Red Hat Enterprise Linux 8

RHEL 8 の cloud-init の設定と管理

cloud-init を使用したクラウドインスタンスの初期化の自動化

Last Updated: 2023-01-22

Red Hat Enterprise Linux 8 RHEL 8 の cloud-init の設定と管理

cloud-init を使用したクラウドインスタンスの初期化の自動化

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

概要

クラウドインスタンスの初期化を自動化するには、cloud-init パッケージを使用できます。仮想マシンに cloud-init をインストールするか、cloud-init がすでにインストールされている Red Hat Enterprise Linux イメージを選択できます。cloud-init は、多くの Red Hat 製品と共に使用できます。詳細については、このドキュメントの次のセクションを参照してください。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

当社の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。

特定の文章に関するコメントの送信

1. **Multi-page HTML** 形式でドキュメントを表示し、ページが完全にロードされてから右上隅に **Feedback** ボタンが表示されていることを確認します。
2. カーソルを使用して、コメントを追加するテキスト部分を強調表示します。
3. 強調表示されたテキストの近くに表示される **Add Feedback** ボタンをクリックします。
4. フィードバックを追加し、**Submit** をクリックします。

Bugzilla からのフィードバック送信 (アカウントが必要)

1. Bugzilla の Web サイトにログインします。
2. **Version** メニューから正しいバージョンを選択します。
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. **Submit Bug** をクリックします。

第1章 CLOUD-INIT の概要

cloud-init は、システムの起動時にクラウドインスタンスの初期化を自動化するソフトウェアパッケージです。**cloud-init** は、さまざまなタスクを実行するように設定できます。以下は、**cloud-init** が実行できるタスクの例です。

- ホスト名の設定
- インスタンスへのパッケージのインストール
- スクリプトの実行
- デフォルトの仮想マシン (VM) 動作の抑制

cloud-init の設定用にイメージを取得する場所は、その使用方法によって異なります。

- **cloud-init** パッケージは、Red Hat カスタマーポータル からダウンロードした KVM ゲストイメージにインストールされます。インスタンスを起動すると、**cloud-init** が有効になります。Red Hat カスタマーポータルからダウンロードした KVM ゲストイメージは、Red Hat Virtualization (RHV) および Red Hat OpenStack Platform を対象として使用されます。RHV および RHOSP 用のイメージをゼロから作成することもできます。
- 別の方法として、Red Hat カスタマーポータルから ISO イメージをダウンロードするか、イメージを作成することができます。この場合は、**cloud-init** を ISO イメージにインストールする必要があります。
- クラウドプロバイダー (AWS または Azure など) でイメージを使用する予定の場合は、Red Hat Image Builder を使用してイメージを作成します。Image Builder のイメージは、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ます。イメージタイプ AMI、VHD、および qcow2 には、**cloud-init** がすでにインストールされています。Image Builder の詳細は、RHEL システムイメージのカスタマイズ を参照してください。

ほとんどのクラウドプラットフォームは **cloud-init** をサポートしますが、設定手順とサポートされるオプションは異なります。また、NoCloud 環境向けに **cloud-init** を設定できます。

cloud-init を 1 台の仮想マシンに設定し、その仮想マシンを追加の仮想マシンまたは仮想マシンのクラスター用のテンプレートとして使用できます。

特定の Red Hat 製品 (Red Hat Virtualization など) では、これらの製品で使用するために **cloud-init** を設定する手順が文書化されています。

本ガイドでは、**cloud-init** のドキュメントを参照している箇所が多数あります。**cloud-init** の完全な情報は、参照用の **cloud-init** ドキュメントを参照してください。

前提条件

- Red Hat カスタマーポータル のアカウントにサインアップします。

1.1. 関連情報

- cloud-init のドキュメント

1.2. CLOUD-INIT の設定

cloud-init は YAML 形式のファイル命令を使用してタスクを実行します。YAML ファイル内で命令を指

定して、**cloud-init** が実行する初期設定を決定します。インスタンスが起動すると、**cloud-init** サービスが起動して、命令を検索して実行します。タスクは、**cloud-init** 設定に基づいて、仮想マシンの初回起動時または後続の起動時に完了します。

/etc/cloud/cloud.cfg ファイルを設定し、**/etc/cloud/cloud.cfg.d/** ディレクトリーの下にディレクティブを追加して、タスクを定義します。

- **cloud.cfg** ファイルには、ユーザーアクセス、認証、システム情報用などのディレクティブが含まれます。
  ファイルには、**cloud-init** のデフォルトおよびオプションのモジュールも含まれています。モジュールは、**cloud-init** 初期化フェーズ、設定フェーズ、最終フェーズを含む 3 つのフェーズ内で順番に実行されます。**cloud.cfg** ファイル内では、3 つのフェーズのモジュールが、**cloud_init_modules**、**cloud_config_modules**、および **cloud_final_modules** の下にそれぞれ一覧表示されます。

- **cloud.cfg.d** ディレクトリーでは、**cloud-init** の追加ディレクティブを追加できます。**cloud.cfg.d** ディレクトリーにディレクティブを追加する場合、通常は ***.cfg** という名前のファイルに追加し、ファイルの上部に **#cloud-config** を常に含めます。

1.3. CLOUD-INIT はステージごとに動作する

cloud-init は、システム起動時に 5 つのステージで動作します。これらの段階では、**cloud-init** が実行するかどうか、および他のタスクの中からそのデータソースを見つける場所が決定されます。ステージは次のとおりです。

1. **cloud-init** ジェネレーターステージでは、**systemd** サービスを介して、起動時に **cloud-init** を実行するかどうかを判断します。

2. ローカルステージでは、**cloud-init** はローカルデータソースを検索し、ネットワーク設定を適用します。

3. ネットワークステージでは、**cloud-init** はユーザーデータを処理し、**cloud.cfg** ファイルの **cloud_init_modules** の下に一覧表示されるモジュールを実行します。**cloud_init_modules** セクションにモジュールを有効化、無効化、または追加できます。

4. config ステージでは、**cloud-init** は **cloud.cfg** ファイルの **cloud_config_modules** の下に一覧表示されるモジュールを実行します。**cloud_config_modules** セクションにモジュールを有効化、無効化、または追加できます。

5. 最終ステージでは、**cloud-init** は **cloud.cfg** ファイルの **cloud_final_modules** に追加したものを実行できます。システムの起動後に通常実行するパッケージインストールを追加できます。また、設定管理プラグインおよびユーザースクリプトを追加することもできます。**cloud_final_modules** セクションにモジュールを有効化、無効化、または追加できます。

