Red Hat Enterprise Linux 9

高度な RHEL 9 インストールの実行

キックスタートを使用した RHEL のインストール

Last Updated: 2024-07-26

Red Hat Enterprise Linux 9 高度な RHEL 9 インストールの実行

キックスタートを使用した RHEL のインストール

法律上の通知

概要

キックスタートを使用して、高度な RHEL インストールを実行できます。キックスタートを使用すると、シンプルなテキストファイルを使用してインストールを自動化できます。

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 概要

Red Hat Enterprise Linux 9 は、より少ない労力でより迅速にワークロードを提供するために必要なツールを使用して、ハイブリッドクラウドのデプロイメントにまたがって、安定した安全で一貫性のある基盤を提供します。対応しているハイパーバイザー環境やクラウドプロバイダー環境にゲストとしてデプロイすることも、物理インフラストラクチャーにデプロイすることもできるため、アプリケーションは、主要なハードウェアアーキテクチャープラットフォームの革新的な機能を利用できます。

1.1. サポートされているアーキテクチャー

Red Hat Enterprise Linux では、次のアーキテクチャーに対応します。

- AMD アーキテクチャーおよび Intel 64 ビットアーキテクチャー
- 64 ビット ARM アーキテクチャー
- IBM Power Systems、リトルエンディアン
- 64 ビット IBM Z アーキテクチャー

注記

IBM Power Server へのインストール手順は、**IBM installation documentation** を参照してください。システムで RHEL のインストールがサポートされていることを確認するには、https://catalog.redhat.com および https://access.redhat.com/articles/rhel-limits を参照してください。

1.2. インストールの用語

本セクションでは、Red Hat Enterprise Linux インストールの用語を説明します。概念が同じでも、その概念が使用されているのがアップストリームかダウンストリームかによって、用語が異なる可能性もあります。

Anaconda - Fedora、Red Hat Enterprise Linux、およびその他の派生製品に使用されるオペレーティングシステムインストーラーです。Anaconda は、Gtk ウィジェット (C で記述)、systemd ユニット、dracut ライブラリーなどの追加ファイルが含まれる、一連の Python モジュールおよびスクリプトです。これは、同時に、作成される (ターゲット) システムのパラメーターを設定するツールを形成します。**インストールプログラム** という用語は、この文書では **Anaconda** のインストールに関する機能を指しています。

第2章 インストール方法

要件に応じて、複数の方法で Red Hat Enterprise Linux をインストールできます。以下のセクションを参照して、要件に最適なインストール方法を判断してください。

2.1. 利用可能なインストール方法

Red Hat Enterprise Linux は、以下のいずれかの方法でインストールできます。

- GUI ベースのインストール
- システムまたはクラウドイメージベースのインストール
- 高度なインストール

GUI ベースのインストール

以下の GUI ベースのインストール方法から選択できます。

- **カスタマーポータルから ISO イメージを使用した RHEL のインストール** - カスタマーポータルから **DVD ISO** イメージファイルをダウンロードして Red Hat Enterprise Linux をインストールします。登録は、インストールの完了後に行われます。このインストール方法は、GUI およびキックスタートで対応しています。
- **コンテンツ配信ネットワークから RHEL の登録およびインストール** - すべてのシステムを登録し、サブスクリプションを割り当て、コンテンツ配信ネットワーク (CDN) から Red Hat Enterprise Linux をインストールします。このインストール方法は、**Boot ISO** イメージファイルおよび **DVD ISO** イメージファイルに対応します。ただし、Boot ISO イメージファイルのインストールソースのデフォルトは CDN であるため、**Boot ISO** イメージファイルが推奨されます。システムの登録後、インストーラーは CDN からパッケージをダウンロードしてインストールします。このインストール方法は、キックスタートでも対応しています。
- **VNC を使用してリモートの RHEL インストールを実行** - RHEL インストールプログラムには、Direct と Connect の 2 つの Virtual Network Computing (VNC) インストールモードがあります。接続が確立されると、2 つのモードに違いはありません。選択するモードは、環境によって異なります。
- **PXE を使用してネットワークから RHEL をインストール:** PXE (Preboot eXecution Environment) を使用するネットワークインストールでは、インストールサーバーへのアクセスがあるシステムに、Red Hat Enterprise Linux をインストールできます。ネットワークインストールには、少なくとも 2 つのシステムが必要です。

システムまたはクラウドイメージベースのインストール

システムまたはクラウドイメージベースのインストール方法は、仮想環境およびクラウド環境でのみ使用できます。システムまたはクラウドイメージベースのインストールを実行するには、Red Hat Image Builder を使用します。Image Builder は、クラウドデプロイメントのシステムイメージを含む、Red Hat Enterprise Linux のカスタマイズされたシステムイメージを作成します。

Image Builder を使用して RHEL をインストールする方法の詳細は、Composing a customized RHEL system image を参照してください。

高度なインストール

以下の高度なインストール方法から選択できます。

- **キックスタートを使用した自動 RHEL インストールの実行:** キックスタートは、ファイルの要件

と設定をすべて指定して、オペレーティングシステムのインストールに役立つ自動化されたプロセスです。キックスタートファイルには、RHEL インストールオプション (タイムゾーン、ドライブパーティション、インストールするパッケージなど) が含まれます。事前に準備したキックスタートファイルを使用すると、ユーザーによる操作を必要とせずにインストールが完了します。これは、一度に多数のシステムに Red Hat Enterprise Linux をデプロイする場合に便利です。

- **コンテンツ配信ネットワークから RHEL の登録およびインストール** - すべてのアーキテクチャーに、コンテンツ配信ネットワーク (CDN) から Red Hat Enterprise Linux を登録してインストールします。登録は、インストールパッケージが CDN からダウンロードされてから、インストールされるまでの間に行われます。このインストール方法は、グラフィカルユーザーインターフェイスおよびキックスタートで対応しています。

パート I. キックスタートを使用した自動インストールの実行

第3章 キックスタートインストールの基礎

以下は、キックスタートの基本情報と、それを使用して Red Hat Enterprise Linux のインストールを自動化する方法を説明します。

3.1. キックスタートを使用したインストールの概要

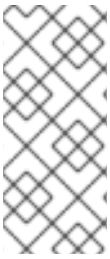
キックスタートは、RHEL インストールプロセスを部分的または完全に自動化する方法を提供します。

キックスタートファイルには、RHEL インストールオプションの一部またはすべてが含まれます。たとえば、タイムゾーン、ドライブのパーティション設定方法、インストールするパッケージなどです。事前に準備したキックスタートファイルを使用すると、ユーザーによる操作を必要としないインストールが可能になります。これは、Red Hat Enterprise Linux を多数のシステムに一度にデプロイする場合などに特に便利です。

キックスタートファイルによりソフトウェア選択の幅を広げることができます。グラフィカルインストールインターフェイスで Red Hat Enterprise Linux を手動でインストールする場合、ソフトウェアの選択は事前定義されている環境とアドオンの選択に限られます。キックスタートファイルを使用すると、パッケージを個別にインストールしたり、除外したりできます。

キックスタートファイルを 1 つのサーバーに置くことで、インストール時に各コンピューターが読み込むことができます。この方法を使用すると、1 つのキックスタートファイルで複数のマシンに Red Hat Enterprise Linux をインストールできるため、ネットワークおよびシステム管理者には理想的な方法になります。

キックスタートスクリプトおよびそのスクリプトの実行により生成されるログファイルは、インストール問題のデバッグの手助けとなるよう、新たにインストールしたシステムの **/tmp** ディレクトリーにすべて保存されます。インストールに使用されるキックスタートおよび Anaconda が生成した出力キックスタートは、ターゲットシステムの **/root** に保存され、キックスタートスクリプトレット実行のログは **/var/log/anaconda** に保存されます。



注記

キックスタートは、Red Hat Enterprise Linux の以前のバージョンではシステムをアップグレードするのに使用できました。Red Hat Enterprise Linux 7 以降では、この機能は削除されており、システムのアップグレードではなく、特殊なツールにより処理されます。Red Hat Enterprise Linux 9 へのアップグレードの詳細は、Upgrading from RHEL 8 to RHEL 9 および Considerations in adopting RHEL を参照してください。

3.2. 自動インストールのワークフロー

キックスタートを使用したインストールは、ローカルの DVD またはディスクを使用するか、NFS、FTP、HTTP、または HTTPS サーバーで実行できます。本セクションでは、キックスタートの使用方法の概要を説明します。

1. キックスタートファイルを作成します。手動で作成したり、手動インストール後に保存したキックファイルファイルをコピーしたり、オンライン生成ツールを使用してファイルを作成したりして、後で編集したりできます。Creating Kickstart files を参照してください。

2. リムーバブルメディア、ディスク、または HTTP (S) サーバー、FTP サーバー、または NFS サーバーに置いたインストールプログラムでキックスタートファイルを使用できるようにします。Making Kickstart files available to the installation program を参照してください。

3. インストール開始に使用する起動用メディアを作成します。インストールメディアの作成 および PXE を使用してネットワークからインストールするための準備 を参照してください。

4. インストールソースをインストールプログラムに利用できるようにします。Creating installation sources for Kickstart installations を参照してください。

5. ブートメディアおよびキックスタートファイルを使用して、インストールを開始します。Starting Kickstart installations を参照してください。

これは、キックスタートファイルが必須のコマンドおよびセクションをすべて含む場合に、インストールが自動的に行われます。必須部分が 1 つ以上欠けている場合、またはエラーが発生した場合は、インストールを手動で行う必要があります。

第4章 キックスタートファイルの作成

次の方法を使用してキックスタートファイルを作成できます。

- オンラインのキックスタート設定ツールを使用する。
- 手動インストールのログとして作成したキックスタートファイルをコピーする。
- キックスタートファイル全体を手動で書き込む。
- Red Hat Enterprise Linux 9 インストール用に Red Hat Enterprise Linux 8 キックスタートファイルを変換します。
  変換ツールの詳細は、Kickstart generator lab を参照してください。
- 仮想環境およびクラウド環境では、Image Builder を使用してカスタムシステムイメージを作成します。

一部の詳細なインストールオプションは、キックスタートファイルを手動で編集しないと設定できないことに注意してください。

4.1. キックスタート設定ツールを使用したキックスタートファイルの作成

Red Hat カスタマーポータルのアカウントをお持ちの場合は、カスタマーポータルで提供している Labs の Kickstart Generator ツールを使用して、キックスタートファイルをオンラインで生成できます。このツールは基本的な設定を段階的に説明し、作成したキックスタートファイルのダウンロードを可能にします。

前提条件

- Red Hat カスタマーポータルアカウントとアクティブな Red Hat サブスクリプションを持っている。

手順

1. Lab で提供されている Kickstart Generator の情報は https://access.redhat.com/labsinfo/kickstartconfig を参照してください。
2. 見出しの左にある **Go to Application** ボタンをクリックし、次のページが読み込まれるのを待ちます。
3. ドロップダウンメニューで **Red Hat Enterprise Linux 9**を選択し、ページが更新するのを待ちます。
4. フォーム内のフィールドを使用して、インストールするシステムを記述します。
   フォームの左側にあるリンクを使用すれば、フォームのセクション間をすばやく移動できます。
5. 生成されたキックスタートファイルをダウンロードするには、ページの先頭に戻り、赤色の **Download** ボタンをクリックします。
   Web ブラウザーによりファイルが保存されます。

4.2. 手動インストールを実行したキックスタートファイルの作成

キックスタートファイルの作成方法としては、Red Hat Enterprise Linux の手動インストールにより作成されたファイルを使用することが推奨される方法となります。インストールが完了すると、インス

トール中に選択したものがすべて、インストール済みシステムの **/root/** ディレクトリーに置かれているキックスタートファイル **anaconda-ks.cfg** に保存されます。このファイルを使用して、以前とまったく同じ方法でインストールを行えます。または、このファイルをコピーして必要な変更を加え、その後のインストールで使用することもできます。

手順

1. RHEL をインストールします。詳細は、Performing a standard RHEL 9 installation を参照してください。
   インストール時に、管理者権限を持つユーザーを作成します。
2. インストール済みシステムでインストールを完了し、再起動します。
3. 管理者アカウントでシステムにログインします。
4. **/root/anaconda-ks.cfg** ファイルを、任意の場所にコピーします。

重要

ファイルには、ユーザーとパスワードの情報が含まれます。

- 端末内のファイルの内容を表示するには、次のコマンドを実行します。

```
# cat /root/anaconda-ks.cfg
```

  出力をコピーして、別のファイルに選択を保存できます。

- 別の場所にファイルをコピーするには、ファイルマネージャーを使用します。root 以外のユーザーがそのファイルを読み込めるように、コピーしたファイルのアクセス権を忘れずに変更してください。

関連情報

- 標準的な RHEL 9 インストールの実行

4.3. 以前の RHEL インストールからキックスタートファイルを変換する

Kickstart Converter ツールを使用して、RHEL 7 Kickstart ファイルを RHEL 8 または 9 インストールで使用するために変換したり、RHEL 8 Kickstart ファイルを RHEL 9 で使用するために変換したりできます。ツールの詳細と、そのツールで RHEL キックスタートファイルを変換する方法は、https://access.redhat.com/labs/kickstartconvert/ を参照してください。

4.4. IMAGE BUILDER を使用したカスタムイメージの作成

Red Hat Image Builder を使用して、仮想デプロイメント用およびクラウドデプロイメント用にカスタマイズされたシステムイメージを作成できます。

Image Builder を使用したカスタムイメージの作成の詳細は、**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。

第5章 インストールプログラムでキックスタートファイルの準備

以下では、ターゲットシステムのインストールプログラムでキックスタートファイルを使用できるようにする方法を説明します。

5.1. ネットワークインストール用のポート

次の表は、ネットワークベースの各種インストールにファイルを提供するためにサーバーで開く必要があるポートの一覧です。

表5.1 ネットワークインストール用のポート

使用プロトコル	開くべきポート
HTTP	80
HTTPS	443
FTP	21
NFS	2049、111、20048
TFTP	69

関連情報

- ネットワークのセキュリティー保護

5.2. NFS サーバーでキックスタートファイルの準備

この手順では、キックスタートスクリプトファイルを NFS サーバーに格納する方法を説明します。この方法を使用すると、キックスタートファイルに物理メディアを使用することなく、1 つのソースから複数のシステムをインストールできます。

前提条件

- ローカルネットワーク上の Red Hat Enterprise Linux 9 を使用するサーバーへの管理者レベルのアクセス権限がある。
- インストールするシステムがサーバーに接続できる。
- サーバー上のファイアウォールがインストール先のシステムからの接続を許可している。

手順

1. root で以下のコマンドを実行して、**nfs-utils** パッケージをインストールします。

```
# dnf install nfs-utils
```

2. キックスタートファイルを、NFS サーバーのディレクトリーにコピーします。

3. テキストエディターで **/etc/exports** ファイルを開き、以下の構文の行を追加します。

```
/exported_directory/ clients
```

4. **/exported_directory/** を、キックスタートファイルを保存しているディレクトリーのフルパスに置き換えます。**clients** の代わりに、この NFS サーバーからインストールするコンピューターのホスト名または IP アドレス、すべてのコンピューターが ISO イメージにアクセスするためのサブネットワーク、またはネットワークアクセスのあるコンピューターが NFS サーバーにアクセスして ISO イメージを使用できるようにする場合はアスタリスク記号 (*) を使用します。このフィールドの形式に関する詳細は、**exports(5)** の man ページを参照してください。**/rhel9-install/** ディレクトリーを、すべてのクライアントに対する読み取り専用として使用できるようにする基本設定は次のようになります。

```
/rhel9-install *
```

5. **/etc/exports** ファイルを保存して、テキストエディターを終了します。

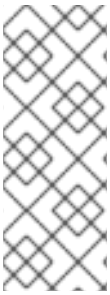
6. nfs サービスを起動します。

```
# systemctl start nfs-server.service
```

/etc/exports ファイルに変更を加える前にサービスを稼働していた場合は、以下のコマンドを実行して、稼働中の NFS サーバーで設定を再ロードします。

```
# systemctl reload nfs-server.service
```

キックスタートファイルは NFS 経由でアクセス可能になり、インストールに使用できるようになりました。



注記

キックスタートソースを指定する場合は、プロトコルに **nfs:** を使用して、サーバーのホスト名または IP アドレス、コロン記号 (**:**)、およびそのファイルを保存しているディレクトリーを指定します。たとえば、サーバーのホスト名が **myserver.example.com** で、そのファイルを **/rhel9-install/my-ks.cfg** に保存した場合、指定するインストールソースの起動オプションは **inst.ks=nfs:myserver.example.com:/rhel9-install/my-ks.cfg** となります。

関連情報

- PXE を使用してネットワークからインストールするための準備

5.3. HTTP サーバーまたは HTTPS サーバーで使用できるキックスタートファイルの準備

この手順では、キックスタートスクリプトファイルを HTTP サーバーまたは HTTPS サーバーに格納する方法を説明します。この方法を使用すると、キックスタートファイルに物理メディアを使用することなく、1 つのソースから複数のシステムをインストールできます。

前提条件

- ローカルネットワーク上の Red Hat Enterprise Linux 9 を使用するサーバーへの管理者レベルのアクセス権限がある。

- インストールするシステムがサーバーに接続できる。
- サーバー上のファイアウォールがインストール先のシステムからの接続を許可している。

手順

1. キックスタートファイルを HTTP に保存するには、**httpd** パッケージをインストールします。

```
# dnf install httpd
```

HTTPS にキックスタートファイルを保存するには、**httpd** パッケージおよび **mod_ssl** パッケージをインストールします。

```
# dnf install httpd mod_ssl
```

警告

Apache Web サーバー設定で SSL セキュリティーが有効になっている場合は、TLSv1 プロトコルのみが有効で、SSLv2 と SSLv3 は無効になっていることを確認してください。POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受けないようにするためです。詳細は https://access.redhat.com/solutions/1232413 を参照してください。

重要

自己署名証明書付きの HTTPS サーバーを使用する場合は、**inst.noverifyssl** オプションを指定してインストールプログラムを起動する必要があります。

2. **/var/www/html/** ディレクトリーのサブディレクトリーに、HTTP(S) サーバーへのキックスタートファイルをコピーします。
3. httpd サービスを起動します。

```
# systemctl start httpd.service
```

キックスタートファイルはアクセス可能になり、インストールとして使用できるようになりました。

注記

キックスタートファイルの場所を指定する場合は、プロトコルに **http://** または **https://** を使用して、サーバーのホスト名または IP アドレス、キックスタートファイルのパス (HTTP サーバーの root への相対パス) を指定します。たとえば、HTTP を使用して、サーバーのホスト名が **myserver.example.com** で、キックスタートファイルを **/var/www/html/rhel9-install/my-ks.cfg** にコピーした場合、指定するインストールソースは **http://myserver.example.com/rhel9-install/my-ks.cfg** となります。

関連情報

- Deploying Web Servers and Proxies
- Configuring and using Database Servers

5.4. FTP サーバーでキックスタートファイルの準備

この手順では、キックスタートスクリプトファイルを FTP サーバーに格納する方法を説明します。この方法を使用すると、キックスタートファイルに物理メディアを使用することなく、1 つのソースから複数のシステムをインストールできます。

前提条件

- ローカルネットワーク上の Red Hat Enterprise Linux 9 を使用するサーバーへの管理者レベルのアクセス権限がある。
- インストールするシステムがサーバーに接続できる。
- サーバー上のファイアウォールがインストール先のシステムからの接続を許可している。

手順

1. root で以下のコマンドを実行して、**vsftpd** パッケージをインストールします。

```
# dnf install vsftpd
```

2. 必要に応じて、**/etc/vsftpd/vsftpd.conf** 設定ファイルをテキストエディターで開いて編集します。
   a. **anonymous_enable=NO** の行を **anonymous_enable=YES** に変更します。
   b. **write_enable=YES** の行を **write_enable=NO** に変更します。
   c. **pasv_min_port=min_port** と **pasv_max_port=max_port** の行を追加します。**min_port** と **max_port** を、FTP サーバーがパッシブモードで使用するポート番号の範囲 (**10021** と **10031** など) に置き換えます。
   このステップは、各種のファイアウォール/NAT 設定を採用するネットワーク環境に必要です。
   d. オプションで、カスタムの変更を設定に追加します。利用可能なオプションは、**vsftpd.conf(5)** の man ページを参照してください。この手順では、デフォルトのオプションが使用されていることを前提としています。

