Red Hat Enterprise Linux 9

Google Cloud Platform への RHEL 9 のデプロイ

RHEL システムイメージの取得と GCP 上での RHEL インスタンスの作成

Last Updated: 2024-07-27

Red Hat Enterprise Linux 9 Google Cloud Platform への RHEL 9 のデプロイ

RHEL システムイメージの取得と GCP 上での RHEL インスタンスの作成

法律上の通知

概要

Red Hat Enterprise Linux (RHEL) をパブリッククラウド環境で使用するには、RHEL システムイメージを作成し、Google Cloud Platform (GCP) などのさまざまなクラウドプラットフォームにデプロイできます。GCP 上で Red Hat High Availability (HA) クラスターを作成および設定することもできます。 次の章では、GCP 上でクラウド RHEL インスタンスと HA クラスターを作成する手順について説明します。これらのプロセスには、必要なパッケージとエージェントのインストール、フェンシングの設定、ネットワークリソースエージェントのインストールが含まれます。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

1. Jira の Web サイトにログインします。
2. 上部のナビゲーションバーで **Create** をクリックします。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. ダイアログの下部にある **Create** をクリックします。

第1章 パブリッククラウドプラットフォームでの RHEL の導入

パブリッククラウドプラットフォームは、コンピューティングリソースをサービスとして提供します。オンプレミスのハードウェアを使用する代わりに、Red Hat Enterprise Linux (RHEL) システムなどの IT ワークロードをパブリッククラウドインスタンスとして実行できます。

1.1. パブリッククラウドで RHEL を使用する利点

パブリッククラウドプラットフォーム上に配置されたクラウドインスタンスとしての RHEL には、RHEL オンプレミスの物理システムまたは仮想マシン (VM) に比べて次の利点があります。

- **リソースの柔軟性と詳細な割り当て**
  RHEL のクラウドインスタンスは、クラウドプラットフォーム上の仮想マシンとして実行されます。この仮想マシンは通常、クラウドサービスのプロバイダーによって維持管理されるリモートサーバーのクラスターです。したがって、特定のタイプの CPU やストレージなどのハードウェアリソースのインスタンスへの割り当ては、ソフトウェアレベルで行われ、簡単にカスタマイズできます。

  また、ローカルの RHEL システムと比較すると、物理ホストの機能によって制限されることがありません。むしろ、クラウドプロバイダーが提供する選択肢に基づいて、さまざまな機能から選択できます。

- **領域とコスト効率**
  クラウドワークロードをホストするためにオンプレミスサーバーを所有する必要がありません。これにより、物理ハードウェアに関連するスペース、電力、メンテナンスの要件が回避されます。

  代わりに、パブリッククラウドプラットフォームでは、クラウドインスタンスの使用料をクラウドプロバイダーに直接支払います。通常、コストはインスタンスに割り当てられたハードウェアとその使用時間に基づきます。したがって、要件に基づいてコストを最適化できます。

- **ソフトウェアで制御される設定**
  クラウドインスタンスの設定全体がクラウドプラットフォーム上にデータとして保存され、ソフトウェアによって制御されます。したがって、インスタンスの作成、削除、クローン作成、または移行を簡単に行うことができます。また、クラウドインスタンスは、クラウドプロバイダーのコンソールでリモートで操作され、デフォルトでリモートストレージに接続されます。

  さらに、クラウドインスタンスの現在の状態をいつでもスナップショットとしてバックアップできます。その後、スナップショットをロードしてインスタンスを保存した状態に復元できます。

- **ホストからの分離とソフトウェアの互換性**
  ローカルの仮想マシンと同様に、クラウドインスタンス上の RHEL ゲストオペレーティングシステムは仮想化されたカーネル上で実行されます。このカーネルは、ホストオペレーティングシステムや、インスタンスへの接続に使用する **クライアント** システムとは別のものです。

  したがって、任意のオペレーティングシステムをクラウドインスタンスにインストールできます。つまり、RHEL パブリッククラウドインスタンスでは、ローカルオペレーティングシステムでは使用できない RHEL 固有のアプリケーションを実行できます。

  さらに、インスタンスのオペレーティングシステムが不安定になったり侵害されたりした場合でも、クライアントシステムには一切影響がありません。

関連情報

- パブリッククラウドとは
- ハイパースケーラーとは
- クラウドコンピューティングの種類
- RHEL のパブリッククラウドのユースケース
- パブリッククラウドデプロイメント用の RHEL の入手

1.2. RHEL のパブリッククラウドのユースケース

パブリッククラウドへのデプロイには多くの利点がありますが、すべてのシナリオにおいて最も効率的なソリューションであるとは限りません。RHEL デプロイメントをパブリッククラウドに移行するかどうかを評価している場合は、ユースケースがパブリッククラウドの利点を享受できるかどうかを検討してください。

有益なユースケース

- パブリッククラウドインスタンスのデプロイは、デプロイメントのアクティブなコンピューティング能力を柔軟に増減する (**スケールアップ** および **スケールダウン** とも呼ばれます) 場合に非常に効果的です。したがって、次のシナリオではパブリッククラウドで RHEL を使用することを推奨します。
  - ピーク時のワークロードが高く、一般的なパフォーマンス要件が低いクラスター。要求に応じてスケールアップおよびスケールダウンすることで、リソースコストの面で高い効率が得られる場合があります。
  - クラスターを迅速にセットアップまたは拡張できます。これにより、ローカルサーバーのセットアップにかかる高額な初期費用が回避されます。
- クラウドインスタンスは、ローカル環境で何が起こっても影響を受けません。したがって、バックアップや障害復旧に使用できます。

問題が発生する可能性のあるユースケース

- 調整不可能な既存の環境を運用している場合。既存のデプロイメントの特定のニーズに合わせてクラウドインスタンスをカスタマイズすることは、現在のホストプラットフォームと比較して費用対効果が低い可能性があります。
- 厳しい予算制限の中で運用している場合。通常、ローカルデータセンターでデプロイメントを維持管理すると、パブリッククラウドよりも柔軟性は低くなりますが、最大リソースコストをより細かく制御できます。

次のステップ

- パブリッククラウドデプロイメント用の RHEL の入手

関連情報

- アプリケーションをクラウドに移行すべきか?また、その決定方法

1.3. パブリッククラウドへの移行時によくある懸念事項

RHEL ワークロードをローカル環境からパブリッククラウドプラットフォームに移行すると、それに伴う変更について懸念が生じる可能性があります。よくある質問は次のとおりです。

RHEL は、クラウドインスタンスとして動作する場合、ローカル仮想マシンとして動作する場合とは異なる動作になりますか?

パブリッククラウドプラットフォーム上の RHEL インスタンスは、ほとんどの点で、オンプレミスサーバーなどのローカルホスト上の RHEL 仮想マシンと同じように機能します。注目すべき例外には次のようなものがあります。

- パブリッククラウドインスタンスは、プライベートオーケストレーションインターフェイスの代わりに、プロバイダー固有のコンソールインターフェイスを使用してクラウドリソースを管理します。
- ネストされた仮想化などの特定の機能が正しく動作しない可能性があります。特定の機能がデプロイメントにとって重要な場合は、選択したパブリッククラウドプロバイダーとその機能の互換性を事前に確認してください。

ローカルサーバーと比べて、パブリッククラウドではデータは安全に保たれますか?

RHEL クラウドインスタンス内のデータの所有権はユーザーにあり、パブリッククラウドプロバイダーはデータにアクセスできません。さらに、主要なクラウドプロバイダーは転送中のデータ暗号化をサポートしているため、仮想マシンをパブリッククラウドに移行する際のデータのセキュリティーが向上します。

RHEL パブリッククラウドインスタンスの一般的なセキュリティーは次のように管理されます。

- パブリッククラウドプロバイダーは、クラウドハイパーバイザーのセキュリティーを担当します。
- Red Hat は、RHEL ゲストオペレーティングシステムのセキュリティー機能をインスタンスに提供します。
- ユーザーは、クラウドインフラストラクチャーにおける特定のセキュリティー設定とプラクティスを管理します。

ユーザーの地理的リージョンは、RHEL パブリッククラウドインスタンスの機能にどのように影響しますか?