起動時の 5 つのステージは、**cloud-init** ドキュメントの Boot Stages セクションで説明されています。

1.4. CLOUD-INIT モジュールはフェーズごとに実行される

cloud-init を実行すると、**cloud.cfg** 内のモジュールが次の 3 つのフェーズで順番に実行されます。

1. ネットワークフェーズ (**cloud_init_modules**)

2. 設定フェーズ (**cloud_config_modules**)

3. 最終フェーズ (**cloud_final_modules**)

仮想マシンで **cloud-init** が初めて実行されると、設定したすべてのモジュールがそれぞれのフェーズで実行されます。**cloud-init** の後続の実行では、モジュールがフェーズ内で実行されるかどうかは、個々のモジュールの **モジュール頻度** により異なります。**cloud-init** が実行されるたびに実行されるモジュールもあれば、インスタンス ID が変更された場合でも、**cloud-init** の初回実行時にしか実行されないモジュールもあります。

注記

インスタンス ID はインスタンスを一意に識別します。インスタンス ID が変更されると、**cloud-init** はそのインスタンスを新しいインスタンスとして処理します。

可能な **モジュール周波数** の値は次のとおりです。

- **Per instance** とは、モジュールがインスタンスの初回起動時に実行されることを意味します。たとえば、インスタンスのクローンを作成したり、保存したイメージから新しいインスタンスを作成したりすると、インスタンス別と指定されたモジュールは再度実行されます。
- **Per once** とは、モジュールが 1 回だけ実行されることを意味します。たとえば、インスタンスのクローンを作成したり、保存したイメージから新しいインスタンスを作成したりすると、1 回と指定されたモジュールは、それらのインスタンスでは再度実行されません。
- **Per always** とは、モジュールが起動ごとに実行されることを意味します。

注記

モジュールの設定時またはコマンドラインを使用して、モジュールの頻度を上書きできます。

1.5. CLOUD-INIT は、ユーザーデータ、メタデータ、およびベンダーデータに対応する

cloud-init は、ユーザーデータ、メタデータ、ベンダーデータを消費し、これらに対応します。

- ユーザーデータには、**cloud.cfg** ファイルと **cloud.cfg.d** ディレクトリーで指定するディレクティブが含まれます。たとえば、ユーザーデータには、実行するファイル、インストールするパッケージ、およびシェルスクリプトを含むことができます。**cloud-init** で許可されているユーザーデータのタイプの詳細は、**cloud-init** ドキュメントの User-Data Formats セクションを参照してください。
- メタデータには、特定のデータソースに関連付けられたデータが含まれます。たとえば、メタデータにはサーバー名とインスタンス ID を含むことができます。特定のクラウドプラットフォームを使用している場合、インスタンスがユーザーデータとメタデータを見つける場所をプラットフォームが決定します。プラットフォームでは、メタデータとユーザーデータの HTTP サービスへの追加が必要な場合があります。この場合、**cloud-init** を実行すると、HTTP サービスからメタデータとユーザーデータが消費されます。
- ベンダーデータは、組織 (クラウドプロバイダーなど) がオプションで提供し、イメージが実行される環境に合わせてイメージをカスタマイズできる情報が含まれます。**cloud-init** は、メタデータを読み込んでシステムを初期化した後に、オプションのベンダーデータおよびユーザーデータに対応します。デフォルトでは、ベンダーデータは初回起動時に実行されます。ベンダーデータの実行を無効にすることができます。
  cloud-init ドキュメントのメタデータ (Instance Metadata)、データソース (Datasources)、ベンダーデータ (Vendor Data) の各セクションを参照してください。

1.6. CLOUD-INIT はクラウドプラットフォームを識別する

cloud-init は、**ds-identify** スクリプトを使用してクラウドプラットフォームの特定を試みます。スクリプトは、インスタンスの初回起動時に実行されます。

データソースディレクティブを追加すると、**cloud-init** の実行時に時間を節約できます。ディレクティブは、**/etc/cloud/cloud.cfg** ファイルまたは **/etc/cloud/cloud.cfg.d** ディレクトリーに追加します。以下に例を示します。

```
datasource_list:[Ec2]
```

クラウドプラットフォームのディレクティブを追加すること以外に、メタデータ URL などの追加の設定詳細を追加して **cloud-init** をさらに設定できます。

```
datasource_list: [Ec2]
datasource:
  Ec2:
    metadata_urls: ['http://169.254.169.254']
```

cloud-init の実行後に、プラットフォームに関する詳細情報を提供するログファイル (**run/cloud-init/ds-identify.log**) を表示できます。

関連情報

- Datasources
- What datasource am I using?

第2章 CLOUD-INIT の RED HAT サポート

本章では、**cloud-init** の Red Hat サポートについて説明します。これには、 **cloud-init** を使用する Red Hat 製品、Red Hat がサポートする **cloud-init** モジュール、デフォルトのディレクトリーおよびファイルに関する情報が含まれます。

2.1. CLOUD-INIT の重要なディレクトリーおよびファイル

以下の表には、重要なディレクトリーおよびファイルが記載されています。これらのディレクトリーおよびファイルを確認します。これらにより、以下のようなタスクを実行することができます。

- **cloud-init** の設定
- **cloud-init** 実行後の設定に関する情報の検索
- ログファイルの検証
- テンプレートの検索

シナリオおよびデータソースに応じて、お使いの設定にとって重要な追加のファイルとディレクトリーが存在する場合があります。

表2.1 cloud-init ディレクトリーおよびファイル

ディレクトリーまたはファイル	説明
/etc/cloud/cloud.cfg	**cloud.cfg** ファイルには基本的な **cloud-init** 設定が含まれ、各モジュールが実行するフェーズを確認できます。
/etc/cloud/cloud.cfg.d	**cloud.cfg.d** ディレクトリーでは、**cloud-init** の追加ディレクティブを追加できます。
/var/lib/cloud	**cloud-init** を実行すると、**/var/lib/cloud** の下にディレクトリーレイアウトが作成されます。このレイアウトには、インスタンス設定の詳細情報を提供するディレクトリーとファイルが含まれます。
/usr/share/doc/cloud-init/examples	**examples** ディレクトリーには、複数の例が含まれています。これらを使用すると、独自のディレクティブを作成できます。
/etc/cloud/templates	このディレクトリーには、特定のシナリオの **cloud-init** で有効にできるテンプレートが含まれています。テンプレートは、有効にするための指示を提供します。
/var/log/cloud-init.log	**cloud-init.log** ファイルは、デバッグに役立つログ情報を提供します。

ディレクトリーまたはファイル	説明
/run/cloud-init	**/run/cloud-init** ディレクトリーには、データソースおよび **ds-identify** スクリプトのログ記録情報が含まれます。