警告

vsftpd.conf ファイルで SSL/TLS セキュリティーを設定している場合は、TLSv1 プロトコルのみを有効にし、SSLv2 と SSLv3 は無効にしてください。POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受けないようにするためです。詳細は、https://access.redhat.com/solutions/1234773 を参照してください。

3. サーバーのファイアウォールを設定します。

   a. ファイアウォールを有効にします。

   ```
   # systemctl enable firewalld
   # systemctl start firewalld
   ```

   b. 直前の手順の FTP ポートおよびポート範囲のファイアウォールで有効にします。

   ```
   # firewall-cmd --add-port min_port-max_port/tcp --permanent
   # firewall-cmd --add-service ftp --permanent
   # firewall-cmd --reload
   ```

   min_port-max_port を、**/etc/vsftpd/vsftpd.conf** 設定ファイルに入力したポート番号に置き換えます。

4. **/var/ftp/** ディレクトリーまたはそのサブディレクトリーに、FTP サーバーへのキックスタートファイルをコピーします。

5. 正しい SELinux コンテキストとアクセスモードがファイルに設定されていることを確認してください。

   ```
   # restorecon -r /var/ftp/your-kickstart-file.ks
   # chmod 444 /var/ftp/your-kickstart-file.ks
   ```

6. **vsftpd** サービスを開始します。

   ```
   # systemctl start vsftpd.service
   ```

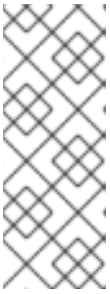
   /etc/vsftpd/vsftpd.conf ファイルを変更する前から、このサービスがすでに実行されていた場合は、サービスを再起動して必ず編集後のファイルを読み込ませてください。

   ```
   # systemctl restart vsftpd.service
   ```

   vsftpd サービスを有効にして、システムの起動プロセス時に開始するようにします。

   ```
   # systemctl enable vsftpd
   ```

   キックスタートファイルはアクセス可能になり、同じネットワークのシステムからのインストールとして使用できるようになりました。



注記

インストールソースを設定するには、プロトコルに **ftp://** を使用して、サーバーのホスト名または IP アドレス、キックスタートファイルのパス (FTP サーバーの root への相対パス) を指定します。たとえば、サーバーのホスト名が **myserver.example.com** で、ファイルを **/var/ftp/my-ks.cfg** にコピーした場合、指定するインストールソースは **ftp://myserver.example.com/my-ks.cfg** となります。

5.5. ローカルボリュームでキックスタートファイルの準備

この手順では、インストールするシステムのボリュームにキックスタートスクリプトファイルを保存する方法を説明します。この方法により、別のシステムは必要なくなります。

前提条件

- USB スティックなど、インストールするマシンに移動できるドライブがある。
- ドライブには、インストールプログラムで読み取ることができるパーティションが含まれている。対応しているタイプは、**ext2**、**ext3**、**ext4**、**xfs**、および **fat** です。
- ドライブがシステムに接続されており、そのボリュームがマウントされている。

手順

1. ボリューム情報のリストを表示し、キックスタートファイルをコピーするボリュームの UUID をメモします。

```
# lsblk -l -p -o name,rm,ro,hotplug,size,type,mountpoint,uuid
```

2. ボリュームのファイルシステムに移動します。
3. このファイルシステムにキックスタートファイルをコピーします。
4. **inst.ks=** オプションを使用して後で使用する文字列をメモしておきます。この文字列の形式は **hd:UUID=volume-UUID:path/to/kickstart-file.cfg** です。パスは、ファイルシステムシステム階層の / (root) ではなく、ファイルシステムの root に相対的になります。 **volume-UUID** を、上記の UUID に置き換えます。
5. ドライブボリュームのマウントをすべて解除します。

```
# umount /dev/xyz ...
```

スペースで区切って、コマンドにすべてのボリュームを追加します。

5.6. 自動読み込みのローカルボリュームでキックスタートファイルを使用可能に

特別な名前が付けられたキックスタートファイルを、インストールするシステムで特別な名前が付けられたボリュームの root に置くことができます。これにより、別のシステムが必要なくなり、インストールプログラムが自動的にファイルを読み込むことができるようになります。

前提条件

- USB スティックなど、インストールするマシンに移動できるドライブがある。
- ドライブには、インストールプログラムで読み取ることができるパーティションが含まれている。対応しているタイプは、**ext2**、**ext3**、**ext4**、**xfs**、および **fat** です。
- ドライブがシステムに接続されており、そのボリュームがマウントされている。

手順

1. キックスタートファイルをコピーするボリューム情報をリスト表示します。

```
# lsblk -l -p
```

2. ボリュームのファイルシステムに移動します。
3. このファイルシステムの root にキックスタートファイルをコピーします。
4. キックスタートファイルの名前を **ks.cfg** に変更します。
5. ボリュームの名前を **OEMDRV** に変更します。
   - **ext2**、**ext3**、および **ext4** のファイルシステムの場合:

     ```
     # e2label /dev/xyz OEMDRV
     ```

   - XFS ファイルシステムの場合:

     ```
     # xfs_admin -L OEMDRV /dev/xyz
     ```

   /dev/xyz を、ボリュームのブロックデバイスのパスに置き換えます。
6. ドライブボリュームのマウントをすべて解除します。

   ```
   # umount /dev/xyz ...
   ```

   スペースで区切って、コマンドにすべてのボリュームを追加します。

第6章 キックスタートインストール用のインストールソースの作成

本セクションでは、必要なリポジトリーおよびソフトウェアパッケージを含む DVD ISO イメージを使用して、Boot ISO イメージのインストールソースを作成する方法を説明します。

6.1. インストールソースの種類

最小限のブートイメージには、以下のいずれかのインストールソースを使用できます。

- **DVD:** DVD に DVD ISO イメージを書き込みます。DVD はインストールソース (ソフトウェアパッケージソース) として自動的に使用されます。
- **ディスクまたは USB ドライブ**:DVD ISO イメージをディスクにコピーして、ドライブからソフトウェアパッケージをインストールするように、インストールプログラムを設定します。USB ドライブを使用する場合は、インストールを開始する前に、USB ドライブがシステムに接続されていることを確認してください。インストールプログラムは、インストールの開始後にメディアを検出することができません。
  - **ディスクの制限**: ディスクの DVD ISO イメージは、インストールプログラムがマウントできるファイルシステムを使用しているパーティションに置く必要があります。対応するファイルシステムは、**xfs**、**ext2**、**ext3**、**ext4**、および **vfat (FAT32)** となります。

警告

Microsoft Windows システムでは、ディスクをフォーマットする際に使用されるデフォルトのファイルシステムは NTFS です。exFAT ファイルシステムも利用できます。ただし、このファイルシステムは、いずれもインストール時に変更することができません。Microsoft Windows でインストールソースとしてディスクまたは USB ドライブを作成している場合は、ドライブを FAT32 としてフォーマットするようにしてください。FAT32 ファイルシステムは、4 GiB を超えるファイルを保存できません。

Red Hat Enterprise Linux 9 では、ローカルディスクのディレクトリーからインストールできます。これを行うには、DVD ISO イメージの内容をディスクのディレクトリーにコピーし、ISO イメージの代わりに、そのディレクトリーをインストールソースとして指定します。たとえば、**inst.repo=hd:<device>:<path to the directory>** です。

- **ネットワーク経由:** DVD ISO イメージまたはインストールツリー (DVD ISO イメージから抽出したコンテンツ) をネットワーク上の場所にコピーし、次のプロトコルを使用して、ネットワーク経由でインストールを実行します。
  - **NFS:** DVD ISO イメージは、ネットワークファイルシステム (NFS) 共有にあります。
  - **HTTPS、HTTP、または FTP** - インストールツリーは、HTTP、HTTPS、または FTP 経由でアクセス可能なネットワーク上にあります。

6.2. ネットワークインストール用のポート

次の表は、ネットワークベースの各種インストールにファイルを提供するためにサーバーで開く必要があるポートの一覧です。

表6.1 ネットワークインストール用のポート

使用プロトコル	開くべきポート
HTTP	80
HTTPS	443
FTP	21
NFS	2049、111、20048
TFTP	69

関連情報

- ネットワークのセキュリティー保護

6.3. NFS サーバーへのインストールソースの作成

この方法を使用して、物理メディアに接続しなくても、1つのソースから複数のシステムをインストールできます。

前提条件

- Red Hat Enterprise Linux 9 を搭載したサーバーへの管理者レベルのアクセス権限があり、このサーバーがインストールするシステムと同じネットワーク上にある。
- Binary DVD イメージをダウンロードしている。詳しくは、インストール ISO イメージのダウンロード を参照してください。
- イメージファイルから、起動可能な CD、DVD、または USB デバイスを作成している。詳細は、インストールメディアの作成 を参照してください。
- ファイアウォールにより、インストールしようとしているシステムがリモートインストールソースにアクセスできることを確認している。詳細については、ネットワークベースのインストール用のポート を参照してください。

手順

1. **nfs-utils** パッケージをインストールします。

```
# dnf install nfs-utils
```

2. DVD ISO イメージを、NFS サーバーのディレクトリーにコピーします。
3. テキストエディターで **/etc/exports** ファイルを開き、以下の構文の行を追加します。

```
/exported_directory/ clients
```

- **/exported_directory/** を、ISO イメージが含まれるディレクトリーのフルパスに置き換えます。
- **clients** を次のいずれかに置き換えます。
  - ターゲットシステムのホスト名または IP アドレス
  - すべてのターゲットシステムが ISO イメージへのアクセスに使用できるサブネットワーク
  - NFS サーバーへのネットワークアクセスを持つすべてのシステムが ISO イメージを使用できるようにするためのアスタリスク記号 (*)

  このフィールドの形式に関する詳細は、**exports(5)** の man ページを参照してください。

  たとえば、**/rhel9-install/** ディレクトリーを、すべてのクライアントに対する読み取り専用として使用できるようにする基本設定は次のようになります。

```
/rhel9-install *
```

4. **/etc/exports** ファイルを保存して、テキストエディターを終了します。
5. nfs サービスを起動します。

```
# systemctl start nfs-server.service
```

/etc/exports ファイルを変更する前に サービスが稼働していた場合は、NFS サーバーの設定をリロードします。

```
# systemctl reload nfs-server.service
```

ISO イメージは、NFS 経由でアクセス可能になり、インストールソースとして使用できるようになりました。

注記

インストールソースを設定するには、プロトコルに **nfs:** を使用し、サーバーのホスト名または IP アドレス、コロン記号 **(:)**、および ISO イメージを保存しているディレクトリーを指定します。たとえば、サーバーのホスト名が **myserver.example.com** で、ISO イメージを **/rhel9-install/** に保存した場合、指定するインストールソースは **nfs:myserver.example.com:/rhel9-install/** となります。

6.4. HTTP または HTTPS を使用するインストールソースの作成

インストールツリー (DVD ISO イメージから抽出したコンテンツと、有効な **.treeinfo** ファイル含むディレクトリー) を使用したネットワークベースのインストール用のインストールソースを作成できます。インストールソースには、HTTP、または HTTPS でアクセスします。

前提条件

- Red Hat Enterprise Linux 9 を搭載したサーバーへの管理者レベルのアクセス権限があり、このサーバーがインストールするシステムと同じネットワーク上にある。

- Binary DVD イメージをダウンロードしている。詳しくは、インストール ISO イメージのダウンロード を参照してください。
- イメージファイルから、起動可能な CD、DVD、または USB デバイスを作成している。詳細は、インストールメディアの作成 を参照してください。
- ファイアウォールにより、インストールしようとしているシステムがリモートインストールソースにアクセスできることを確認している。詳細については、ネットワークベースのインストール用のポート を参照してください。
- **httpd** パッケージがインストールされている。
- **https** インストールソースを使用すると、**mod_ssl** パッケージがインストールされます。

警告

Apache Web サーバー設定で SSL セキュリティーが有効になっている場合は、TLSv1.3 プロトコルを有効にすることが推奨されます。デフォルトでは、TLSv1.2 (LEGACY) が有効になっています。

重要

自己署名証明書付きの HTTPS サーバーを使用する場合は、**noverifyssl** オプションを指定してインストールプログラムを起動する必要があります。

手順

1. HTTP(S) サーバーに DVD ISO イメージをコピーします。
2. DVD ISO イメージをマウントするのに適したディレクトリーを作成します。以下はその例です。

```
# mkdir /mnt/rhel9-install/
```

3. DVD ISO イメージをディレクトリーにマウントします。

```
# mount -o loop,ro -t iso9660 /image_directory/image.iso /mnt/rhel9-install/
```

/image_directory/image.iso を DVD ISO イメージへのパスに置き換えます。

4. マウントされたイメージから、HTTP(S) サーバーの root にファイルをコピーします。

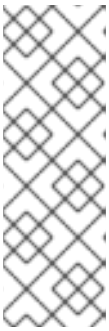
```
# cp -r /mnt/rhel9-install/ /var/www/html/
```

このコマンドでは、イメージのコンテンツが格納された **/var/www/html/rhel9-install/** ディレクトリーが作成されます。他の一部のコピー方法は、有効なインストールソースに必要な **.treeinfo** ファイルを省略する可能性があることに注意してください。この手順で示されているように、ディレクトリー全体に対して **cp** コマンドを入力すると、**.treeinfo** が正しくコピーされます。

5. **httpd** サービスを開始します。

```
# systemctl start httpd.service
```

これにより、インストールツリーにアクセスできるようになり、インストールソースとして使用できるようになります。



注記

インストールソースを設定するには、プロトコルに **http://** または **https://** を使用して、サーバーのホスト名または IP アドレス、および ISO イメージのファイルを保存するディレクトリー (HTTP サーバーの root への相対パス) を指定します。たとえば、HTTP を使用し、サーバーのホスト名が **myserver.example.com** で、イメージのファイルが **/var/www/html/rhel9-install/** にコピーされた場合、指定するインストールソースは **http://myserver.example.com/rhel9-install/** となります。

関連情報

- さまざまな種類のサーバーのデプロイメント

6.5. FTP を使用するインストールソースの作成

インストールツリー (DVD ISO イメージから抽出したコンテンツと、有効な **.treeinfo** ファイル含むディレクトリー) を使用したネットワークベースのインストール用のインストールソースを作成できます。インストールソースには、FTP を使用してアクセスします。

前提条件

- Red Hat Enterprise Linux 9 を搭載したサーバーへの管理者レベルのアクセス権限があり、このサーバーがインストールするシステムと同じネットワーク上にある。
- Binary DVD イメージをダウンロードしている。詳細は、起動可能なインストールメディアの作成 を参照してください。
- ファイアウォールにより、インストールしようとしているシステムがリモートインストールソースにアクセスできることを確認している。詳細については、ネットワークベースのインストール用のポート を参照してください。
- **vsftpd** パッケージがインストールされている。

手順

1. 必要に応じて、**/etc/vsftpd/vsftpd.conf** 設定ファイルをテキストエディターで開いて編集します。
   a. **anonymous_enable=NO** の行を **anonymous_enable=YES** に変更します。
   b. **write_enable=YES** の行を **write_enable=NO** に変更します。
   c. **pasv_min_port=<min_port>** および **pasv_max_port=<max_port>** の行を追加します。<min_port> と <max_port> を、FTP サーバーがパッシブモードで使用するポート番号の範囲 (**10021** と **10031** など) に置き換えます。
   この手順は、各種のファイアウォール/NAT 設定を採用するネットワーク環境で必要になる可能性があります。

d. オプション: カスタムの変更を設定に追加します。利用可能なオプションは、**vsftpd.conf(5)** の man ページを参照してください。この手順では、デフォルトのオプションが使用されていることを前提としています。

警告

vsftpd.conf ファイルで SSL/TLS セキュリティーを設定している場合は、TLSv1 プロトコルのみを有効にし、SSLv2 と SSLv3 は無効にしてください。POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受けないようにするためです。詳細は、https://access.redhat.com/solutions/1234773 を参照してください。

2. サーバーのファイアウォールを設定します。

a. ファイアウォールを有効にします。

```
# systemctl enable firewalld
```

b. ファイアウォールを起動します。

```
# systemctl start firewalld
```

c. 前の手順で設定した FTP ポートとポート範囲を許可するようにファイアウォールを設定します。

```
# firewall-cmd --add-port min_port-max_port/tcp --permanent
# firewall-cmd --add-service ftp --permanent
```

<**min_port**> と <**max_port**> を **/etc/vsftpd/vsftpd.conf** 設定ファイルに入力したポート番号に置き換えます。

d. ファイアウォールをリロードして、新しいルールを適用します。

```
# firewall-cmd --reload
```

3. DVD ISO イメージを FTP サーバーにコピーします。

4. DVD ISO イメージをマウントするのに適したディレクトリーを作成します。以下はその例です。

```
# mkdir /mnt/rhel9-install
```

5. DVD ISO イメージをディレクトリーにマウントします。

```
# mount -o loop,ro -t iso9660 /image-directory/image.iso /mnt/rhel9-install
```

/image-directory/image.iso を DVD ISO イメージへのパスに置き換えます。

6. マウントされたイメージから、FTP サーバーのルートにファイルをコピーします。

```
# mkdir /var/ftp/rhel9-install
# cp -r /mnt/rhel9-install/ /var/ftp/
```

これでイメージのコンテンツが格納された **/var/ftp/rhel9-install/** ディレクトリーが作成されます。一部のコピー方法は、有効なインストールソースに必要な **.treeinfo** ファイルを省略できることに注意してください。この手順で示されているように、ディレクトリー全体に対して **cp** コマンドを入力しても、**.treeinfo** が正しくコピーされます。

7. 正しい SELinux コンテキストとアクセスモードが、コピーされたコンテンツに設定されていることを確認します。

```
# restorecon -r /var/ftp/rhel9-install
# find /var/ftp/rhel9-install -type f -exec chmod 444 {} \;
# find /var/ftp/rhel9-install -type d -exec chmod 755 {} \;
```

8. **vsftpd** サービスを開始します。

```
# systemctl start vsftpd.service
```

/etc/vsftpd/vsftpd.conf ファイルを変更する前から、このサービスがすでに実行されていた場合は、サービスを再起動して必ず編集後のファイルを読み込ませてください。

```
# systemctl restart vsftpd.service
```

vsftpd サービスを有効にして、システムの起動プロセス時に開始するようにします。

```
# systemctl enable vsftpd
```

これにより、インストールツリーにアクセスできるようになり、インストールソースとして使用できるようになります。

注記

インストールソースを設定するには、プロトコルに **ftp://** を使用して、サーバーのホスト名または IP アドレス、および ISO イメージのファイルを保存するディレクトリー (FTP サーバーの root への相対パス) を指定します。たとえば、サーバーのホスト名が **myserver.example.com** で、イメージからコピーしたファイルを **/var/ftp/rhel9-install/** に置いた場合、指定するインストールソースは **ftp://myserver.example.com/rhel9-install/** となります。

第7章 KERNEL-64K を使用した ARM への RHEL のインストール

デフォルトでは、RHEL 9 は 4k ページサイズをサポートするカーネルとともに配布されます。この 4k カーネルは、スペース、電力、コストの制約により 64k ページカーネルの使用が現実的ではない小規模な環境や小規模なクラウドインスタンスでメモリーを効率的に使用するには十分なものです。

重要

初回起動後に、OS を再インストールせずに 4k と 64k のページサイズのカーネルを切り替えることは推奨されません。

7.1. キックスタートを使用した ARM への KERNEL-64K のインストール

RHEL は、最適なパフォーマンスを得るために大規模な物理メモリー設定を必要とするワークロードをサポートする ARM64 ハードウェアアーキテクチャーを提供します。このような大規模なメモリー設定では、大きな MMU ページサイズ (64k) を使用する必要があります。