RHEL インスタンスは、地理的な場所に関係なく、パブリッククラウドプラットフォームで使用できます。したがって、オンプレミスサーバーと同じリージョンでインスタンスを実行できます。

ただし、物理的に離れたリージョンでインスタンスをホストすると、操作時に待ち時間が長くなる可能性があります。さらに、パブリッククラウドプロバイダーによっては、特定のリージョンで、追加機能が提供される場合や、より高いコスト効率が得られる場合があります。RHEL インスタンスを作成する前に、選択したクラウドプロバイダーで利用可能なホスティングリージョンのプロパティーを確認してください。

1.4. パブリッククラウドデプロイメント用の RHEL の入手

RHEL システムをパブリッククラウド環境にデプロイするには、次の手順を実行する必要があります。

1. 要件と市場の現在のオファーに基づいて、ユースケースに最適なクラウドプロバイダーを選択します。
   現在、RHEL インスタンスの実行が認定されているクラウドプロバイダーは次のとおりです。
   - Amazon Web Services (AWS)
   - Google Cloud Platform (GCP)

- Microsoft Azure

注記

このドキュメントでは、GCP への RHEL のデプロイについて特に説明します。

2. 選択したクラウドプラットフォーム上に RHEL クラウドインスタンスを作成します。詳細は、RHEL クラウドインスタンスを作成する方法 を参照してください。
3. RHEL デプロイメントを最新の状態に保つには、Red Hat Update Infrastructure (RHUI) を使用します。

関連情報

- RHUI ドキュメント
- Red Hat Open Hybrid Cloud

1.5. RHEL クラウドインスタンスを作成する方法

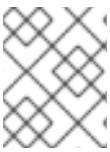
RHEL インスタンスをパブリッククラウドプラットフォームにデプロイするには、次のいずれかの方法を使用できます。

RHEL のシステムイメージを作成し、クラウドプラットフォームにインポートします。

- システムイメージを作成するには、RHEL Image Builder を使用するか、イメージを手動で構築します。
- これは既存の RHEL サブスクリプションを使用する方法で、**bring your own subscription** (BYOS) とも呼ばれます。
- 年間サブスクリプションを前払いすると、Red Hat お客様割引を利用できます。
- カスタマーサービスは Red Hat によって提供されます。
- 複数のイメージを効率的に作成するには、**cloud-init** ツールを使用できます。

RHEL インスタンスをクラウドプロバイダーマーケットプレイスから直接購入します。

- サービスの利用に対して時間料金を後払いで支払います。したがって、この方法は **従量課金制** (PAYG) とも呼ばれます。
- カスタマーサービスはクラウドプラットフォームプロバイダーによって提供されます。



注記

さまざまな方法を使用して Google Cloud Platform に RHEL インスタンスをデプロイする詳細な手順については、このドキュメントの次の章を参照してください。

関連情報

- ゴールデンイメージとは

- RHEL 9 の cloud-init の設定および管理

第2章 RHEL IMAGE BUILDER を使用した GCP へのイメージのアップロード

RHEL Image Builder を使用すると、**gce** イメージをビルドし、ユーザーまたは GCP サービスアカウントの認証情報を指定して、**gce** イメージを GCP 環境に直接アップロードできます。

2.1. CLI を使用した GCE イメージの GCP へのアップロード

gce イメージを GCP にアップロードするための認証情報を含む設定ファイルを設定します。

警告

イメージが起動しなくなるため、**gce** イメージを GCP に手動でインポートすることはできません。アップロードするには、**gcloud** または RHEL Image Builder を使用する必要があります。

前提条件

- イメージを GCP にアップロードするための有効な Google アカウントと認証情報がある。認証情報は、ユーザーアカウントまたはサービスアカウントから取得できます。認証情報に関連付けられたアカウントには、少なくとも次の IAM ロールが割り当てられている必要があります。
  - **roles/storage.admin** - ストレージオブジェクトの作成と削除
  - **roles/compute.storageAdmin** - 仮想マシンイメージの Compute Engine へのインポート
- 既存の GCP バケットがあります。

手順

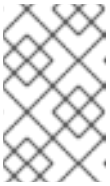
1. テキストエディターを使用して、次の内容の **gcp-config.toml** 設定ファイルを作成します。

```
provider = "gcp"

[settings]
bucket = "GCP_BUCKET"
region = "GCP_STORAGE_REGION"
object = "OBJECT_KEY"
credentials = "GCP_CREDENTIALS"
```

   - **GCP_BUCKET** は既存のバケットを指します。アップロード中のイメージの中間ストレージオブジェクトを格納するために使用されます。
   - **GCP_STORAGE_REGION** は、通常の Google ストレージリージョンであると同時に、デュアルリージョンまたはマルチリージョンでもあります。
   - **OBJECT_KEY** は、中間ストレージオブジェクトの名前です。アップロード前に存在してはならず、アップロードプロセスが完了すると削除されます。オブジェクト名が **.tar.gz** で終わらない場合、拡張子がオブジェクト名に自動的に追加されます。

- **GCP_CREDENTIALS** は、GCP からダウンロードした認証情報 JSON ファイルの **Base64** エンコードされたスキームです。認証情報によって、GCP がイメージをアップロードするプロジェクトが決まります。



注記

GCP での認証に別のメカニズムを使用する場合、**gcp-config.toml** ファイルでの **GCP_CREDENTIALS** の指定は任意です。他の認証方法については、Authenticating with GCP を参照してください。

2. GCP からダウンロードした JSON ファイルから **GCP_CREDENTIALS** を取得します。

```
$ sudo base64 -w 0 cee-gcp-nasa-476a1fa485b7.json
```

3. 追加のイメージ名とクラウドプロバイダープロファイルを使用して Compose を作成します。

```
$ sudo composer-cli compose start BLUEPRINT-NAME gce IMAGE_KEY gcp-config.toml
```

イメージビルド、アップロード、およびクラウド登録プロセスは、完了に最大 10 分かかる場合があります。

検証

- イメージのステータスが FINISHED であることを確認します。

```
$ sudo composer-cli compose status
```

関連情報

- Identity and Access Management
- Create storage buckets

2.2. RHEL IMAGE BUILDER によるさまざまな GCP 認証情報の認証順序の並べ替え

RHEL Image Builder でいくつかの異なる種類の認証情報を使用して、GCP で認証できます。複数の認証情報セットを使用して GCP で認証するように RHEL Image Builder が設定されている場合、次の優先順位で認証情報が使用されます。

1. 設定ファイルで **composer-cli** コマンドで指定された認証情報。
2. **osbuild-composer** ワーカー設定で設定された認証情報。
3. 次の方法で認証方法を自動的に見つけようとする、**Google GCP SDK** ライブラリーからの **Application Default Credentials**。
    a. **GOOGLE_APPLICATION_CREDENTIALS** 環境変数が設定されている場合、Application Default Credentials は、変数が指すファイルから認証情報を読み込んで使用しようとします。

b. Application Default Credentials は、コードを実行しているリソースに関連付けられたサービスアカウントを使用して認証を試みます。たとえば、Google Compute Engine 仮想マシンです。

注記

イメージをアップロードする GCP プロジェクトを決定するには、GCP 認証情報を使用する必要があります。したがって、すべてのイメージを同じ GCP プロジェクトにアップロードする場合を除き、**composer-cli** コマンドを使用して **gcp-config.toml** 設定ファイルに認証情報を指定する必要があります。

2.2.1. composer-cli コマンドで GCP 認証情報を指定する

アップロードターゲット設定の **gcp-config.toml** ファイルで、GCP 認証情報を指定できます。時間を節約するために、Google アカウント認証情報の JSON ファイルの **Base64** エンコードスキームを使用します。