2.2. CLOUD-INIT を使用する RED HAT 製品

以下の Red Hat 製品で **cloud-init** を使用できます。

- **Red Hat Virtualization.cloud-init** を仮想マシンにインストールすると、テンプレートを作成し、そのテンプレートから作成したすべての仮想マシンに **cloud-init** 機能を利用できます。仮想マシンでの **cloud-init** の使用に関する詳細は、Cloud-Init を使用した仮想マシンの設定の自動化 を参照してください。
- **Red Hat OpenStack Platform.cloud-init** を使用して、OpenStack のイメージの設定をサポートできます。詳細は、インスタンスおよびイメージガイド を参照してください。
- **Red Hat CloudForms.cloud-init** を使用して、Red Hat CloudForms 用の仮想マシンをプロビジョニングできます。詳細は、Customizing Templates for Virtual Machine and Instance Provisioning を参照してください。
- **Red Hat Satellite.cloud-init** を Red Hat Satellite で使用することができます。詳細は、Preparing Cloud-init Images in Red Hat Virtualization を参照してください。
- **Red Hat OpenShift.**OpenShift 用の仮想マシンを作成する際に **cloud-init** を使用できます。詳細は、Creating virtual machines を参照してください。

2.3. RED HAT はこれらの CLOUD-INIT モジュールをサポートします。

Red Hat は、ほとんどの **cloud-init** モジュールをサポートしています。個々のモジュールには、複数の設定オプションを含めることができます。以下の表は、Red Hat が現在サポートしているすべての **cloud-init** モジュールの一覧で、簡単な説明とデフォルトのモジュール頻度を記載しています。これらのモジュールの完全な説明およびオプションは、cloud-init ドキュメントの Modules セクションを参照してください。

表2.2 サポートされている cloud-init モジュール

cloud-init モジュール	説明	デフォルトのモジュール頻度
bootcmd	ブートプロセスの初期段階でコマンドを実行します。	常時
ca_certs	CA 証明書を追加します。	インスタンス別
debug	デバッグを支援するために内部情報の出力を有効または無効にします。	インスタンス別
disable_ec2_metadata	AWS EC2 メタデータを有効または無効にします。	常時

cloud-init モジュール	説明	デフォルトのモジュール頻度
disk_setup	シンプルなパーティションテーブルとファイルシステムを設定します。	インスタンス別
final_message	**cloud-init** の完了後に出力メッセージを指定します。	常時
foo	モジュール構造の例 (モジュールは何もしません)。	インスタンス別
growpart	パーティションのサイズを変更して、利用可能なディスク領域を埋めます。	常時
keys_to_console	コンソールに書き込むことができるフィンガープリントとキーの制御を許可します。	インスタンス別
landscape	ランドスケープクライアントをインストールおよび設定します。	インスタンス別
locale	システムロケールを設定し、システム全体に適用します。	インスタンス別
mcollective	**mcollective** をインストール、設定、および起動します。	インスタンス別
migrator	**cloud-init** の古いバージョンを新しいバージョンへに移動します。	常時
mounts	マウントポイントとスワップファイルを設定します。	インスタンス別
phone_home	起動完了後にリモートホストにデータを投稿します。	インスタンス別
power_state_change	すべての設定モジュールの実行後にシャットダウンを完了し、再起動します。	インスタンス別
puppet	puppet をインストールおよび設定します。	インスタンス別
resizefs	ファイルシステムのサイズを変更し、パーティションで利用可能な領域をすべて使用します。	常時

cloud-init モジュール	説明	デフォルトのモジュール頻度
resolv_conf	**resolv.conf** を設定します。	インスタンス別
rh_subscription	Red Hat Enterprise Linux システムを登録します。	インスタンス別
rightscale_userdata	**cloud-init** に RightScale 設定フックのサポートを追加します。	インスタンス別
rsyslog	**rsyslog** を使用してリモートシステムロギングを設定します。	インスタンス別
runcmd	任意のコマンドを実行します。	インスタンス別
salt_minion	**salt minion** をインストール、設定、および開始します。	インスタンス別
scripts_per_boot	起動スクリプトごとに実行します。	常時
scripts_per_instance	インスタンススクリプトごとに実行します。	インスタンス別
scripts_per_once	スクリプトを 1 回実行します。	1 回
scripts_user	ユーザースクリプトを実行します。	インスタンス別
scripts_vendor	ベンダースクリプトを実行します。	インスタンス別
seed_random	ランダムなシードデータを提供します。	インスタンス別
set_hostname	ホスト名および完全修飾ドメイン名 (FQDN) を設定します。	常時
set_passwords	ユーザーパスワードを設定し、SSH パスワード認証を有効または無効にします。	インスタンス別
ssh_authkey_fingerprints	ユーザーの SSH 鍵のフィンガープリントをログに記録します。	インスタンス別
ssh_import_id	SSH 鍵をインポートします。	インスタンス別
ssh	SSH、ホスト、および認可された SSH 鍵を設定します。	インスタンス別

cloud-init モジュール	説明	デフォルトのモジュール頻度
timezone	システムのタイムゾーンを設定します。	インスタンス別
update_etc_hosts	**/etc/hosts** を更新します。	常時
update_hostname	ホスト名および FQDN を更新します。	常時
users_groups	ユーザーおよびグループを設定します。	インスタンス別
write_files	任意のファイルを書き込みます。	インスタンス別
yum_add_repo	yum リポジトリー設定をシステムに追加します。	常時

以下の表は、現在 Red Hat がサポートしていないモジュールを一覧表示しています。

表2.3 モジュールはサポートされません。

モジュール
apt_configure
apt_pipeline
byobu
chef
emit_upstart
grub_dpkg
ubuntu_init_switch

2.4. デフォルトの CLOUD.CFG ファイル

/etc/cloud/cloud.cfg ファイルは、**cloud-init** の基本設定を設定するモジュールを一覧表示します。

ファイルのモジュールは、**cloud-init** のデフォルトのモジュールです。お使いの環境にモジュールを設定したり、不要なモジュールを削除したりすることができます。**cloud.cfg** に含まれるモジュールは、ファイルに一覧表示されているだけで、必ずしもなんでも実行する訳ではありません。**cloud-init** フェーズのいずれかでアクションを実行する必要がある場合は、それらを個別に設定する必要があります。

cloud.cfg ファイルは、個別のモジュールを実行するための時系列を提供します。Red Hat が追加するモジュールをサポートする限り、追加のモジュールを **cloud.cfg** に追加できます。

