● AWS::EC2::InternetGateway● AWS::EC2::VPCGatewayAttachment● AWS::EC2::RouteTable● AWS::EC2::Route● AWS::EC2::SubnetRouteTableAssociation● 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。提供されるテンプレートでは、各パブリックサブネットに EIP アドレスと NAT ゲートウェイがあります。これらの NAT ゲートウェイは、プライベートサブネットインスタンスなどのクラスターリソースがインターネットに到達できるようにするもので、一部のネットワークが制限された環境またはプロキシのシナリオでは必要ありません。	
ネットワークアクセス制御	<ul style="list-style-type: none">● AWS::EC2::NetworkAcl● AWS::EC2::NetworkAclEntry	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。	
		ポート	理由
		80	インバウンド HTTP トラフィック
		443	インバウンド HTTPS トラフィック
		22	インバウンド SSH トラフィック
		1024 - 65535	インバウンド一時 (ephemeral) トラフィック
		0 - 65535	アウトバウンド一時 (ephemeral) トラフィック

コンポーネント	AWS タイプ	説明
プライベートサブネット	<ul style="list-style-type: none"> ● AWS::EC2::Subnet ● AWS::EC2::RouteTable ● AWS::EC2::SubnetRouteTableAssociation 	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。

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 エントリを作成する必要があります。**api.<cluster_name>.<domain>** のエントリ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する必要があります。ポート 22623 はクラスター内のノードからアクセスする必要があります。

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。

コンポーネント	AWS タイプ	説明
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。
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

グループ	タイプ	IP プロトコル	ポート範囲
		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
MasterIngressEtc	etcd	tcp	2379- 2380
MasterIngressVxlan	Vxlan パケット	udp	4789
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
WorkerIngress Vxlan	Vxlan パケット	udp	4789
WorkerIngress WorkerVxlan	Vxlan パケット	udp	4789
WorkerIngress Internal	内部クラスター通信	tcp	9000 - 9999
WorkerIngress WorkerInternal	内部クラスター通信	tcp	9000 - 9999
WorkerIngress Kube	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 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 WorkerIngressServices	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	*

ロール	結果	アクション	リソース
	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	*

1.7.3.4. 必要な AWS パーミッション

AdministratorAccess ポリシーを、Amazon Web Services (AWS) で作成する 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 クラスターのすべてのコンポーネント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。

例1.11 インストールに必要な EC2 パーミッション

- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2>CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2>CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2>CreateVolume**
- **ec2>DeleteSecurityGroup**
- **ec2>DeleteSnapshot**
- **ec2>DeleteTags**
- **ec2:DeregisterImage**

- **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:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**

- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例1.12 インストール時のネットワークリソースの作成に必要なパーミッション

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



注記

既存の VPC を使用する場合、アカウントではネットワークリソースの作成にこれらのパーミッションを必要としません。

例1.13 インストールに必要な Elastic Load Balancing のパーミッション

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

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

例1.14 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**



注記

AWS アカウントに Elastic Load Balancer (ELB) を作成していない場合、IAM ユーザーには **iam:CreateServiceLinkedRole** パーミッションも必要です。

例1.15 インストールに必要な Route 53 パーミッション

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

例1.16 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:GetBucketAcl**

- **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**

例1.17 クラスター Operator が必要とする S3 パーミッション

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

例1.18 ベースクラスターリソースの削除に必要なパーミッション

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**

- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**
- **tag:GetResources**

例1.19 ネットワークリソースの削除に必要なパーミッション

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**



注記

既存の VPC を使用する場合、アカウントではネットワークリソースの削除にこれらのパーミッションを必要としません。

例1.20 マニフェストの作成に必要な追加の IAM および S3 パーミッション

- **iam:CreateAccessKey**

- **iam:CreateUser**
- **iam>DeleteAccessKey**
- **iam>DeleteUser**
- **iam>DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

1.7.4. 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

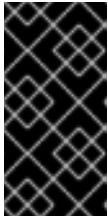
- Linux または macOS を使用するコンピューターからクラスターをインストールする必要があります。
- インストールプログラムをダウンロードするには、500 MB のローカルディスク領域が必要です。

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。

**重要**

インストールプログラムで作成されたファイルを削除しても、クラスターがインストール時に失敗した場合でもクラスターは削除されません。特定のクラウドプロバイダー用に記載された OpenShift Container Platform のアンインストール手順を完了して、クラスターを完全に削除する必要があります。

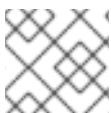
3. インストールプログラムを展開します。たとえば、Linux オペレーティングシステムを使用するコンピューターで以下のコマンドを実行します。

```
$ 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イメージを提供する Quay.io など、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。

1.7.5. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。

**注記**

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
  -f <path>/<file_name> 1
```

- 1 `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。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。

1.7.6. 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。

1.7.6.1. インストール設定ファイルの作成

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インストール設定ファイルを生成し、カスタマイズします。

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します。

手順

1. **install-config.yaml** ファイルを取得します。

- a. 以下のコマンドを実行します。

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

- 1 **<installation_directory>** の場合、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

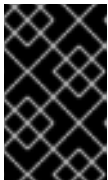
インストールのデバッグまたは障害復旧を実行する必要がある実稼働用の OpenShift Container Platform クラスターでは、**ssh-agent** プロセスが使用する SSH キーを指定します。

- 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 Route 53 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** ファイルを編集し、以下の **compute** スタンザに示されるようにコンピュートレプリカ (ワーカーレプリカとしても知られる) 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。

1.7.6.2. インストール時のクラスター全体のプロキシの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシを使用することができます。プロキシ設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルが必要です。
- クラスターがアクセスする必要があるサイトを確認し、プロキシをバイパスする必要があるかどうかを判別します。デフォルトで、すべてのクラスター egress トラフィック (クラスターをホストするクラウドについてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。**Proxy** オブジェクトの **spec.noProxy** フィールドにサイトを追加し、必要に応じてプロキシをバイパスします。



注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Google Cloud Platform (GCP)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシ設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> ❶
  httpsProxy: http://<username>:<pswd>@<ip>:<port> ❷
  noProxy: example.com ❸
```

```
additionalTrustBundle: | 4
-----BEGIN CERTIFICATE-----
<MY_TRUSTED_CA_CERT>
-----END CERTIFICATE-----
...
```

- 1 クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpProxy** 値を指定することはできません。
- 2 クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。このフィールドが指定されていない場合、HTTP および HTTPS 接続の両方に **httpProxy** が使用されます。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpsProxy** 値を指定することはできません。
- 3 プロキシを除外するための宛先ドメイン名、ドメイン、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に . を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。* を使用し、すべての宛先のプロキシをバイパスします。
- 4 指定されている場合、インストールプログラムは HTTPS 接続のプロキシに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** 設定マップを作成し、この設定マップは **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、MITM CA 証明書を指定する必要があります。



注記

インストールプログラムは、プロキシの **readinessEndpoints** フィールドをサポートしません。

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシ設定を使用する **cluster** という名前のクラスター全体のプロキシを作成します。プロキシ設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。



注記

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシを作成することはできません。

1.7.6.3. Kubernetes マニフェストおよび Ignition 設定ファイルの作成

一部のクラスター定義ファイルを変更し、クラスターマシンを手動で起動するため、クラスターがマシンを作成するために必要な Kubernetes マニフェストと 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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリー についてのドキュメントを参照してください。

前提条件

- OpenShift Container Platform インストールプログラムを取得します。
- **install-config.yaml** インストール設定ファイルを作成します。

手順

1. 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
```

出力例

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

- 1 **<installation_directory>** については、作成した **install-config.yaml** ファイルが含まれる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タ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。

2. コントロールプレーンマシンを定義する Kubernetes マニフェストファイルを削除します。

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

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。

3. 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 <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

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。

4. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルを変更し、Pod がコントロールプレーンマシンにスケジュールされないようにします。
 - a. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** ファイルを開きます。

- b. **mastersSchedulable** パラメーターを見つけ、その値を **False** に設定します。
 - c. ファイルを保存し、終了します。
5. オプション: **Ingress Operator** を DNS レコードを作成するよう設定する必要がない場合は、**<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 設定ファイルから **privateZone** および **publicZone** セクションを削除します。

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: ❶
    id: mycluster-100419-private-zone
  publicZone: ❷
    id: example.openshift.com
status: {}
```

- ❶ ❷ このセクションを完全に削除します。

これを実行する場合、後のステップで Ingress DNS レコードを手動で追加する必要があります。

6. Ignition 設定ファイルを取得します。

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

- ❶ **<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
```

1.7.7. インフラストラクチャー名の抽出

Ignition 設定ファイルには、Amazon Web Services (AWS) 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インフラストラクチャー名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します。

- クラスターの Ignition 設定ファイルを生成します。
- **jq** パッケージをインストールします。

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し、表示するには、以下のコマンドを実行します。

```
$ jq -r .infraID <installation_directory>/metadata.json ❶
```

- ❶ **<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。

出力例

```
openshift-vw9j6 ❶
```

- ❶ 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。

1.7.8. 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。

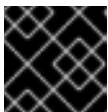
```
[
  {
    "ParameterKey": "VpcCidr", ❶
    "ParameterValue": "10.0.0.0/16" ❷
  },
  {
    "ParameterKey": "AvailabilityZoneCount", ❸
```

```

    "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その名前です。
 - 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。

1.7.8.1. 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。

例1.21 VPC の CloudFormation テンプレート

```

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"]]
    ]

```

1.7.9. 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

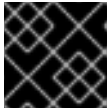
提供される CloudFormation テンプレートを使用して AWS インフラストラクチャーを使用しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせする必要がある可能性があります。

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 Route 53 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>." \ ❶
-r '.HostedZones | .[] | select(.Name=="($name)") | .Id'
```

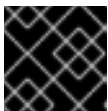
- ❶ **<route53_domain>** について、クラスターの **install-config.yaml** ファイルを生成した時に作成した Route 53 ベースドメインを指定します。

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", ❶
    "ParameterValue": "mycluster" ❷
  },
  {
    "ParameterKey": "InfrastructureName", ❸
    "ParameterValue": "mycluster-<random_string>" ❹
  },
  {
    "ParameterKey": "HostedZoneId", ❺
    "ParameterValue": "<random_string>" ❻
  },
  {
    "ParameterKey": "HostedZoneName", ❼
    "ParameterValue": "example.com" ❽
  },
  {
    "ParameterKey": "PublicSubnets", ❾
    "ParameterValue": "subnet-<random_string>" ❿
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

- ❶ 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 Ignition 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 Route 53 パブリックゾーン ID。
 - 6 **Z21IXYZABCZ2A4** に類する形式の Route 53 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 Route 53 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 Route 53 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オド (.) 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その名前です。

- 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。
RegisterNlbTargetGroupsLambda	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。

1.7.9.1. 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。

例1.22 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

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 Z21XYZABCZ2A4.