手順

1. **GOOGLE_APPLICATION_CREDENTIALS** 環境変数に保存されているパスを使用して、Google アカウント認証情報ファイルのエンコードされたコンテンツを取得するには、次のコマンドを実行します。

```
$ base64 -w 0 "${GOOGLE_APPLICATION_CREDENTIALS}"
```

2. アップロードターゲット設定の **gcp-config.toml** ファイルで、認証情報を設定します。

```
provider = "gcp"

[settings]
provider = "gcp"

[settings]
...
credentials = "GCP_CREDENTIALS"
```

2.2.2. osbuild-composer ワーカー設定で認証情報を指定する

すべてのイメージビルドでグローバルに GCP に使用される GCP 認証認証情報を設定できます。このようにして、イメージを同じ GCP プロジェクトにインポートする場合、GCP へのすべてのイメージのアップロードに同じ認証情報を使用できます。

手順

- **/etc/osbuild-worker/osbuild-worker.toml** ワーカー設定で、次の認証情報の値を設定します。

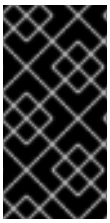
```
[gcp]
credentials = "PATH_TO_GCP_ACCOUNT_CREDENTIALS"
```

第3章 GOOGLE CLOUD PLATFORM での GOOGLE COMPUTE ENGINE インスタンスとしての RED HAT ENTERPRISE LINUX イメージのデプロイメント

Google Cloud Platform (GCP) 上に Red Hat Enterprise Linux 9 (RHEL 9) のデプロイメントを設定するには、RHEL 9 を GCP 上の Google Compute Engine (GCE) インスタンスとしてデプロイします。

注記

GCP 向けの Red Hat 製品認定のリストは、Red Hat on Google Cloud Platform を参照してください。



重要

ISO イメージからカスタム仮想マシンを作成することは可能ですが、**Red Hat Image Builder** 製品を使用して、特定のクラウドプロバイダーで使用するようにカスタマイズされたイメージを作成することを推奨します。詳細は Composing a Customized RHEL System Image を参照してください。

前提条件

- Red Hat カスタマーポータル のアカウントがある (本章の手順を完了するのに必要)。
- Google Cloud Platform コンソールにアクセスするために、GCP でアカウントを作成している。詳細は Google Cloud を参照してください。

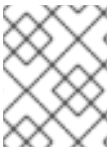
3.1. GCP での RED HAT ENTERPRISE LINUX イメージオプション

Google Cloud Platform に RHEL 9 をデプロイする場合、複数のタイプのイメージを使用できます。要件に基づいて、ユースケースに最適なオプションを検討してください。

表3.1 イメージオプション

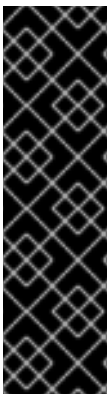
イメージオプション	サブスクリプション	サンプルシナリオ	留意事項
Red Hat Gold Image をデプロイする	既存の Red Hat サブスクリプションを使用する	Google Cloud Platform で Red Hat Gold Image を選択します。Gold イメージの詳細、および Google Cloud Platform で Gold Image にアクセスする方法は、Red Hat Cloud Access Reference Guide を参照してください。	その他のすべてのインスタンスコストを Google 社に支払うことになりますが、サブスクリプション自体には Red Hat 製品コストが含まれます。Red Hat は、カスタム RHEL イメージのサポートを直接提供します。

イメージオプション	サブスクリプション	サンプルシナリオ	留意事項
GCP に移動するカスタムイメージをデプロイする	既存の Red Hat サブスクリプションを使用する	カスタムイメージをアップロードし、サブスクリプションを割り当てます。	その他のすべてのインスタンスコストを支払うことになりますが、サブスクリプション自体には Red Hat 製品コストが含まれます。Red Hat は、カスタム RHEL イメージのサポートを直接提供します。
RHEL を含む既存の GCP イメージをデプロイする	GCP イメージには、Red Hat 製品が含まれる	GCP Compute Engine でインスタンスを起動する時に RHEL イメージを選択するか、Google Cloud Platform Marketplace からイメージを選択します。	従量課金モデルでは、GCP に 1 時間ごとに支払います。このようなイメージは "オンデマンド" イメージと呼ばれています。GCP は、サポート契約に基づいてオンデマンドイメージのサポートを提供します。



注記

Red Hat Image Builder を使用して、GCP 用のカスタムイメージを作成できます。詳細は Composing a Customized RHEL System Image を参照してください。



重要

オンデマンドインスタンスをカスタム RHEL インスタンスに変換することはできません。オンデマンドイメージからカスタム RHEL **bring-your-own-subscription** (BYOS) イメージに変更するには、次の手順を実行します。

1. 新しいカスタム RHEL インスタンスを作成し、オンデマンドインスタンスからデータを移行します。
2. データを移行した後に、オンデマンドのインスタンスをキャンセルして二重請求を回避します。

関連情報

- Red Hat in the Public Cloud
- コンピュートエンジンのイメージ
- Creating an instance from a custom image

3.2. ベースイメージの理解

ISO イメージからベース仮想マシンを作成するには、事前設定されたベースイメージとその設定を使用できます。

3.2.1. カスタムベースイメージの使用

仮想マシン (VM) を手動で設定するには、まずベース (スターター) となる仮想マシンイメージを作成します。続いて、設定を変更して、仮想マシンがクラウドで動作するために必要なパッケージを追加できます。イメージのアップロード後に、特定のアプリケーションに追加の設定変更を行うことができます。

関連情報

- Red Hat Enterprise Linux

3.2.2. 仮想マシン設定

クラウド仮想マシンには、以下の設定が必要です。

表3.2 仮想マシンの設定

設定	推奨事項
ssh	仮想マシンへのリモートアクセスを確立するには、SSH を有効にする必要があります。
dhcp	プライマリー仮想アダプターは、dhcp 用に設定する必要があります。

3.3. ISO イメージからのベース仮想マシンの作成

ISO イメージから RHEL 9 ベースイメージを作成するには、ホストマシンの仮想化を有効にし、RHEL 仮想マシン (VM) を作成します。

前提条件

- ホストマシンで 仮想化が有効 になっている。
- Red Hat カスタマーポータル から最新の Red Hat Enterprise Linux ISO イメージをダウンロードし、イメージを **/var/lib/libvirt/images** に移動しました。

3.3.1. RHEL ISO イメージからの仮想マシンの作成

手順

1. 仮想化用のホストマシンを有効にしていることを確認します。設定および手順は、RHEL 9 で仮想化を有効にする を参照してください。
2. 基本的な Red Hat Enterprise Linux 仮想マシンを作成し、起動します。手順は、仮想マシンの作成 を参照してください。
   a. コマンドラインを使用して仮想マシンを作成する場合は、デフォルトのメモリーと CPU を仮想マシンの容量に設定するようにしてください。仮想ネットワークインターフェイスを **virtio** に設定します。
   たとえば、次のコマンドは **/home/username/Downloads/rhel9.iso** イメージを使用して仮想マシン **kvmtest** を作成します。

```
# virt-install \
    --name kvmtest --memory 2048 --vcpus 2 \
    --cdrom /home/username/Downloads/rhel9.iso,bus=virtio \
    --os-variant=rhel9.0
```

b. Web コンソールを使用して仮想マシンを作成する場合は、Web コンソールで仮想マシンの作成 の手順を行います。以下の点に注意してください。

- **仮想マシンをすぐに起動** は選択しないでください。
- **メモリー** を希望のサイズに変更します。
- インストールを開始する前に、**仮想ネットワークインターフェイス設定** で **モデル** を **virtio** に変更し、**vCPUs** を仮想マシンの容量設定に変更していることを確認します。

3.3.2. RHEL インストールの完了

Amazon Web Services (AWS) にデプロイする RHEL システムのインストールを完了するには、**インストールの概要** ビューをカスタマイズし、インストールを開始し、仮想マシンの起動後にルートアクセスを有効にします。