Red Hat Enterprise Linux (RHEL) のファイルのデフォルトコンテンツは、以下のとおりです。

注記

- モジュールは、**cloud.cfg** で指定された順序で実行されます。通常、この順序は変更しません。
- **cloud.cfg** ディレクティブは、ユーザーデータによって上書きされることができます。
- **cloud-init** を手動で実行する場合は、コマンドラインオプションで **cloud.cfg** を上書きできます。
- 各モジュールには独自の設定オプションが含まれており、この設定オプションに特定の情報を追加することができます。

```
users: 1
 - default

disable_root: 1 2
ssh_pwauth:   0 3

mount_default_fields: [~, ~, 'auto', 'defaults,nofail,x-systemd.requires=cloud-init.service', '0', '2'] 4
ssh_deletekeys:   1 5
ssh_genkeytypes: ['rsa', 'ecdsa', 'ed25519'] 6
syslog_fix_perms: ~ 7
disable_vmware_customization: false 8

cloud_init_modules: 9
 - disk_setup
 - migrator
 - bootcmd
 - write-files
 - growpart
 - resizefs
 - set_hostname
 - update_hostname
 - update_etc_hosts
 - rsyslog
 - users-groups
 - ssh

cloud_config_modules: 10
 - mounts
 - locale
 - set-passwords
 - rh_subscription
 - yum-add-repo
 - package-update-upgrade-install
 - timezone
 - puppet
```

```
 - chef
 - salt-minion
 - mcollective
 - disable-ec2-metadata
 - runcmd

cloud_final_modules: 11
 - rightscale_userdata
 - scripts-per-once
 - scripts-per-boot
 - scripts-per-instance
 - scripts-user
 - ssh-authkey-fingerprints
 - keys-to-console
 - phone-home
 - final-message
 - power-state-change

system_info:
  default_user: 12
    name: cloud-user
    lock_passwd: true
    gecos: Cloud User
    groups: [adm, systemd-journal]
    sudo: ["ALL=(ALL) NOPASSWD:ALL"]
    shell: /bin/bash
  distro: rhel 13
  paths:
    cloud_dir: /var/lib/cloud 14
    templates_dir: /etc/cloud/templates 15
  ssh_svcname: sshd 16

# vim:syntax=yaml
```

1. システムのデフォルトユーザーを指定します。詳細は、Users and Groups を参照してください。
2. root ログインを有効または無効にします。詳細は、Authorized Keys を参照してください。
3. **ssh** がパスワード認証を受け入れるよう設定されているかどうかを指定します。詳細は、Set Passwords を参照してください。
4. マウントポイントを設定します。6 つの値を含む一覧でなければなりません。詳細は、Mounts を参照してください。
5. デフォルトのホスト SSH 鍵を削除するかどうかを指定します。詳細は、Host Keys を参照してください。
6. 生成する鍵のタイプを指定します。詳細は、Host Keys を参照してください。RHEL 8.4 以前の場合、この行のデフォルト値は ~ であることに注意してください。
7. **cloud-init** は、起動時に複数のステージで実行されます。**cloud-init** が、すべてのステージをログファイルにログ記録できるように、このオプションを設定します。このオプションの詳細は、**usr/share/doc/cloud-init/examples** ディレクトリーの **cloud-config.txt** ファイルを参照してください。

8. VMware vSphere のカスタマイズを有効または無効にします。
9. 本セクションのモジュールは、起動プロセスの初期段階における **cloud-init** サービスの起動時に実行されるサービスです。
10. これらのモジュールは、初回起動後の **cloud-init** 設定時に実行されます。
11. これらのモジュールは、設定完了後に **cloud-init** の最終フェーズで実行されます。
12. デフォルトユーザーの詳細を指定します。詳細は、Users and Groups を参照してください。
13. ディストリビューションを指定します。
14. **cloud-init** 固有のサブディレクトリーが含まれるメインディレクトリーを指定します。詳細は、Directory layout を参照してください。
15. テンプレートの場所を指定します。
16. SSH サービスの名前

関連情報

- Where are the Configuration Files?
- Modules

2.5. CLOUD.CFG.D ディレクトリー

cloud-init は、ユーザーが提供および設定するディレクティブに対応します。通常、これらのディレクティブは **cloud.cfg.d** ディレクトリーに含まれています。

注記

cloud.cfg ファイル内でユーザーデータディレクティブを追加することでモジュールを設定できますが、**cloud.cfg** を未変更のままにすること (ベストプラクティス) をご検討ください。ディレクティブを **/etc/cloud/cloud.cfg.d** ディレクトリーに追加します。このディレクトリーにディレクティブを追加することで、今後の変更およびアップグレードを容易にすることができます。

ディレクティブを追加する方法は複数あります。ディレクティブは、見出し **#cloud-config** を含む ***.cfg** という名前のファイルに追加できます。通常、ディレクトリーには複数の ***cfg** ファイルが含まれます。ディレクティブを追加するオプションは他にもあります。たとえば、ユーザーデータスクリプトを追加できます。詳細は、User-Data Formats を参照してください。

関連情報

- Where are the Configuration Files?
- Cloud config examples

2.6. デフォルトの 05_LOGGING.CFG ファイル

05_logging.cfg ファイルは、**cloud-init** のログ情報を設定します。**/etc/cloud/cloud.cfg.d** ディレクトリーには、追加する他の **cloud-init** ディレクティブと共にこのファイルが含まれます。

cloud-init は、デフォルトで **05_logging.cfg** のロギング設定を使用します。Red Hat Enterprise Linux (RHEL) のファイルのデフォルトコンテンツは、以下のとおりです。

```
## This yaml formatted config file handles setting
## logger information.  The values that are necessary to be set
## are seen at the bottom.  The top '_log' are only used to remove
## redundancy in a syslog and fallback-to-file case.
##
## The 'log_cfgs' entry defines a list of logger configs
## Each entry in the list is tried, and the first one that
## works is used.  If a log_cfg list entry is an array, it will
## be joined with '\n'.
_log:
 - &log_base |
  [loggers]
  keys=root,cloudinit

  [handlers]
  keys=consoleHandler,cloudLogHandler

  [formatters]
  keys=simpleFormatter,arg0Formatter

  [logger_root]
  level=DEBUG
  handlers=consoleHandler,cloudLogHandler

  [logger_cloudinit]
  level=DEBUG
  qualname=cloudinit
  handlers=
  propagate=1

  [handler_consoleHandler]
  class=StreamHandler
  level=WARNING
  formatter=arg0Formatter
  args=(sys.stderr,)

  [formatter_arg0Formatter]
  format=%(asctime)s - %(filename)s[%(levelname)s]: %(message)s

  [formatter_simpleFormatter]
  format=[CLOUDINIT] %(filename)s[%(levelname)s]: %(message)s
 - &log_file |
  [handler_cloudLogHandler]
  class=FileHandler
  level=DEBUG
  formatter=arg0Formatter
  args=('/var/log/cloud-init.log',)
 - &log_syslog |
  [handler_cloudLogHandler]
  class=handlers.SysLogHandler
  level=DEBUG
  formatter=simpleFormatter
  args=("/dev/log", handlers.SysLogHandler.LOG_USER)
```

```
log_cfgs:
# Array entries in this list will be joined into a string
# that defines the configuration.
#
# If you want logs to go to syslog, uncomment the following line.
# - [ *log_base, *log_syslog ]
#
# The default behavior is to just log to a file.
# This mechanism that does not depend on a system service to operate.
 - [ *log_base, *log_file ]
# A file path can also be used.
# - /etc/log.conf

# This tells cloud-init to redirect its stdout and stderr to
# 'tee -a /var/log/cloud-init-output.log' so the user can see output
# there without needing to look on the console.
output: {all: '| tee -a /var/log/cloud-init-output.log'}
```