RHEL 9 のインストール時に、**kernel-64k** パッケージを選択して、64k ページサイズをサポートするカーネルを備えた RHEL をインストールできます。

手順

- キックスタートファイルの **%packages** セクションに、次のパッケージリストを追加します。

```
%packages
kernel-64k
-kmod-kvdo
-vdo
-kernel
%end
```

検証手順

- ページサイズを確認するには、インストールが完了してシステムが再起動された後、ターミナルを開いて次を実行します。

```
$ getconf PAGESIZE
65536
```

出力 **65536** は、64k カーネルが使用されていることを示します。

- スワップパーティションが有効になっていることを確認するには、次のように入力します。

```
$ free
               total        used        free      shared  buff/cache   available
Mem:        35756352     3677184    34774848       25792      237120    32079168
Swap:        6504384           0     6504384
```

total 列と free 列がゼロ以外の値です。これは、スワップが正常に有効になっていることを示します。

7.2. コマンドラインを使用した ARM への KERNEL-64K のインストール

デフォルトのカーネル (4k ページサイズをサポート) を使用して RHEL をすでにインストールしている場合は、インストール後にコマンドラインを使用して **kernel-64k** をインストールできます。

手順

1. root ユーザーとしてターミナルを開き、次のように入力します。

```
# dnf -y install kernel-64k
```

2. **kernel-64k** をデフォルトとして設定するには、次のように入力します。

```
# k=$(echo /boot/vmlinuz*64k)
# grubby --set-default=$k \
        --update-kernel=$k \
        --args="crashkernel=2G-:640M"
```

3. システムの起動順序を、デフォルトオプションとして RHEL を使用するように設定します。

   a. 現在の起動順序を取得します。以下に例を示します。

```
# efibootmgr
BootCurrent: 0000
Timeout: 5 seconds
BootOrder: 0003,0004,0001,0000,0002,0005
Boot0000\* Red Hat Enterprise Linux
```

   b. RHEL を優先するように起動順序を設定します。たとえば、前の手順の出力の場合は、次のコマンドを使用します。

```
# efibootmgr -o 0000,0001,0002,0003,0004,0005
```

4. システムを再起動します。

```
# reboot
```

5. オプション: 再起動後、4k カーネルを削除します。

```
# dnf erase kernel
```

   両方のバージョンを誤って保持すると、将来 **yum update** コマンドを使用してカーネルを更新したときに、4k カーネルがデフォルトになる可能性があります。

検証

- ページサイズを確認するには、ターミナルを開き、任意のユーザーとして次のコマンドを実行します。

```
$ getconf PAGESIZE
65536
```

  出力 **65536** は、64k カーネルが使用されていることを示します。

- スワップが有効であることを確認するには、次のように入力します。

```
$ free
              total        used        free      shared  buff/cache   available
Mem:       35756352     3677184    34774848       25792      237120    32079168
Swap:       6504384           0     6504384
```

total 列と free 列がゼロ以外の値です。これは、スワップが正常に有効になっていることを示します。

第8章 キックスタートインストールの開始

キックスタートインストールは、複数の方法で開始できます。

- 手動でインストールプログラムの起動メニューに入り、そこにキックスタートファイルを含むオプションを指定します。
- 自動的に PXE ブートで起動オプションを編集することもできます。
- 特定の名前を持つボリュームに、自動的にファイルを提供することもできます。

次のセクションでは、各メソッドの実行方法を説明します。

8.1. 手動でのキックスタートインストールの開始

本セクションでは、キックスタートを手動で起動する方法を説明します。この場合は、(**boot:** プロンプトで起動オプションを追加することで) ユーザーとの対話が必要になります。インストールシステムを起動する場合は、起動オプション **inst.ks=location** を使用します。location は、キックスタートファイルの場所に置き換えます。ブートオプションとブートプロンプトの形式を指定する正確な方法は、システムのアーキテクチャーによって異なります。詳細については、Boot options for RHEL installer ガイドを参照してください。

前提条件

- インストールするシステムからアクセスできる場所に、キックスタートファイルを用意しておきます。

手順

1. ローカルメディア (CD、DVD、USB フラッシュドライブなど) を使用してシステムを起動します。
2. 起動プロンプトで、必要な起動オプションを指定します。
   a. キックスタートファイルまたは必要なリポジトリーがネットワークの場所にある場合は、**ip=** オプションを使用したネットワークの設定が必要になる場合があります。インストーラーは、このオプションを使用せずに、デフォルトで DHCP プロトコルを使用するすべてのネットワークデバイスを設定しようとします。
   b. 起動オプション **inst.ks=** と、キックスタートファイルの場所を追加します。
   c. 必要なパッケージがインストールされるソフトウェアソースにアクセスするには **inst.repo=** オプションを追加しないといけない場合があります。このオプションを指定しないと、キックスタートファイルでインストールソースを指定する必要があります。
3. 追加した起動オプションを確認してインストールを開始します。
   これにより、キックスタートファイルで指定されているインストールオプションを使用したインストールが開始します。キックスタートファイルに問題がなく、必要なコマンドがすべて含まれていれば、この時点からインストールは完全に自動化で行われます。

注記

UEFI セキュアブートが有効になっているシステムに、Red Hat Enterprise Linux ベータ版リリースをインストールした場合は、システムの Machine Owner Key (MOK) リストにベータ版の公開鍵を追加します。

8.2. PXE を使用した自動キックスタートインストールの開始

AMD64、Intel 64、および 64 ビット ARM システム、ならびに IBM Power Systems サーバーでは、PXE サーバーを使用して起動する機能があります。PXE サーバーの設定時に、ブートローダー設定ファイルに起動オプションを追加できます。これにより、インストールを自動的に開始できるようになります。このアプローチにより、ブートプロセスを含めたインストールを完全に自動化できるようになります。

この手順は一般的な参照です。詳細な手順はシステムのアーキテクチャーによって異なります。すべてのオプションが、すべてのアーキテクチャーで使用できるわけではありません (たとえば、64 ビットの IBM Z で PXE ブートを使用することはできません)。

前提条件

- インストールするシステムからアクセスできる場所に、キックスタートファイルを用意しておきます。
- システムを起動してインストールを開始するために使用できる PXE サーバーが用意されています。

手順

1. PXE サーバー上でブートローダー設定ファイルを開き、**inst.ks=** 起動オプションを適切な行に追加します。ファイル名と構文は、システムのアーキテクチャーおよびハードウェアにより異なります。
   - BIOS が搭載される AMD64 システムおよび Intel 64 システムのファイル名は、デフォルトまたはシステムの IP アドレスをベースにしたもののいずれかになります。このケースでは、インストールエントリーにある append 行に、**inst.ks=** オプションを追加します。設定ファイルの append 行は以下のようになります。

     ```
     append initrd=initrd.img inst.ks=http://10.32.5.1/mnt/archive/RHEL-9/9.x/x86_64/kickstarts/ks.cfg
     ```

   - GRUB2 ブートローダーを使用しているシステム (UEFI ファームウェアが搭載されている AMD64、Intel 64、および 64 ビット ARM システム、ならびに IBM Power Systems サーバー) のファイル名は **grub.cfg** になります。このファイルのインストールエントリーに含まれる kernel 行に、**inst.ks=** オプションを追加します。設定ファイルの kernel 行の例を以下に示します。

     ```
     kernel vmlinuz inst.ks=http://10.32.5.1/mnt/archive/RHEL-9/9.x/x86_64/kickstarts/ks.cfg
     ```

2. ネットワークサーバーからインストールを起動します。
   これでキックスタートファイルで指定されているインストールオプションを使用したインストールが開始します。キックスタートファイルに問題がなく、必要なコマンドがすべて含まれていれば、インストールは完全に自動で行われます。

注記

UEFI セキュアブートが有効になっているシステムに、Red Hat Enterprise Linux ベータ版リリースをインストールした場合は、システムの Machine Owner Key (MOK) リストにベータ版の公開鍵を追加します。

8.3. ローカルボリュームを使用した自動キックスタートインストールの開始

特別にラベルが追加されたストレージボリュームで、特定の名前が付いたキックスタートファイルを置くことで、キックスタートインストールを開始できます。

前提条件

- ラベル **OEMDRV** で準備されたボリューム、およびそのルートに **ks.cfg** として存在するキックスタートファイルがあります。
- このボリュームを含むドライブは、インストールプログラムの起動時にシステムで使用できます。

手順

1. ローカルメディア (CD、DVD、USB フラッシュドライブなど) を使用してシステムを起動します。
2. 起動プロンプトで、必要な起動オプションを指定します。
   a. 必要なリポジトリーがネットワーク上にある場合は、**ip=** オプションを使用したネットワークの設定が必要になる場合があります。インストーラーは、このオプションを使用せずに、デフォルトで DHCP プロトコルを使用するすべてのネットワークデバイスを設定しようとします。
   b. 必要なパッケージがインストールされるソフトウェアソースにアクセスするには **inst.repo=** オプションを追加しないといけない場合があります。このオプションを指定しないと、キックスタートファイルでインストールソースを指定する必要があります。インストールソースの詳細は、インストールプログラム設定およびフロー制御のためのキックスタートコマンド を参照してください。
3. 追加した起動オプションを確認してインストールを開始します。
   インストールが開始し、キックスタートファイルが自動的に検出され、自動化されたキックスタートインストールを開始します。

注記

UEFI セキュアブートが有効になっているシステムに、Red Hat Enterprise Linux ベータ版リリースをインストールした場合は、システムの Machine Owner Key (MOK) リストにベータ版の公開鍵を追加します。

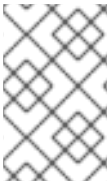
第9章 インストール中のコンソールとロギング

Red Hat Enterprise Linux インストーラーは、**tmux** 端末マルチプレクサーを使用して、メインのインターフェイスのほかに複数の画面を表示し、制御します。この画面は、それぞれ目的が異なり、インストールプロセス中に発生した問題をトラブルシューティングするのに使用できるさまざまなログを表示します。画面の 1 つでは、起動オプションまたはキックスタートコマンドを使用して明示的に無効にしない限り、**root** 権限で使用できる対話式シェルプロンプトを使用できます。

注記

一般的に、インストール関連の問題を診断する必要がなければ、デフォルトのグラフィカルインストール環境から、他の環境に移動する必要はありません。

端末マルチプレクサーは、仮想コンソール 1 で実行しています。インストール環境を、**tmux** に変更する場合は、**Ctrl**+**Alt**+**F1** を押します。仮想コンソール 6 で実行されているメインのインストールインターフェイスに戻るには、**Ctrl**+**Alt**+**F6** を押します。



注記

テキストモードのインストールを選択するには、仮想コンソール 1 (**tmux**) を開始し、その後にコンソール 6 に切り替えると、グラフィカルインターフェイスではなくシェルプロンプトが開きます。

tmux を実行しているコンソールには、利用可能な画面が 5 つあります。その内容と、キーボードショートカットは、以下の表で説明します。キーボードショートカットは 2 段階となっており、最初に **Ctrl**+**b** を押し、両方のキーを離してから、使用する画面で数字キーを押す必要があります。

また、**Ctrl**+**b n**、**Alt+ Tab**、および **Ctrl**+**b p** を使用して、次または前の **tmux** 画面に切り替えることもできます。

表9.1 利用可能な tmux 画面

ショートカット	内容
Ctrl+**b 1**	メインのインストールプログラム画面。テキストベースのプロンプト (テキストモードのインストール中もしくは VNC Direct モードを使用の場合) とデバッグ情報があります。
Ctrl+**b 2**	**root** 権限のある対話式シェルプロンプト。
Ctrl+**b 3**	インストールログ - **/tmp/anaconda.log** に保存されているメッセージを表示します。
Ctrl+**b 4**	ストレージログ - **/tmp/storage.log** に保存されているストレージデバイスおよび設定に関連するメッセージを表示します。
Ctrl+**b 5**	プログラムログ - **/tmp/program.log** に保存されている、インストールプロセス時に実行するユーティリティーのメッセージを表示します。

第10章 キックスタートファイルの維持

キックスタートファイルで自動チェックを実行できます。通常、新規または問題のあるキックスタートファイルが有効であることを確認します。

10.1. キックスタートのメンテナンスツールのインストール

キックスタートのメンテナンスツールを使用するには、それを含むパッケージをインストールする必要があります。

手順

- **pykickstart** パッケージをインストールします。

```
# dnf install pykickstart
```

10.2. キックスタートファイルの確認

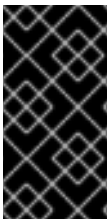
ksvalidator コマンドラインユーティリティーを使用して、キックスタートファイルが有効であることを確認します。これは、キックスタートファイルに大規模な変更を加える際に便利です。**ksvalidator** コマンドで **-v RHEL9** オプションを使用して、RHEL9 クラスの新しいコマンドを認識します。

手順

- キックスタートファイルで **ksvalidator** を実行します。

```
$ ksvalidator -v RHEL9 /path/to/kickstart.ks
```

/path/to/kickstart.ks を、確認するキックスタートファイルのパスに置き換えます。



重要

検証ツールは、インストールの成功を保証しているわけではありません。このツールは、構文が正しく、ファイルに非推奨のオプションが含まれていないことだけを保証します。キックスタートファイルの **%pre** セクション、**%post** セクション、および **%packages** セクションは検証されません。

関連情報

- **ksvalidator(1)** の man ページ

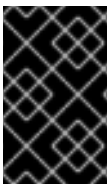
パート II. コンテンツ配信ネットワーク (CDN) と SATELLITE から RHEL を登録してインストールする場合。

第11章 キックスタートを使用した CDN を介した RHEL の登録およびインストール

本セクションでは、キックスタートを使用して、システムを登録し、RHEL サブスクリプションを割り当て、Red Hat コンテンツ配信ネットワーク (CDN) からインストールする方法を説明します。

11.1. CDN から RHEL の登録およびインストール

この手順に従って、システムを登録して、RHEL サブスクリプションを割り当て、キックスタートコマンドの **rhsm** を使用して Red Hat コンテンツ配信ネットワーク (CDN) からインストールします。これは、**syspurpose** コマンドと Red Hat Insights に対応しています。キックスタートコマンド **rhsm** は、システムの登録時にカスタムの **%post** スクリプトを使用する要件を削除します。

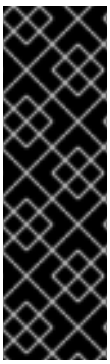


重要

CDN 機能は、**Boot ISO** および **DVD ISO** のイメージファイルでサポートされています。ただし、**Boot ISO** イメージファイルのインストールソースのデフォルトは CDN であるため、Boot ISO イメージファイルを使用することが推奨されます。

前提条件

- CDN にアクセスできるネットワークに接続されている。
- キックスタートファイルを作成し、リムーバブルメディア、ディスク、または HTTP(S)、FTP、NFS サーバーを使用するネットワーク上の場所のインストールプログラムから使用できるようにしてある。
- インストールするシステムからアクセス可能な場所にキックスタートファイルがある。
- インストールの開始に使用するブートメディアを作成し、インストールソースをインストールプログラムで使用できるようにしました。



重要

- システム登録後に使用されるインストールソースリポジトリーは、システムの起動方法により異なります。詳細は、Performing a standard RHEL 9 installation の **Installation source repository after system registration** のセクションを参照してください。
- サブスクリプションは、システムがアクセスできる CDN サブセットとリポジトリーを管理するため、キックスタートファイルではリポジトリー設定は必要ありません。

手順

1. キックスタートファイルを開きます。
2. このファイルに、**rhsm** キックスタートコマンドとそのオプションを追加します。

   組織 (必須)

   組織 ID を入力します。以下に例を示します。

   ```
   --organization=1234567
   ```

注記

セキュリティー上の理由から、CDN から登録してインストールする場合、Red Hat のユーザー名およびパスワードアカウントの詳細はキックスタートでは対応していません。

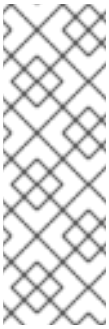
アクティベーションキー (必須)

アクティベーションキーを入力します。サブスクリプションにアクティベーションキーが登録されている限り、複数の鍵を使用できます。以下に例を示します。

```
--activation-key="Test_key_1" --activation-key="Test_key_2"
```

Red Hat Insights (任意)

ターゲットシステムを Red Hat Insights に接続します。



注記

Red Hat Insights は SaaS (Software-as-a-Service) 製品で、継続的に、登録済みの Red Hat ベースのシステムに詳細な分析を提供し、物理環境、仮想環境、クラウド環境、およびコンテナーデプロイメントでセキュリティー、パフォーマンス、および安定性に関する脅威をプロアクティブに特定します。インストーラー GUI を使用した手動インストールとは異なり、キックスタートの使用時には、Red Hat Insights への接続はデフォルトで有効になっていません。

以下に例を示します。

```
--connect-to-insights
```

HTTP プロキシー (任意)

HTTP プロキシーを設定します。以下に例を示します。

```
--proxy="user:password@hostname:9000"
```

注記

ホスト名のみが必須です。認証のないデフォルトポートでプロキシーを実行する必要がある場合は、オプションが **--proxy="hostname"** になります。

システムの目的 (任意)

次のコマンドを使用して、システムの目的のロール、SLA、使用方法を設定します。

```
subscription-manager syspurpose role --set="Red Hat Enterprise Linux Server" --
sla="Premium" --usage="Production"
```

例

次の例では、すべてのキックスタートコマンドの **rhsm** オプションを含む最小限のキックスタートファイルを表示しています。

```
graphical
lang en_US.UTF-8
keyboard us
rootpw 12345
timezone America/New_York
zerombr
clearpart --all --initlabel
autopart
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --
usage="Production"
rhsm --organization="12345" --activation-key="test_key" --connect-to-insights --
proxy="user:password@hostname:9000"
reboot
%packages
vim
%end
```

3. キックスタートファイルを保存し、インストールプロセスを開始します。

関連情報

- システムの目的の設定
- キックスタートインストールの開始
- **Red Hat Insights product documentation**
- Understanding Activation Keys
- Subscription Manager の HTTP プロキシーの設定は、**subscription-manager** man ページの **PROXY CONFIGURATION** セクションを参照してください。

11.2. CDN からシステム登録の確認

以下の手順に従って、システムが CDN に登録されていることを確認します。

前提条件

- CDN を使用した登録とインストール に記載されているように、登録とインストールのプロセスを完了している。
- **Starting Kickstart installations** に従って、キックスタートインストールを開始しています。
- インストール済みシステムを再起動して、端末画面が開いている。

手順

1. 端末画面で、**root** ユーザーとしてログインして、登録を確認します。

```
# subscription-manager list
```

出力には、割り当てられているサブスクリプションの詳細が表示されます。以下に例を示します。

```
Installed Product Status

Product Name: Red Hat Enterprise Linux for x86_64
Product ID: 486
Version: X
Arch: x86_64
Status: Subscribed
Status Details
Starts: 11/4/2019
Ends: 11/4/2020
```

2. 詳細なレポートを表示するには、次のコマンドを実行します。

```
# subscription-manager list --consumed
```

11.3. CDN からシステムの登録解除

以下の手順に従って、Red Hat CDN からシステムの登録を解除します。

前提条件

- Registering and installing RHEL from the CDN に従って、登録およびインストールプロセスを完了している。
- Starting Kickstart installations に従って、キックスタートインストールを開始しています。
- インストール済みシステムを再起動して、端末画面が開いている。

手順

- 端末画面で、**root** ユーザーとしてログインし、登録を解除します。

```
# subscription-manager unregister
```

システムに割り当てられていたサブスクリプションが削除され、CDN への接続が削除されます。

第12章 キックスタートを使用した SATELLITE からの RHEL の登録およびインストール

本セクションでは、キックスタートを使用して、システムを登録し、RHEL サブスクリプションを割り当て、Red Hat Satellite からインストールする方法を説明します。

12.1. SATELLITE からの RHEL の登録およびインストール

この手順では、キックスタートコマンドの **rhsm** を使用して、システムを登録し、RHEL サブスクリプションを割り当て、Satellite インスタンスからインストールする方法を説明します。また、システムの目的を設定し、システムを Red Hat Insights に接続する方法についても説明します。キックスタートコマンド **rhsm** は、システムの登録時にカスタムの **%post** スクリプトを使用する要件を削除します。