手順

1. インストールプロセス中に使用する言語を選択します。
2. **インストール概要** ビューで、以下を行います。
   a. **ソフトウェアの選択** をクリックし、**最小インストール** を選択します。
   b. **完了** をクリックします。
   c. **インストール先** をクリックし、**ストレージ設定** で **カスタム** を選択します。
      - **/boot** で、500 MB 以上であることを確認してください。残りの領域は、root / に使用できます。
      - 標準のパーティションが推奨されますが、論理ボリューム管理 (LVM) を使用することも可能です。
      - ファイルシステムには、xfs、ext4、ext3 などを使用できます。
      - 変更が完了したら、**完了** をクリックします。
3. **インストールの開始** をクリックします。
4. **Root パスワード** を設定します。必要に応じて、他のユーザーを作成します。
5. インストールが完了したら、仮想マシンを再起動して **root** でログインします。
6. イメージを設定します。
   a. 仮想マシンを登録し、Red Hat Enterprise Linux 9 リポジトリーを有効にします。

```
# subscription-manager register --auto-attach
```

   b. **cloud-init** パッケージがインストールされ、有効になっていることを確認します。

```
# dnf install cloud-init
# systemctl enable --now cloud-init.service
```

7. 仮想マシンの電源をオフにします。

関連情報

- cloud-init の概要

3.4. RHEL イメージの GCP へのアップロード

Google Cloud Platform (GCP) で RHEL 9 インスタンスを実行するには、RHEL 9 イメージを GCP にアップロードする必要があります。

3.4.1. GCP での新規プロジェクトの作成

Red Hat Enterprise Linux 9 イメージを Google Cloud Platform (GCP) にアップロードするには、まず GCP で新しいプロジェクトを作成する必要があります。

前提条件

- GCP のアカウントを持っている必要があります。そうでない場合は、Google Cloud を参照してください。

手順

1. GCP コンソール を起動します。
2. **Google Cloud Platform** の右側にあるドロップダウンメニューをクリックします。
3. ポップアップメニューから **新しいプロジェクト** をクリックします。
4. **新しいプロジェクト** ウィンドウに、新規プロジェクトの名前を入力します。
5. **Organization**のチェックボックスを選択します。ドロップダウンメニューをクリックして、必要に応じて組織を変更します。
6. 親組織またはフォルダーの **場所** を確認します。**参照** をクリックして検索し、必要に応じてこの値を変更します。
7. **作成** をクリックして、新しい GCP プロジェクトを作成します。

注記

Google Cloud SDK をインストールしたら、CLI コマンドの **gcloud projects create** を使用してプロジェクトを作成できます。以下に例を示します。

```
# gcloud projects create my-gcp-project3 --name project3
```

この例では、プロジェクト ID **my-gcp-project3** とプロジェクト名 **project3** のプロジェクトを作成します。詳細は、gcloud project create を参照してください。

関連情報

- Creating and Managing Resources in Google Cloud

3.4.2. Google Cloud SDK のインストール

Google Cloud Platform (GCP) で HA クラスターを管理する際に、多くの手順で Google Cloud SDK のツールが必要になります。

手順

1. GCP の説明に従って、Google Cloud SDK アーカイブをダウンロードし、抽出します。詳細は、GCP ドキュメント Quickstart for Linux を参照してください。
2. Google Cloud SDK の初期化と同じ手順に従います。

注記

Google Cloud SDK を初期化すると、**gcloud** CLI コマンドを使用してタスクを実行でき、プロジェクトとインスタンスに関する情報を取得できます。たとえば、**gcloud compute project-info describe --project <project-name>** コマンドを使用してプロジェクト情報を表示できます。

関連情報

- Linux 用のクイックスタート
- gcloud command reference
- gcloud コマンドライン ツールの概要

3.4.3. Google Compute Engine の SSH 鍵の作成

GCE で SSH 鍵を生成して登録し、パブリック IP アドレスを使用してインスタンスに直接 SSH 接続できるようにします。

手順

1. **ssh-keygen** コマンドを使用して、GCE で使用する SSH 鍵ペアを生成します。

```
# ssh-keygen -t rsa -f ~/.ssh/google_compute_engine
```

2. GCP Console Dashboard ページ から、Google の **Cloud Console バナー** の左側にある **ナビゲーション** メニューをクリックし、**Compute Engine** を選択して **Metadata** を選択します。
3. **SSH 鍵** をクリックして、**編集** をクリックします。
4. **~/.ssh/google_compute_engine.pub** ファイルから生成された出力を入力し、**保存** をクリックします。
これで、標準の SSH を使用してインスタンスに接続できます。

```
# ssh -i ~/.ssh/google_compute_engine <username>@<instance_external_ip>
```

注記

gcloud compute config-ssh コマンドを実行すると、インスタンスのエイリアスを設定ファイルに追加できます。エイリアスは、インスタンス名による単純な SSH 接続を許可します。**gcloud compute config-ssh** コマンドの詳細は、gcloud compute config-ssh を参照してください。

関連情報

- gcloud compute config-ssh
- インスタンスへの接続

3.4.4. GCP ストレージでのストレージバケットの作成

RHEL 9 イメージを GCP にインポートするには、まず GCP ストレージバケットを作成する必要があります。

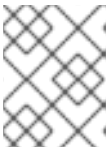
手順

1. GCP にログインしていない場合は、次のコマンドを実行してログインします。

```
# gcloud auth login
```

2. ストレージバケットを作成します。

```
# gsutil mb gs://bucket_name
```



注記

Google Cloud コンソールを使用してバケットを作成することもできます。詳細は、バケットの作成 を参照してください。

関連情報

- バケットの作成

3.4.5. GCP バケットへのイメージの変換およびアップロード

ローカルの RHEL 9 イメージを GCP にデプロイするには、まずイメージを変換して GCP バケットにアップロードする必要があります。次の手順では、**qcow2** イメージを **raw** 形式に変換し、そのイメージを **tar** アーカイブとしてアップロードする方法について説明します。ただし、異なる形式を使用することも可能です。

手順

1. **qemu-img** コマンドを実行してイメージを変換します。変換されたイメージは、**disk.raw** という名前になるはずです。

```
# qemu-img convert -f qcow2 -O raw rhel-9.0-sample.qcow2 disk.raw
```

2. イメージに tar コマンドを実行します。

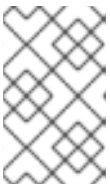
```
# tar --format=oldgnu -Sczf disk.raw.tar.gz disk.raw
```

3. そのイメージを、以前作成したバケットにアップロードします。アップロードには数分かかる場合があります。

```
# gsutil cp disk.raw.tar.gz gs://bucket_name
```

4. **Google Cloud Platform** のホーム画面で、折りたたまれたメニューアイコンをクリックし、**ストレージ** を選択してから **ブラウザー** を選択します。

5. バケットの名前をクリックします。
バケット名の下に、tar コマンドを実行したイメージが表示されます。



注記

GCP コンソール を使用してイメージをアップロードすることもできます。これを行うには、バケットの名前をクリックしてから **ファイルのアップロード** をクリックします。

関連情報

- 仮想ディスクの手動インポート
- インポート方法の選択

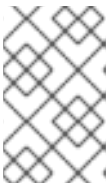
3.4.6. GCP バケットのオブジェクトからイメージの作成

GCP バケットにアップロードしたオブジェクトから GCE イメージを作成するには、オブジェクトを GCE イメージに変換する必要があります。

手順

1. 以下のコマンドを実行して GCE のイメージを作成します。作成するイメージの名前、バケット名、および tar コマンドを実行したイメージの名前を指定します。

```
# gcloud compute images create my-image-name --source-uri gs://my-bucket-
name/disk.raw.tar.gz
```