関連情報

- Logging

2.7. CLOUD-INIT /VAR/LIB/CLOUD ディレクトリーのレイアウト

cloud-init を最初に実行すると、インスタンスおよび **cloud-init** 設定に関する情報が含まれるディレクトリーレイアウトが作成されます。

ディレクトリーには、/**scripts**/**vendor** などのオプションのディレクトリーを追加できます。

以下は、**cloud-init** のサンプルディレクトリーレイアウトです。

```
/var/lib/cloud/
    - data/
       - instance-id
       - previous-instance-id
       - previous-datasource
       - previous-hostname
       - result.json
       - set-hostname
       - status.json
    - handlers/
    - instance
       - boot-finished
       - cloud-config.txt
       - datasource
       - handlers/
       - obj.pkl
       - scripts/
       - sem/
       - user-data.txt
       - user-data.txt.i
       - vendor-data.txt
       - vendor-data.txt.i
    - instances/
```

```
    f111ee00-0a4a-4eea-9c17-3fa164739c55/
      - boot-finished
      - cloud-config.txt
      - datasource
      - handlers/
      - obj.pkl
      - scripts/
      - sem/
      - user-data.txt
      - user-data.txt.i
      - vendor-data.txt
      - vendor-data.txt.i
  - scripts/
    - per-boot/
    - per-instance/
    - per-once/
    - vendor/
  - seed/
  - sem/
    - config_scripts_per_once.once
```

関連情報

- Directory layout

第3章 CLOUD-INIT の設定

本章では、**cloud-init** で最も一般的な設定タスクの例を紹介します。

cloud-init 設定では、**cloud.cfg** ファイルおよび **cloud.cfg.d** ディレクトリーへのディレクティブの追加を必要とすることがあります。あるいは、特定のデータソースでは、ユーザーデータファイルやメタデータファイルなどのファイルにディレクティブを追加する必要がある場合があります。データソースでは、ディレクティブの HTTP サーバーへのアップロードが必要な場合があります。データソースの要件を確認し、それに応じてディレクティブを追加します。

3.1. NOCLOUD データソースの CLOUD-INIT を含む仮想マシンの作成

このセクションでは、**cloud-init** を含む新しい仮想マシン (VM) を作成するためのサンプル手順を示します。この手順では、**meta-data** および **user-data** ファイルを作成します。

- **meta-data** ファイルには、インスタンスの詳細が含まれます。
- **user-data** には、ユーザーを作成し、アクセスを付与するための情報が含まれます。

次に、これらのファイルを新しい ISO イメージに追加し、KVM ゲストイメージから作成した新しい仮想マシンに ISO ファイルをアタッチします。このシナリオでは、データソースは NoCloud です。

手順

1. **cloudinitiso** という名前のディレクトリーを作成し、これに移動します。

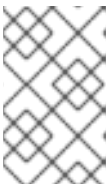
```
$ mkdir cloudinitiso
$ cd cloudinitiso
```

2. **meta-data** という名前のファイルを作成します。以下の情報をファイルに追加します。

```
instance-id: citest
local-hostname: citest-1
```

3. **user-data** という名前のファイルを作成します。ファイルに以下の情報を追加します。

```
#cloud-config
password: cilogon
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...fhHQ== sample@redhat.com
```



注記

user-data ファイルの最後の行は、SSH 公開鍵を参照します。**~/.ssh/id_rsa.pub** で SSH 公開鍵を検索します。このサンプル手順を行う場合は、行を変更して公開鍵の 1 つを含めます。

4. **genisoimage** コマンドを使用して、**user-data** および **meta-data** を含む ISO イメージを作成します。

```
# genisoimage -output ciiso.iso -volid cidata -joliet -rock user-data meta-data
```

```
I: -input-charset not specified, using utf-8 (detected in locale settings)
Total translation table size: 0
Total rockridge attributes bytes: 331
Total directory bytes: 0
Path table size(bytes): 10
Max brk space used 0
183 extents written (0 MB)
```

5. Red Hat カスタマーポータルから、**/var/lib/libvirt/images** ディレクトリーに KVM ゲストイメージをダウンロードします。

6. **virt-install** コマンドを使用して、KVM ゲストイメージから新しい仮想マシンを作成します。イメージへの添付として、作成した ISO イメージを含めます。

```
virt-install \
   --memory 4096 \
   --vcpus 4 \
   --name mytestcivm \
   --disk /var/lib/libvirt/images/rhel-8.1-x86_64-
kvm.qcow2,device=disk,bus=virtio,format=qcow2 \
   --disk /home/sample/cloudinitiso/ciiso.iso,device=cdrom \
   --os-type Linux \
   --os-variant rhel8.0 \
   --virt-type kvm \
   --graphics none \
   --import
```

7. **cloud-user** としてイメージにログインします。パスワードは **cilogon** です。

```
citest-1 login: cloud-user
Password:
[cloud-user@citest-1 ~]$
```

検証

- **cloud-init** ステータスを確認して、タスクが完了したことを確認します。

```
[cloud-user@citest-1 instance]$ cloud-init status
status: done
```

- **cloud-init** は、実行時に **/var/lib/cloud** の下の **cloud-init** ディレクトリーレイアウトを作成し、指定したディレクティブに基づいて特定のディレクトリーコンテンツを更新または変更します。
  たとえば、データソースファイルをチェックして、データソースが **NoCloud** であることを確認できます。