重要

- Satellite のインストールは、Boot ISO イメージファイルおよび DVD ISO イメージファイルでサポートされています。ただし、Boot ISO イメージファイルのインストールソースのデフォルトは Satellite であるため、Boot ISO イメージファイルを使用することが推奨されます。
- システム登録後に使用されるインストールソースリポジトリーは、システムの起動方法により異なります。詳細は、Installation source repository after the system registration を参照してください。
- サブスクリプションは、システムがアクセスできる Satellite ホストリポジトリーを管理するため、キックスタートファイルではリポジトリー設定は必要ありません。

前提条件

- システムが Satellite インスタンスにアクセスできるネットワークに接続されている。
- Red Hat Satellite Server のバージョンが 6.11 以降である
- キックスタートファイルを作成し、リムーバブルメディア、ディスク、または HTTP(S)、FTP、NFS サーバーを使用するネットワーク上の場所のインストールプログラムから使用できるようにしてある。
- インストールするシステムからアクセス可能な場所にキックスタートファイルがある。
- 組織 ID、アクティベーションキー、および使用する Satellite 6.11 インスタンスの URL が手元にあります。
- 必要な BaseOS および AppStream RPM リポジトリーを有効にし、同期し、コンテンツビューに追加した。
- アクティベーションキーのリリースバージョンが 9.x に設定されており、その中で関連するコンテンツビューが選択されている。

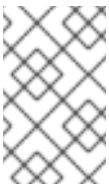
手順

1. キックスタートファイルを開きます。
2. このファイルに、**rhsm** キックスタートコマンドとそのオプションを追加します。

組織 (必須)

組織 ID を入力します。以下に例を示します。

```
--organization=1234567
```



注記

セキュリティー上の理由から、Satellite から登録してインストールする場合、Red Hat のユーザー名およびパスワードアカウントの詳細はキックスタートではサポートされていません。

アクティベーションキー (必須)

アクティベーションキーを入力します。サブスクリプションにアクティベーションキーが登録されている限り、複数の鍵を使用できます。以下に例を示します。

```
--activation-key="Test_key_1" --activation-key="Test_key_2"
```

Red Hat Insights (任意)

ターゲットシステムを Red Hat Insights に接続します。

注記

Red Hat Insights は SaaS (Software-as-a-Service) 製品で、継続的に、登録済みの Red Hat ベースのシステムに詳細な分析を提供し、物理環境、仮想環境、クラウド環境、およびコンテナーデプロイメントでセキュリティー、パフォーマンス、および安定性に関する脅威をプロアクティブに特定します。インストーラー GUI を使用した手動インストールとは異なり、キックスタートの使用時には、Red Hat Insights への接続はデフォルトで有効になっていません。

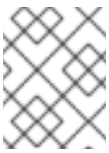
以下に例を示します。

```
--connect-to-insights
```

HTTP プロキシー (任意)

HTTP プロキシーを設定します。以下に例を示します。

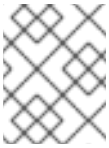
```
--proxy="user:password@hostname:9000"
```



注記

ホスト名のみが必須です。認証のないデフォルトポートでプロキシーを実行する必要がある場合は、オプションが **--proxy="hostname"** になります。

サーバーのホスト名



注記

サーバーのホスト名には HTTP プロトコル (例: **nameofhost.com**) は必要ありません。

Satellite インスタンスに登録する場合はサーバーのホスト名を設定します。以下に例を示します。

```
--server-hostname="nameofhost.com"
```

システムの目的 (任意)

次のコマンドを使用して、システムの目的のロール、SLA、使用方法を設定します。

```
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --
usage="Production"
```

例

次の例では、すべてのキックスタートコマンドの **rhsm** オプションを含む最小限のキックスタートファイルを表示しています。

```
graphical
lang en_US.UTF-8
keyboard us
rootpw 12345
timezone America/New_York
zerombr
clearpart --all --initlabel
autopart
syspurpose --role="Red Hat Enterprise Linux Server" --sla="Premium" --
usage="Production"
rhsm --organization="12345" --activation-key="test_key" --connect-to-insights --server-
hostname="nameofhost.com" --proxy="user:password@hostname:9000"
reboot
%packages
vim
%end
```

3. キックスタートファイルを保存し、インストールプロセスを開始します。

検証手順

システムがインストールされ、再起動し、端末ウィンドウが開いたら、システムが Satellite に登録されていることを確認できます。

1. 端末ウィンドウで、root ユーザーとして以下のコマンドを入力します。

```
# subscription-manager list
Installed Product Status
Product Name: Red Hat Enterprise Linux for x86_64
Product ID: 486
Version: 9
Arch: x86_64
Status: Subscribed
```

```
Status Details
Starts: 11/4/2019
Ends: 11/4/2020
```

2. 詳細なレポートを表示するには、以下を実行します。

```
# subscription-manager list --consumed
```

関連情報

- システムの目的の設定
- キックスタートインストールの開始
- Red Hat Insights product documentation
- Red Hat Subscription Management を使用するためのアクティベーションキーについて 。Subscription Manager の HTTP プロキシーの設定については、**subscription-manager** man ページの **PROXY CONFIGURATION** セクションを参照してください。

12.2. SATELLITE からのシステムの登録解除

この手順では、Satellite からシステムの登録を解除する方法を説明します。

前提条件

- 登録およびインストールプロセスを完了している。
- キックスタートインストールを開始している。
- インストール済みシステムを再起動して、端末画面が開いている。

手順

- 端末ウィンドウで、root ユーザーとして以下のコマンドを入力します。

```
# subscription-manager unregister
```

システムに割り当てられていたサブスクリプションが登録解除され、Satellite への接続が削除されます。

パート III. 高度な設定オプション

第13章 システムの目的の設定

システムの目的を使用して、Red Hat Enterprise Linux 9 システムの使用目的を記録します。システムの目的を設定すると、エンタイトルメントサーバーは最も適切なサブスクリプションを自動接続できます。本セクションは、キックスタートを使用して、システムの目的を設定する方法を説明します。

次の利点があります。

- システム管理および事業運営に関する詳細なシステムレベルの情報
- システムを調達した理由とその目的を判断する際のオーバーヘッドを削減
- Subscription Manager の自動割り当てと、システムの使用状況の自動検出および調整のカスタマーエクスペリエンスの向上

13.1. 概要

以下のいずれかの方法でシステムの目的のデータを入力できます。

- イメージの作成時
- **Connect to Red Hat**画面を使用してシステムを登録し、Red Hat サブスクリプションを割り当てる際の GUI インストール時
- **syspurpose Kickstart** コマンドを使用したキックスタートインストール時
- **subscription-manager** コマンドライン (CLI) ツールを使用したインストール後

システムの目的を記録するために、システムの目的の以下のコンポーネントを設定できます。選択された値は、登録時にエンタイトルメントサーバーが、システムに最適なサブスクリプションを割り当てるのに使用されます。

ロール

- Red Hat Enterprise Linux Server
- Red Hat Enterprise Linux Workstation
- Red Hat Enterprise Linux Compute Node

サービスレベルアグリーメント

- Premium
- Standard
- Self-Support

用途

- Production
- Development/Test
- Disaster Recovery

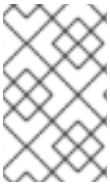
関連情報

- RHEL システムイメージのカスタマイズ
- 高度な RHEL 9 インストールの実行
- Red Hat Subscription Manager の使用および設定

13.2. キックスタートでシステムの目的の設定

以下の手順に従って、インストール時にシステムの目的を設定します。これを行うには、キックスタート設定ファイルで、キックスタートコマンドの **syspurpose** を使用します。

システムの目的は Red Hat Enterprise Linux インストールプログラムでは任意の機能ですが、最適なサブスクリプションを自動的にアタッチするためにシステムの目的を設定することを強く推奨します。



注記

インストール完了後にシステムの目的を有効にすることもできます。これを行うには、**subscription-manager** コマンドラインツールを使用します。**subscription-manager** ツールコマンドは、**syspurpose** キックスタートコマンドとは異なります。

キックスタートコマンド **syspurpose** では、以下のアクションが利用可能です。

ロール

システムで計画しているロールを設定します。このアクションは以下の形式を使用します。

```
syspurpose --role=
```

割り当てられるロールは以下のとおりです。

- **Red Hat Enterprise Linux Server**
- **Red Hat Enterprise Linux Workstation**
- **Red Hat Enterprise Linux Compute Node**

SLA

システムで計画している SLA を設定します。このアクションは以下の形式を使用します。

```
syspurpose --sla=
```

割り当てられる SLA は以下の通りです。

- **Premium**
- **Standard**
- **Self-Support**

使用方法

システムで計画している使用目的を設定します。このアクションは以下の形式を使用します。

```
syspurpose --usage=
```

割り当てられる使用方法は以下の通りです。

- **Production**
- **Development/Test**
- **障害復旧**

アドオン

追加のレイヤード製品または機能。複数のアイテムを追加するには、階層化製品/機能ごとに 1 回使用する **--addon** を複数回指定します。このアクションは以下の形式を使用します。

```
syspurpose --addon=
```

13.3. 関連情報

- Configuring System Purpose using the **subscription-manager** command-line tool

第14章 インストール時のドライバーの更新

本セクションは、Red Hat Enterprise Linux インストールプロセス時にドライバーの更新を完了する方法を説明します。

注記

これは、インストールプロセスの任意の手順です。Red Hat は、必要な場合を除いて、ドライバーの更新は行わないことを推奨します。

前提条件

- Red Hat、ハードウェアベンダー、または信頼できるサードパーティーベンダーから、Red Hat Enterprise Linux のインストール時に、ドライバー更新が必要になることが通知されている。

14.1. 概要

Red Hat Enterprise Linux は、多数のハードウェアデバイス用のドライバーに対応していますが、新たにリリースしたドライバーには対応していない可能性があります。ドライバーの更新は、そのドライバーが対応していないために、インストールが完了できない場合に限り、実行する必要があります。インストール中にドライバーを更新することは、通常、特定の設定に対応する場合に限り必要になります。たとえば、システムのストレージデバイスへのアクセスを提供するストレージアダプター用ドライバーをインストールします。

警告

ドライバー更新ディスクは、競合するカーネルドライバーを無効にする場合があります。この方法でカーネルモジュールをアンロードすると、インストールエラーが発生することがあります。

14.2. ドライバー更新の種類

Red Hat、ハードウェアベンダー、信頼できるサードパーティーは、ドライバー更新を ISO イメージファイルとして提供します。ISO イメージファイルを受け取ったら、ドライバー更新の種類を選択してください。

ドライバー更新の種類

自動

推奨されるドライバーの更新方法です。**OEMDRV** とラベルが付いたストレージデバイス (CD、DVD、または USB フラッシュドライブ) が、そのシステムに物理的に接続されます。インストールの開始時に、**OEMDRV** ストレージデバイスが存在する場合は、それがドライバー更新ディスクのように扱われ、インストールプログラムはそのドライバーを自動的に読み込みます。

アシスト付き

このインストールプログラムは、ドライバーの更新を指定するように促します。**OEMDRV** 以外の任意のローカルストレージデバイスラベルを使用できます。インストールを開始するときに、**inst.dd** ブートオプションが指定されます。このオプションにパラメーターを付けずに使用すると、インス

トールプログラムはシステムに接続されているすべてのストレージデバイスを表示し、ドライバー更新を含むデバイスを選択するように促します。

手動

ドライバー更新イメージまたは RPM パッケージのパスを手動で指定します。**OEMDRV** 以外のラベルを持つ任意のローカルストレージ、またはインストールシステムからアクセス可能なネットワークの場所を使用できます。**inst.dd=location** 起動オプションは、インストールの開始時に指定しますが、**location** は、ドライバー更新ディスクまたは ISO イメージのパスになります。このオプションを指定すると、インストールプログラムは特定の場所にあるドライバー更新を読み込みます。手動でドライバーを更新する場合は、ローカルストレージデバイス、またはネットワークの場所 (HTTP、HTTPS、または FTP サーバー) を指定できます。

注記

- **inst.dd=location** と **inst.dd** の両方を同時に使用できます。**location** は、ドライバー更新ディスクまたは ISO イメージのパスになります。このシナリオでは、インストールプログラムは、その場所から、利用可能なドライバーの更新を読み込み、ドライバーの更新が含まれるデバイスを選択するように求められます。

制限事項

セキュアブート技術を使用する UEFI システムでは、すべてのドライバーが有効な証明書で署名されている必要があります。Red Hat ドライバーは、Red Hat の秘密鍵のいずれかで署名され、カーネルで対応する公開鍵により認証されます。追加で別のドライバーを読み込む場合は、それが署名されていることを確認してください。

14.3. ドライバー更新の準備

この手順では、CD および DVD でドライバー更新の準備を行う方法を説明します。

前提条件

- Red Hat、ハードウェアベンダー、または信頼できるサードパーティーベンダーからドライバー更新の ISO イメージを受け取っている。
- ドライバー更新の ISO イメージを CD または DVD に焼き付けている。

警告

CD または DVD で、**.iso** で終了する ISO イメージファイルが 1 つしか利用できない場合、書き込み処理は成功していません。CD または DVD に ISO イメージを作成する方法は、システムの書き込みソフトウェアのドキュメントを参照してください。

手順

1. ドライバー更新用 CD または DVD をシステムの CD/DVD ドライブに挿入し、システムのファイルマネージャーツールで参照します。

2. **rhdd3** ファイルが 1 つ利用できることを確認します。**rhdd3** は、ドライバーの説明が含まれる署名ファイルと、ディレクトリーの **rpms** です。このディレクトリーには、さまざまなアーキテクチャー用のドライバーが同梱される RPM パッケージが含まれます。

14.4. 自動ドライバー更新の実行

この手順では、インストール時にドライバーの自動更新を行う方法を説明します。

前提条件

- **OEMDRV** ラベルの付いた標準のディスクパーティションにドライバーの更新イメージを置くか、**OEMDRV** ドライバー更新イメージを CD または DVD に作成します。RAID や LVM ボリュームなどの高度なストレージは、ドライバーの更新プロセス中はアクセスできない可能性があります。
- インストールプロセスを開始する前に、ボリュームラベル **OEMDRV** が付いたブロックデバイスをシステムに接続しているか、事前に準備した CD または DVD をシステムの CD/DVD ドライブに挿入している。

手順

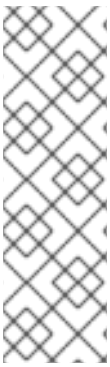
- 前提条件の手順を完了すると、インストールプログラムの起動時にドライバーが自動的にロードされ、システムのインストールプロセス中にインストールされます。

14.5. アシスト付きドライバー更新の実行

この手順では、インストール時に、ドライバーのアシスト付き更新を行う方法を説明します。

前提条件

- インストールプロセスを開始する前に、**OEMDRV** ボリュームラベルのないブロックデバイスをシステムに接続し、ドライバーディスクイメージをこのデバイスにコピーしたか、ドライバー更新の CD または DVD を準備して、システムの CD または DVD ドライブに挿入しました。



注記

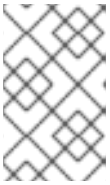
CD または DVD に ISO イメージファイルを書き込んだにもかかわらず、**OEMDRV** ボリュームラベルがない場合は、引数を追加せずに **inst.dd** オプションを使用できます。インストールプログラムは、CD または DVD からドライバーをスキャンして選択するオプションを提供します。このシナリオでは、インストールプログラムから、ドライバー更新用 ISO イメージを選択するように求められません。別のシナリオでは、起動オプション **inst.dd=location** で CD または DVD を使用します。これにより、インストールプログラムが、ドライバー更新に CD または DVD を自動的にスキャンできるようになります。詳細は、Performing a manual driver update を参照してください。

手順

1. ブートメニューウィンドウで **Tab** キーを押して、ブートコマンドラインを表示します。
2. 起動オプション **inst.dd** をコマンドラインに追加し、**Enter** を押して起動プロセスを実行します。

3. メニューから、ローカルディスクパーティション、もしくは CD デバイスまたは DVD デバイスを選択します。インストールプログラムが ISO ファイル、または ドライバー更新 RPM パッケージをスキャンします。

4. 必要に応じて、ドライバー更新 ISO ファイルを選択してください。



注記

選択したデバイスまたはパーティション (ドライバー更新 CD または DVD を含む光学ドライブなど) に、ISO イメージファイルではなく、ドライバー更新 RPM パッケージが含まれる場合は、この手順は必要ありません。

5. 必要なドライバーを選択します。

   a. キーボードの数字キーを使用して、ドライバー選択を切り替えます。

   b. **c** を押して、選択したドライバーをインストールします。選択したドライバーが読み込まれ、インストールプロセスが始まります。

14.6. 手動によるドライバー更新の実行

この手順では、インストール時にドライバーを手動で更新する方法を説明します。

前提条件

- ドライバー更新の ISO イメージファイルを USB フラッシュドライブまたは Web サーバーに配置し、コンピューターに接続しました。

手順

1. ブートメニューウィンドウで **Tab** キーを押して、ブートコマンドラインを表示します。

2. **inst.dd=location** 起動オプションをコマンドに追加します。場所は、ドライバー更新のファイルがある場所です。通常、イメージファイルは Web サーバー http://server.example.com/dd.iso など、または USB フラッシュドライブ **/dev/sdb1** などに置きます。ドライバー更新を含む RPM パッケージ http://server.example.com/dd.rpm などを指定することもできます。

3. **Enter** を押して、起動プロセスを実行してください。指定した場所で利用可能なドライバーが自動的に読み込まれ、インストールプロセスが始まります。

関連情報

- The **inst.dd** boot option

14.7. ドライバーの無効

この手順では、誤動作しているドライバーを無効にする方法を説明します。

前提条件

- インストールプログラムブートメニューを起動している。

手順

1. ブートメニューで **Tab** キーを押して、ブートコマンドラインを表示します。
2. 起動オプション **modprobe.blacklist=driver_name** をコマンドラインに追加します。
3. **driver_name** を、無効にするドライバーの名前に置き換えます。以下に例を示します。

```
modprobe.blacklist=ahci
```

   起動オプション **modprobe.blacklist=** を使用して無効にしたドライバーは、インストール済みシステムで無効になり、**/etc/modprobe.d/anaconda-blacklist.conf** ファイルに表示されます。

4. **Enter** を押して、起動プロセスを実行してください。

第15章 UEFI セキュアブートを使用したベータシステムの起動

オペレーティングシステムのセキュリティーを強化するには、UEFI セキュアブートが有効になっているシステムで Red Hat Enterprise Linux ベータ版リリースを起動したときに、署名の検証に UEFI セキュアブート機能を使用します。

15.1. UEFI セキュアブートおよび RHEL ベータ版リリース

UEFI セキュアブートでは、オペレーティングシステムカーネルが、認識された秘密キーで署名されている必要があります。UEFI セキュアブートは、対応する公開キーを使用して署名を検証します。

Red Hat Enterprise Linux 8 のベータリリースの場合には、カーネルは Red Hat ベータ固有の秘密鍵で署名されます。UEFI セキュアブートは、対応する公開鍵を使用して署名を検証しようとしますが、このハードウェアはベータ版の秘密鍵を認識しないため、Red Hat Enterprise Linux ベータ版のリリースシステムは起動に失敗します。そのため、ベータリリースで UEFI セキュアブートを使用するには、MOK (Machine Owner Key) 機能を使用して Red Hat ベータ公開キーをシステムに追加します。

15.2. UEFI セキュアブートのベータ公開鍵の追加

このセクションでは、UEFI セキュアブート用に Red Hat Enterprise Linux ベータ版の公開鍵を追加する方法を説明します。

前提条件

- システムで UEFI セキュアブートが無効になっています。
- Red Hat Enterprise Linux ベータ版リリースがインストールされており、システムの再起動もセキュアブートが無効になっている。
- システムにログインし、**初期セットアップ** 画面でタスクを完了します。