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:
  HealthCheckIntervalSeconds: 10
  HealthCheckPath: "/readyz"
  HealthCheckPort: 6443

```


HealthCheckProtocol: HTTPS
HealthyThresholdCount: 2
UnhealthyThresholdCount: 2
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:
 HealthCheckIntervalSeconds: 10
 HealthCheckPath: "/readyz"
 HealthCheckPort: 6443
 HealthCheckProtocol: HTTPS
 HealthyThresholdCount: 2
 UnhealthyThresholdCount: 2
 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:

HealthCheckIntervalSeconds: 10
HealthCheckPath: "/healthz"
HealthCheckPort: 22623
HealthCheckProtocol: HTTPS
HealthyThresholdCount: 2
UnhealthyThresholdCount: 2
Port: 22623
Protocol: TCP
TargetType: ip
VpcId:
 Ref: VpcId
TargetGroupAttributes:
 - Key: deregistration_delay.timeout_seconds
 Value: 60

RegisterTargetLambdalaRole:

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

```

RegisterNlbTargets:
  Type: "AWS::Lambda::Function"
  Properties:
    Handler: "index.handler"
    Role:
      Fn::GetAtt:
        - "RegisterTargetLambdalaRole"
        - "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']['TargetId']})
            elif event['RequestType'] == 'Create':
                elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetId']})
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetId'])
    Runtime: "python3.7"
    Timeout: 120

```

```

RegisterSubnetTagsLambdalaRole:
  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:

- "RegisterSubnetTagsLambdaRole"

- "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.

Value: !GetAtt ExtApiElb.LoadBalancerFullName

InternalApiLoadBalancerName:
 Description: Full name of the internal API load balancer.
 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]]
 RegisterNlbTargetsLambda:
 Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.
 Value: !GetAtt RegisterNlbTargets.Arn
 ExternalApiTargetGroupArn:
 Description: ARN of the external API target group.
 Value: !Ref ExternalApiTargetGroup
 InternalApiTargetGroupArn:
 Description: ARN of the internal API target group.
 Value: !Ref InternalApiTargetGroup
 InternalServiceTargetGroupArn:
 Description: ARN of the internal service target group.
 Value: !Ref InternalServiceTargetGroup

1.7.10. 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。

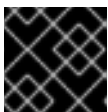
```
[
  {
    "ParameterKey": "InfrastructureName", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "VpcCidr", ❸
    "ParameterValue": "10.0.0.0/16" ❹
  }
]
```

```

    },
    {
      "ParameterKey": "PrivateSubnets", ❸
      "ParameterValue": "subnet-<random_string>" ❹
    },
    {
      "ParameterKey": "VpcId", ❺
      "ParameterValue": "vpc-<random_string>" ❻
    }
  ]
}

```

- ❶ クラスターの Ignition 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。
 - ❷ 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。
 - ❸ VPC の CIDR ブロック。
 - ❹ **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。
 - ❺ VPC 用に作成したプライベートサブネット。
 - ❻ VPC の CloudFormation テンプレートの出力から **PrivateSubnetIds** 値を指定します。
 - ❼ クラスター用に作成した VPC。
 - ❽ 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> ❶
  --template-body file://<template>.yaml ❷
  --parameters file://<parameters>.json ❸
  --capabilities CAPABILITY_NAMED_IAM

```

- ❶ **<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。
- ❷ **<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その名前です。
- ❸ **<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グループ ID
WorkerSecurityGroupID	ワーカーセキュリティグループ 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

1.7.10.1. セキュリティオブジェクトの 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オブジェクトをデプロイすることができます。

例1.23 セキュリティオブジェクトの CloudFormation テンプレート

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

MasterIngressEtc:

Type: AWS::EC2::SecurityGroupIngress

Properties:

- GroupId: !GetAtt MasterSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
- Description: etc
- 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

MasterIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

- GroupId: !GetAtt MasterSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
- Description: Geneve packets
- FromPort: 6081
- ToPort: 6081
- IpProtocol: udp

MasterIngressWorkerGeneve:

Type: AWS::EC2::SecurityGroupIngress

Properties:

- GroupId: !GetAtt MasterSecurityGroup.GroupId
- SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Geneve packets

FromPort: 6081

ToPort: 6081

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

MasterIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

MasterIngressWorkerInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

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

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

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

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

ToPort: 4789
IpProtocol: udp

WorkerIngressMasterVxlan:

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

WorkerIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
 Description: Geneve packets
 FromPort: 6081
 ToPort: 6081
 IpProtocol: udp

WorkerIngressMasterGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
 Description: Geneve packets
 FromPort: 6081
 ToPort: 6081
 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

WorkerIngressMasterInternal:

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

WorkerIngressInternalUDP:

Type: AWS::EC2::SecurityGroupIngress
Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Internal cluster communication
FromPort: 9000
ToPort: 9999
IpProtocol: udp

WorkerIngressMasterInternalUDP:

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

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

WorkerIngressMasterIngressServices:

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

WorkerIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

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:AttachVolume"

- "ec2:AuthorizeSecurityGroupIngress"

- "ec2:CreateSecurityGroup"

- "ec2:CreateTags"

- "ec2:CreateVolume"

- "ec2>DeleteSecurityGroup"

- "ec2>DeleteVolume"

- "ec2:Describe*"

- "ec2:DetachVolume"

- "ec2:ModifyInstanceAttribute"

- "ec2:ModifyVolume"

- "ec2:RevokeSecurityGroupIngress"

- "elasticloadbalancing:AddTags"

- "elasticloadbalancing:AttachLoadBalancerToSubnets"

- "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"

- "elasticloadbalancing:CreateListener"

- "elasticloadbalancing:CreateLoadBalancer"
- "elasticloadbalancing:CreateLoadBalancerPolicy"
- "elasticloadbalancing:CreateLoadBalancerListeners"
- "elasticloadbalancing:CreateTargetGroup"
- "elasticloadbalancing:ConfigureHealthCheck"
- "elasticloadbalancing>DeleteListener"
- "elasticloadbalancing>DeleteLoadBalancer"
- "elasticloadbalancing>DeleteLoadBalancerListeners"
- "elasticloadbalancing>DeleteTargetGroup"
- "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
- "elasticloadbalancing:DeregisterTargets"
- "elasticloadbalancing:Describe*"
- "elasticloadbalancing:DetachLoadBalancerFromSubnets"
- "elasticloadbalancing:ModifyListener"
- "elasticloadbalancing:ModifyLoadBalancerAttributes"
- "elasticloadbalancing:ModifyTargetGroup"
- "elasticloadbalancing:ModifyTargetGroupAttributes"
- "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
- "elasticloadbalancing:RegisterTargets"
- "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
- "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
- "kms:DescribeKey"

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:DescribeInstances"

- "ec2:DescribeRegions"

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

1.7.11. 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1.19 RHCOS AMI

AWS ゾーン	AWS AMI
ap-northeast-1	ami-0530d04240177f118
ap-northeast-2	ami-09e4cd700276785d2
ap-south-1	ami-0754b15d212830477
ap-southeast-1	ami-03b46cc4b1518c5a8
ap-southeast-2	ami-0a5b99ab2234a4e6a
ca-central-1	ami-012bc4ee3b6c673bc
eu-central-1	ami-02e08df1201f1c2f8
eu-north-1	ami-0309c9d2fadcb2d5a
eu-west-1	ami-0bdd69d8e7cd18188
eu-west-2	ami-0e610e967a62dbdfa
eu-west-3	ami-0e817e26f638a71ac

AWS ゾーン	AWS AMI
me-south-1	ami-024117d7c87b7ff08
sa-east-1	ami-08e62f746b94950c1
us-east-1	ami-077ede5bed2e431ea
us-east-2	ami-0f4ecf819275850dd
us-west-1	ami-0c4990e435bc6c5fe
us-west-2	ami-000d6e92357ac605c

1.7.12. 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

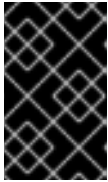
提供される CloudFormation テンプレートを使用してブートストラップノードを作成しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせる必要がある可能性があります。

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リールを作成します。

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を提供します。追加のセキュリティを提供するには、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 ❶
```

- ❶ <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/
```

出力例

```
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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "RhcosAmi", ❸
    "ParameterValue": "ami-<random_string>" ❹
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", ❺
    "ParameterValue": "0.0.0.0/0" ❻
  },
  {
    "ParameterKey": "PublicSubnet", ❼
  }
]
```

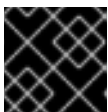
```

    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroup", 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": "RegisterNlbTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<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 Ignition 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 **RegisterNlbTargetsLambda** 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その名前です。
- 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。

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

1.7.12.1. 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。

例1.24 ブートストラップマシンの CloudFormation テンプレート

```
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[0-9]|2[0-4][0-9]|25[0-5])$
  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
RegisterNlbTargetsLambdaArn:
  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
 - RegisterNlbTargetsLambdaArn
 - 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 "MasterSecurityGroup"
    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 RegisterNLBTargetsLambdaArn

```



```
TargetArn: !Ref ExternalApiTargetGroupArn
TargetIp: !GetAtt BootstrapInstance.PrivateIp
```

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

```
RegisterBootstrapInternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    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
```

1.7.13. 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": "RegisterNlbTargetsLambdaArn", 29
      "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<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 Ignition 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 Route 53 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 Route 53 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 Route 53 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オド (.) 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グループ ID。
- 18 セキュリティ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**

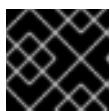
- **r4.xlarge**
- **r4.8xlarge**
- **r4.16xlarge**



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 **RegisterNlbTargetsLambda** 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
```

- ❶ **<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。
- ❷ **<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その名前です。
- ❸ **<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>
```