注記

または、Google Cloud コンソールを使用して、イメージを作成することもできます。詳細は、カスタムイメージの作成、削除、および中止 を参照してください。

2. 必要に応じて、GCP コンソールでイメージを検索します。

   a. **Google Cloud Console** バナーの左側にある **ナビゲーション** メニューをクリックします。

   b. **Compute Engine** を選択し、**イメージ** を選択します。

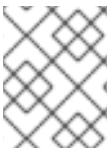
関連情報

- カスタム イメージの作成、削除、使用中止

- gcloud compute images create

3.4.7. イメージからの Google Compute Engine インスタンスの作成

イメージから GCE 仮想マシンインスタンスを設定するには、GCP コンソールを使用します。



注記

GCE 仮想マシンインスタンスおよびその設定オプションの詳細は、Creating and starting a VM instance を参照してください。

手順

1. GCP Console Dashboard ページ から、Google の **Cloud Console バナー** の左側にある **ナビゲーション** メニューをクリックし、**Compute Engine** を選択して **イメージ** を選択します。
2. イメージを選択します。
3. **インスタンスの作成** をクリックします。
4. **インスタンスの作成** ページで、インスタンスの **名前** を入力します。
5. **地域** と **ゾーン** を選択します。
6. ワークロードの要件を満たす、もしくは超過する **マシン設定** を選択します。
7. **ブートディスク** にイメージ名が指定されていることを確認します。
8. 必要に応じて、**ファイアウォール** で **HTTP トラフィックを許可する** か、**HTTPS トラフィックを許可する** を選択します。
9. **作成** をクリックします。

注記

これらは、基本インスタンスの作成に必要な最小限の設定オプションです。アプリケーション要件に応じて追加オプションを確認します。

10. **VM インスタンス** にあるイメージを見つけます。
11. GCP Console Dashboard から、Google の **Cloud Console バナー** の左側にある **ナビゲーション** メニューをクリックし、**Compute Engine** を選択してから **仮想マシンインスタンス** を選択します。

注記

または、CLI コマンド **gcloud compute instances create** を使用して、イメージから GCE 仮想マシンインスタンスを作成することもできます。簡単な例を以下に示します。

```
gcloud compute instances create myinstance3 --zone=us-central1-a --image
test-iso2-image
```

この例では、既存のイメージ **test-iso2-image** に基づいて、ゾーン **us-central1-a** に **myinstance3** という名前の仮想マシンインスタンスを作成します。詳細は、gcloud compute instances create を参照してください。

3.4.8. インスタンスへの接続

パブリック IP アドレスを使用して GCE インスタンスに接続します。

手順

1. インスタンスが実行中であることを確認します。次のコマンドは、インスタンスが実行中かどうか、実行中の場合は実行中のインスタンスのパブリック IP アドレスなど、GCE インスタンスに関する情報をリスト表示します。

```
# gcloud compute instances list
```

2. 標準の SSH を使用してインスタンスに接続します。この例では、上述の手順で作成した **google_compute_engine** キーを使用します。

```
# ssh -i ~/.ssh/google_compute_engine <user_name>@<instance_external_ip>
```

注記

GCP は、インスタンスに対して SSH を多数実行する方法を提供します。詳細は、Connecting to instances を参照してください。以前に設定した root アカウントおよびパスワードを使用して、インスタンスに接続することもできます。

関連情報

- gcloud compute instances list
- インスタンスへの接続

3.4.9. Red Hat サブスクリプションの割り当て

subscription-manager コマンドを使用すると、Red Hat サブスクリプションを登録して RHEL インスタンスに割り当てることができます。

前提条件

- サブスクリプションが有効になっている。

手順

1. システムを登録します。

```
# subscription-manager register --auto-attach
```

2. サブスクリプションを割り当てます。

   - アクティベーションキーを使用して、サブスクリプションを割り当てることができます。詳細は、カスタマーポータルのアクティベーションキーを作成する を参照してください。
   - または、サブスクリプションプール (Pool ID) の ID を使用してサブスクリプションを手動で割り当てることができます。ホストベースのサブスクリプションのハイパーバイザーへの割り当て を参照してください。

3. **オプション:** Red Hat Hybrid Cloud Console でインスタンスに関するさまざまなシステムメトリクスを収集するには、インスタンスを Red Hat Insights に登録します。

```
# insights-client register --display-name <display-name-value>
```

   Red Hat Insights の詳細な設定については、Red Hat Insights のクライアント設定ガイド を参照してください。

関連情報

- カスタマーポータルのアクティベーションキーを作成する
- ホストベースのサブスクリプションのハイパーバイザーへの割り当て
- Red Hat Insights のクライアント設定ガイド

3.5. 関連情報

- Red Hat in the Public Cloud
- Google Cloud

第4章 GOOGLE CLOUD PLATFORM での RED HAT HIGH AVAILABILITY クラスターの設定

ノード障害が発生した場合に RHEL ノードがワークロードを自動的に再分配するクラスターを作成するには、Red Hat High Availability Add-On を使用します。このような高可用性 (HA) クラスターは、Google Cloud Platform (GCP) などのパブリッククラウドプラットフォームでもホストできます。GCP 上での RHEL HA クラスターの作成は、特定の詳細を除けば、非クラウド環境での HA クラスターの作成と同様です。

Google Compute Engine (GCE) 仮想マシン (VM) インスタンスをクラスターノードとして使用して、Google Cloud Platform (GCP) で Red Hat HA クラスターを設定するには、以下のセクションを参照してください。

これらは、次の情報を提供します。

- GCP 用の環境を設定するための前提手順。環境を設定したら、仮想マシンインスタンスを作成および設定できます。
- 個別のノードを GCP の HA ノードのクラスターに変換する HA クラスターの作成に固有の手順。これには、各クラスターノードに高可用性パッケージおよびエージェントをインストールし、フェンシングを設定し、ネットワークリソースエージェントをインストールする手順が含まれます。

前提条件

- Red Hat Enterprise Linux 9 Server: rhel-9-server-rpms/8Server/x86_64
- Red Hat Enterprise Linux 9 Server (High Availability): rhel-9-server-ha-rpms/8Server/x86_64
  - アクティブな GCP プロジェクトに属し、プロジェクトでリソースを作成するのに十分なパーミッションを持っている必要があります。
  - プロジェクトには、個別ユーザーではなく、仮想マシンインスタンスに属する サービスアカウント が必要です。別のサービスアカウントを作成する代わりに、デフォルトのサービスアカウントを使用する方法は、Using the Compute Engine Default Service Account を参照してください。

またはプロジェクト管理者がカスタムサービスアカウントを作成する場合、サービスアカウントは以下のロールに設定する必要があります。

- クラウドトレースエージェント
- コンピュート管理者
- コンピュートネットワーク管理者
- クラウドデータベースユーザー
- ロギング管理者
- 監視エディター
- 監視メトリックライター
- サービスアカウント管理者

- ストレージ管理者

4.1. パブリッククラウドプラットフォームで高可用性クラスターを使用する利点

高可用性 (HA) クラスターは、特定のワークロードを実行するためにリンクされた一連のコンピューター (**ノード** と呼ばれます) です。HA クラスターの目的は、ハードウェアまたはソフトウェア障害が発生した場合に備えて、冗長性を提供することです。HA クラスター内のノードに障害が発生しても、Pacemaker クラスターリソースマネージャーがワークロードを他のノードに分散するため、クラスター上で実行されているサービスに顕著なダウンタイムが発生することはありません。

HA クラスターはパブリッククラウドプラットフォームで実行することもできます。この場合、クラウド内の仮想マシン (VM) インスタンスを個々のクラスターノードとして使用します。パブリッククラウドプラットフォームで HA クラスターを使用すると、次の利点があります。

- 可用性の向上: VM に障害が発生した場合、ワークロードがすぐに他のノードに再分散されるため、実行中のサービスが中断されません。
- スケーラビリティー: 需要が高いときに追加のノードを起動し、需要が低いときにノードを停止することができます。
- 費用対効果: 従量課金制の料金設定では、実行中のノードに対して料金を支払うだけで済みます。
- 管理の簡素化: 一部のパブリッククラウドプラットフォームでは、HA クラスターの設定を容易にする管理インターフェイスが提供されています。