手順

1. システムの Machine Owner Key (MOK) リストに Red Hat ベータ版の公開鍵の登録を開始します。

   ```
   # mokutil --import /usr/share/doc/kernel-keys/$(uname -r)/kernel-signing-ca.cer
   ```

   $(uname -r) はカーネルバージョン (**4.18.0-80.el8.x86_64** など) に置き換えられます。

2. プロンプトが表示されたらパスワードを入力します。
3. システムを再起動し、任意のキーを押して起動を続行します。Shim UEFI キー管理ユーティリティーは、システム起動時に起動します。
4. **Enroll MOK** を選択します。
5. **Continue** を選択します。
6. **Yes** を選択し、パスワードを入力します。この鍵はシステムのファームウェアにインポートされます。
7. **Reboot** を選択します。
8. システムでセキュアブートを有効にします。

15.3. ベータ版公開鍵の削除

Red Hat Enterprise Linux ベータ版リリースを削除し、Red Hat Enterprise Linux General Availability (GA) リリースをインストールするか、別のオペレーティングシステムをインストールする予定の場合は、ベータ版の公開鍵を削除します。

この手順では、ベータ版の公開鍵を削除する方法を説明します。

手順

1. システムの Machine Owner Key (MOK) リストから Red Hat ベータ版の公開鍵の削除を開始します。

```
# mokutil --reset
```

2. プロンプトが表示されたらパスワードを入力します。
3. システムを再起動し、任意のキーを押して起動を続行します。Shim UEFI キー管理ユーティリティーは、システム起動時に起動します。
4. **Reset MOK** を選択します。
5. **Continue** を選択します。
6. **Yes** を選択し、手順 2 で指定したパスワードを入力します。この鍵はシステムのファームウェアから削除されます。
7. **Reboot** を選択します。

パート IV. キックスタートの参照

付録A キックスタートスクリプトのファイル形式の参照

この参照は、キックスタートファイルの形式を詳細に説明します。

A.1. キックスタートファイルの形式

キックスタートスクリプトは、インストールプログラムが認識するキーワードが含まれ、インストールの指示を提供するプレーンテキストのファイルです。ファイルを ASCII テキストとして保存できるテキストエディター (例: Linux システムの **Gedit** または **vim**、Windows システムの **メモ帳**) は、キックスタートファイルの作成や編集に使用できます。キックスタート設定ファイルには好きな名前を付けることができますが、後で他の設定ファイルやダイアログでこの名前を指定する必要があるため、シンプルな名前にしておくことが推奨されます。

コマンド

コマンドは、インストールの命令として役に立つキーワードです。各コマンドは 1 行で記載する必要があります。コマンドにはオプションを指定できます。コマンドとオプションの指定方法は、シェルで Linux コマンドを使用するのと似ています。

セクション

パーセント **%** 文字で始まる特殊コマンドは、セクションを開始します。セクションのコマンドの解釈は、セクションの外に置かれたコマンドとは異なります。すべてのセクションは、**%end** コマンドで終了する必要があります。

セクションタイプ

利用可能なセクションは以下のとおりです。

- **アドオンセクション**。これらのセクションは、**%addon addon_name** コマンドを使用します。
- **パッケージの選択セクション**。**%packages** から始まります。これを使用してインストールするパッケージを指定します。これには、パッケージグループやモジュールなど、間接的な指定も含まれます。
- **スクリプトセクション**。これは、**%pre**、**%pre-install**、**%post**、および **%onerror** で開始します。これらのセクションは必須ではありません。

コマンドセクション

コマンドセクションは、スクリプトセクションや **%packages** セクション以外の、キックスタートファイルのコマンドに使用される用語です。

スクリプトセクション数および順序付け

コマンドセクションを除くすべてのセクションはオプションであり、複数回表示できます。特定タイプのスクリプトセクションが評価される際に、キックスタートにあるそのタイプのセクションがすべて、表示順に評価されます。たとえば、**%post** が 2 つある場合は、表示されている順に評価されます。ただし、さまざまなタイプのスクリプトセクションを任意の順序で指定する必要はありません。**%pre** セクションの前に、**%post** セクションがあるかどうかは問題ありません。

コメント

キックスタートコマンドは、ハッシュ文字 **#** 始まる行です。このような行は、インストールプログラムには無視されます。

必須項目以外は省略しても構いません。必須項目を省略すると、インストールプログラムがインタラクティブモードに変更され、通常の対話型インストールと同じように、ユーザーが関連する項目に回答できるようになります。キックスタートスクリプトは、**cmdline** コマンドで非対話的に宣言することもできます。非対話モードでは、回答していない項目があるとインストールプロセスが中断します。

注記

テキストまたはグラフィカルモードのキックスタートインストール時にユーザーの対話が必要な場合は、インストールを完了するために更新が必須であるウィンドウのみに入力してください。スポークを入力すると、キックスタートの設定がリセットされる可能性があります。設定のリセットは、インストール先ウィンドウの入力後に、ストレージに関連するキックスタートコマンドに特化して適用されます。

A.2. キックスタートでのパッケージ選択

キックスタートは、インストールするパッケージを選択するために、**%packages** コマンドで始まるセクションを使用します。この方法で、パッケージ、グループ、環境、モジュールストリーム、およびモジュールプロファイルをインストールできます。

A.2.1. パッケージの選択セクション

%packages コマンドを使用して、インストールするソフトウェアパッケージを説明するキックスタートセクションを開始します。**%packages** セクションは、**%end** コマンドで終了する必要があります。

パッケージは、環境、グループ、モジュールストリーム、モジュールプロファイル、またはパッケージ名で指定できます。関連パッケージを含むいくつかの環境およびグループが定義されます。環境およびグループのリストは、Red Hat Enterprise Linux 9 インストール DVD の **repository/repodata/*-comps-repository.architecture.xml** ファイルを参照してください。

***-comps-repository.architecture.xml** ファイルには、利用可能な環境 (**<environment>** タグでマーク) およびグループ (**<group>** タグ) を記述した構造が含まれています。各エントリーには、ID、ユーザー可視性の値、名前、説明、パッケージリストがあります。グループがインストールに選択されていると、パッケージリストで **mandatory** とマークされたパッケージが常にインストールされ、**default** とマークされたパッケージは、他で個別に除外されていない場合に限りインストールされます。また、**optional** とマークされたパッケージは、グループが選択されている場合でも、他で明確に含める必要があります。

パッケージグループや環境は、その ID (**<id>** タグ) もしくは名前 (**<name>** タグ) を使用して指定できます。

どのパッケージをインストールするべきかわからない場合は、Minimal Install 環境を選択することが推奨されます。**最小インストール** では、Red Hat Enterprise Linux 9 の実行に必須のパッケージのみが提供されます。これにより、システムが脆弱性の影響を受ける可能性が大幅に減ります。必要な場合は、インストール後に追加パッケージをインストールできます。**最小インストール** の詳細は、**セキュリティーの強化** の 必要なパッケージの最小限のインストール のセクションを参照してください。**初期セットアップ** は、デスクトップ環境と X Window System がインストールに含まれ、グラフィカルログインが有効になっていないと、キックスタートファイルからシステムをインストールしてから実行することができません。

重要

64 ビットシステムに 32 ビットパッケージをインストールするには、次を行います。

- **%packages** セクションに **--multilib** オプションを指定します。
- **glibc.i686** のように、そのパッケージの構築対象である 32 ビットアーキテクチャーをパッケージ名に追記します。

A.2.2. パッケージの選択コマンド

このコマンドは、キックスタートファイルの **%packages** セクションで使用できます。

環境の指定

@^ 記号で開始する行で、インストールする環境全体を指します。

```
%packages
@^Infrastructure Server
%end
```

これは、**インフラストラクチャーサーバー** 環境の一部となるパッケージをすべてインストールします。利用可能なすべての環境は、Red Hat Enterprise Linux 9 インストール DVD の **repository/repodata/*-comps-repository.architecture.xml** ファイルで説明されています。

キックスタートファイルに指定する必要があるのは、1 つの環境だけです。追加の環境を指定すると、最後に指定した環境のみが使用されます。

グループの指定

1 行に 1 エントリーずつグループを指定します。***-comps-repository.architecture.xml** ファイルに指定したとおりに、**@** 記号に続いてグループのフルネームまたはグループ ID を指定します。以下に例を示します。

```
%packages
@X Window System
@Desktop
@Sound and Video
%end
```

Core グループは常に選択されるため、**%packages** セクションで指定する必要はありません。

個別パッケージの指定

1 行に 1 エントリーで、名前で個別のパッケージを指定します。アスタリスク記号 (*) をパッケージ名のワイルドカードとして使用できます。以下に例を示します。

```
%packages
sqlite
curl
aspell
docbook*
%end
```

docbook* エントリーには、ワイルドカードを使用したパターンに適合する **docbook-dtds** パッケージおよび **docbook-style** パッケージが含まれます。

モジュールストリームのプロファイルの指定

プロファイルの構文を使用して、モジュールストリープのポリシーを、1 行ごとに指定します。

```
%packages
@module:stream/profile
%end
```

これにより、モジュールストリームで指定したプロファイルに記載されているパッケージがすべてインストールされます。

- モジュールにデフォルトのストリームが指定されている場合は、削除できます。デフォルトのストリームが指定されていない場合は、指定する必要があります。
- モジュールストリームにデフォルトのプロファイルが指定されている場合は、削除できます。デフォルトのプロファイルが指定されていない場合は、指定する必要があります。
- 異なるストリームでモジュールを複数回インストールすることはできません。
- 同じモジュールおよびストリームの複数プロファイルをインストールできます。

モジュールおよびグループは、@ 記号で始まる同じ構文を使用します。同じ名前のモジュールとパッケージグループが存在する場合は、モジュールが優先されます。

Red Hat Enterprise Linux 9 では、モジュールは AppStream リポジトリーにのみ存在します。利用可能なモジュールのリストを表示するには、インストールされている Red Hat Enterprise Linux 9 システムで **dnf module list** コマンドを使用します。

キックスタートコマンド **module** を使用して、モジュールストリームを有効にし、直接命名して、モジュールストリームに含まれるパッケージをインストールすることもできます。

環境、グループ、パッケージの除外

ダッシュ (**-**) を先頭に付け、インストールから除外するパッケージやグループを指定します。以下に例を示します。

```
%packages
-@Graphical Administration Tools
-autofs
-ipa*compat
%end
```

重要

キックスタートファイルで * のみを使用して、利用可能なパッケージをすべてインストールする方法はサポートされていません。

%packages セクションのデフォルト動作は、オプションを使用して変更する方法がいくつかあります。オプションの中には、全パッケージの選択で機能するものと、特定のグループにのみ機能するものがあります。

関連情報

- DNF ツールを使用したソフトウェアの管理

A.2.3. 一般的なパッケージ選択のオプション

%packages では、以下のオプションが使用できます。オプションを使用するには、パッケージ選択セクションの最初に追加します。以下に例を示します。

```
%packages --multilib --ignoremissing
```

--default

パッケージのデフォルトセットをインストールします。これは、対話式インストールの **パッケージの選択** 画面でその他を選択しない場合にインストールされるパッケージセットに対応するものです。

--excludedocs

パッケージに含まれているドキュメンテーションをインストールしません。ほとんどの場合、**/usr/share/doc** ディレクトリーにインストールされるファイルは除外されますが、個別に除外されるファイルは個別のパッケージによります。

--ignoremissing

インストールを停止してインストールの中断または続行を確認する代わりに、インストールソースにないパッケージ、グループ、モジュールストリーム、モジュールプロファイル、および環境を無視します。

--inst-langs

インストールする言語リストを指定します。これはパッケージグループレベルでの選択とは異なることに注意してください。このオプションでは、インストールするパッケージグループを記述するのではなく、RPM マクロを設定して、個別パッケージからインストールする翻訳ファイルを制御します。

--multilib

64 ビットのシステムに 32 ビットのパッケージをインストールできるように、multilib パッケージ用にインストールされたシステムを設定し、本セクションで説明しているようにパッケージをインストールします。
通常、AMD64 および Intel 64 のシステムでは、x86_64 パッケージおよび noarch パッケージのみをインストールできます。ただし、--multilib オプションを使用すると、32 ビット AMD および i686 Intel のシステムパッケージが存在する場合は自動的にインストールされます。

これは **%packages** セクションで明示的に指定されているパッケージにのみ適用されます。キックスタートファイルで指定されずに依存関係としてのみインストールされるパッケージは、他のアーキテクチャーで利用可能な場合でも、必要とされるアーキテクチャーのバージョンにのみインストールされます。

システムのインストール時に、Anaconda が **multilib** モードでパッケージをインストールするように設定できます。以下のいずれかのオプションを使用して **multilib** モードを有効にします。

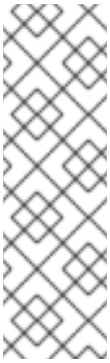
1. 以下の行でキックスタートファイルを設定します。

```
%packages --multilib --default
%end
```

2. インストールイメージの起動中に、inst.multilib 起動オプションを追加します。

--nocore

@Core パッケージグループのインストールを無効にします。これを使用しない場合は、デフォルトでインストールされます。**--nocore** での **@Core** パッケージグループの無効化は、軽量コンテナーの作成にのみ使用してください。**--nocore** を指定してデスクトップやサーバーのシステムをインストールすると、システムが使用できなくなります。



注記

- **@Core** パッケージグループ内のパッケージを、**-@Core** を使用して除外することはできません。**@Core** パッケージグループを除外する唯一の方法は、**--nocore** オプションを使用することです。
- **@Core** パッケージグループは、作業 system のインストールに必要なパッケージの最小セットとして定義されています。これは、Package Manifest および Scope of Coverage Details で定義されているコアパッケージには関係ありません。

--exclude-weakdeps

弱い依存関係からのパッケージのインストールを無効にします。これは、Recommends フラグおよび Supplements フラグで選択したパッケージセットにリンクされたパッケージです。デフォルトでは、弱い依存関係がインストールされます。

--retries=

DNF がパッケージのダウンロードを試みる回数を設定します (再試行)。デフォルト値は 10 です。このオプションはインストール時にのみ適用され、インストールされているシステムの DNF 設定には影響を及ぼしません。

--timeout=

DNF タイムアウトを秒単位で設定します。デフォルト値は 30 です。このオプションはインストール時にのみ適用され、インストールされているシステムの DNF 設定には影響を及ぼしません。

A.2.4. 特定パッケージグループ用のオプション

以下のオプションは、単一パッケージグループにのみ適用されます。キックスタートファイルの **%packages** コマンドで使用する代わりに、グループ名に追加します。以下に例を示します。

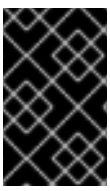
```
%packages
@Graphical Administration Tools --optional
%end
```

--nodefaults

デフォルト選択ではなく、グループの必須パッケージのみをインストールします。

--optional

デフォルトの選択に加えて、***-comps-repository.architecture.xml** ファイルのグループ定義でオプションの印が付けられているパッケージをインストールします。
Scientific Support のようなパッケージグループは、必須もしくはデフォルトのパッケージが指定されておらず、オプションのパッケージのみであることに注意してください。この場合は、**--optional** オプションを常に使用する必要があり、このオプションを使用しないと、このグループからパッケージをインストールすることができません。



重要

--nodefaults および **--optional** オプションは併用できません。**--nodefaults** を使用して、インストール中に必須パッケージのみをインストールし、インストール後にインストール済みシステムにオプションのパッケージをインストールできます。

A.3. キックスタートファイル内のスクリプト

キックスタートファイルには以下のスクリプトを追加できます。

- **%pre**
- **%pre-install**
- **%post**

本セクションでは、スクリプトに関する以下の情報を提供します。

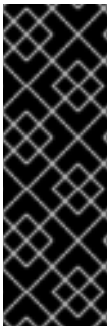
- 実行時間
- スクリプトに追加できるコマンドのタイプ
- スクリプトの目的
- スクリプトオプション

A.3.1. %pre スクリプト

%pre スクリプトは、キックスタートファイルの読み込み直後 (スクリプトが完全に解析され、インストールが開始する前) にシステムで実行されます。各セクションは、**%pre** で開始し、**%end** で終了する必要があります。

%pre スクリプトは、ネットワークおよびストレージデバイスのアクティベートおよび設定に使用できます。また、インストール環境で利用可能なインタープリターを使用して、スクリプトを実行することもできます。インストールを進める前に特定の設定を必要とするネットワークやストレージがある場合や、追加のログパラメーターや環境変数などを設定するスクリプトがある場合には、**%pre** スクリプトを追加すると便利です。

%pre スクリプトでの問題のデバッグは難しくなる可能性があるため、**%pre** スクリプトは必要な場合にのみ使用することが推奨されます。



重要

キックスタートの **%pre** セクションは、インストーラーイメージ (**inst.stage2**) がフェッチされた後に発生するインストールの段階で実行されます。これは、root がインストーラー環境 (インストーラーイメージ) に切り替わった **後**、および **Anaconda** インストーラー自体が起動した **後** に実行されます。次に、**%pre** の設定が適用され、キックスタートの URL などで設定されたインストールリポジトリーからパッケージを取得するために使用できます。ただし、ネットワークからイメージ (**inst.stage2**) をフェッチするようにネットワークを設定するために使用する **ことはできません**。

インストール環境の **/sbin** ディレクトリーおよび **/bin** ディレクトリーにあるほとんどのユーティリティーの他に、**%pre** スクリプトでは、ネットワーク、ストレージ、およびファイルシステムに関連するコマンドを使用できます。

%pre セクションのネットワークにはアクセスできます。この時点では name サービスが設定されていないため、URL ではなく IP アドレスだけが有効です。

注記

pre スクリプトは、chroot 環境では実行しません。

A.3.1.1. %pre スクリプトセクションのオプション

以下のオプションを使用して、インストール前のスクリプトの動作を変更できます。オプションを使用するには、スクリプトの最初の部分で **%pre** 行にオプションを追加してください。以下に例を示します。

```
%pre --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Python などの別のスクリプト言語を指定できます。システムで利用可能なスクリプト言語は、どれでも使用できます。ほとんどの場合は、**/usr/bin/sh**、**/usr/bin/bash**、および **/usr/libexec/platform-python** になります。
platform-python インタープリターは、Python バージョン 3.6 を使用することに注意してください。新しいパスおよびバージョン用に、Python スクリプトを以前の RHEL バージョンから変更する必要があります。また、**platform-python** は、システムツールを対象としています。インストール環境外では **python36** パッケージを使用してください。Red Hat Enterprise Linux における Python の詳細は、**動的プログラミング言語のインストールおよび使用** の Python の概要 を参照してください。

--erroronfail

スクリプトが失敗するとエラーを表示し、インストールを停止します。エラーメッセージは、失敗の原因がログ記録されている場所を示します。インストールされたシステムは、不安定で起動できない状態になる可能性があります。**inst.nokill** オプションを使用して、スクリプトをデバッグできます。

--log=

スクリプトの出力を、指定したログファイルに記録します。以下に例を示します。

```
%pre --log=/tmp/ks-pre.log
```

A.3.2. %pre-install スクリプト

pre-install スクリプトのコマンドは、以下のタスクの完了後に実行されます。

- システムのパーティションを設定した。
- ファイルシステムは /mnt/sysroot の下に作成およびマウントされます
- ネットワークが起動オプションとキックスタートコマンドに従って設定されている。

各 **%pre-install** セクションは、**%pre-install** で開始し、**%end** で終了します。

%pre-install スクリプトを使用してインストールを修正して、パッケージのインストール前に保証されている ID があるユーザーとグループを追加できます。

インストールに必要な変更には、**%post** スクリプトを使用することが推奨されます。**%pre-install** スクリプトは、**%post** スクリプトが必要な変更に満たない場合に限り使用します。

注記: **pre-install** スクリプトは、chroot 環境では実行しません。

A.3.2.1. %pre-install スクリプトセクションオプション

以下のオプションを使用して、**pre-install** のスクリプトの動作を変更できます。オプションを使用する場合は、スクリプトの先頭にある **%pre-install** 行に追加してください。以下に例を示します。