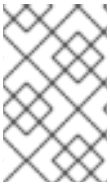
```
$ cd /var/lib/cloud/instance
$ cat datasource
DataSourceNoCloud: DataSourceNoCloud [seed=/dev/sr0][dsmode=net]
```

  cloud-init は、user-data を **/var/lib/cloud/instance/user-data.txt** にコピーします。

```
$cat user-data.txt
```

```
#cloud-config
password: cilogon
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...fhHQ== sample@redhat.com
```

これらはサンプルです。**cloud-init** ディレクトリーレイアウトには、もっと多くの情報が含まれます。



注記

OpenStack の場合は、インスタンスおよびイメージガイド に、**cloud-init** を使用してインスタンスを設定するための情報が含まれています。特定の手順は、Creating a customized instance を参照してください。

関連情報

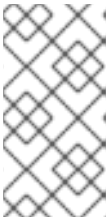
- NoCloud

3.2. CLOUD-INIT を使用してクラウドユーザーパスワードを期限切れにする

初回ログイン時に **cloud-user** パスワードを変更するよう **cloud-user** に強制することができます。パスワードを失効させるには、以下の手順を実行します。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。



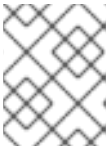
注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **chpasswd: {expire: False}** の行を **chpasswd: {expire: True}** に変更します。

```
#cloud-config
password: mypassword
chpasswd: {expire: True}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
```

これはパスワードを失効させます。なぜなら、**password** と **chpasswd** は特に指定がない限り、デフォルトのユーザーで動作するからです。



注記

これはグローバル設定です。**chpasswd** を **True** に設定すると、作成するすべてのユーザーが、ログイン時にパスワードを変更する必要があります。

3.3. CLOUD-INIT でのデフォルトユーザー名の変更

デフォルトのユーザー名は、**cloud-user** 以外のものに変更できます。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **user: <username>** の行を追加します。<username> は新しいデフォルトのユーザー名に置き換えます。

```
#cloud-config
user: username
password: mypassword
chpasswd: {expire: False}
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
```

3.4. CLOUD-INIT を使用した ROOT パスワードの設定

root パスワードを設定するには、ユーザーの一覧を作成します。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. ファイルの **chpasswd** セクションで、ユーザー一覧を作成します。以下の例に形式を示します。

注記

空白は重要です。ユーザー一覧のコロンの前後に空白を含めないでください。空白が含まれている場合、パスワードは空白を入れた設定となります。

```
#cloud-config
ssh_pwauth: True
ssh_authorized_keys:
  - ssh-rsa AAA...SDvz user1@yourdomain.com
  - ssh-rsa AAB...QTuo user2@yourdomain.com
chpasswd:
  list: |
     root:myrootpassword
     cloud-user:mypassword
  expire: False
```

注記

この方法を使用してユーザーパスワードを設定する場合は、本セクションの **すべてのパスワード** を設定する必要があります。

3.5. CLOUD-INIT を使用した RED HAT サブスクリプションの管理

rh_subscription ディレクティブを使用してシステムを登録できます。以下に例を示します。サブスクリプションごとに、ユーザーデータを編集します。

手順

以下の例では、**auto-attach** オプションと **service-level** オプションを使用しています。

- **rh_subscription** の下に **username** と **password** を追加して、**auto-attach** を **True** に設定し、**service-level** を **self-support** に設定します。

```
rh_subscription:
  username: sample@redhat.com
  password: 'mypassword'
  auto-attach: True
  service-level: self-support
```

注記

service-level オプションでは、**auto-attach** オプションを使用する必要があります。

以下の例では、**activation-key** オプションおよび **org** オプションを使用しています。

- **rh_subscription** の下に **activation key** と **org** の番号を追加し、**auto-attach** を **True** に設定します。

```
rh_subscription:
  activation-key: example_key
  org: 12345
  auto-attach: True
```

以下の例では、サブスクリプションプールを追加します。

- **rh_subscription** の下に、**username**、**password**、およびプール番号を追加します。

```
rh_subscription:
  username: sample@redhat.com
  password: 'password'
  add-pool: XYZ01234567
```

注記

このサンプルは、**subscription-manager attach --pool=XYZ01234567** コマンドに相当します。

以下の例では、サーバーのホスト名を **/etc/rhsm/rhsm.conf** ファイルに設定します。

- **rh_subscription** の下に **username**、**password**、**server-hostname** を追加し、**auto-attach** を **True** に設定します。

```
rh_subscription:
  username: sample@redhat.com
  password: 'password'
  server-hostname: test.example.com
  auto-attach: True
```

3.6. CLOUD-INIT を使用したユーザーおよびユーザーオプションの追加

users セクション でユーザーを作成し、説明します。セクションを変更して初期システム設定にユーザーをさらに追加でき、追加のユーザーオプションを設定できます。

users セクションを追加する場合、本セクションのデフォルトのユーザーオプションを設定する必要もあります。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

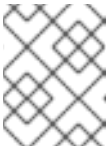
cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **users** セクションを追加または変更し、ユーザーを追加します。

   - **cloud-user** を、指定する他のユーザーと共に作成したデフォルトユーザーにするには、セクションの最初のエントリーとして **default** を追加することを確認してください。これが最初のエントリーでない場合は、**cloud-user** は作成されません。

   - デフォルトでは、**selinux-user** 値がない場合、ユーザーは **unconfined_u** とラベル付けされます。

```
#cloud-config
users:
  - default
  - name: user2
    gecos: User N. Ame
    selinux-user: staff_u
    groups: users,wheel
    ssh_pwauth: True
    ssh_authorized_keys:
      - ssh-rsa AA..vz user@domain.com
chpasswd:
  list: |
    root:password
    cloud-user:mypassword
    user2:mypassword2
  expire: False
```



注記

- この例では、ユーザー **user2** を 2 つのグループ (**users** と **wheel**) に配置します。

3.7. CLOUD-INIT を使用した最初の起動コマンドの実行

runcmd セクションおよび **bootcmd** セクションを使用して、起動および初期化中にコマンドを実行できます。

bootcmd セクションは、初期化プロセスの早い段階で実行され、デフォルトでは起動ごとに実行されます。**runcmd** セクションは、プロセスの最後の方で実行され、初回起動時および初期化中にのみ実行されます。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **bootcmd** および **runcmd** のセクションを追加します。**cloud-init** が実行するコマンドを含めます。

```
#cloud-config
users:
  - default
  - name: user2
    gecos: User N. Ame
    groups: users
chpasswd:
```

```
  list: |
    root:password
    fedora:myfedpassword
    user2:mypassword2
  expire: False
bootcmd:
 - echo New MOTD >> /etc/motd
runcmd:
 - echo New MOTD2 >> /etc/motd
```