Red Hat Enterprise Linux (RHEL) システムで HA を有効にするために、Red Hat は High Availability Add-On を提供しています。High Availability Add-On は、RHEL システム上で HA クラスターを作成するために必要なすべてのコンポーネントを提供します。コンポーネントには、高可用性サービス管理ツールとクラスター管理ツールが含まれます。

関連情報

- High Availability Add-On の概要

4.2. 必要なシステムパッケージ

RHEL の基本イメージを作成して設定するには、ホストシステムに次のパッケージがインストールされている必要があります。

表4.1 システムパッケージ

パッケージ	リポジトリー	説明
libvirt	rhel-9-for-x86_64-appstream-rpms	プラットフォーム仮想化を管理するためのオープンソース API、デーモン、および管理ツール。
virt-install	rhel-9-for-x86_64-appstream-rpms	仮想マシンを構築するためのコマンドラインユーティリティー

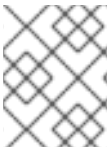
パッケージ	リポジトリー	説明
libguestfs	rhel-9-for-x86_64-appstream-rpms	仮想マシンファイルシステムにアクセスして変更するためのライブラリー
guestfs-tools	rhel-9-for-x86_64-appstream-rpms	仮想マシン用のシステム管理ツール。**virt-customize** ユーティリティーが含まれています

4.3. GCP での RED HAT ENTERPRISE LINUX イメージオプション

Google Cloud Platform に RHEL 9 をデプロイする場合、複数のタイプのイメージを使用できます。要件に基づいて、ユースケースに最適なオプションを検討してください。

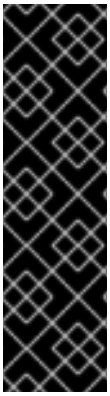
表4.2 イメージオプション

イメージオプション	サブスクリプション	サンプルシナリオ	留意事項
Red Hat Gold Image をデプロイする	既存の Red Hat サブスクリプションを使用する	Google Cloud Platform で Red Hat Gold Image を選択します。Gold イメージの詳細、および Google Cloud Platform で Gold Image にアクセスする方法は、Red Hat Cloud Access Reference Guide を参照してください。	その他のすべてのインスタンスコストを Google 社に支払うことになりますが、サブスクリプション自体には Red Hat 製品コストが含まれます。Red Hat は、カスタム RHEL イメージのサポートを直接提供します。
GCP に移動するカスタムイメージをデプロイする	既存の Red Hat サブスクリプションを使用する	カスタムイメージをアップロードし、サブスクリプションを割り当てます。	その他のすべてのインスタンスコストを支払うことになりますが、サブスクリプション自体には Red Hat 製品コストが含まれます。Red Hat は、カスタム RHEL イメージのサポートを直接提供します。
RHEL を含む既存の GCP イメージをデプロイする	GCP イメージには、Red Hat 製品が含まれる	GCP Compute Engine でインスタンスを起動する時に RHEL イメージを選択するか、Google Cloud Platform Marketplace からイメージを選択します。	従量課金モデルでは、GCP に 1 時間ごとに支払います。このようなイメージは "オンデマンド" イメージと呼ばれています。GCP は、サポート契約に基づいてオンデマンドイメージのサポートを提供します。



注記

Red Hat Image Builder を使用して、GCP 用のカスタムイメージを作成できます。詳細は Composing a Customized RHEL System Image を参照してください。



重要

オンデマンドインスタンスをカスタム RHEL インスタンスに変換することはできません。オンデマンドイメージからカスタム RHEL **bring-your-own-subscription** (BYOS) イメージに変更するには、次の手順を実行します。

1. 新しいカスタム RHEL インスタンスを作成し、オンデマンドインスタンスからデータを移行します。
2. データを移行した後に、オンデマンドのインスタンスをキャンセルして二重請求を回避します。

関連情報

- Red Hat in the Public Cloud
- コンピュートエンジンのイメージ
- Creating an instance from a custom image

4.4. GOOGLE CLOUD SDK のインストール

Google Cloud Platform (GCP) で HA クラスターを管理する際に、多くの手順で Google Cloud SDK のツールが必要になります。

手順

1. GCP の説明に従って、Google Cloud SDK アーカイブをダウンロードし、抽出します。詳細は、GCP ドキュメント Quickstart for Linux を参照してください。
2. Google Cloud SDK の初期化と同じ手順に従います。

注記

Google Cloud SDK を初期化すると、**gcloud** CLI コマンドを使用してタスクを実行でき、プロジェクトとインスタンスに関する情報を取得できます。たとえば、**gcloud compute project-info describe --project <project-name>** コマンドを使用してプロジェクト情報を表示できます。

関連情報

- Linux 用のクイックスタート
- gcloud command reference
- gcloud コマンドライン ツールの概要

4.5. GCP イメージバケットの作成

以下のドキュメントには、デフォルトの場所に 複数のリージョン のバケットを作成する最小要件が記載されています。

前提条件

- GCP ストレージユーティリティー (gsutil)

手順

1. Google Cloud Platform にログインしていない場合は、次のコマンドを実行してログインします。

```
# gcloud auth login
```

2. ストレージバケットを作成します。

```
$ gsutil mb gs://BucketName
```

以下に例を示します。

```
$ gsutil mb gs://rhel-ha-bucket
```

関連情報

- Make buckets

4.6. カスタムの仮想プライベートクラウドネットワークおよびサブネットの作成

クラスターを高可用性 (HA) 機能を使用して設定するには、Virtual Private Cloud (VPC) ネットワークとサブネットが必要です。

手順

1. GCP コンソールを起動します。
2. 左側のナビゲーションペインで **Networking** の下にある **VPC ネットワーク** を選択します。
3. **VPC ネットワークの作成** をクリックします。
4. VPC ネットワークの名前を入力します。
5. **新規サブネット** で、クラスターを作成するリージョンに **カスタムサブネット** を作成します。
6. **作成** をクリックします。

4.7. ベース GCP イメージの準備およびインポート

ローカルの RHEL 9 イメージを GCP にデプロイするには、まずイメージを変換して GCP バケットにアップロードする必要があります。

手順

1. ファイルを変換します。GCP にアップロードしたイメージは **raw** 形式であり、**disk.raw** という名前である必要があります。

```
$ qemu-img convert -f qcow2 ImageName.qcow2 -O raw disk.raw
```

2. **raw** ファイルを圧縮します。GCP にアップロードしたイメージを圧縮する必要があります。

```
$ tar -Sczf ImageName.tar.gz disk.raw
```

3. 圧縮されたイメージを先に作成したバケットにインポートします。

```
$ gsutil cp ImageName.tar.gz gs://BucketName
```

4.8. ベース GCP インスタンスの作成および設定

GCP の運用およびセキュリティー要件に準拠する GCP インスタンスを作成して設定するには、次の手順を実行します。

手順

1. バケット内の圧縮ファイルからイメージを作成します。

```
$ gcloud compute images create BaseImageName --source-uri
gs://BucketName/BaseImageName.tar.gz
```

以下に例を示します。

```
[admin@localhost ~] $ gcloud compute images create rhel-76-server --source-uri gs://user-
rhelha/rhel-server-76.tar.gz
Created [https://www.googleapis.com/compute/v1/projects/MyProject/global/images/rhel-
server-76].
NAME          PROJECT                FAMILY  DEPRECATED  STATUS
rhel-76-server  rhel-ha-testing-on-gcp                     READY
```

2. イメージからテンプレートインスタンスを作成します。ベース RHEL インスタンスに必要な最小サイズは n1-standard-2 です。追加の設定オプションは、gcloud compute instances create を参照してください。

```
$ gcloud compute instances create BaseInstanceName --can-ip-forward --machine-type n1-
standard-2 --image BaseImageName --service-account ServiceAccountEmail
```