```
%pre-install --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

複数の **%pre-install** セクションを複数設定できます。インタープリターは同じものを複数回使用することもできます。設定したものは、キックスタートファイル内の参照順に評価されます。

--interpreter=

Python などの別のスクリプト言語を指定できます。システムで利用可能なスクリプト言語は、どれでも使用できます。ほとんどの場合は、**/usr/bin/sh**、**/usr/bin/bash**、および **/usr/libexec/platform-python** になります。
platform-python インタープリターは、Python バージョン 3.6 を使用することに注意してください。新しいパスおよびバージョン用に、Python スクリプトを以前の RHEL バージョンから変更する必要があります。また、**platform-python** は、システムツールを対象としています。インストール環境外では **python36** パッケージを使用してください。Red Hat Enterprise Linux における Python の詳細は、**動的プログラミング言語のインストールおよび使用** の Python の概要 を参照してください。

--erroronfail

スクリプトが失敗するとエラーを表示し、インストールを停止します。エラーメッセージは、失敗の原因がログ記録されている場所を示します。インストールされたシステムは、不安定で起動できない状態になる可能性があります。**inst.nokill** オプションを使用して、スクリプトをデバッグできます。

--log=

スクリプトの出力を、指定したログファイルに記録します。以下に例を示します。

```
%pre-install --log=/mnt/sysroot/root/ks-pre.log
```

A.3.3. %post スクリプト

%post スクリプトは、インストールが完了した後、システムが最初に再起動する前に実行されるインストール後のスクリプトです。本セクションでは、システムのサブスクリプションなどのタスクを実行できます。

インストールが完了し、システムを最初に再起動する前に、システムで実行するコマンドを追加するオプションがあります。このセクションは、**%post** で始まり、**%end** で終了します。

%post セクションは、追加ソフトウェアのインストールや、追加のネームサーバーの設定といった機能に役に立ちます。インストール後のスクリプトは **chroot** 環境で実行するため、インストールメディアからスクリプトや RPM をコピーするなどの作業はデフォルトでは機能しません。この動作は、以下に記載されるように **--nochroot** オプションを使用することで変更できます。その後、**%post** スクリプトはインストール環境で実行し、インストール済みのターゲットシステムの **chroot** で実行することはありません。

インストール後のスクリプトは **chroot** 環境で実行されるため、ほとんどの **systemctl** コマンドはいかなるアクションも拒否します。

%post セクションの実行中にも、インストールメディアが挿入される必要があることに注意してください。

A.3.3.1. %post スクリプトセクションオプション

以下のオプションを使用して、インストール後のスクリプトの動作を変更できます。オプションを使用するには、スクリプトの最初の部分で **%post** 行にオプションを追加してください。以下に例を示します。

```
%post --interpreter=/usr/libexec/platform-python
-- Python script omitted --
%end
```

--interpreter=

Python などの別のスクリプト言語を指定できます。以下に例を示します。

```
%post --interpreter=/usr/libexec/platform-python
```

システムで利用可能なスクリプト言語は、どれでも使用できます。ほとんどの場合は、**/usr/bin/sh**、**/usr/bin/bash**、および **/usr/libexec/platform-python** になります。

platform-python インタープリターは、Python バージョン 3.6 を使用することに注意してください。新しいパスおよびバージョン用に、Python スクリプトを以前の RHEL バージョンから変更する必要があります。また、**platform-python** は、システムツールを対象としています。インストール環境外では **python36** パッケージを使用してください。Red Hat Enterprise Linux における Python の詳細は、**動的プログラミング言語のインストールおよび使用** の Python の概要 を参照してください。

--nochroot

chroot 環境外で実行するコマンドを指定できます。
以下の例では、/etc/resolv.conf ファイルを、インストールしたばかりのファイルシステムにコピーします。

```
%post --nochroot
cp /etc/resolv.conf /mnt/sysroot/etc/resolv.conf
%end
```

--erroronfail

スクリプトが失敗するとエラーを表示し、インストールを停止します。エラーメッセージは、失敗の原因がログ記録されている場所を示します。インストールされたシステムは、不安定で起動できない状態になる可能性があります。**inst.nokill** オプションを使用して、スクリプトをデバッグできます。

--log=

スクリプトの出力を、指定したログファイルに記録します。ログファイルのパスは、ユーザーが **--nochroot** オプションを使用しているかどうかを考慮に入れる必要があることに注意して下さい。**--nochroot** がない場合の例を示します。

```
%post --log=/root/ks-post.log
```

--nochroot を使用した場合は、以下のようになります。

```
%post --nochroot --log=/mnt/sysroot/root/ks-post.log
```

A.3.3.2. 例: インストール後スクリプトで NFS のマウント

この **%post** セクション例では、NFS 共有をマウントし、共有の **/usr/new-machines/** に置かれた **runme** スクリプトを実行します。キックスタートモードでは NFS ファイルのロックがサポートされていないため、**-o nolock** オプションが必要となることに注意してください。

```
# Start of the %post section with logging into /root/ks-post.log
%post --log=/root/ks-post.log

# Mount an NFS share
mkdir /mnt/temp
mount -o nolock 10.10.0.2:/usr/new-machines /mnt/temp
openvt -s -w -- /mnt/temp/runme
umount /mnt/temp

# End of the %post section
%end
```

A.4. キックスタートでのエラー処理セクション

Red Hat Enterprise Linux 7 から、インストールプログラムが致命的なエラーに遭遇した場合に実行するカスタムスクリプトをキックスタートインストールに含めることができるようになりました。たとえば、インストールが要求されたパッケージにエラーがあったり、指定した VNC が起動に失敗したり、ストレージデバイスのスキャン中にエラーが発生する場合などです。このようなエラーが発生すると、インストールが続行できません。インストールプログラムは、キックスタートファイルで提供された順番で、すべての **%onerror** スクリプトを実行します。また、**%onerror** スクリプトは、トレースバックの際にも実行されます。

それぞれの **%onerror** スクリプトが、**%end** で終了する必要があります。

エラー処理のセクションでは、次のオプションを受け入れます。

--erroronfail

スクリプトが失敗するとエラーを表示し、インストールを停止します。エラーメッセージは、失敗の原因がログ記録されている場所を示します。インストールされたシステムは、不安定で起動できない状態になる可能性があります。**inst.nokill** オプションを使用して、スクリプトをデバッグできます。

--interpreter=

Python などの別のスクリプト言語を指定できます。以下に例を示します。

```
%onerror --interpreter=/usr/libexec/platform-python
```

システムで利用可能なスクリプト言語は、どれでも使用できます。ほとんどの場合は、**/usr/bin/sh**、**/usr/bin/bash**、および **/usr/libexec/platform-python** になります。

platform-python インタープリターは、Python バージョン 3.6 を使用することに注意してください。新しいパスおよびバージョン用に、Python スクリプトを以前の RHEL バージョンから変更する必要があります。また、**platform-python** は、システムツールを対象としています。インストール環境外では **python36** パッケージを使用してください。Red Hat Enterprise Linux における Python の詳細は、**動的プログラミング言語のインストールおよび使用** の Python の概要 を参照してください。

--log=

スクリプトの出力を、指定したログファイルに記録します。

A.5. キックスタートのアドオンセクション

Red Hat Enterprise Linux 7 以降は、キックスタートインストールでアドオンをサポートするようになりました。これらのアドオンは、多くの方法で基本的なキックスタート (および Anaconda) の機能を拡張できます。

キックスタートファイルでアドオンを使用するには、**%addon addon_name options** コマンドを使用し、**%end** ステートメントでコマンドを終了します。これはインストール前およびインストール後スクリプトのセクションと似ています。たとえば、デフォルトで Anaconda で提供される Kdump アドオンを使用する場合は、次のコマンドを使用します。

```
%addon com_redhat_kdump --enable --reserve-mb=auto
%end
```

%addon コマンドには、独自のオプションが含まれていません。すべてのオプションは実際のアドオンに依存しています。

付録B キックスタートのコマンドおよびオプションの参照

ここでは、Red Hat Enterprise Linux インストールプログラムがサポートするキックスタートコマンドのリストを提供します。コマンドは、いくつかのカテゴリーに分かれ、アルファベット順に記載されています。コマンドが複数のカテゴリーに該当する場合は、該当するすべてのカテゴリーに記載されます。

B.1. キックスタートの変更

以下のセクションでは、Red Hat Enterprise Linux 9 におけるキックスタートコマンドおよびオプションの変更を説明します。

B.1.1. RHEL 8 で auth または authconfig が非推奨に

authconfig ツールおよびパッケージが削除されたため、Red Hat Enterprise Linux 8 では、キックスタートコマンドの **auth** または **authconfig** が非推奨になっています。

コマンドラインで実行した **authconfig** コマンドと同様、キックスタートスクリプトの **authconfig** コマンドが **authselect-compat** ツールを使用して、新しい **authselect** ツールを実行するようになりました。この互換性層や、その既知の問題の説明は、**authselect-migration(7)** の man ページを参照してください。このインストールプログラムは、非推奨のコマンドの使用を自動的に検出し、互換性層を提供する **authselect-compat** パッケージをインストールします。

B.1.2. 以前の RHEL リリースのキックスタートファイルの使用

以前の RHEL リリースのキックスタートファイルを使用する場合は、Red Hat Enterprise Linux 8 BaseOS リポジトリーおよび AppStream リポジトリーの詳細について、Considerations in adopting RHEL 8 の **Repositories** のセクションを参照してください。

B.1.3. キックスタートで非推奨になったコマンドおよびオプション

以下のキックスタートのコマンドとオプションが RHEL 9 では非推奨になりました。

- **timezone --ntpservers** - 代わりに **timesource** コマンドを使用します。
- **timezone --nontp**
- **logging --level**
- **%packages --excludeWeakdeps** - 代わりに **--exclude-weakdeps** を使用します。
- **%packages --instLangs** - 代わりに **--inst-langs** を使用します。
- **%Anaconda**
- **pwpolicy** - 代わりに Anaconda 設定ファイルを使用します。
- **syspurpose** - 代わりに **subscription-manager syspurpose** を使用してください

特定のオプションだけがリスト表示されている場合は、基本コマンドおよびその他のオプションは引き続き利用でき、非推奨ではありません。キックスタートファイルで非推奨のコマンドを使用すると、ログに警告が出力されます。**inst.ksstrict** 起動オプションを使用して、非推奨のコマンド警告をエラーにすることもできます。

B.1.4. キックスタートから削除されたコマンドおよびオプション

RHEL 9 では、以下のキックスタートのコマンドとオプションが完全に削除されました。キックスタートファイルでこれを使用すると、エラーが発生します。

- **device**
- **deviceprobe**
- **dmraid**
- **install** (サブコマンドまたはメソッドをコマンドとして直接使用)
- **multipath**
- **bootloader --upgrade**
- **ignoredisk --interactive**
- **partition --active**
- **harddrive --biospart**
- **autostep**

特定のオプションおよび値だけが表示されている場合は、基本コマンドおよびその他のオプションは引き続き利用でき、削除されません。

B.2. インストールプログラムの設定とフロー制御のためのキックスタートコマンド

このリストのキックスタートコマンドは、インストールのモードとコースを制御し、最後に何が起こるかを制御します。

B.2.1. cdrom

キックスタートコマンドの **cdrom** は任意です。これは、システムの最初の光学ドライブからインストールを実行します。

構文

```
cdrom
```

注記

- このコマンドにはオプションはありません。
- 実際にインストールを実行するには、カーネルコマンドラインで **inst.repo** オプションが指定されていない限り、**cdrom**、**harddrive**、**hmc**、**nfs**、**liveimg**、**ostreesetup**、**rhsm**、または **url** のいずれかを指定する必要があります。

B.2.2. cmdline

キックスタートコマンドの **cmdline** は任意です。完全に非対話式のコマンドラインモードでインストールを実行します。対話のプロンプトがあるとインストールは停止します。

構文

```
cmdline
```

注記

- 完全に自動となるインストールでは、キックスタートファイルで利用可能なモード (**graphical**、**text**、または **cmdline**) のいずれかを指定するか、起動オプション **console=** を使用する必要があります。モードが指定されていないと、可能な場合はグラフィカルモードが使用されるか、VNC モードおよびテキストモードからの選択が求められます。
- このコマンドにはオプションはありません。
- このモードは、x3270 端末と共に 64 ビットの IBM Z システムで使用する場合に便利です。

B.2.3. driverdisk

キックスタートコマンドの **driverdisk** は任意です。このコマンドを使用して、インストールプログラムに追加ドライバーを提供します。

ドライバーディスクは、キックスタートを使用したインストール中に、デフォルトでは含まれていないドライバーを追加する場合に使用します。ドライバーディスクのコンテンツを、システムのディスクにあるパーティションのルートディレクトリーにコピーする必要があります。次に、**driverdisk** コマンドを使用して、インストールプログラムがドライバーディスクとその場所を検索するように指定する必要があります。

構文

```
driverdisk [partition|--source=url|--biospart=biospart]
```

オプション

この方法のいずれかで、ドライバーディスクの場所を指定する必要があります。

- partition - ドライバーディスクを含むパーティション。パーティションを指定する場合はパーティション名 (**sdb1** など) だけでは **なく**、完全パス (/**dev**/**sdb1** など) を使用してください。
- **--source=** - ドライバーディスクの URL。以下のようになります。

```
driverdisk --source=ftp://path/to/dd.img
driverdisk --source=http://path/to/dd.img
driverdisk --source=nfs:host:/path/to/dd.img
```

- **--biospart=** - ドライバーディスクを含む BIOS パーティション (**82p2** など)。

注記

ドライバーディスクは、ネットワーク経由や **initrd** から読み込むのではなく、ローカルディスクまたは同様のデバイスから読み込むこともできます。以下の手順に従います。

1. ディスクドライブ、USB、または同様のデバイスにドライバーディスクを読み込みます。
2. このデバイスにラベルを設定します (**DD** など)。
3. キックスタートファイルに以下の行を追加します。

```
driverdisk LABEL=DD:/e1000.rpm
```

DD を具体的なラベルに置き換え、e1000.rpm を具体的な名前に置き換えます。LABEL ではなく、**inst.repo** コマンドがサポートするものを使用して、ディスクドライブを指定してください。

B.2.4. eula

キックスタートコマンドの **eula** は任意です。ユーザーとの対話なしでエンドユーザーライセンス契約 (EULA) に同意するには、このオプションを使用します。このオプションを使用すると、インストールを終了して、システムを最初に再起動した後に、ライセンス契約に同意するように求められなくなります。

構文

```
eula [--agreed]
```

オプション

- **--agreed** (必須) - EULA に同意します。このオプションは必ず使用する必要があります。使用しないと **eula** コマンド自体を使用する意味がなくなります。

B.2.5. firstboot

キックスタートコマンドの **firstboot** は任意です。初めてシステムを起動した時に、**初期セットアップ** アプリケーションを開始するかどうかを指定します。有効にする場合は、**initial-setup** パッケージをインストールする必要があります。何も指定しないとデフォルトで無効になるオプションです。

構文

```
firstboot OPTIONS
```

オプション

- **--enable** または **--enabled** - システムの初回起動時に、初期セットアップを開始します。
- **--disable** または **--disabled** - システムの初回起動時に、初期セットアップを開始しません。
- **--reconfig** - システムの起動時に、初期セットアップが再設定モードで開始します。このモードでは、デフォルトのオプションに加えて、root パスワード、時刻と日付、ネットワークとホスト名の設定オプションが有効になります。

B.2.6. graphical

キックスタートコマンドの **graphical** は任意です。これは、グラフィカルモードでインストールを実行します。これがデフォルトになります。

構文

```
graphical [--non-interactive]
```

オプション

- **--non-interactive** - 完全に非対話式のモードでインストールを実行します。このモードでは、ユーザーの対話が必要になるとインストールを終了します。

注記

- 完全に自動となるインストールでは、キックスタートファイルで利用可能なモード (**graphical**、**text**、または **cmdline**) のいずれかを指定するか、起動オプション **console=** を使用する必要があります。モードが指定されていないと、可能な場合はグラフィカルモードが使用されるか、VNC モードおよびテキストモードからの選択が求められます。

B.2.7. halt

キックスタートコマンドの **halt** は任意です。

インストールが正常に完了するとシステムを一時停止します。手動インストールと同じく、Anaconda のメッセージが表示され、ユーザーがキーを押すのを待ってから再起動が行われます。キックスタートを使用したインストールで、完了方法が指定されない場合は、このオプションがデフォルトとして使用されます。

構文

```
halt
```

注記

- **halt** コマンドは **shutdown -H** コマンドと同じです。詳細は、**shutdown(8)** の man ページを参照してください。
- 他の完了方法は、**poweroff**、**reboot**、**shutdown** などのコマンドをご覧ください。
- このコマンドにはオプションはありません。

B.2.8. harddrive

キックスタートコマンドの **harddrive** は任意です。ローカルドライブにある完全インストール用の ISO イメージまたは Red Hat インストールツリーからインストールします。ドライブは、インストールプログラムがマウントできるファイルシステムでフォーマットする必要があります (**ext2**、**ext3**、**ext4**、**vfat**、または **xfs**)。

構文

```
harddrive OPTIONS
```

オプション

- **--partition=** - インストールするパーティションを指定する場合に使用します (**sdb2** など)。
- **--dir=** - 完全インストール用 DVD の ISO イメージやインストールツリーの **variant** ディレクトリーを格納しているディレクトリーを指定する場合に使用します。

例

```
harddrive --partition=hdb2 --dir=/tmp/install-tree
```

注記

- **harddrive** コマンドは、**install** コマンドとともに使用する必要がありました。**install** コマンドが非推奨になり、(**install** が暗黙的に使用されるようになったため) **harddrive** は独立して使用できるようになりました。
- 実際にインストールを実行するには、カーネルコマンドラインで **inst.repo** オプションが指定されていない限り、**cdrom**、**harddrive**、**hmc**、**nfs**、**liveimg**、**ostreesetup**、**rhsm**、または **url** のいずれかを指定する必要があります。

B.2.9. liveimg

キックスタートコマンドの **liveimg** は任意です。パッケージの代わりに、ディスクイメージからインストールを実行します。

構文

```
liveimg --url=SOURCE [OPTIONS]
```

必須オプション

- **--url=** - インストール元となる場所です。**HTTP**、**HTTPS**、**FTP**、**file** が対応プロトコルになります。

任意のオプション

- **--url=** - インストール元となる場所です。**HTTP**、**HTTPS**、**FTP**、**file** が対応プロトコルになります。
- **--proxy=** - インストール実行時に使用するプロキシー (**HTTP**、**HTTPS**、または **FTP**) を指定します。
- **--checksum=** - 検証に使用するイメージファイルのチェックサム **SHA256** を使用するオプションの引数です。
- **--noverifyssl** - **HTTPS** サーバーへの接続の際に、SSL 確認を無効にします。

例

```
liveimg --url=file:///images/install/squashfs.img --
checksum=03825f567f17705100de3308a20354b4d81ac9d8bed4bb4692b2381045e56197 --
noverifyssl
```

注記

- イメージは、ライブ ISO イメージの **squashfs.img** ファイル、圧縮 tar ファイル (**.tar**、**.tbz**、**.tgz**、**.txz**、**.tar.bz2**、**.tar.gz**、または **.tar.xz**)、もしくはインストールメディアでマウントできるファイルシステムであればどれでも構いません。**ext2**、**ext3**、**ext4**、**vfat**、**xfs** などが対応ファイルシステムになります。
- ドライバーディスクで **liveimg** インストールモードを使用している場合、ディスク上のドライバーがインストールされるシステムに自動的に含まれることはありません。これらのドライバーが必要な場合は、手動でインストールするか、キックスタートスクリプトの **%post** セクションでインストールします。

- 実際にインストールを実行するには、カーネルコマンドラインで **inst.repo** オプションが指定されていない限り、**cdrom**、**harddrive**、**hmc**、**nfs**、**liveimg**、**ostreesetup**、**rhsm**、または **url** のいずれかを指定する必要があります。

B.2.10. logging

キックスタートコマンドの **logging** は任意です。インストール時に Anaconda に記録されるエラーログを制御します。インストール済みのシステムには影響しません。