3.8. CLOUD-INIT を使用した SUDOERS の追加

sudo および **groups** エントリーを **users** セクションに追加することで、ユーザーを sudoer として設定できます。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **sudo** エントリーを追加し、ユーザーアクセスを指定します。たとえば、**sudo: ALL=(ALL) NOPASSWD:ALL** は、ユーザーに無制限のユーザーアクセスを許可します。

3. **groups** エントリーを追加し、ユーザーを含むグループを指定します。

```
#cloud-config
users:
  - default
  - name: user2
    gecos: User D. Two
    sudo: ["ALL=(ALL) NOPASSWD:ALL"]
    groups: wheel,adm,systemd-journal
    ssh_pwauth: True
    ssh_authorized_keys:
      - ssh-rsa AA...vz user@domain.com
chpasswd:
  list: |
    root:password
    cloud-user:mypassword
    user2:mypassword2
  expire: False
```

3.9. CLOUD-INIT を使用した静的ネットワーク設定

メタデータに **network-interfaces** セクションを追加することで、**cloud-init** を使用したネットワーク設定を設定できます。

Red Hat Enterprise Linux は、**NetworkManager** を介してデフォルトのネットワークサービスを提供します。これは、動的ネットワークを制御および設定するデーモンで、利用可能な場合にはネットワークデバイスと接続が起動してアクティブな状態を維持します。**NetworkManager** の詳細は、NetworkManager の使用 を参照してください。

データソースがネットワーク設定を提供する場合があります。詳細は、**cloud-init** ドキュメントの Network Configuration Sources を参照してください。

cloud-init のネットワーク設定を指定しておらず、ネットワーク設定を無効にしていない場合、**cloud-init** は割り当てられているデバイスに接続があるかどうかを判別しようとします。接続されたデバイスを見つけると、インターフェイスで DHCP 要求を発行するネットワーク設定が生成されます。詳細は、**cloud-init** ドキュメントの Fallback Network Configuration セクションを参照してください。

手順

以下の例では、静的ネットワーク設定を追加します。

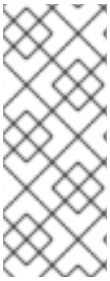
1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **network-interfaces** セクションを追加します。

```
network:
  version: 1
  config:
    - type: physical
      name: eth0
      subnets:
        - type: static
          address: 192.168.1.10/24
          gateway: 192.168.1.254
```



注記

メタデータに以下の情報を追加することで、ネットワーク設定を無効にすることができます。

```
network
  config: disabled
```

関連情報

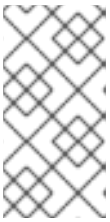
- Network Configuration
- NoCloud

3.10. CLOUD-INIT を使用した ROOT ユーザーのみの設定

root ユーザーがあり、他のユーザーはないようにユーザーデータを設定することができます。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。



注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **users** セクションに、ユーザー **root** のエントリーを作成します。
以下の簡単な例には、**name** オプションのみを持つ **users** セクションが含まれます。

```
users:
  - name: root
chpasswd:
  list: |
    root:password
  expire: False
```

3. オプションで、root ユーザーの SSH 鍵を設定します。

```
users:
  - name: root
    ssh_pwauth: True
    ssh_authorized_keys:
      - ssh-rsa AA..vz user@domain.com
```

3.11. CLOUD-INIT で CONTAINER-STORAGE-SETUP を使用したストレージの設定

write_files モジュール内で、**container-storage-setup** ユーティリティーを参照してストレージを設定できます。

手順

1. データソースの要件に応じて、user-data ファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg.d** ディレクトリーに追加します。

注記

cloud-init が、ユーザーディレクティブを含むファイルを認識できるように、すべてのユーザーディレクティブはファイルの最上部に **#cloud-config** が含まれます。**cloud.cfg.d** ディレクトリーにディレクティブを含める場合は、ファイル名を ***.cfg** とし、ファイルの最上部に常に **#cloud-config** を含めます。

2. **write_files** モジュールを追加または変更して、**container-storage-setup** ユーティリティーへのパスを追加します。

以下の例では、root 論理ボリュームのサイズを、デフォルトの 3 GB ではなく 6GB に設定します。

```
write_files:
  - path: /etc/sysconfig/docker-storage-setup
    permissions: 0644
    owner: root
    content: |
    ROOT_SIZE=6G
```

注記

RHEL 7.4 より前のバージョンでは、**container-storage-setup** は **docker-storage-setup** と呼ばれていました。RHEL 7.4 の時点で、ストレージに OverlayFS を使用している場合は、SELinux でのこのタイプのファイルシステムを Enforcing モードで使用できるようになりました。

3.12. CLOUD-INIT を使用したシステムロケールの変更

locale モジュールを使用して、システムの場所を設定できます。

手順

1. データソースの要件に応じて、メタデータファイルを開いて編集します。編集しない場合は、以下のディレクティブを **cloud.cfg** ファイルまたは **cloud.cfg.d** ディレクトリーに追加します。
2. 場所を指定して **locale** ディレクティブを追加します。以下の例では、**UTF-8** エンコーディングで **locale** を **ja_JP** (日本) に設定しています。

```
#cloud-config
locale: ja_JP.UTF-8
```

関連情報

- Locale

3.13. CLOUD-INIT およびシェルスクリプト

bootcmd または **runcmd** に、リスト値または文字列の値を追加できます。また、ユーザーデータ内にシェルスクリプトを指定することもできます。

- **bootcmd** または **runcmd** のリスト値を使用する場合は、各リスト項目は **execve** を使用して順番に実行されます。
- 文字列の値を使用する場合、文字列全体がシェルスクリプトとして実行されます。
- **cloud-init** を使用してシェルスクリプトを実行する場合は、**cloud-init** に **.yaml** ファイルを指定する代わりに、(シバン (#!) 機能を備えた) シェルスクリプトを指定できます。

シェルスクリプトを **bootcmd** および **runcmd** に配置する方法の例は、Run commands on first boot を参照してください。

3.14. CLOUD-INIT による設定ファイルの更新の阻止

バックアップイメージからインスタンスを作成または復元すると、インスタンス ID が変更されます。インスタンス ID の変更により、**cloud-init** が設定ファイルを更新する可能性があります。