1.7.13.1. 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。

例1.25 コントロールプレーンマシンの CloudFormation テンプレート

```
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](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

RegisterNlbTargetsLambdaArn:

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

- RegisterNLbpTargetsLambdaArn

- 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 "MasterSecurityGroup"
              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 RegisterNlbTargetsLambdaArn
 TargetArn: !Ref ExternalApiTargetGroupArn
 TargetIp: !GetAtt Master0.PrivateIp

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

RegisterMaster0InternalServiceTarget:
 Condition: DoRegistration
 Type: Custom::NLBRegister
 Properties:
 ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 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:
 - AssociatePublicIp: "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, } } } } } } } } } }'
 - {
 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 RegisterNlbTargetsLambdaArn
 TargetArn: !Ref ExternalApiTargetGroupArn
 TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalApiTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:

ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 TargetArn: !Ref InternalApiTargetGroupArn
 TargetIp: !GetAtt Master1.PrivateIp

RegisterMaster1InternalServiceTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:

ServiceToken: !Ref RegisterNlbTargetsLambdaArn
 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 RegisterNlbTargetsLambdaArn

TargetArn: !Ref ExternalApiTargetGroupArn

TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalApiTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:

ServiceToken: !Ref RegisterNlbTargetsLambdaArn

TargetArn: !Ref InternalApiTargetGroupArn

TargetIp: !GetAtt Master2.PrivateIp

RegisterMaster2InternalServiceTarget:

Condition: DoRegistration

Type: Custom::NLBRegister

Properties:

ServiceToken: !Ref RegisterNlbTargetsLambdaArn

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]
]

```

1.7.14. 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 AWS 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リールを作成します。
- 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。

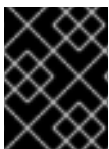
```
$ ./openshift-install wait-for bootstrap-complete --dir=<installation_directory> \ ❶
--log-level=info ❷
```

- ❶ **<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。
- ❷ 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。

1.7.14.1. 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。これらのノードを手動で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リールを作成します。
- 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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "RhcosAmi", ❸
    "ParameterValue": "ami-<random_string>" ❹
  },
  {
    "ParameterKey": "Subnet", ❺
    "ParameterValue": "subnet-<random_string>" ❻
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", ❼
    "ParameterValue": "sg-<random_string>" ❽
  },
  {
    "ParameterKey": "IgnitionLocation", ❾
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
  } ❿
  {
    "ParameterKey": "CertificateAuthorities", ⓫
    "ParameterValue": "" ⓬
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", ⓭
    "ParameterValue": "" ⓮
  },
  {
    "ParameterKey": "WorkerInstanceType", ⓯
    "ParameterValue": "m4.large" ⓰
  }
]
```

- ❶ クラスターの Ignition 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。
- ❷ 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。
- ❸ ワーカーノードに使用する最新の Red Hat Enterprise Linux CoreOS(RHCOS)AMI。
- ❹ **AWS::EC2::Image::Id** 値を指定します。
- ❺ 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)。
- ❻ 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。
- ❼ ワーカーノードに関連付けるワーカーセキュリティグループ ID。
- ❽ セキュリティ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。

workerSecurityGroup 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 **WorkerInstanceProfile** 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> ❶
--template-body file://<template>.yaml \ ❷
--parameters file://<parameters>.json ❸
```

- ❶ **<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。
- ❷ **<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その名前です。
- ❸ **<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。

1.7.14.1.1. 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。

例1.26 ワーカーマシンの CloudFormation テンプレート

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

```

1.7.15. バイナリーのダウンロードによる CLI のインストール

コマンドラインインターフェイスを使用して OpenShift Container Platform と対話するために CLI (**oc**) をインストールすることができます。**oc** は Linux、Windows、または macOS にインストールできます。



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 4.5 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、インストールします。

1.7.15.1. Linux への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Linux にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Linux** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開します。

```
$ tar xvzf <file>
```

5. **oc** バイナリーを、**PATH** にあるディレクトリーに配置します。**PATH** を確認するには、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.7.15.2. Windows での CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを Windows にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **Windows** を選択し、**Download command-line tools** をクリックします。
4. ZIP プログラムでアーカイブを解凍します。
5. **oc** バイナリーを、**PATH** にあるディレクトリーに移動します。
PATH を確認するには、コマンドプロンプトを開いて以下のコマンドを実行します。

```
C:\> path
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
C:\> oc <command>
```

1.7.15.3. macOS への CLI のインストール

以下の手順を使用して、OpenShift CLI (**oc**) バイナリーを macOS にインストールできます。

手順

1. Red Hat OpenShift Cluster Manager サイトの [Infrastructure Provider](#) ページに移動します。
2. インフラストラクチャプロバイダーを選択し、(該当する場合は) インストールタイプを選択します。
3. **Command-line interface** セクションで、ドロップダウンメニューの **MacOS** を選択し、**Download command-line tools** をクリックします。
4. アーカイブを展開し、解凍します。
5. **oc** バイナリーをパスにあるディレクトリーに移動します。
PATH を確認するには、ターミナルを開き、以下のコマンドを実行します。

```
$ echo $PATH
```

CLI のインストール後は、**oc** コマンドを使用して利用できます。

```
$ oc <command>
```

1.7.16. 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 クラスターをデプロイします。
- **oc** CLI をインストールします。

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
```

- 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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.7.17. マシンの証明書署名要求の承認

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。最初にクライアント要求を承認し、次にサーバー要求を承認する必要があります。

前提条件

- マシンがクラスターに追加されています。

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
```

出力例

```
NAME    STATUS    ROLES    AGE    VERSION
master-0 Ready     master   63m    v1.18.3
master-1 Ready     master   63m    v1.18.3
master-2 Ready     master   64m    v1.18.3
worker-0 NotReady   worker   76s    v1.18.3
worker-1 NotReady   worker   70s    v1.18.3
```

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
```

出力例

```
NAME    AGE    REQUESTOR                                CONDITION
csr-8b2br 15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

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。クライアントの CSR が承認されたら、Kubelet は提供証明書のセカンダリー CSR を作成します。これには、手動の承認が必要です。次に、後続の提供証明書の更新要求は、Kubelet が同じパラメーターを持つ新規証明書を要求する場合に **machine-approver** によって自動的に承認されます。

- 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 --no-run-if-empty oc adm certificate approve
```

- クライアント要求が承認されたら、クラスターに追加した各マシンのサーバー要求を確認する必要があります。

```
$ oc get csr
```

出力例

```
NAME      AGE  REQUESTOR                                CONDITION
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

- 残りの CSR が承認されず、それらが **Pending** ステータスにある場合、クラスターマシンの CSR を承認します。
 - 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。

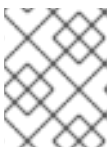
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

- すべてのクライアントおよびサーバーの CSR が承認された後に、マシンのステータスが **Ready** になります。以下のコマンドを実行して、これを確認します。

```
$ oc get nodes
```

出力例

```
NAME      STATUS  ROLES  AGE  VERSION
master-0  Ready   master  73m  v1.20.0
master-1  Ready   master  73m  v1.20.0
master-2  Ready   master  74m  v1.20.0
worker-0  Ready   worker  11m  v1.20.0
worker-1  Ready   worker  11m  v1.20.0
```



注記

サーバー CSR の承認後にマシンが **Ready** ステータスに移行するまでに数分の時間がかかる場合があります。

関連情報

- CSR の詳細は、[Certificate Signing Requests](#) を参照してください。

1.7.18. 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ます。

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
```

出力例

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED	SINCE
authentication	4.5.4	True	False	False	69s
cloud-credential	4.5.4	True	False	False	12m
cluster-autoscaler	4.5.4	True	False	False	11m
console	4.5.4	True	False	False	46s
dns	4.5.4	True	False	False	11m
image-registry	4.5.4	True	False	False	5m26s
ingress	4.5.4	True	False	False	5m36s
kube-apiserver	4.5.4	True	False	False	8m53s
kube-controller-manager	4.5.4	True	False	False	7m24s
kube-scheduler	4.5.4	True	False	False	12m
machine-api	4.5.4	True	False	False	12m
machine-config	4.5.4	True	False	False	7m36s
marketplace	4.5.4	True	False	False	7m54m
monitoring	4.5.4	True	False	False	7h54s
network	4.5.4	True	False	False	5m9s
node-tuning	4.5.4	True	False	False	11m
openshift-apiserver	4.5.4	True	False	False	11m
openshift-controller-manager	4.5.4	True	False	False	5m943s
openshift-samples	4.5.4	True	False	False	3m55s
operator-lifecycle-manager	4.5.4	True	False	False	11m
operator-lifecycle-manager-catalog	4.5.4	True	False	False	11m
service-ca	4.5.4	True	False	False	11m
service-catalog-apiserver	4.5.4	True	False	False	5m26s
service-catalog-controller-manager	4.5.4	True	False	False	5m25s
storage	4.5.4	True	False	False	5m30s

2. 利用不可の Operator を設定します。

1.7.18.1. イメージレジストリーストレージの設定

Amazon Web Services はデフォルトのストレージを提供します。つまり、Image Registry Operator はインストール後に利用可能になります。ただし、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、レジストリーストレージを手動で設定する必要があります。

実稼働クラスターに必要な永続ボリュームの設定についての手順が示されます。該当する場合、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。これは、実稼働以外のクラスターでのみ利用できます。

アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

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のレジストリーストレージを設定し、OpenShift Container Platform を非表示のリージョンにデプロイできます。詳細は、[Configuring the registry for AWS user-provisioned infrastructure](#) を参照してください。

1.7.18.1.1. 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を使用した AWS のレジストリーストレージの設定

インストール時に、Amazon 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。
- Amazon S3 ストレージの場合、シークレットには以下のキーが含まれることが予想されます。
 - **REGISTRY_STORAGE_S3_ACCESSKEY**
 - **REGISTRY_STORAGE_S3_SECRETKEY**

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
```

設定例

```
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。

1.7.18.1.2. 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

1. 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. イメージのビルドおよびプッシュを有効にするためにレジストリーが **managed** に設定されていることを確認します。