注記

ロギングは TCP でのみサポートされています。リモートロギングの場合は、**--port=** オプションで指定するポート番号がリモートサーバーで開いていることを確認してください。デフォルトのポートは 514 です。

構文

```
logging OPTIONS
```

任意のオプション

- **--host=** - 指定したリモートホストにログ情報を送信します。ログを受け取るには、リモートホストで設定した syslogd プロセスが実行している必要があります。
- **--port=** - リモートの syslogd プロセスがデフォルト以外のポートを使用する場合は、このオプションを使用して設定します。

B.2.11. mediacheck

キックスタートコマンドの **mediacheck** は任意です。このコマンドを使用すると、インストール開始前にメディアチェックの実行が強制されます。インストール時の介入が必要となるため、デフォルトでは無効になっています。

構文

```
mediacheck
```

注記

- このキックスタートコマンドは、**rd.live.check** 起動オプションに相当します。
- このコマンドにはオプションはありません。

B.2.12. nfs

キックスタートコマンドの **nfs** は任意です。指定した NFS サーバーからインストールを実行します。

構文

```
nfs OPTIONS
```

オプション

- **--server=** - インストール元となるサーバーを指定します (ホスト名または IP)。
- **--dir=** - インストールツリーの **variant** ディレクトリーを格納しているディレクトリーを指定する場合に使用します。
- **--opts=** - NFS エクスポートのマウントに使用するマウントポイントを指定します (オプション)。

例

```
nfs --server=nfsserver.example.com --dir=/tmp/install-tree
```

注記

- 実際にインストールを実行するには、カーネルコマンドラインで **inst.repo** オプションが指定されていない限り、**cdrom**、**harddrive**、**hmc**、**nfs**、**liveimg**、**ostreesetup**、**rhsm**、または **url** のいずれかを指定する必要があります。

B.2.13. ostreesetup

キックスタートコマンドの **ostreesetup** は任意です。これは、OStree ベースのインストールを設定するのに使用されます。

構文

```
ostreesetup --osname=OSNAME [--remote=REMOTE] --url=URL --ref=REF [--nogpg]
```

必須オプション:

- **--osname=OSNAME** - OS インストール用の root の管理
- **--url=URL** - インストール元となるリポジトリーの URL
- **--ref=REF** - インストールに使用するリポジトリーのブランチー名

任意のオプション:

- **--remote=REMOTE** - リモートリポジトリーの場所。
- **--nogpg** - GPG 鍵の検証の無効化

注記

- OStree ツールの詳細は、アップストリームのドキュメント https://ostreedev.github.io/ostree/ を参照してください。

B.2.14. poweroff

キックスタートコマンドの **poweroff** は任意です。インストールが正常に完了したら、システムをシャットダウンして電源を切ります。通常、手動のインストールでは Anaconda によりメッセージが表示され、ユーザーがキーを押すのを待ってから再起動が行われます。

構文

-

```
poweroff
```

注記

- **poweroff** オプションは **shutdown -P** コマンドと同じです。詳細は、**shutdown(8)** の man ページを参照してください。
- 他の完了方法は、**halt**、**reboot**、**shutdown** などのキックスタートコマンドをご覧ください。キックスタートファイルに完了方法が明示的には指定されていない場合は、**halt** オプションがデフォルトの完了方法になります。
- **poweroff** オプションは、使用中のハードウェアに大きく依存します。特に、BIOS、APM (advanced power management)、ACPI (advanced configuration and power interface) などの特定ハードウェアコンポーネントは、システムカーネルと対話できる状態にする必要があります。使用システムの APM/ACPI 機能の詳細に関しては、ハードウェアのマニュアルを参照してください。
- このコマンドにはオプションはありません。

B.2.15. reboot

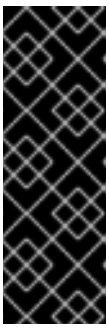
キックスタートコマンドの **reboot** は任意です。インストールが正常に完了したらシステムを再起動するように、インストールプログラムに指示します (引数なし)。通常、キックスタートは、メッセージを表示し、ユーザーがキーを押してから再起動します。

構文

```
reboot OPTIONS
```

オプション

- **--eject** - 再起動の前に起動可能なメディア (DVD、USB、またはその他のメディア) の取り出しを試みます。
- **--kexec** - 完全な再起動を実行する代わりに **kexec** システムコールを使用します。BIOS やファームウェアが通常実行するハードウェアの初期化を行わずに、インストールしたシステムを即座にメモリーに読み込みます。



重要

このオプションは非推奨になっており、テクノロジープレビューとしてのみ利用できます。テクノロジープレビュー機能に対する Red Hat のサポート範囲の詳細は、テクノロジープレビュー機能のサポート範囲 を参照してください。

kexec の使用時には、(完全なシステム再起動では通常クリアされる) デバイスレジスターにデータが残ります。デバイスドライバーによってはこれが問題になる可能性もあります。

注記

- インストールメディアやインストール方法によっては、**reboot** オプションを使用するとインストールプロセスがループして**完了しなくなる場合があります**。

- **reboot** オプションは **shutdown -r** コマンドと同じです。詳細は、**shutdown(8)** の man ページを参照してください。
- 64 ビットの IBM Z でコマンドラインによるインストールを行う際は、**reboot** を指定してインストールを完全自動化します。
- その他の完了方法は、**halt**、**poweroff**、**shutdown** などのキックスタートオプションをご覧ください。キックスタートファイルに完了方法が明示的には指定されていない場合は、**halt** オプションがデフォルトの完了方法になります。

B.2.16. rhsm

キックスタートコマンドの **rhsm** は任意です。ここでは、インストールプログラムにより、CDN から RHEL が登録されインストールされるようになっています。

注記

キックスタートコマンド **rhsm** は、システムの登録時にカスタムの **%post** スクリプトを使用する要件を削除します。

オプション

- **--organization=** - 組織 ID を使用して CDN から RHEL を登録してインストールします。
- **--activation-key=** - アクティベーションキーを使用して、CDN から RHEL を登録してインストールします。使用するアクティベーションキーがサブスクリプションに登録されている限り、アクティベーションキーごとに 1 回使用するオプションを複数回使用できます。
- **--connect-to-insights** - ターゲットシステムを Red Hat Insights に接続します。
- **--proxy=** - HTTP プロキシーを設定します。
- **--server-hostname=** - 登録用の Satellite インスタンスのホスト名を設定します。
- **rhsm** キックスタートコマンドを使用してインストールソースリポジトリーを CDN に切り替えるには、次の条件を満たす必要があります。
  - カーネルコマンドラインで、**inst.stage2=<URL>** を使用してインストールイメージを取得したが、**inst.repo=** を使用してインストールソースを指定していない。
  - キックスタートファイルで、**url**、**cdrom**、**harddrive**、**liveimg**、**nfs**、および **ostree** セットアップコマンドを使用してインストールソースを指定していない。
- 起動オプションを使用して指定したインストールソース URL、またはキックスタートファイルに含まれるインストールソース URL は、キックスタートファイルに有効な認証情報を持つ **rhsm** コマンドが含まれている場合でも CDN よりも優先されます。システムが登録されていますが、URL インストールソースからインストールされています。これにより、以前のインストールプロセスが通常通りに動作するようになります。

B.2.17. shutdown

キックスタートコマンドの **shutdown** は任意です。インストールが正常に完了したら、システムをシャットダウンします。

構文

```
shutdown
```

注記

- キックスタートオプションの **shutdown** は、**shutdown** コマンドと同じです。詳細は、**shutdown(8)** の man ページを参照してください。
- その他の完了方法は、**halt**、**poweroff**、**reboot** などのキックスタートオプションをご覧ください。キックスタートファイルに完了方法が明示的には指定されていない場合は、**halt** オプションがデフォルトの完了方法になります。
- このコマンドにはオプションはありません。

B.2.18. sshpw

キックスタートコマンドの **sshpw** は任意です。

インストール中に、**SSH** 接続によりインストールプログラムと対話操作を行い、その進捗状況を監視できます。**sshpw** コマンドを使用して、ログオンに使用する一時的なアカウントを作成します。コマンドの各インスタンスにより、インストール環境でしか存在しない個別アカウントが作成されます。ここで作成されたアカウントは、インストールが完了したシステムには転送されません。

構文

```
sshpw --username=name [OPTIONS] password
```

必須オプション

- **--username**=name - ユーザー名を入力します。このオプションは必須です。
- password - このユーザーに使用するパスワードです。このオプションは必須です。

任意のオプション

- **--iscrypted** - このオプションを追加すると、パスワード引数はすでに暗号化済みと仮定されます。**--plaintext** と相互排他的になります。暗号化したパスワードを作成する場合は Python を使用します。

  ```
  $ python3 -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if (pw==getpass.getpass("Confirm: ")) else exit())'
  ```

  上記の例では、ランダムの salt を使用して、パスワードの sha512 暗号と互換性があるハッシュが生成されます。

- **--plaintext** - このオプションを使用すると、パスワードの引数はプレーンテキストであると仮定されます。**--iscrypted** と相互排他的になります。
- **--lock** - このオプションを指定すると、このアカウントはデフォルトでロックされます。つまり、ユーザーはコンソールからログインできなくなります。
- **--sshkey** - このオプションを指定すると、**<password>** 文字列が ssh 鍵の値として解釈されます。

注記

- デフォルトでは、**ssh** サーバーは、インストール時に起動しません。インストール時に **ssh** を使用できるようにするには、カーネル起動オプション **inst.sshd** を使用してシステムを起動します。
- インストール中、別のユーザーの **ssh** アクセスを許可する一方で、root の **ssh** アクセスを無効にする場合は、次のコマンドを実行します。

```
sshpw --username=example_username example_password --plaintext
sshpw --username=root example_password --lock
```

- 単に root の **ssh** アクセスを無効にするには、以下のコマンドを使用します。

```
sshpw --username=root example_password --lock
```

B.2.19. text

キックスタートコマンドの **text** は任意です。テキストモードでキックスタートインストールを実行します。キックスタートインストールは、デフォルトでグラフィカルモードで実行します。

構文

```
text [--non-interactive]
```

オプション

- **--non-interactive** - 完全に非対話式のモードでインストールを実行します。このモードでは、ユーザーの対話が必要になるとインストールを終了します。

注記

- 完全に自動となるインストールでは、キックスタートファイルで利用可能なモード (**graphical**、**text**、または **cmdline**) のいずれかを指定するか、起動オプション **console=** を使用する必要がある点に注意してください。モードが指定されていないと、可能な場合はグラフィカルモードが使用されるか、VNC モードおよびテキストモードからの選択が求められます。

B.2.20. url

キックスタートコマンドの **url** は任意です。これは、FTP、HTTP、または HTTPS プロトコルを使用して、リモートサーバーのインストールツリーイメージからインストールするのに使用されます。URL は 1 つだけ指定できます。

--url、**--metalink**、または **--mirrorlist** オプションのいずれかを指定する必要があります。

構文

```
url --url=FROM [OPTIONS]
```

オプション

- **--url=FROM** - インストール元となる **HTTP**、**HTTPS**、**FTP**、または **ファイル** の場所を指定します。

- **--mirrorlist=** - インストール元となるミラー URL を指定します。
- **--proxy=** - インストール時に使用する **HTTP**、**HTTPS**、または **FTP** プロキシーを指定します。
- **--noverifyssl** - **HTTPS** サーバーへの接続時に SSL 検証を無効にします。
- **--metalink=URL** - インストール元となるメタリンク URL を指定します。変数の置換は、**URL**の **$releasever** および **$basearch** で行います。

例

- HTTP サーバーからインストールするには、以下を行います。

```
url --url=http://server/path
```

- FTP サーバーからインストールするには、以下を行います。

```
url --url=ftp://username:password@server/path
```

注記

- 実際にインストールを実行するには、カーネルコマンドラインで **inst.repo** オプションが指定されていない限り、**cdrom**、**harddrive**、**hmc**、**nfs**、**liveimg**、**ostreesetup**、**rhsm**、または **url** のいずれかを指定する必要があります。

B.2.21. vnc

キックスタートコマンドの **vnc** は任意です。これにより、VNC を介して、リモートにグラフィカルインストールを表示できます。

テキストインストールではサイズと言語の一部が制限されるため、通常はテキストモードよりもこの方法が好まれます。追加のオプション指定がないと、このコマンドは、パスワードを使用せずに、インストールシステムで VNC サーバーを開始し、接続に必要な詳細を表示します。

構文

```
vnc [--host=host_name] [--port=port] [--password=password]
```

オプション

--host=

指定したホスト名でリッスンしている VNC ビューアープロセスに接続します。

--port=

リモート VNC ビューアープロセスがリッスンしているポートを指定します。このオプションを使用しないと、Anaconda は VNC のデフォルトポートである 5900 を使用します。

--password=

VNC セッションへの接続に必要なパスワードを設定します。これはオプションですが、推奨されます。

関連情報

- PXE を使用してネットワークからインストールするための準備

B.2.22. %include

キックスタートコマンドの **%include** は任意です。

%include コマンドを使用して、キックスタートファイル内の別のファイルのコンテンツが、キックスタートファイルの **%include** コマンドの場所にあるかのように設定します。

この包含は、**%pre** スクリプトセクションの後にのみ評価されるため、**%pre** セクションでスクリプトにより生成されたファイルに使用できます。**%pre** セクションを評価する前にファイルを指定するには、**%ksappend** コマンドを使用します。

構文

```
%include path/to/file
```

B.2.23. %ksappend

キックスタートコマンドの **%ksappend** は任意です。

%ksappend コマンドを使用して、キックスタートファイル内の別のファイルのコンテンツが、キックスタートファイルの **%ksappend** コマンドの場所にあるかのように設定します。

この包含は、**%include** コマンドで使用するのとは異なり、**%pre** スクリプトセクションの前に評価されます。

構文

```
%ksappend path/to/file
```

B.3. システム設定用キックスタートコマンド

このリストのキックスタートコマンドは、ユーザー、リポジトリー、サーバーなど、システムの詳細を設定します。

B.3.1. auth または authconfig (非推奨)

重要

非推奨になった **auth** または **authconfig** Kickstart コマンドではなく、新しい **authselect** コマンドを使用します。**auth** および **authconfig** は、一部の後方互換性としてのみ利用できます。

キックスタートコマンドの **auth** または **authconfig** は任意です。**authconfig** ツールを使用してシステムの認証オプションを設定します。インストール完了後もコマンドラインで実行できます。

構文

```
authconfig [OPTIONS]
```

注記

- キックスタートコマンドの **auth** または **authconfig** コマンドは、以前は **authconfig** ツールと

呼ばれていました。このツールは、Red Hat Enterprise Linux 8 では非推奨になりました。このキックスタートコマンドは、**authselect-compat** ツールを使用して、新しい **authselect** ツールを呼び出せるようになりました。互換性層の説明と、その既知の問題は、**authselect-migration(7)** の man ページを参照してください。インストールプログラムが自動的に非推奨のコマンドの使用を検出し、互換性層を提供するために、システムに **authselect-compat** パッケージをインストールします。

- デフォルトでは、パスワードがシャドウ化されています。
- 安全対策上、**SSL** プロトコルで OpenLDAP を使用する場合は、サーバー設定内の **SSLv2** および **SSLv3** のプロトコルを必ず無効にしてください。POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受けないようにするためです。詳細は https://access.redhat.com/solutions/1234843 を参照してください。

B.3.2. authselect

キックスタートコマンドの **authselect** は任意です。**authselect** コマンドを使用してシステムの認証オプションを設定します。インストール完了後もコマンドラインで実行できます。

構文

```
authselect [OPTIONS]
```

注記

- このコマンドは、すべてのオプションを **authselect** コマンドに渡します。詳細は、**authselect(8)** の man ページ、および **authselect --help** コマンドを参照してください。
- このコマンドは、Red Hat Enterprise Linux 8 で非推奨になった **auth** または **authconfig** コマンドを、**authconfig** ツールに置き換えます。
- デフォルトでは、パスワードがシャドウ化されています。
- 安全対策上、**SSL** プロトコルで OpenLDAP を使用する場合は、サーバー設定内の **SSLv2** および **SSLv3** のプロトコルを必ず無効にしてください。POODLE SSL 脆弱性 (CVE-2014-3566) の影響を受けないようにするためです。詳細は https://access.redhat.com/solutions/1234843 を参照してください。

B.3.3. firewall

キックスタートコマンドの **firewall** は任意です。インストール済みシステムにファイアウォール設定を指定します。

構文

```
firewall --enabled|--disabled [incoming] [OPTIONS]
```

必須オプション

- **--enabled** または **--enable** - DNS 応答や DHCP 要求など、発信要求に対する応答ではない着信接続を拒否します。このマシンで実行中のサービスへのアクセスが必要な場合は、特定サービスに対してファイアウォールの通過許可を選択できます。
- **--disabled** または **--disable** - iptable ルールを一切設定しません。

任意のオプション

- **--trust** - **em1** などのデバイスを指定することで、ファイアウォールを通過するこのデバイスへの着信トラフィックおよびこのデバイスからの発信トラフィックをすべて許可します。複数のデバイスをリスト表示するには、**--trust em1 --trust em2** などのオプションをさらに使用します。**--trust em1, em2** などのようなコンマ区切りは使用しないでください。

- **--remove-service** - サービスがファイアウォールを通過するのを許可しません。

- incoming - 指定したサービスがファイアウォールを通過できるように、以下のいずれかに置き換えます (複数のサービスを指定できます)。

  - **--ssh**

  - **--smtp**

  - **--http**

  - **--ftp**

- **--port=** - port:protocol の形式で指定したポートのファイアウォール通過を許可できます。たとえば、IMAP アクセスがファイアウォールを通過できるようにする場合は、**imap:tcp** と指定します。ポート番号を明示的に指定することもできます。ポート 1234 の UDP パケットを許可する場合は **1234:udp** と指定します。複数のポートを指定する場合は、コンマで区切って指定します。

- **--service=** - このオプションは、高レベルでサービスのファイアウォール通過を許可する方法です。サービスの中には複数のポートを開く必要があったり (**cups**、**avahi** など)、サービスが正常に動作するように特殊な設定を必要とするものがあります。このような場合は、**--port** オプションでポート単位での指定を行ったり、**--service=** を使用して必要なポートをすべて一度に開くことが可能です。
  firewalld パッケージ内の **firewall-offline-cmd** プログラムで認識できるオプションは、すべて使用できます。**firewalld** サービスを実行している場合は、**firewall-cmd --get-services** を実行すると、認識できるサービス名のリストが表示されます。

- **--use-system-defaults** - ファイアウォールを設定しません。このオプションにより、anaconda では何も実行せず、システムが、パッケージまたは ostree で提供されるデフォルトに依存するようになります。このオプションを他のオプションと共に使用すると、他のすべてのオプションは無視されます。

B.3.4. group

キックスタートコマンドの **group** は任意です。システムに新しいユーザーグループを作成します。

```
group --name=name [--gid=gid]
```

必須オプション

- **--name=** - グループ名を指定します。

任意のオプション

- **--gid=** - グループの GID です。指定しないとシステムの GID 以外で次に使用可能な GID がデフォルト設定されます。

注記

- 指定された名前や GID を持つグループが存在すると、このコマンドは失敗します。
- **user** コマンドは、新たに作成したユーザーに新しいグループを作成するのに使用できます。

B.3.5. keyboard (必須)

キックスタートコマンド **keyboard** が必要です。これは、システムに利用可能なキーボードレイアウトを 1 つまたは複数設定します。

構文

```
keyboard --vckeymap|--xlayouts OPTIONS
```

オプション

- **--vckeymap=** - 使用する **VConsole** キーマップを指定します。**/usr/lib/kbd/keymaps/xkb/** ディレクトリーの各ファイル名から **.map.gz** 拡張子を外したものが、有効なキーマップ名になります。
- **--xlayouts=** - 使用する X のレイアウトを、空白なしのコンマで区切ったリストで指定します。**setxkbmap(1)** と同じ形式 (**layout** 形式 (**cz** など)、または **layout (variant)** 形式 (**cz (qwerty)** など)) の値をとります。
  使用できるレイアウトは、man ページ **xkeyboard-config(7)** の **Layouts** を参照してください。
- **--switch=** - レイアウト切り替えのオプションリストを指定します (複数のキーボードレイアウト切り替え用のショートカット)。複数のオプションは、空白なしのコンマで区切ってください。**setxkbmap(1)** と同じ形式の値を受け取ります。
  使用できる切り替えオプションは、**xkeyboard-config(7)** の man ページの **Options** をご覧ください。