バックアップから作成または復元する際に、**cloud-init** が特定の設定ファイルを更新しないようにするには、以下の手順を実行します。

手順

1. **/etc/cloud/cloud.cfg** ファイルを開いて編集します。
2. インスタンスの復元時に、**cloud-init** が更新しない設定をコメントアウトまたは削除します。たとえば、SSH 鍵ファイルの更新を回避するには、**cloud_init_modules** セクションから **-ssh** を削除します。

```
cloud_init_modules:
 - disk_setup
 - migrator
 - bootcmd
 - write-files
 - growpart
 - resizefs
 - set_hostname
 - update_hostname
 - update_etc_hosts
 - rsyslog
 - users-groups
 # - ssh
```

検証

cloud-init が更新した設定ファイルを確認することができます。確認するには、**/var/log/cloud/cloud-init.log** ファイルを調べます。更新されたファイルは、インスタンスの起動時に **Writing to** で始まるメッセージでログに記録されます。以下に例を示します。

```
2019-09-03 00:16:07,XXX - util.py[DEBUG]: Writing to /root/.ssh/authorized_keys - wb: [XXX] 554
bytes
2019-09-03 00:16:08,XXX - util.py[DEBUG]: Writing to /etc/ssh/sshd_config - wb: [XXX] 3905 bytes
```

3.15. CLOUD-INIT の実行後に KVM ゲストイメージから作成された仮想マシンの変更

本セクションでは、**cloud-init** を再度実行する前に、**cloud-init** 設定を変更する場合の手順の例を説明します。インストールされ、有効になっている **cloud-init** パッケージを含む仮想マシンを起動すると、**cloud-init** は仮想マシンの初回起動時にデフォルト状態で実行されます。

手順

1. 仮想マシンにログインします。
2. ディレクティブを追加または変更します。たとえば、**/etc/cloud** ディレクトリーの **cloud.cfg** ファイルを変更するか、ディレクティブを **/etc/cloud/cloud.cfg.d** ディレクトリーに追加します。

3. **cloud-init clean** コマンドを実行してディレクトリーをクリーンアップし、**cloud-init** が再実行できるようにします。root で以下のコマンドを実行して、仮想マシンをクリーンアップすることもできます。

```
`rm -Rf /var/lib/cloud/instances/*`
`rm -Rf /var/lib/cloud/instance`
`rm -Rf /var/lib/cloud/data/*`
```

注記

クリーニングしたイメージを新しいイメージとして保存し、そのイメージを複数の仮想マシンに使用できます。新しい仮想マシンは、更新された **cloud-init** 設定を使用して、**cloud-init** を実行します。

4. **cloud-init** を再実行するか、仮想マシンを再起動します。
cloud-init が再実行し、変更した設定を実装します。

3.16. CLOUD-INIT 実行後の特定データソースの仮想マシンの変更

本セクションでは、**cloud-init** を再度実行する前に、**cloud-init** 設定を変更する場合の手順の例を説明します。以下の手順では、例として OpenStack を使用しています。手順は、データソースによって異なる点に留意してください。

手順

1. OpenStack Platform のインスタンスを作成して起動します。OpenStack のインスタンス作成に関する詳細は、仮想マシンインスタンス を参照してください。この例では、仮想マシンには **cloud-init** が含まれており、これは仮想マシンの起動時に実行されます。

2. ディレクティブを追加または変更します。たとえば、OpenStack HTTP サーバー上に保管されている **user-data.file** ファイルを変更します。

3. 仮想マシンをクリーンアップします。以下のコマンドを root 権限で実行します。

```
`rm -rf /etc/resolv.conf /run/cloud-init`
`userdel -rf cloud-user`
`hostnamectl set-hostname localhost.localdomain`
`rm /etc/NetworkManager/conf.d/99-cloud-init.conf`
```

注記

クリーニングしたイメージを新しいイメージとして保存し、そのイメージを複数の仮想マシンに使用できます。新しい仮想マシンは、更新された **cloud-init** 設定を使用して、**cloud-init** を実行します。

4. **cloud-init** を再実行するか、仮想マシンを再起動します。
cloud-init が再実行し、設定変更を実装します。

3.17. CLOUD-INIT のトラブルシューティング

cloud-init の実行後にインスタンスをトラブルシューティングするには、設定ファイルとログファイルを調べます。問題を特定したら、インスタンスで **cloud-init** を再実行できます。

cloud-init コマンドを使用して、コマンドラインから **cloud-init** を実行できます。コマンド構文とオプションの引数およびサブコマンドの説明を表示するには、**cloud-init --help** コマンドを実行します。基本的な構文は以下のとおりです。

```
cloud-init [-h] [--version] [--file FILES] [--debug] [--force]
{init,modules,single,query,dhclient-hook,features,analyze,devel,collect-logs,clean,status}
```

以下の手順は、**cloud-init** の問題を特定するアイデアおよびプログラムを再実行するサンプルを提供しています。

手順

1. **cloud-init** 設定ファイルを確認します。
   a. **/etc/cloud/cloud.cfg** 設定ファイルを検証します。**cloud_init_modules**、**cloud_config_modules**、および **cloud_final_modules** に含まれるモジュールを確認します。
   b. **/etc/cloud/cloud.cfg.d** ディレクトリーで、ディレクティブ (***.cfg** files) を確認します。
2. 特定の問題の詳細については、**/var/log/cloud-init.log** ファイルおよび **/var/log/cloud-init-output.log** ファイルを確認してください。たとえば、root パーティションが自動的に拡張されないという問題があった場合は、**growpart** のログメッセージを確認してください。ファイルシステムが拡張されなかった場合は、ログメッセージで **resizefs** を確認します。以下に例を示します。

   ```
   # grep resizefs /var/log/cloud-init.log
   ```

   注記

   growpart は LVM をサポートしません。root パーティションが LVM をベースとしている場合は、root パーティションは初回起動時に自動的に拡張されません。

3. **cloud-init** を再実行します。サンプルシナリオを以下に示します。root でコマンドを実行します。
   - init モジュールのみで **cloud-init** を再実行します。

     ```
     /usr/bin/cloud-init -d init
     ```

   - 設定内のすべてのモジュールで **cloud-init** を再実行します。

     ```
     /usr/bin/cloud-init -d modules
     ```

   - **cloud-init** キャッシュを削除し、起動後に **cloud-init** を強制的に実行します。

     ```
     rm -rf /var/lib/cloud/* && /usr/bin/cloud-init -d init
     ```

   - 以下のコマンドを実行してディレクトリーをクリーンアップし、クリーンなインスタンスをシミュレートします。

     ```
     rm -Rf /var/lib/cloud/instances/*
     ```

```
rm -Rf /var/lib/cloud/instance
rm -Rf /var/lib/cloud/data/*
reboot
```

- 以下のコマンドを実行して、**cloud-init** を再実行します。

```
cloud-init init --local
cloud-init init
```

関連情報

- CLI Interface