- 以下を実行します。

```
$ oc edit configs.imageregistry/cluster
```

次に、行を変更します。

```
managementState: Removed
```

上記を以下のように変更します。

```
managementState: Managed
```

1.7.19. ブートストラップリソースの削除

クラスターの初期 Operator 設定の完了後に、Amazon Web Services (AWS) からブートストラップリソースを削除します。

前提条件

- クラスターの初期 Operator 設定が完了済みです。

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。

1.7.20. Ingress DNS レコードの作成

DNS ゾーン設定を削除した場合には、Ingress ロードバランサーを参照する DNS レコードを手動で作成します。ワイルドカードレコードまたは特定のレコードのいずれかを作成できます。以下の手順では A レコードを使用しますが、CNAME やエイリアスなどの必要な他のレコードタイプを使用できます。

前提条件

- 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する OpenShift Container Platform クラスターを Amazon Web Services (AWS) にデプロイしています。
- OpenShift CLI (**oc**) をインストールします。
- **jq** パッケージをインストールします。
- 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。[Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#) を参照してください。

手順

1. 作成するルートを決めます。
 - ワイルドカードレコードを作成するには、***.apps.<cluster_name>.<domain_name>** を使用します。ここで、**<cluster_name>** はクラスター名で、**<domain_name>** は OpenShift Container Platform クラスターの Route 53 ベースドメインです。
 - 特定のレコードを作成するには、以下のコマンドの出力にあるように、クラスターが使用する各ルートにレコードを作成する必要があります。

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}\n'} routes
```

出力例

```
oauth-openshift.apps.<cluster_name>.<domain_name>
```

```
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

2. Ingress Operator ロードバランサーのステータスを取得し、使用する外部 IP アドレスの値をメモします。これは **EXTERNAL-IP** 列に表示されます。

```
$ oc -n openshift-ingress get service router-default
```

出力例

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
router-default	LoadBalancer	172.30.62.215	ab3...28.us-east-2.elb.amazonaws.com	80:31499/TCP,443:30693/TCP

3. ロードバランサーのホストゾーン ID を見つけます。

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName == "<external_ip>").CanonicalHostedZoneNameID' ❶
```

- ❶ **<external_ip>** については、取得した Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。

出力例

```
Z3AADJGX6KTTL2
```

このコマンドの出力は、ロードバランサーのホストゾーン ID です。

4. クラスターのドメインのパブリックホストゾーン ID を取得します。

```
$ aws route53 list-hosted-zones-by-name \
  --dns-name "<domain_name>" ❶ \
  --query 'HostedZones[? Config.PrivateZone != `true` && Name == `<domain_name>.`].Id' ❷ \
  --output text
```

- ❶ **<domain_name>** については、OpenShift Container Platform クラスターの Route 53 ベースドメインを指定します。

出力例

```
/hostedzone/Z3URY6TWQ91KVV
```

ドメインのパブリックホストゾーン ID がコマンド出力に表示されます。この例では、これは **Z3URY6TWQ91KVV** になります。

5. プライベートゾーンにエイリアスレコードを追加します。

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>",
>         "Type": "A",
>         "AliasTarget": {
>           "HostedZoneId": "<hosted_zone_id>",
>           "DNSName": "<external_ip>.",
>           "EvaluateTargetHealth": false
>         }
>       }
>     ]
>   }
> }'
```

- ❶ **<private_hosted_zone_id>** については、DNS および負荷分散の CloudFormation テンプレートから値を指定します。
- ❷ **<cluster_domain>** については、OpenShift Container Platform クラスターで使用するドメインまたはサブドメインを指定します。
- ❸ **<hosted_zone_id>** については、取得したロードバランサーのパブリックホストゾーン ID を指定します。
- ❹ **<external_ip>** については、Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。このパラメーターの値に末尾のピリオド (.) が含まれていることを確認します。

6. パブリックゾーンにレコードを追加します。

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --
change-batch '{
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>",
>         "Type": "A",
>         "AliasTarget": {
>           "HostedZoneId": "<hosted_zone_id>",
>           "DNSName": "<external_ip>.",
>           "EvaluateTargetHealth": false
>         }
>       }
>     ]
>   }
> }'
```

- ❶ **<public_hosted_zone_id>** については、ドメインのパブリックホストゾーンを指定します。

- 2 **<cluster_domain>** については、OpenShift Container Platform クラスターで使用するドメインまたはサブドメインを指定します。
- 3 **<hosted_zone_id>** については、取得したロードバランサーのパブリックホストゾーン ID を指定します。
- 4 **<external_ip>** については、Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。このパラメーターの値に末尾のピリオド (.) が含まれていることを確認します。

1.7.21. 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に、デプロイメントを完了するまでモニターします。

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で削除しています。
- **oc** CLI をインストールし、ログインします。

手順

- 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
```

- 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。

出力例

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



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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書からの回復 についてのドキュメントを参照してください。

1.7.22. 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、[リモートの健全性レポートをオプトアウト](#) することができます。

- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

1.8. 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ネットワークが制限された環境での AWS へのクラスターのインストール

OpenShift Container Platform バージョン 4.5 では、各自でプロビジョニングするインフラストラクチャーおよびインストールリリースコンテンツの内部ミラーを使用して、クラスターを Amazon Web Services (AWS) にインストールできます。



重要

ミラーリングされたインストールリリースのコンテンツを使用して OpenShift Container Platform クラスターをインストールすることは可能ですが、クラスターが AWS API を使用するにはインターネットへのアクセスが必要になります。

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。

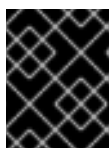


重要

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インストールする手順は、例としてのみ提供されます。独自にプロビジョニングするインフラストラクチャーでクラスターをインストールするには、クラウドプロバイダーおよび OpenShift Container Platform のインストールプロセスについて理解する必要があります。これらの手順を実行するか、独自の手順を作成するのに役立つ複数の CloudFormation テンプレートが提供されます。他の方法を使用して必要なリソースを作成することもできます。これらのテンプレートはサンプルとしてのみ提供されます。

1.8.1. 前提条件

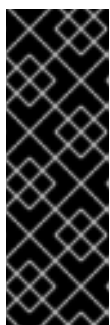
- [ミラーホストでミラーレジストリーを作成](#) しており、使用しているバージョンの OpenShift Container Platform の **imageContentSources** データを取得している。



重要

インストールメディアはミラーホストにあるため、そのコンピューターを使用してすべてのインストール手順を完了することができます。

- [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。
- [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し、Telemetry を使用する予定がある場合は、クラスターがアクセスする必要のある [サイトを許可するようにファイアウォールを設定](#) する必要があります。



注記

プロキシを設定する場合は、このサイト一覧も確認してください。

- システムが IAM(アイデンティティおよびアクセス管理) を管理できない場合、クラスター管理者は [IAM 認証情報を手動で作成し、維持](#) できます。手動モードは、クラウド IAM API に到達できない環境でも使用できます。

1.8.2. ネットワークが制限された環境でのインストールについて

OpenShift Container Platform 4.5 では、ソフトウェアコンポーネントを取得するためにインターネットへのアクティブな接続を必要としないインストールを実行できます。ネットワークが制限された環境のインストールは、クラスターのインストール先となるクラウドプラットフォームに応じて、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また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して実行できます。

クラウドプラットフォーム上でネットワークが制限されたインストールの実行を選択した場合でも、そのクラウド API へのアクセスが必要になります。Amazon Web Service の IAM サービスなどの一部のクラウド機能はインターネットアクセスを必要とするため、インターネットアクセスが依然として必要になる場合があります。ネットワークによっては、ベアメタルハードウェアまたは VMware vSphere へのインストールには、インターネットアクセスが必要になる場合があります。

ネットワークが制限されたインストールを完了するには、OpenShift Container Platform レジストリーのコンテンツをミラーリングし、インストールメディアを含むレジストリーを作成する必要があります。このミラーは、インターネットと制限されたネットワークの両方にアクセスできるミラーホストで、または制限に対応する他の方法を使用して作成できます。



重要

ユーザーによってプロビジョニングされるインストールの設定は複雑であるため、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ネットワークが制限されたインストールを試行する前に、標準的な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実行することを検討してください。このテストが完了すると、ネットワークが制限されたインストール時に発生する可能性のある問題の切り分けやトラブルシューティングがより容易になります。

1.8.2.1. その他の制限

ネットワークが制限された環境のクラスターには、以下の追加の制限および制約があります。

- **ClusterVersion** ステータスには **Unable to retrieve available updates** エラーが含まれます。
- デフォルトで、開発者カタログのコンテンツは、必要とされるイメージストリームタグにアクセスできないために使用できません。

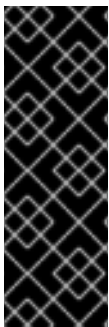
1.8.3. OpenShift Container Platform のインターネットアクセスおよび Telemetry アクセス

OpenShift Container Platform 4.5 では、クラスターをインストールするために必要なイメージを取得するために、インターネットアクセスが必要になります。クラスターの健全性および正常に実行された更新についてのメトリクスを提供するためにデフォルトで実行される Telemetry サービスにもインターネットアクセスが必要です。クラスターがインターネットに接続されている場合、Telemetry は自動的に実行され、クラスターは [Red Hat OpenShift Cluster Manager \(OCM\)](#) に登録されます。

Red Hat OpenShift Cluster Manager インベントリーが Telemetry によって自動的に維持されるか、または OCM を手動で使用しているかのいずれによって正常であることを確認した後に、[subscription watch](#) を使用して、アカウントまたはマルチクラスターレベルで OpenShift Container Platform サブスクリプションを追跡します。

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](#) ページにアクセスし、インストールプログラムをダウンロードし、サブスクリプション管理を実行します。クラスターにインターネットアクセスがあり、Telemetry を無効にしない場合、そのサービスは有効なサブスクリプションでクラスターを自動的に使用します。
- 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に [Quay.io](#) にアクセスします。
- クラスターの更新を実行するために必要なパッケージを取得します。



重要

クラスターでインターネットに直接アクセスできない場合、プロビジョニングする一部のタイプのインフラストラクチャーでネットワークが制限されたインストールを実行できます。このプロセスで、必要なコンテンツをダウンロードし、これを使用してミラーレジストリーにクラスターのインストールおよびインストールプログラムの生成に必要なパッケージを設定します。インストールタイプによっては、クラスターのインストール環境でインターネットアクセスが不要となる場合があります。クラスターを更新する前に、ミラーレジストリーのコンテンツを更新します。

1.8.4. 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 のテスト済みインテグレーション](#) のページを参照してください。

提供される Cloud Formation 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、Cloud Formation テンプレートを参照してください。