以下に例を示します。

```
[admin@localhost ~] $ gcloud compute instances create rhel-76-server-base-instance --can-
ip-forward --machine-type n1-standard-2 --image rhel-76-server --service-account
account@project-name-on-gcp.iam.gserviceaccount.com
Created [https://www.googleapis.com/compute/v1/projects/rhel-ha-testing-on-gcp/zones/us-
east1-b/instances/rhel-76-server-base-instance].
NAME   ZONE   MACHINE_TYPE   PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP
STATUS
rhel-76-server-base-instance  us-east1-bn1-standard-2          10.10.10.3   192.227.54.211
RUNNING
```

3. SSH 端末セッションでインスタンスに接続します。

```
$ ssh root@PublicIPaddress
```

4. RHEL ソフトウェアを更新します。

   a. Red Hat Subscription Manager (RHSM) に登録します。

   b. サブスクリプションプール ID を有効にします (または **--auto-attach** コマンドを使用します)。

   c. すべてのリポジトリーを無効にします。

```
# subscription-manager repos --disable=*
```

   d. 以下のリポジトリーを有効にします。

```
# subscription-manager repos --enable=rhel-9-server-rpms
```

   e. **dnf update** コマンドを実行します。

```
# dnf update -y
```

5. 実行中のインスタンスに GCP Linux ゲスト環境をインストールします (インプレースインストール)。
   手順は、Install the guest environment in-place を参照してください。

6. **CentOS/RHEL** オプションを選択します。

7. コマンドスクリプトをコピーして、コマンドプロンプトに貼り付けて、スクリプトを即座に実行します。

8. インスタンスに以下の設定変更を行います。これらの変更は、カスタムイメージの GCP の推奨事項に基づいています。詳細は、gcloudcompute images list を参照してください。

   a. **/etc/chrony.conf** ファイルを編集し、すべての NTP サーバーを削除します。

   b. 以下の NTP サーバーを追加します。

```
metadata.google.internal iburst Google NTP server
```

   c. 永続的なネットワークデバイスルールを削除します。

```
# rm -f /etc/udev/rules.d/70-persistent-net.rules

# rm -f /etc/udev/rules.d/75-persistent-net-generator.rules
```

   d. ネットワークサービスが自動的に開始するように設定します。

```
# chkconfig network on
```

   e. **sshd service** が自動的に起動するように設定します。

```
# systemctl enable sshd
# systemctl is-enabled sshd
```

f. タイムゾーンを UTC に設定します。

```
# ln -sf /usr/share/zoneinfo/UTC /etc/localtime
```

g. (必要に応じて) **/etc/ssh/ssh_config** ファイルを編集し、以下の行をファイルの最後に追加します。アクティブではない期間が長くなった場合も、SSH セッションをアクティブな状態に保ちます。

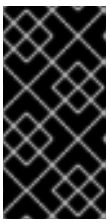
```
# Server times out connections after several minutes of inactivity.
# Keep alive ssh connections by sending a packet every 7 minutes.
ServerAliveInterval 420
```

h. **/etc/ssh/sshd_config** ファイルを編集し、必要に応じて以下の変更を加えます。**ClientAliveInterval 420** 設定は任意です。これにより、アクティブではない期間が長くなった場合にも SSH セッションをアクティブな状態に保ちます。

```
PermitRootLogin no
PasswordAuthentication no
AllowTcpForwarding yes
X11Forwarding no
PermitTunnel no
# Compute times out connections after 10 minutes of inactivity.
# Keep ssh connections alive by sending a packet every 7 minutes.
ClientAliveInterval 420
```

9. パスワードアクセスを無効にします。

```
ssh_pwauth from 1 to 0.
ssh_pwauth: 0
```



重要

以前は、SSH セッションアクセスを許可するパスワードアクセスを有効にして、インスタンスを設定していました。パスワードアクセスを無効にする必要があります。すべての SSH セッションアクセスはパスワードなしにする必要があります。

10. サブスクリプションマネージャーからインスタンスの登録を解除します。

```
# subscription-manager unregister
```

11. シェルの履歴を消去します。次の手順でインスタンスを実行した状態にします。

```
# export HISTSIZE=0
```

4.9. スナップショットイメージの作成

GCP HA インスタンスの設定とディスクデータを保存するには、そのスナップショットを作成します。

手順

1. 実行中のインスタンスで、データをディスクに同期します。

```
# sync
```

2. ホストシステムで、スナップショットを作成します。

```
$ gcloud compute disks snapshot InstanceName --snapshot-names SnapshotName
```

3. ホストシステムで、スナップショットから設定済みのイメージを作成します。

```
$ gcloud compute images create ConfiguredImageFromSnapshot --source-snapshot
SnapshotName
```

関連情報

- Creating Persistent Disk Snapshots

4.10. HA ノードテンプレートインスタンスおよび HA ノードの作成

スナップショットからイメージを設定したら、ノードテンプレートを作成できます。その後、このテンプレートを使用してすべての HA ノードを作成できます。

手順

1. インスタンステンプレートを作成します。

```
$ gcloud compute instance-templates create InstanceTemplateName --can-ip-forward --
machine-type n1-standard-2 --image ConfiguredImageFromSnapshot --service-account
ServiceAccountEmailAddress
```

以下に例を示します。

```
[admin@localhost ~] $ gcloud compute instance-templates create rhel-91-instance-template
--can-ip-forward --machine-type n1-standard-2 --image rhel-91-gcp-image --service-account
account@project-name-on-gcp.iam.gserviceaccount.com
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-
gcp/global/instanceTemplates/rhel-91-instance-template].
NAME  MACHINE_TYPE   PREEMPTIBLE  CREATION_TIMESTAMP
rhel-91-instance-template   n1-standard-2          2018-07-25T11:09:30.506-07:00
```

2. 1 つのゾーンに複数のノードを作成します。

```
# gcloud compute instances create NodeName01 NodeName02 --source-instance-template
InstanceTemplateName --zone RegionZone --network=NetworkName --
subnet=SubnetName
```

以下に例を示します。

```
[admin@localhost ~] $ gcloud compute instances create rhel81-node-01 rhel81-node-02
rhel81-node-03 --source-instance-template rhel-91-instance-template --zone us-west1-b --
network=projectVPC --subnet=range0
```

```
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-01].
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-02].
Created [https://www.googleapis.com/compute/v1/projects/project-name-on-gcp/zones/us-west1-b/instances/rhel81-node-03].
NAME          ZONE        MACHINE_TYPE  PREEMPTIBLE  INTERNAL_IP  EXTERNAL_IP     STATUS
rhel81-node-01  us-west1-b  n1-standard-2               10.10.10.4   192.230.25.81   RUNNING
rhel81-node-02  us-west1-b  n1-standard-2               10.10.10.5   192.230.81.253  RUNNING
rhel81-node-03  us-east1-b  n1-standard-2               10.10.10.6   192.230.102.15  RUNNING
```

4.11. HA パッケージおよびエージェントのインストール

Google Cloud Platform (GCP) 上で Red Hat High Availability クラスターを設定できるようにするには、各ノードに高可用性パッケージとエージェントをインストールする必要があります。

手順

1. Google Cloud コンソールで **Compute Engine** を選択し、**VM instance** を選択します。
2. インスタンスを選択し、**SSH** の横にある矢印をクリックして、gcloud コマンドオプションの **表示** を選択します。
3. インスタンスへのパスワードなしアクセスのために、コマンドプロンプトでこのコマンドを貼り付けます。
4. sudo アカウントアクセスを有効にし、Red Hat Subscription Manager に登録します。
5. サブスクリプションプール ID を有効にします (または **--auto-attach** コマンドを使用します)。
6. すべてのリポジトリーを無効にします。

```
# subscription-manager repos --disable=*
```

7. 以下のリポジトリーを有効にします。

```
# subscription-manager repos --enable=rhel-9-server-rpms
# subscription-manager repos --enable=rhel-9-for-x86_64-highavailability-rpms
```

8. **pcs pacemaker**、フェンスエージェント、およびリソースエージェントをインストールします。

```
# dnf install -y pcs pacemaker fence-agents-gce resource-agents-gcp
```