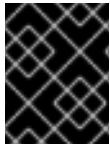
1.8.4.1. クラスターマシン

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マシンセット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。これらのマシンはマシンセットによって制御されません。

提供される Cloud Formation テンプレートを使用して、クラスターマシンの以下のインスタンスタイプを使用できます。



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。

表1.20 マシンのインスタンスタイプ

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

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	コンピューティング
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があります。

1.8.4.2. 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。

1.8.4.3. 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
- パブリックおよびプライベート Route 53 ゾーン
- セキュリティグループ
- IAM ロール
- S3 バケット

非接続環境で作業している場合、EC2 および ELB エンドポイントのパブリック IP アドレスに到達することはできません。これを解決するには、VPC エンドポイントを作成し、これをクラスターが使用するサブネットに割り当てる必要があります。エンドポイントの名前は以下のように指定する必要があります。

- **ec2.<region>.amazonaws.com**
- **elasticloadbalancing.<region>.amazonaws.com**
- **s3.<region>.amazonaws.com**

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	<ul style="list-style-type: none">● 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 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を使用して、S3 でホストされているレジストリーとの通信を強化します。	
パブリックサブネット	<ul style="list-style-type: none">● AWS::EC2::Subnet● AWS::EC2::SubnetNetworkAclAssociation	VPC には1から3のアベイラビリティ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	<ul style="list-style-type: none">● AWS::EC2::InternetGateway● AWS::EC2::VPCGatewayAttachment● AWS::EC2::RouteTable● AWS::EC2::Route● AWS::EC2::SubnetRouteTableAssociation● 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。提供されるテンプレートでは、各パブリックサブネットに EIP アドレスと NAT ゲートウェイがあります。これらの NAT ゲートウェイは、プライベートサブネットインスタンスなどのクラスターリソースがインターネットに到達できるようにするもので、一部のネットワークが制限された環境またはプロキシのシナリオでは必要ありません。	
ネットワークアクセス制御	<ul style="list-style-type: none">● AWS::EC2::NetworkAcl● AWS::EC2::NetworkAclEntry	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。	
		ポート	理由
		80	インバウンド HTTP トラフィック
		443	インバウンド HTTPS トラフィック
		22	インバウンド SSH トラフィック
		1024 - 65535	インバウンド一時 (ephemeral) トラフィック

コンポーネント	AWS タイプ	説明	
		0 - 65535	アウトバウンド一時 (ephemeral) トラフィック
プライベートサブネット	<ul style="list-style-type: none"> AWS::EC2::Subnet AWS::EC2::RouteTable AWS::EC2::SubnetRouteTableAssociation 	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。	

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する必要があります。ポート 22623 はクラスター内のノードからアクセスする必要があります。

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。

コンポーネント	AWS タイプ	説明
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。
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

グループ	タイプ	IP プロトコル	ポート範囲
		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
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

Ingress グループ	説明	IP プロトコル	ポート範囲
MasterIngress WorkerIngress Services	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
WorkerIngress Vxlan	Vxlan パケット	udp	4789
WorkerIngress WorkerVxlan	Vxlan パケット	udp	4789
WorkerIngress Internal	内部クラスター通信	tcp	9000 - 9999
WorkerIngress WorkerInternal	内部クラスター通信	tcp	9000 - 9999
WorkerIngress Kube	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 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	*

1.8.4.4. 必要な AWS パーミッション

AdministratorAccess ポリシーを、Amazon Web Services (AWS) で作成する 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 クラスターのすべてのコンポーネント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。

例1.27 インストールに必要な EC2 パーミッション

- **tag:TagResources**
- **tag:UntagResources**
- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2>CreateNetworkInterface**
- **ec2:AttachNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**

- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **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:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**

- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:ReleaseAddress**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例1.28 インストール時のネットワークリソースの作成に必要なパーミッション

- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**



注記

既存の VPC を使用する場合、アカウントではネットワークリソースの作成にこれらのパーミッションを必要としません。

例1.29 インストールに必要な Elastic Load Balancing のパーミッション

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**

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

例1.30 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**



注記

AWS アカウントに Elastic Load Balancer (ELB) を作成していない場合、IAM ユーザーには **iam:CreateServiceLinkedRole** パーミッションも必要です。

例1.31 インストールに必要な Route 53 パーミッション

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

例1.32 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:GetBucketAcl**
- **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**

例1.33 クラスター Operator が必要とする S3 パーミッション

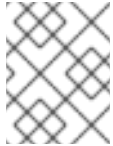
- **s3>DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

例1.34 ベースクラスターリソースの削除に必要なパーミッション

- **autoscaling:DescribeAutoScalingGroups**
- **ec2:DeleteNetworkInterface**
- **ec2:DeleteVolume**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DescribeTargetGroups**
- **iam:DeleteAccessKey**
- **iam:DeleteUser**
- **iam:ListInstanceProfiles**
- **iam:ListRolePolicies**
- **iam:ListUserPolicies**
- **s3:DeleteObject**
- **s3:ListBucketVersions**
- **tag:GetResources**

例1.35 ネットワークリソースの削除に必要なパーミッション

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReplaceRouteTableAssociation**

**注記**

既存の VPC を使用する場合、アカウントではネットワークリソースの削除にこれらのパーミッションを必要としません。

例1.36 マニフェストの作成に必要な追加の IAM および S3 パーミッション

- **iam:CreateAccessKey**
- **iam:CreateUser**
- **iam>DeleteAccessKey**
- **iam>DeleteUser**
- **iam>DeleteUserPolicy**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **s3:PutBucketPublicAccessBlock**
- **s3:GetBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**
- **s3:HeadBucket**
- **s3:ListBucketMultipartUploads**
- **s3:AbortMultipartUpload**

1.8.5. 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

クラスターで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がある場合、**ssh-agent** とインストールプログラムの両方に SSH キーを指定する必要があります。このキーを使用してパブリッククラスターのブートストラップマシンにアクセスし、インストールの問題をトラブルシューティングできます。

**注記**

実稼働環境では、障害復旧およびデバッグが必要です。

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 ed25519 -N "" \
-f <path>/<file_name> ①
```

- ① `~/.ssh/id_rsa` などの、新規 SSH キーのパスおよびファイル名を指定します。既存のキーペアがある場合は、公開鍵が `~/.ssh` ディレクトリーにあることを確認します。

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。



注記

FIPS で検証済み/進行中のモジュール (Modules in Process) 暗号ライブラリーを使用する OpenShift Container Platform クラスターを **x86_64** アーキテクチャーにインストールする予定の場合は、**ed25519** アルゴリズムを使用するキーは作成しないでください。代わりに、**rsa** アルゴリズムまたは **ecdsa** アルゴリズムを使用するキーを作成します。

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)"
```

出力例

```
Agent pid 31874
```

クラスターが FIPS モードにある場合は、FIPS 準拠のアルゴリズムのみを使用して SSH キーを生成します。鍵は RSA または ECDSA のいずれかである必要があります。

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

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

出力例

```
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ルプログラムに指定します。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。

1.8.6. 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。

1.8.6.1. インストール設定ファイルの作成

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インストール設定ファイルを生成し、カスタマイズします。

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します。ネットワークが制限されたインストールでは、これらのファイルがミラーホスト上に置かれます。

手順

1. **install-config.yaml** ファイルを取得します。

- a. 以下のコマンドを実行します。

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

- 1 **<installation_directory>** の場合、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。


```
imageContentSources:
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

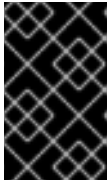
コマンドの出力の **imageContentSources** セクションを使用して、リポジトリ、またはネットワークが制限されたネットワークに取り込んだメディアからのコンテンツをミラーリングする際に使用した値をミラーリングします。

- d. オプション: パブリッシュストラテジーを **Internal** に設定します。

```
publish: Internal
```

このオプションを設定すると、内部 Ingress コントローラーおよびプライベートロードバランサーを作成します。

4. 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。

1.8.6.2. インストール時のクラスター全体のプロキシの設定

実稼働環境では、インターネットへの直接アクセスを拒否し、代わりに HTTP または HTTPS プロキシを使用することができます。プロキシ設定を **install-config.yaml** ファイルで行うことにより、新規の OpenShift Container Platform クラスターをプロキシを使用するように設定できます。

前提条件

- 既存の **install-config.yaml** ファイルが必要です。
- クラスターがアクセスする必要のあるサイトを確認し、プロキシをバイパスする必要があるかどうかを判別します。デフォルトで、すべてのクラスター egress トラフィック (クラスターをホストするクラウドについてのクラウドプロバイダー API に対する呼び出しを含む) はプロキシされます。**Proxy** オブジェクトの **spec.noProxy** フィールドにサイトを追加し、必要に応じてプロキシをバイパスします。



注記

Proxy オブジェクトの **status.noProxy** フィールドには、インストール設定の **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr**、および **networking.serviceNetwork[]** フィールドの値が設定されます。

Amazon Web Services (AWS)、Google Cloud Platform (GCP)、Microsoft Azure、および Red Hat OpenStack Platform (RHOSP) へのインストールの場合、**Proxy** オブジェクトの **status.noProxy** フィールドには、インスタンスメタデータのエンドポイント (**169.254.169.254**) も設定されます。