9. すべてのパッケージを更新します。

```
# dnf update -y
```

4.12. HA サービスの設定

各ノードで HA サービスを設定します。

手順

1. **hacluster** ユーザーは、前の手順で **pcs** および **pacemaker** のインストール時に作成されました。すべてのクラスターノードに **hacluster** ユーザーのパスワードを作成します。すべてのノードに同じパスワードを使用します。

```
# passwd hacluster
```

2. **firewalld** サービスがインストールされている場合は、HA サービスを追加します。

```
# firewall-cmd --permanent --add-service=high-availability

# firewall-cmd --reload
```

3. **pcs** サービスを起動し、システムの起動時に開始できるようにします。

```
# systemctl start pcsd.service

# systemctl enable pcsd.service

Created symlink from /etc/systemd/system/multi-user.target.wants/pcsd.service to
/usr/lib/systemd/system/pcsd.service.
```

検証

1. **pcsd** サービスが実行していることを確認します。

```
# systemctl status pcsd.service

pcsd.service - PCS GUI and remote configuration interface
Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
Active: active (running) since Mon 2018-06-25 19:21:42 UTC; 15s ago
Docs: man:pcsd(8)
man:pcs(8)
Main PID: 5901 (pcsd)
CGroup: /system.slice/pcsd.service
└─5901 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &
```

2. **/etc/hosts** ファイルを編集します。すべてのノードに RHEL ホスト名と内部 IP アドレスを追加します。

関連情報

- How should the /etc/hosts file be set up on RHEL cluster nodes?

4.13. クラスターの作成

複数のノードをクラスターに変換するには、次の手順を使用します。

手順

1. ノードの 1 つで、**pcs** ユーザーを認証します。コマンドのクラスター内の各ノードの名前を指定します。

```
# pcs host auth hostname1 hostname2 hostname3
Username: hacluster
Password:
hostname1: Authorized
hostname2: Authorized
hostname3: Authorized
```

2. クラスターを作成します。

```
# pcs cluster setup cluster-name hostname1 hostname2 hostname3
```

検証

1. 以下のコマンドを実行して、起動時にノードが自動的にクラスターに参加できるようにします。

```
# pcs cluster enable --all
```

2. クラスターを起動します。

```
# pcs cluster start --all
```

4.14. フェンシングデバイスの作成

高可用性 (HA) 環境には、障害発生時に誤動作しているノードを分離し、クラスターの可用性を維持するフェンシングデバイスが必要です。

ほとんどのデフォルト設定では、GCP インスタンス名と RHEL ホスト名が同じである点に注意してください。

手順

1. GCP インスタンス名を取得します。次のコマンドの出力には、インスタンスの内部 ID も表示されることに注意してください。

```
# fence_gce --zone us-west1-b --project=rhel-ha-on-gcp -o list
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# fence_gce --zone us-west1-b --project=rhel-ha-testing-on-gcp
-o list

4435801234567893181,InstanceName-3
4081901234567896811,InstanceName-1
7173601234567893341,InstanceName-2
```

2. フェンスデバイスを作成します。

```
# pcs stonith create FenceDeviceName fence_gce zone=Region-Zone
project=MyProject
```

検証

- フェンスデバイスが起動していることを確認します。

```
# pcs status
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs status

Cluster name: gcp-cluster
Stack: corosync
Current DC: rhel81-node-02 (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
Last updated: Fri Jul 27 12:53:25 2018
Last change: Fri Jul 27 12:51:43 2018 by root via cibadmin on rhel81-node-01

3 nodes configured
3 resources configured

Online: [ rhel81-node-01 rhel81-node-02 rhel81-node-03 ]

Full list of resources:

us-west1-b-fence    (stonith:fence_gce):    Started rhel81-node-01

Daemon Status:
corosync: active/enabled
pacemaker: active/enabled
pcsd: active/enabled
```

4.15. GCP-VCP-MOVE-VIP リソースエージェントの設定

gcp-vpc-move-vip リソースエージェントは、セカンダリー IP アドレス (エイリアス IP) を実行中のインスタンスに割り当てます。これは、クラスター内の異なるノード間で渡すことのできるフローティング IP アドレスです。

このリソースに関する詳細情報を表示します。

```
# pcs resource describe gcp-vpc-move-vip
```

プライマリーサブネットアドレス範囲またはセカンダリーサブネットアドレス範囲を使用するようにリソースエージェントを設定できます。

プライマリーサブネットアドレス範囲

プライマリー VPC サブネットのリソースを設定するには、以下の手順を実行します。

手順

1. **aliasip** リソースを作成します。未使用の内部 IP アドレスを含めます。コマンドに CIDR ブロックを含めます。

```
# pcs resource create aliasip gcp-vpc-move-vip alias_ip=UnusedIPaddress/CIDRblock
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource create aliasip gcp-vpc-move-vip
alias_ip=10.10.10.200/32
```

2. ノード上の IP を管理する **IPaddr2** リソースを作成します。

```
# pcs resource create vip IPaddr2 nic=interface ip=AliasIPaddress cidr_netmask=32
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource create vip IPaddr2 nic=eth0 ip=10.10.10.200
cidr_netmask=32
```

3. **vipgrp** でネットワークリソースをグループ化します。

```
# pcs resource group add vipgrp aliasip vip
```

検証

1. リソースが起動し、**vipgrp** でグループ化されていることを確認します。

```
# pcs status
```

2. リソースが別のノードに移動できることを確認します。

```
# pcs resource move vip Node
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource move vip rhel81-node-03
```

3. 別のノードで **vip** が正常に起動することを確認します。

```
# pcs status
```

セカンダリーサブネットアドレス範囲

セカンダリーサブネットアドレス範囲のリソースを設定するには、以下の手順を実施します。

前提条件

- カスタムネットワークとサブネットを作成している
- **オプション:** Google Cloud SDK がインストールされている。手順については、 Google Cloud SDK のインストール をご覧ください。
  ただし、次の手順の **gcloud** コマンドは、Google Cloud Web コンソールで有効化できるターミナルで使用できることに注意してください。

手順

1. セカンダリーサブネットアドレス範囲を作成します。

```
# gcloud compute networks subnets update SubnetName --region RegionName --add-secondary-ranges SecondarySubnetName=SecondarySubnetRange
```

以下に例を示します。

```
# gcloud compute networks subnets update range0 --region us-west1 --add-secondary-ranges range1=10.10.20.0/24
```

2. **aliasip** リソースを作成します。セカンダリーサブネットアドレス範囲に未使用の内部 IP アドレスを作成します。コマンドに CIDR ブロックを含めます。

```
# pcs resource create aliasip gcp-vpc-move-vip alias_ip=UnusedIPaddress/CIDRblock
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource create aliasip gcp-vpc-move-vip alias_ip=10.10.20.200/32
```

3. ノード上の IP を管理する **IPaddr2** リソースを作成します。

```
# pcs resource create vip IPaddr2 nic=interface ip=AliasIPaddress cidr_netmask=32
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource create vip IPaddr2 nic=eth0 ip=10.10.20.200 cidr_netmask=32
```

4. **vipgrp** でネットワークリソースをグループ化します。

```
# pcs resource group add vipgrp aliasip vip
```

検証

1. リソースが起動し、**vipgrp** でグループ化されていることを確認します。

```
# pcs status
```

2. リソースが別のノードに移動できることを確認します。

```
# pcs resource move vip Node
```

以下に例を示します。

```
[root@rhel81-node-01 ~]# pcs resource move vip rhel81-node-03
```

3. 別のノードで **vip** が正常に起動することを確認します。

```
# pcs status
```

4.16. 関連情報

- Support Policies for RHEL High Availability clusters - Transport Protocols
- VPC network overview
- Exploring RHEL High Availability's Components, Concepts, and Features - Overview of Transport Protocols
- Design Guidance for RHEL High Availability Clusters - Selecting the Transport Protocol