手順

1. **install-config.yaml** ファイルを編集し、プロキシ設定を追加します。以下に例を示します。

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> ❶
  httpsProxy: http://<username>:<pswd>@<ip>:<port> ❷
  noProxy: example.com ❸
additionalTrustBundle: | ❹
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
...
```

- ❶ クラスター外の HTTP 接続を作成するために使用するプロキシ URL。URL スキームは **http** である必要があります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpProxy** 値を指定することはできません。
- ❷ クラスター外で HTTPS 接続を作成するために使用するプロキシ URL。このフィールドが指定されていない場合、HTTP および HTTPS 接続の両方に **httpProxy** が使用されます。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、**httpsProxy** 値を指定することはできません。
- ❸ プロキシを除外するための宛先ドメイン名、ドメイン、IP アドレス、または他のネットワーク CIDR のコンマ区切りの一覧。サブドメインのみと一致するように、ドメインの前に **.** を付けます。たとえば、**.y.com** は **x.y.com** に一致しますが、**y.com** には一致しません。***** を使用し、すべての宛先のプロキシをバイパスします。
- ❹ 指定されている場合、インストールプログラムは HTTPS 接続のプロキシに必要な 1 つ以上の追加の CA 証明書が含まれる **user-ca-bundle** という名前の設定マップを **openshift-config** namespace に生成します。次に Cluster Network Operator は、これらのコンテンツを Red Hat Enterprise Linux CoreOS (RHCOS) 信頼バンドルにマージする **trusted-ca-bundle** 設定マップを作成し、この設定マップは **Proxy** オブジェクトの **trustedCA** フィールドで参照されます。**additionalTrustBundle** フィールドは、プロキシのアイデンティティ証明書が RHCOS 信頼バンドルからの認証局によって署名されない限り必要になります。追加のプロキシ設定が必要ではなく、追加の CA を必要とする MITM の透過的なプロキシネットワークを使用する場合には、MITM CA 証明書を指定する必要があります。

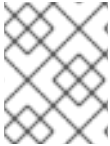


注記

インストールプログラムは、プロキシの **readinessEndpoints** フィールドをサポートしません。

2. ファイルを保存し、OpenShift Container Platform のインストール時にこれを参照します。

インストールプログラムは、指定の **install-config.yaml** ファイルのプロキシ設定を使用する **cluster** という名前のクラスター全体のプロキシを作成します。プロキシ設定が指定されていない場合、**cluster Proxy** オブジェクトが依然として作成されますが、これには **spec** がありません。



注記

cluster という名前の **Proxy** オブジェクトのみがサポートされ、追加のプロキシーを作成することはできません。

1.8.6.3. Kubernetes マニフェストおよび Ignition 設定ファイルの作成

一部のクラスター定義ファイルを変更し、クラスターマシンを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 Kubernetes マニフェストと 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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリー についてのドキュメントを参照してください。

前提条件

- OpenShift Container Platform インストールプログラムを取得します。ネットワークが制限されたインストールでは、これらのファイルがミラーホスト上に置かれます。
- **install-config.yaml** インストール設定ファイルを作成します。

手順

1. 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
```

出力例

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for
Scheduler cluster settings
```

- 1 **<installation_directory>** については、作成した **install-config.yaml** ファイルが含まれる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タマシンを作成するため、この警告を無視しても問題はありません。

2. コントロールプレーンマシンを定義する Kubernetes マニフェストファイルを削除します。

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

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。

3. 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 <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

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。

4. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes マニフェストファイルを変更し、Pod がコントロールプレーンマシンにスケジュールされないようにします。
 - a. **<installation_directory>/manifests/cluster-scheduler-02-config.yml** ファイルを開きます。
 - b. **mastersSchedulable** パラメーターを見つけ、その値を **False** に設定します。
 - c. ファイルを保存し、終了します。
5. オプション: [Ingress Operator](#) を DNS レコードを作成するよう設定する必要がない場合は、**<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 設定ファイルから **privateZone** および **publicZone** セクションを削除します。

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: ❶
    id: mycluster-100419-private-zone
  publicZone: ❷
    id: example.openshift.com
status: {}
```

- ❶ ❷ このセクションを完全に削除します。

これを実行する場合、後のステップで Ingress DNS レコードを手動で追加する必要があります。

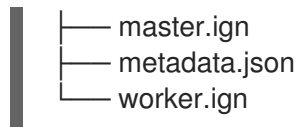
6. Ignition 設定ファイルを取得します。

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

- ❶ **<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
```

1.8.7. インフラストラクチャー名の抽出

Ignition 設定ファイルには、Amazon Web Services (AWS) 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インフラストラクチャー名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します。
- クラスターの Ignition 設定ファイルを生成します。
- **jq** パッケージをインストールします。

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し、表示するには、以下のコマンドを実行します。

```
$ jq -r .infraID <installation_directory>/metadata.json ❶
```

- ❶ **<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。

出力例

```
openshift-vw9j6 ❶
```

- ❶ 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。

1.8.8. 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。

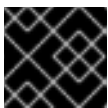
手順

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", ❶
    "ParameterValue": "10.0.0.0/16" ❷
  },
  {
    "ParameterKey": "AvailabilityZoneCount", ❸
    "ParameterValue": "1" ❹
  },
  {
    "ParameterKey": "SubnetBits", ❺
    "ParameterValue": "12" ❻
  }
]
```

- ❶ VPC の CIDR ブロック。
- ❷ **x.x.x.x/16-24** 形式で CIDR ブロックを指定します。
- ❸ VPC をデプロイするアベイラビリティゾーンの数。
- ❹ 1 から 3 の間の整数を指定します。
- ❺ 各アベイラビリティゾーン内の各サブネットのサイズ。
- ❻ 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> ❶
  --template-body file://<template>.yaml ❷
  --parameters file://<parameters>.json ❸
```

- ❶ **<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。
- ❷ **<template>** は、保存した CloudFormation テンプレート YAML ファイルへの相対パスです。

たはその名前です。

- 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。

1.8.8.1. 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。

例1.37 VPC の CloudFormation テンプレート

```
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"]]

]

1.8.9. 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 Route 53 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 Route 53 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
  },
  {

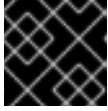
```

```

    "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 Ignition 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 Route 53 パブリックゾーン ID。
 - 6 **Z21IXYZABCZ2A4** に類する形式の Route 53 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 Route 53 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 Route 53 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オド (.) 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> ❶
--template-body file://<template>.yaml ❷
--parameters file://<parameters>.json ❸
--capabilities CAPABILITY_NAMED_IAM
```

- ❶ **<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。
- ❷ **<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その名前です。
- ❸ **<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。
RegisterNlbTargetLambda	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。

1.8.9.1. 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。

例1.38 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

```
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:

HealthCheckIntervalSeconds: 10

HealthCheckPath: "/readyz"

HealthCheckPort: 6443

HealthCheckProtocol: HTTPS

HealthyThresholdCount: 2

UnhealthyThresholdCount: 2

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:

HealthCheckIntervalSeconds: 10

HealthCheckPath: "/readyz"

HealthCheckPort: 6443

HealthCheckProtocol: HTTPS

HealthyThresholdCount: 2

UnhealthyThresholdCount: 2

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:
HealthCheckIntervalSeconds: 10
HealthCheckPath: "/healthz"
HealthCheckPort: 22623
HealthCheckProtocol: HTTPS
HealthyThresholdCount: 2
UnhealthyThresholdCount: 2
Port: 22623
Protocol: TCP
TargetType: ip
VpcId:
Ref: VpcId
TargetGroupAttributes:
- Key: deregistration_delay.timeout_seconds
Value: 60

RegisterTargetLambdalaRole:
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:
            - "RegisterTargetLambdalaRole"
            - "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

RegisterSubnetTagsLambdalaRole:
    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:
        - "RegisterSubnetTagsLambdaramRole"
        - "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.

Value: !GetAtt ExtApiElb.LoadBalancerFullName

InternalApiLoadBalancerName:

Description: Full name of the internal API load balancer.

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]]

RegisterNlbPTargetsLambda:

Description: Lambda ARN useful to help register or deregister IP targets for these load balancers.

Value: !GetAtt RegisterNlbPTargets.Arn

ExternalApiTargetGroupArn:

Description: ARN of the external API target group.

Value: !Ref ExternalApiTargetGroup

InternalApiTargetGroupArn:

Description: ARN of the internal API target group.

Value: !Ref InternalApiTargetGroup

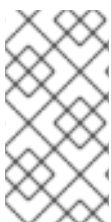
InternalServiceTargetGroupArn:

Description: ARN of the internal service target group.

Value: !Ref InternalServiceTargetGroup

1.8.10. 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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "VpcCidr", ❸
    "ParameterValue": "10.0.0.0/16" ❹
  },
  {
    "ParameterKey": "PrivateSubnets", ❺
    "ParameterValue": "subnet-<random_string>" ❻
  },
  {
    "ParameterKey": "VpcId", ❼
    "ParameterValue": "vpc-<random_string>" ❽
  }
]
```

- ❶ クラスターの Ignition 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。
- ❷ 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。
- ❸ VPC の CIDR ブロック。
- ❹ **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。
- ❺ VPC 用に作成したプライベートサブネット。
- ❻ VPC の CloudFormation テンプレートの出力から **PrivateSubnetIds** 値を指定します。
- ❼ クラスター用に作成した VPC。
- ❽ 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> ❶
--template-body file://<template>.yaml ❷
--parameters file://<parameters>.json ❸
--capabilities CAPABILITY_NAMED_IAM
```

- ❶ **<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。
- ❷ **<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その名前です。
- ❸ **<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。

MasterSecurityGroup Id	マスターセキュリティグループ ID
WorkerSecurityGroup Id	ワーカーセキュリティグループ 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

1.8.10.1. セキュリティオブジェクトの 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オブジェクトをデプロイすることができます。

例1.39 セキュリティオブジェクトの CloudFormation テンプレート

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])|(V(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

MasterIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
IpProtocol: udp

MasterIngressWorkerGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
GroupId: !GetAtt MasterSecurityGroup.GroupId
SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
Description: Geneve packets
FromPort: 6081
ToPort: 6081
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

MasterIngressInternalUDP:

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

MasterIngressWorkerInternalUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Internal cluster communication

FromPort: 9000

ToPort: 9999

IpProtocol: udp

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

MasterIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt MasterSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services
FromPort: 30000
ToPort: 32767
IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:

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

WorkerIngressVxlan:

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

WorkerIngressMasterVxlan:

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

WorkerIngressGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
 Description: Geneve packets
 FromPort: 6081
 ToPort: 6081
 IpProtocol: udp

WorkerIngressMasterGeneve:

Type: AWS::EC2::SecurityGroupIngress
Properties:
 GroupId: !GetAtt WorkerSecurityGroup.GroupId
 SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
 Description: Geneve packets
 FromPort: 6081
 ToPort: 6081
 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

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

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

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

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

WorkerIngressMasterIngressServices:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: tcp

WorkerIngressIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:

Type: AWS::EC2::SecurityGroupIngress

Properties:

GroupId: !GetAtt WorkerSecurityGroup.GroupId

SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId

Description: Kubernetes ingress services

FromPort: 30000

ToPort: 32767

IpProtocol: udp

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:AttachVolume"
    - "ec2:AuthorizeSecurityGroupIngress"
    - "ec2:CreateSecurityGroup"
    - "ec2:CreateTags"
    - "ec2:CreateVolume"
    - "ec2>DeleteSecurityGroup"
    - "ec2>DeleteVolume"
    - "ec2:Describe*"
    - "ec2:DetachVolume"
    - "ec2:ModifyInstanceAttribute"
    - "ec2:ModifyVolume"
    - "ec2:RevokeSecurityGroupIngress"
    - "elasticloadbalancing:AddTags"
    - "elasticloadbalancing:AttachLoadBalancerToSubnets"
    - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
    - "elasticloadbalancing:CreateListener"
    - "elasticloadbalancing:CreateLoadBalancer"
    - "elasticloadbalancing:CreateLoadBalancerPolicy"
    - "elasticloadbalancing:CreateLoadBalancerListeners"
    - "elasticloadbalancing:CreateTargetGroup"
    - "elasticloadbalancing:ConfigureHealthCheck"
    - "elasticloadbalancing>DeleteListener"
    - "elasticloadbalancing>DeleteLoadBalancer"
    - "elasticloadbalancing>DeleteLoadBalancerListeners"
    - "elasticloadbalancing>DeleteTargetGroup"
    - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
    - "elasticloadbalancing:DeregisterTargets"
    - "elasticloadbalancing:Describe*"
    - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
    - "elasticloadbalancing:ModifyListener"
    - "elasticloadbalancing:ModifyLoadBalancerAttributes"
    - "elasticloadbalancing:ModifyTargetGroup"
    - "elasticloadbalancing:ModifyTargetGroupAttributes"
    - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
    - "elasticloadbalancing:RegisterTargets"
    - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
    - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
    - "kms:DescribeKey"
  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:DescribeInstances"
            - "ec2:DescribeRegions"
          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

```

1.8.11. 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1.21 RHCOS AMI

AWS ゾーン	AWS AMI
ap-northeast-1	ami-0530d04240177f118

AWS ゾーン	AWS AMI
ap-northeast-2	ami-09e4cd700276785d2
ap-south-1	ami-0754b15d212830477
ap-southeast-1	ami-03b46cc4b1518c5a8
ap-southeast-2	ami-0a5b99ab2234a4e6a
ca-central-1	ami-012bc4ee3b6c673bc
eu-central-1	ami-02e08df1201f1c2f8
eu-north-1	ami-0309c9d2fadcb2d5a
eu-west-1	ami-0bdd69d8e7cd18188
eu-west-2	ami-0e610e967a62dbdfa
eu-west-3	ami-0e817e26f638a71ac
me-south-1	ami-024117d7c87b7ff08
sa-east-1	ami-08e62f746b94950c1
us-east-1	ami-077ede5bed2e431ea
us-east-2	ami-0f4ecf819275850dd
us-west-1	ami-0c4990e435bc6c5fe
us-west-2	ami-000d6e92357ac605c

1.8.12. 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

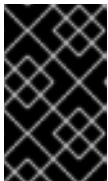
提供される CloudFormation テンプレートを使用してブートストラップノードを作成しない場合、提供される情報を確認し、インフラストラクチャーを手動で作成する必要があります。クラスターが適切に初期化されない場合、インストールログを用意して Red Hat サポートに問い合わせる必要がある可能性があります。

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リールを作成します。

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を提供します。追加のセキュリティを提供するには、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/
```

出力例

```
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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "RhcosAmi", ❸
    "ParameterValue": "ami-<random_string>" ❹
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", ❺
    "ParameterValue": "0.0.0.0/0" ❻
  },
  {
    "ParameterKey": "PublicSubnet", ❼
    "ParameterValue": "subnet-<random_string>" ❽
  },
  {
    "ParameterKey": "MasterSecurityGroup", ❾
    "ParameterValue": "sg-<random_string>" ❿
  },
  {
    "ParameterKey": "VpcId", ⓫
    "ParameterValue": "vpc-<random_string>" ⓬
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", ⓭
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" ⓮
  },
  {
    "ParameterKey": "AutoRegisterELB", ⓯
    "ParameterValue": "yes" ⓰
  },
  {
    "ParameterKey": "RegisterNlbTargetsLambdaArn", ⓱
    "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<random_string>" ⓲
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", ⓳
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" ⓴
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", ⓵
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" ⓶
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", ⓷
    "ParameterValue": "arn:aws:elasticloadbalancing:<region>:"
  }
]
```

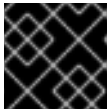
```

    <account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]

```

- 1 クラスターの Ignition 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 **RegisterNlbTargetsLambda** 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その名前です。
- 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。

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

1.8.12.1. 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。

例1.40 ブートストラップマシンの CloudFormation テンプレート

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

RegisterNlbTargetsLambdaArn:

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

- RegisterNlbTargetsLambdaArn

- 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 "MasterSecurityGroup"
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 RegisterNlbTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp

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

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

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

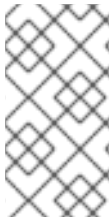
  BootstrapPublicIp:
    Description: The bootstrap node public IP address.
    Value: !GetAtt BootstrapInstance.PublicIp

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

```

1.8.13. 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リールを作成します。
- 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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "RhcocAmi", ❸
    "ParameterValue": "ami-<random_string>" ❹
  },
  {
    "ParameterKey": "AutoRegisterDNS", ❺
    "ParameterValue": "yes" ❻
  },
  {
    "ParameterKey": "PrivateHostedZoneId", ❼
    "ParameterValue": "<random_string>" ❽
  },
  {
    "ParameterKey": "PrivateHostedZoneName", ❾
    "ParameterValue": "mycluster.example.com" ❿
  },
  {
    "ParameterKey": "Master0Subnet", ⓫
    "ParameterValue": "subnet-<random_string>" ⓬
  },
]
```



```

{
  "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": "RegisterNlbTargetsLambdaArn", 29
  "ParameterValue": "arn:aws:lambda:<region>:<account_number>:function:
<dns_stack_name>-RegisterNlbTargets-<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 Ignition 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 Route 53 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 Route 53 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 Route 53 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オド (.) 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グループ ID。
- 18 セキュリティ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 **RegisterNlbTargetsLambda** 値を指定します。

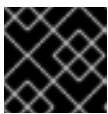
31 外部 API ロードバランサーのターゲットグループの ARN。

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その名前です。
- 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>
```

1.8.13.1. 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。

例1.41 コントロールプレーンマシンの CloudFormation テンプレート

```
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 Z21XYZABCZ2A4.

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

RegisterNlbTargetsLambdaArn:

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
- RegisterNIblpTargetsLambdaArn
- 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 RegisterNlbTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

```

```

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

```

```

RegisterMaster0InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbTargetsLambdaArn
    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"},"network":{"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 RegisterNlbTargetsLambdaArn
  TargetArn: !Ref ExternalApiTargetGroupArn
  TargetIp: !GetAtt Master1.PrivateIp

```

RegisterMaster1InternalApiTarget:

```

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

```

RegisterMaster1InternalServiceTarget:

```

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

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

RegisterMaster2InternalServiceTarget:
Condition: DoRegistration
Type: Custom::NLBRegister
Properties:
  ServiceToken: !Ref RegisterNlbTargetsLambdaArn
  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]
]

```

1.8.14. 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 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リールを作成します。
- 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> \ ❶  
--log-level=info ❷
```

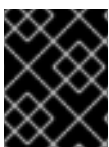
❶ **<installation_directory>** には、インストールファイルを保存したディレクトリーへのパスを指定します。

❷ 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。

1.8.14.1. 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。これらのノードを手動で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リールを作成します。
- 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", ❶
    "ParameterValue": "mycluster-<random_string>" ❷
  },
  {
    "ParameterKey": "RhcosAmi", ❸
    "ParameterValue": "ami-<random_string>" ❹
  },
  {
    "ParameterKey": "Subnet", ❺
    "ParameterValue": "subnet-<random_string>" ❻
  },
  {
    "ParameterKey": "WorkerSecurityGroup", ❼
    "ParameterValue": "sg-<random_string>" ❽
  },
  {
    "ParameterKey": "IgnitionLocation", ❾
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
  },
  {
    "ParameterKey": "CertificateAuthorities", ❿
    "ParameterValue": "" ⓫
  },
]
```

```

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

```

- 1 クラスターの Ignition 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グループ ID。
- 8 セキュリティグループおよびロールの CloudFormation テンプレートの出力から **WorkerSecurityGroup** ID 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 セキュリティグループおよびロールの CloudFormation テンプレートの出力から **WorkerInstanceProfile** 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> ❶
--template-body file://<template>.yaml \ ❷
--parameters file://<parameters>.json ❸
```

- 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その名前です。
- 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続けます。



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。

1.8.14.1.1. 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。

例1.42 ワーカーマシンの CloudFormation テンプレート

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

```

1.8.15. 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。

英語版

前提条件

- OpenShift Container Platform クラスターをデプロイします。
- **oc** CLI をインストールします。

手順

1. **kubeadmin** 認証情報をエクスポートします。

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

- ❶ **<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. エクスポートされた設定を使用して、**oc** コマンドを正常に実行できることを確認します。

```
$ oc whoami
```

出力例

```
system:admin
```

1.8.16. マシンの証明書署名要求の承認

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。最初にクライアント要求を承認し、次にサーバー要求を承認する必要があります。

前提条件

- マシンがクラスターに追加されています。

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
```

出力例

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   63m   v1.18.3
master-1  Ready    master   63m   v1.18.3
master-2  Ready    master   64m   v1.18.3
worker-0  NotReady worker   76s   v1.18.3
worker-1  NotReady worker   70s   v1.18.3
```

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
```

出力例

```
NAME      AGE  REQUESTOR                                     CONDITION
csr-8b2br 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
csr-8vnps 15m  system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper Pending
...
```

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。クライアントの CSR が承認されたら、Kubelet は提供証明書のセカンダリー CSR を作成します。これには、手動の承認が必要です。次に、後続の提供証明書の更新要求は、Kubelet が同じパラメータを持つ新規証明書を要求する場合に **machine-approver** によって自動的に承認されます。

- 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 --no-run-if-empty oc adm certificate approve
```

4. クライアント要求が承認されたら、クラスターに追加した各マシンのサーバー要求を確認する必要があります。

```
$ oc get csr
```

出力例

```
NAME      AGE  REQUESTOR                                     CONDITION
```

```
csr-bfd72 5m26s system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv 5m26s system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 残りの CSR が承認されず、それらが **Pending** ステータスにある場合、クラスターマシンの CSR を承認します。

- 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> ❶
```

- ❶ **<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
```

6. すべてのクライアントおよびサーバーの CSR が承認された後に、マシンのステータスが **Ready** になります。以下のコマンドを実行して、これを確認します。

```
$ oc get nodes
```

出力例

```
NAME      STATUS   ROLES    AGE   VERSION
master-0  Ready    master   73m   v1.20.0
master-1  Ready    master   73m   v1.20.0
master-2  Ready    master   74m   v1.20.0
worker-0  Ready    worker   11m   v1.20.0
worker-1  Ready    worker   11m   v1.20.0
```



注記

サーバー CSR の承認後にマシンが **Ready** ステータスに移行するまでに数分の時間がかかる場合があります。

関連情報

- CSR の詳細は、[Certificate Signing Requests](#) を参照してください。

1.8.17. 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ます。

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
```

出力例

NAME	VERSION	AVAILABLE	PROGRESSING	DEGRADED
SINCE				
authentication	4.5.4	True	False	False 69s
cloud-credential	4.5.4	True	False	False 12m
cluster-autoscaler	4.5.4	True	False	False 11m
console	4.5.4	True	False	False 46s
dns	4.5.4	True	False	False 11m
image-registry	4.5.4	True	False	False 5m26s
ingress	4.5.4	True	False	False 5m36s
kube-apiserver	4.5.4	True	False	False 8m53s
kube-controller-manager	4.5.4	True	False	False 7m24s
kube-scheduler	4.5.4	True	False	False 12m
machine-api	4.5.4	True	False	False 12m
machine-config	4.5.4	True	False	False 7m36s
marketplace	4.5.4	True	False	False 7m54m
monitoring	4.5.4	True	False	False 7h54s
network	4.5.4	True	False	False 5m9s
node-tuning	4.5.4	True	False	False 11m
openshift-apiserver	4.5.4	True	False	False 11m
openshift-controller-manager	4.5.4	True	False	False 5m943s
openshift-samples	4.5.4	True	False	False 3m55s
operator-lifecycle-manager	4.5.4	True	False	False 11m
operator-lifecycle-manager-catalog	4.5.4	True	False	False 11m
service-ca	4.5.4	True	False	False 11m
service-catalog-apiserver	4.5.4	True	False	False 5m26s
service-catalog-controller-manager	4.5.4	True	False	False 5m25s
storage	4.5.4	True	False	False 5m30s

2. 利用不可の Operator を設定します。

1.8.17.1. イメージレジストリーストレージの設定

Amazon Web Services はデフォルトのストレージを提供します。つまり、Image Registry Operator はインストール後に利用可能になります。ただし、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、レジストリーストレージを手動で設定する必要があります。

実稼働クラスターに必要な永続ボリュームの設定についての手順が示されます。該当する場合、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。これは、実稼働以外のクラスターでのみ利用できます。

アップグレード時に **Recreate** ロールアウトストラテジーを使用して、イメージレジストリーがブロックストレージタイプを使用することを許可するための追加の手順が提供されます。

1.8.17.1.1. 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を使用した AWS のレジストリーストレージの設定

インストール時に、Amazon 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。
- Amazon S3 ストレージの場合、シークレットには以下のキーが含まれることが予想されます。
 - **REGISTRY_STORAGE_S3_ACCESSKEY**
 - **REGISTRY_STORAGE_S3_SECRETKEY**

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
```

設定例

```
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。

1.8.17.1.2. 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

1. 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. イメージのビルドおよびプッシュを有効にするためにレジストリーが `managed` に設定されていることを確認します。

- 以下を実行します。

```
$ oc edit configs.imageregistry/cluster
```

次に、行を変更します。

```
managementState: Removed
```

上記を以下のように変更します。

```
managementState: Managed
```

1.8.18. ブートストラップリソースの削除

クラスターの初期 Operator 設定の完了後に、Amazon Web Services (AWS) からブートストラップリソースを削除します。

前提条件

- クラスターの初期 Operator 設定が完了済みです。

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> ❶
```

❶ **<name>** は、ブートストラップスタックの名前です。

1.8.19. Ingress DNS レコードの作成

DNS ゾーン設定を削除した場合には、Ingress ロードバランサーを参照する DNS レコードを手動で作成します。ワイルドカードレコードまたは特定のレコードのいずれかを作成できます。以下の手順では A レコードを使用しますが、CNAME やエイリアスなどの必要な他のレコードタイプを使用できます。

前提条件

- 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する OpenShift Container Platform クラスターを Amazon Web Services (AWS) にデプロイしています。
- OpenShift CLI (**oc**) をインストールします。
- **jq** パッケージをインストールします。
- 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。 [Install the AWS CLI Using the Bundled Installer \(Linux, macOS, or Unix\)](#) を参照してください。

手順

1. 作成するルートを決めます。

- ワイルドカードレコードを作成するには、***.apps.<cluster_name>.<domain_name>** を使用します。ここで、**<cluster_name>** はクラスター名で、**<domain_name>** は OpenShift Container Platform クラスターの Route 53 ベースドメインです。
- 特定のレコードを作成するには、以下のコマンドの出力にあるように、クラスターが使用する各ルートにレコードを作成する必要があります。

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}\n'}{end}{end}' routes
```

出力例

```
oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
grafana-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

2. Ingress Operator ロードバランサーのステータスを取得し、使用する外部 IP アドレスの値をメモします。これは **EXTERNAL-IP** 列に表示されます。

```
$ oc -n openshift-ingress get service router-default
```

出力例

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)
AGE				
router-default	LoadBalancer	172.30.62.215	ab3...28.us-east-2.elb.amazonaws.com	80:31499/TCP,443:30693/TCP
		5m		

3. ロードバランサーのホストゾーン ID を見つけます。

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName == "<external_ip>").CanonicalHostedZoneNameId' ❶
```

- ❶ **<external_ip>** については、取得した Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。

出力例

```
Z3AADJGX6KTTL2
```

このコマンドの出力は、ロードバランサーのホストゾーン ID です。

4. クラスターのドメインのパブリックホストゾーン ID を取得します。

```
$ aws route53 list-hosted-zones-by-name \
  --dns-name "<domain_name>" ❶
  --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' ❷
  --output text
```

- ❶ ❷ **<domain_name>** については、OpenShift Container Platform クラスターの Route 53 ベースドメインを指定します。

出力例

```
/hostedzone/Z3URY6TWQ91KVV
```

ドメインのパブリックホストゾーン ID がコマンド出力に表示されます。この例では、これは **Z3URY6TWQ91KVV** になります。

5. プライベートゾーンにエイリアスレコードを追加します。

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ ❶
> "Changes": [
> {
>   "Action": "CREATE",
>   "ResourceRecordSet": {
>     "Name": "\\052.apps.<cluster_domain>", ❷
>     "Type": "A",
>     "AliasTarget":{
>       "HostedZoneId": "<hosted_zone_id>", ❸
>       "DNSName": "<external_ip>.", ❹
>       "EvaluateTargetHealth": false
>     }
>   }
> }
> ]
> }'
```

- ❶ **<private_hosted_zone_id>** については、DNS および負荷分散の CloudFormation テンプレートから値を指定します。
- ❷ **<cluster_domain>** については、OpenShift Container Platform クラスターで使用するドメインまたはサブドメインを指定します。
- ❸ **<hosted_zone_id>** については、取得したロードバランサーのパブリックホストゾーン ID を指定します。
- ❹ **<external_ip>** については、Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。このパラメーターの値に末尾のピリオド (.) が含まれていることを確認します。

6. パブリックゾーンにレコードを追加します。

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>" --
change-batch '{
❶  > "Changes": [
>   {
>     "Action": "CREATE",
>     "ResourceRecordSet": {
>       "Name": "\\052.apps.<cluster_domain>",
❷     >       "Type": "A",
>       "AliasTarget": {
>         "HostedZoneId": "<hosted_zone_id>",
❸     >         "DNSName": "<external_ip>.",
❹     >         "EvaluateTargetHealth": false
>       }
>     }
>   }
> ]
> }'
```

- ❶ **<public_hosted_zone_id>** については、ドメインのパブリックホストゾーンを指定します。
- ❷ **<cluster_domain>** については、OpenShift Container Platform クラスターで使用するドメインまたはサブドメインを指定します。
- ❸ **<hosted_zone_id>** については、取得したロードバランサーのパブリックホストゾーン ID を指定します。
- ❹ **<external_ip>** については、Ingress Operator ロードバランサーの外部 IP アドレスの値を指定します。このパラメーターの値に末尾のピリオド (.) が含まれていることを確認します。

1.8.20. 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に、デプロイメントを完了するまでモニターします。

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で削除しています。
- **oc** CLI をインストールし、ログインします。

手順

1. 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
```

- 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へのパスを指定します。

出力例

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



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り、その後に更新される証明書が含まれます。証明書を更新する前にクラスターが停止し、24 時間経過した後にクラスターを再起動すると、クラスターは期限切れの証明書を自動的に復元します。例外として、kubelet 証明書を回復するために保留状態の **node-bootstrapper** 証明書署名要求 (CSR) を手動で承認する必要があります。詳細は、コントロールプレーン証明書の期限切れの状態からのリカバリーについてのドキュメントを参照してください。

2. [Cluster registration](#) ページでクラスターを登録します。

1.8.21. 次のステップ

- [クラスターをカスタマイズ](#) します。
- Cluster Samples Operator および **must-gather** ツールの [イメージストリームを設定](#) します。
- [ネットワークが制限された環境での Operator Lifecycle Manager \(OLM\) の使用](#) 方法について参照します。
- クラスターのインストールに使用したミラーレジストリーに信頼される CA がある場合、[信頼ストアを設定](#) してこれをクラスターに追加します。
- 必要な場合は、[リモートの健全性レポートをオプトアウト](#) することができます。
- 必要に応じて、[クラウドプロバイダーの認証情報を削除](#) できます。

1.9. 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。

1.9.1. 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するクラスターの削除

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するクラスターは、クラウドから削除できます。



注記

アンインストール後に、とくに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 (UPI) クラスターで適切に削除されていないリソースがあるかどうかについて、クラウドプロバイダーを確認します。インストーラーが作成されなかったり、インストーラーがアクセスできない場合には、リソースがある可能性があります。

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ります。
- 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ります。

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。