注記

- **--vckeymap=** オプションまたは **--xlayouts=** オプションのいずれかを使用する必要があります。

例

以下の例では、**--xlayouts=** オプションを使用して 2 種類のキーボードレイアウト (**English (US)** と **Czech (qwerty)**) を設定し、切り替えオプションは、**Alt**+**Shift** を使用するように指定しています。

```
keyboard --xlayouts=us,'cz (qwerty)' --switch=grp:alt_shift_toggle
```

B.3.6. lang (必須)

キックスタートコマンドの **lang** が必要です。これは、インストール時に使用する言語と、インストール済みシステムで使用するデフォルト言語を設定します。

構文

```
lang language [--addsupport=language,...]
```

必須オプション

- **language** - この言語のサポートをインストールし、システムのデフォルトとして設定します。

任意のオプション

- **--addsupport=** - 追加言語のサポートを指定します。空白を入れずコンマで区切った形式を受け取ります。以下に例を示します。

```
lang en_US --addsupport=cs_CZ,de_DE,en_UK
```

注記

- **locale -a | grep _** コマンドまたは **localectl list-locales | grep _** コマンドは、ロケールのリストを返します。
- テキストモードのインストールでは、特定の言語には対応していません (中国語、日本語、韓国語、インド系言語など)。**lang** コマンドでこの言語を指定しても、インストールプロセスは英語で続行します。ただし、インストール後のシステムでは選択した言語がデフォルトの言語として使用されます。

例

言語を英語に設定するには、キックスタートファイルに次の行が含まれている必要があります。

```
lang en_US
```

B.3.7. module

キックスタートコマンドの **module** は任意です。このコマンドを使用すると、キックスタートスクリプトでパッケージのモジュールストリームが有効になります。

構文

```
module --name=NAME [--stream=STREAM]
```

必須オプション

--name=

有効にするモジュールの名前を指定します。**NAME** を、実際の名前に置き換えます。

任意のオプション

--stream=

有効にするモジュールストリームの名前を指定します。**STREAM** を、実際の名前に置き換えます。デフォルトストリームが定義されているモジュールには、このオプションを指定する必要はありません。デフォルトストリームのないモジュールの場合、このオプションは必須であり省略するとエラーになります。異なるストリームでモジュールを複数回有効にすることはできません。

注記

- このコマンドと **%packages** セクションを組み合わせて使用すると、モジュールとストリーム

を明示的に指定せずに、有効なモジュールとストリームの組み合わせで提供されるパッケージをインストールできます。モジュールは、パッケージをインストールする前に有効にする必要があります。**module** コマンドでモジュールを有効にしたら、**%packages** セクションにパッケージのリストを追加することで、このモジュールで有効にしたパッケージをインストールできます。

- 1 つの **module** コマンドで、1 つのモジュールとストリームの組み合わせのみを有効にできます。複数のモジュールを有効にするには、複数の **module** コマンドを使用します。異なるストリームでモジュールを複数回有効にすることはできません。
- Red Hat Enterprise Linux 9 では、モジュールは AppStream リポジトリーにのみ存在します。利用可能なモジュールのリストを表示するには、インストールされている Red Hat Enterprise Linux 9 システムで **dnf module list** コマンドを実行します。

関連情報

- DNF ツールを使用したソフトウェアの管理

B.3.8. repo

キックスタートコマンドの **repo** は任意です。パッケージインストール用のソースとして使用可能な追加の dnf リポジトリーを設定します。複数の **repo** 行を追加できます。

構文

```
repo --name=repoid [--baseurl=url|--mirrorlist=url|--metalink=url] [OPTIONS]
```

必須オプション

- **--name=** - リポジトリー ID を入力します。このオプションは必須です。以前に追加したリポジトリーと名前が競合する場合は無視されます。インストールプログラムでは事前設定したリポジトリーのリストが使用されるため、このリストにあるリポジトリーと同じ名前のものは追加できません。

URL オプション

これらのオプションは相互排他的で、オプションです。ここでは、dnf のリポジトリーの設定ファイル内で使用できる変数はサポートされません。文字列 **$releasever** および **$basearch** を使用できます。これは、URL の該当する値に置き換えられます。

- **--baseurl=** - リポジトリーの URL を入力します。
- **--mirrorlist=** - リポジトリーのミラーのリストを指す URL を入力します。
- **--metalink=** - リポジトリーのメタリンクを持つ URL です。

任意のオプション

- **--install** - 指定したリポジトリーの設定を、インストールしたシステムの **/etc/yum.repos.d/** ディレクトリーに保存します。このオプションを使用しない場合は、キックスタートファイルで設定したリポジトリーの使用はインストール中に限られ、インストール後のシステムでは使用できません。

- **--cost=** - このリポジトリーに割り当てるコストを整数で入力します。複数のリポジトリーで同じパッケージを提供している場合に、リポジトリーの使用優先順位がこの数値で決まります。コストの低いリポジトリーは、コストの高いリポジトリーよりも優先されます。

- **--excludepkgs=** - このリポジトリーからは読み出しては**ならない**パッケージ名のリストをコンマ区切りで指定します。複数のリポジトリーで同じパッケージが提供されていて、特定のリポジトリーから読み出す場合に便利なオプションです。(**publican** といった) 完全なパッケージ名と (**gnome-*** といった) グロブの両方が使えます。

- **--includepkgs=** - このリポジトリーから取得できるパッケージ名およびグロブのリストをコンマ区切りで指定します。リポジトリーが提供するその他のパッケージは無視されます。これは、リポジトリーが提供する他のパッケージをすべて除外しながら、リポジトリーから1つのパッケージまたはパッケージセットをインストールする場合に便利です。

- **--proxy=[protocol://][username[:password]@]host[:port]** - このリポジトリーにだけ使用するHTTP/HTTPS/FTP プロキシーを指定します。この設定は他のリポジトリーには影響しません。また、HTTP インストールでは **install.img** の読み込みについても影響はありません。

- **--noverifyssl** - **HTTPS** サーバーへの接続の際に、SSL 確認を無効にします。

注記

- インストールに使用するリポジトリーは安定した状態を維持してください。インストールが終了する前にリポジトリーに変更が加えられると、インストールが失敗する可能があります。

B.3.9. rootpw (必須)

キックスタートコマンドの **rootpw** が必要です。システムの root パスワードを **password** 引数に設定します。

構文

```
rootpw [--iscrypted|--plaintext] [--lock] password
```

必須オプション

- **password** - パスワード指定。プレーンテキストまたは暗号化された文字列。以下の **--iscrypted** および **--plaintext** を参照してください。

オプション

- **--iscrypted** - このオプションを追加すると、パスワード引数はすでに暗号化済みと仮定されます。**--plaintext** と相互排他的になります。暗号化したパスワードを作成する場合は python を使用します。

  ```
  $ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if (pw==getpass.getpass("Confirm: ")) else exit())'
  ```

  上記の例では、ランダムの salt を使用して、パスワードの sha512 暗号と互換性があるハッシュが生成されます。

- **--plaintext** - このオプションを使用すると、パスワードの引数はプレーンテキストであると仮定されます。**--iscrypted** と相互排他的になります。

- **--lock** - このオプションを含めると、アカウントはデフォルトでロックされます。このユ

- **--lock** - このオプションを含めると、root アカウントはデフォルトでロックされます。つまり、root ユーザーはコンソールからログインできなくなります。また、グラフィカルおよびテキストベースの手動インストールで、**Root Password** ウィンドウが無効になります。

- **--allow-ssh** - このオプションを指定すると、root ユーザーは SSH でパスワードを使用してシステムにログインできます。このオプションは、RHEL 9.1 以降でのみ利用できます。

password-based SSH root logins を有効にするには、キックスタートインストールメソッドで次の行をキックスタートファイルに追加します。RHEL 9.0 では、**--allow-ssh** オプションは利用できません。

```
%post
echo "PermitRootLogin yes" > /etc/ssh/sshd_config.d/01-permitrootlogin.conf
%end
```

B.3.10. selinux

キックスタートコマンドの **selinux** は任意です。インストール済みシステムの SELinux の状態を設定します。デフォルトの SELinux ポリシーは **enforcing** です。

構文

```
selinux [--disabled|--enforcing|--permissive]
```

オプション

--enforcing

SELinux をデフォルトの対象ポリシーである **enforcing** で有効にします。

--permissive

SELinux のポリシーに基づく警告を出力します。ただし、実際にはポリシーは実施されません。

--disabled

システムで SELinux を完全に無効にします。

関連情報

- Using SElinux

B.3.11. services

キックスタートコマンドの **services** は任意です。デフォルトの systemd ターゲット下で実行するデフォルトのサービスセットを変更します。無効にするサービスのリストは、有効にするサービスのリストの前に処理されます。したがって、サービスが両方のリストに記載されていると、そのサービスは有効になります。

構文

```
services [--disabled=list] [--enabled=list]
```

オプション

- **--disabled=** - 無効にするサービスをコンマ区切りで指定します。

- **--enabled=** - 有効にするサービスをコンマ区切りで指定します。

注記

- **services** 要素を使用して **systemd** サービスを有効にする場合は、指定されたサービスファイルを含むパッケージを **%packages** セクションに含めるようにしてください。
- 複数のサービスは、スペースを入れずにコンマで区切って含める必要があります。たとえば、4 つのサービスを無効にするには、次のように入力します。

```
services --disabled=auditd,cups,smartd,nfslock
```

スペースを含めると、Kickstart は最初のスペースまでのサービスのみを有効化または無効化します。以下に例を示します。

```
services --disabled=auditd, cups, smartd, nfslock
```

この場合は、**auditd** サービスしか無効になりません。4 つのサービスをすべて無効にするには、エントリーから空白を取り除きます。

B.3.12. skipx

キックスタートコマンドの **skipx** は任意です。存在する場合は、インストール済みシステムで X が設定されていません。

パッケージ選択のオプションでディスプレイマネージャーをインストールすると、このパッケージにより X の設定が作成されるため、インストールが完了したシステムは **graphical.target** にデフォルト設定されることになります。これにより、**skipx** オプションが無効になります。

構文

```
skipx
```

注記

- このコマンドにはオプションはありません。

B.3.13. sshkey

キックスタートコマンドの **sshkey** は任意です。インストール済みシステムで、指定したユーザーの **authorized_keys** ファイルに SSH キーを追加します。

構文

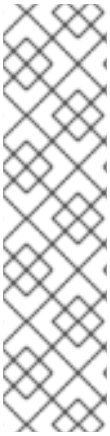
```
sshkey --username=user "ssh_key"
```

必須オプション

- **--username=** - 鍵をインストールするユーザー。
- **ssh_key** - 完全な SSH 鍵のフィンガープリント。引用符でラップする必要があります。

B.3.14. syspurpose

キックスタートコマンドの **syspurpose** は任意です。インストール後にシステムがどのように使用されるかを説明するシステムの目的を設定します。この情報により、適切なサブスクリプションエンタイトルメントがシステムに適用されます。



注記

Red Hat Enterprise Linux 9.0 以降では、1 つの **subscription-manager syspurpose** モジュールで **role**、**service-level**、**usage**、および **addons** サブコマンドを利用可能にすることで、1 つのモジュールでシステムの目的の属性を管理および表示できます。以前は、システム管理者は 4 つのスタンドアロンの **syspurpose** コマンドのいずれかを使用して各属性を管理していました。このスタンドアロンの **syspurpose** コマンドは RHEL 9.0 以降非推奨となり、RHEL 9 以降では削除される予定です。Red Hat は、現在のリリースのライフサイクル中にバグ修正とこの機能に対するバグ修正やサポートを提供しますが、この機能は機能強化の対象外となります。RHEL 9 以降、単一の **subscription-manager syspurpose** コマンドとその関連のサブコマンドは、システムの目的を使用する唯一の方法です。

構文

```
syspurpose [OPTIONS]
```

オプション

- **--role=** - 希望するシステムロールを設定します。利用できる値は次のとおりです。
  - Red Hat Enterprise Linux Server
  - Red Hat Enterprise Linux Workstation
  - Red Hat Enterprise Linux Compute Node
- **--sla=** - サービスレベルアグリーメントを設定します。利用できる値は次のとおりです。
  - Premium
  - Standard
  - Self-Support
- **--usage=** - システムの使用方法。利用できる値は次のとおりです。
  - Production
  - Disaster Recovery
  - Development/Test
- **--addon=** - のレイヤード製品または機能を指定します。このオプションは複数回使用できます。

注記

- スペースで値を入力し、二重引用符で囲みます。

```
syspurpose --role="Red Hat Enterprise Linux Server"
```

- システムの目的を設定することが強く推奨されますが、Red Hat Enterprise Linux インストールプログラムでは任意の機能です。

B.3.15. timezone (必須)

キックスタートコマンド **timezone** が必要です。システムのタイムゾーンを設定します。

構文

```
timezone timezone [OPTIONS]
```

必須オプション

- **timezone** - システムに設定するタイムゾーン

任意のオプション

- **--utc** - これを指定すると、ハードウェアクロックが UTC (グリニッジ標準) 時間に設定されているとシステムは見なします。
- **--nontp** - NTP サービスの自動スタートを無効にします。このオプションは非推奨となりました。
- **--ntpservers=** - 使用する NTP サーバーを空白を入れないコンマ区切りのリストで指定します。このオプションは非推奨になりました。代わりに **timesource** コマンドを使用してください。

注記

Red Hat Enterprise Linux 9 では、タイムゾーン名は **pytz** パッケージにより提供される **pytz.common_timezones** のリストを使用して検証されます。

B.3.16. timesource(任意)

キックスタートコマンドの **timesource** は任意です。これを使用して、タイムデータを提供する NTP、NTS サーバー、およびプールを設定し、システムで NTP サービスを有効または無効にするかどうかも制御します。

構文

```
timesource [--ntp-server NTP_SERVER | --ntp-pool NTP_POOL | --ntp-disable] [--nts]
```

必須オプション

timesource コマンドを使用する場合は、以下のいずれかのオプションを指定する必要があります。

- **--ntp-server** - 1 つの NTP サーバーをタイムソースとして追加します。このオプションは、1 つの NTP タイムソースサーバーを追加するために、1 つのコマンドに 1 回だけ追加できます。複数のソースを追加するには、毎回それぞれ 1 つの --ntp-server オプションまたは --ntp-pool オプションを使用して、複数の timesource コマンドを追加します。たとえば、**Europe** のタイムゾーンに複数のソースを追加するには、以下のコマンドを実行します。

```
timezone Europe
timesource --ntp-server 0.rhel.pool.ntp.org
timesource --ntp-server 1.rhel.pool.ntp.org
```

```
timesource --ntp-server 2.rhel.pool.ntp.org
```

- **--ntp-pool** - タイムソースとして NTP サーバープールを追加します。このオプションは、1 つの NTP タイムソースプールを追加するために 1 回だけ追加できます。timesource コマンドを繰り返し、複数のソースを追加します。
- **--ntp-disable** - インストール済みシステムの NTP タイムソースを無効にします。

任意のオプション

- **--nts** - このコマンドで追加されたサーバーまたはプールは NTS プロトコルを使用します。このオプションは --ntp-disable を使用しても追加できますが、効果はありません。

注記

- **timezone** コマンドの **--ntpservers** オプションが非推奨になりました。Red Hat は、この新しいオプションを **timesource** コマンドの表現機能に使用することを推奨します。
- **timesource** コマンドのみが、サーバーとプールをプレーン **NTP** プロトコルではなく、**NTS** を使用するものとしてマークできます。

B.3.17. user

キックスタートコマンドの **user** は任意です。システムに新しいユーザーを作成します。

構文

```
user --name=username [OPTIONS]
```

必須オプション

- **--name=** - ユーザー名を入力します。このオプションは必須です。

任意のオプション

- **--gecos=** - ユーザーの GECOS 情報を指定します。これは、コンマ区切りのさまざまなシステム固有フィールドの文字列です。ユーザーのフルネームやオフィス番号などを指定するのに使用されます。詳細は、**passwd(5)** の man ページを参照してください。
- **--groups=** - デフォルトグループの他にもユーザーが所属すべきグループ名のコンマ区切りのリストです。このグループは、ユーザーアカウントの作成前に存在する必要があります。詳細は、**group** コマンドを参照してください。
- **--homedir=** - ユーザーのホームディレクトリーです。設定しない場合は、**/home/username** がデフォルトになります。
- **--lock** - このオプションを指定すると、このアカウントはデフォルトでロックされます。つまり、ユーザーはコンソールからログインできなくなります。また、グラフィカルおよびテキストベースの手動インストールで、**ユーザーの作成** ウィンドウが無効になります。
- **--password=** - 新規のユーザーパスワードです。指定しないと、そのアカウントはデフォルトでロックされます。

- **--iscrypted** - このオプションを追加すると、パスワード引数はすでに暗号化済みと仮定されます。**--plaintext** と相互排他的になります。暗号化したパスワードを作成する場合は python を使用します。

```
$ python -c 'import crypt,getpass;pw=getpass.getpass();print(crypt.crypt(pw) if
(pw==getpass.getpass("Confirm: ")) else exit())'
```

  上記の例では、ランダムの salt を使用して、パスワードの sha512 暗号と互換性があるハッシュが生成されます。

- **--plaintext** - このオプションを使用すると、パスワードの引数はプレーンテキストであると仮定されます。**--iscrypted** と相互排他的になります。

- **--shell=** - ユーザーのログインシェルです。指定しないと、システムのデフォルトが使用されます。

- **--uid=** - ユーザーの UID (User ID) です。指定しないと、次に利用可能なシステム以外の UID をデフォルトにします。

- **--gid=** - ユーザーのグループで使用される GID (Group ID) です。指定しないと、次に利用可能なシステム以外のグループ ID をデフォルトにします。

注記

- **--uid** と **--gid** のオプションを使用して、通常のユーザーとそのデフォルトグループに **1000** ではなく **5000** から始まる範囲の ID を設定することを検討してください。これは、システムユーザーおよびグループに予約してある **0**–**999** の範囲が今後広がり、通常のユーザーの ID と重複する可能性があるためです。

- ファイルおよびディレクトリーはさまざまなパーミッションで作成され、パーミッションは、ファイルまたはディレクトリーを作成するアプリケーションによる影響を受けます。たとえば、**mkdir** コマンドは、すべてのパーミッションを有効にしてディレクトリーを作成します。ただし、**user file-creation mask** 設定で指定されたように、アプリケーションは、新規に作成したファイルに特定パーミッションを付与しません。
  user file-creation mask は、**umask** コマンドで管理できます。新規ユーザー向けの **user file-creation mask** のデフォルト設定は、インストール済みシステムの **/etc/login.defs** 設定ファイルの **UMASK** 変数で定義されます。これを設定しない場合は、デフォルト値 **022** を使用します。デフォルト値を使用し、アプリケーションがファイルを作成した場合は、ファイルの所有者以外のユーザーに書き込みパーミッションが付与されません。ただし、これは他の設定やスクリプトで無効にできます。

B.3.18. xconfig

キックスタートコマンドの **xconfig** は任意です。X Window System を設定します。

構文

```
xconfig [--startxonboot]
```

オプション

- **--startxonboot** - インストール済みシステムでグラフィカルログインを使用します。

注記

- Red Hat Enterprise Linux 9 には KDE デスクトップ環境が含まれていないため、アップストリームに記載されている **--defaultdesktop=** を使用しないでください。

B.4. ネットワーク設定用キックスタートコマンド

このリストのキックスタートコマンドにより、システムにネットワークを設定できます。

B.4.1. ネットワーク (任意)

オプションの **network** キックスタートコマンドを使用して、ターゲットシステムのネットワーク情報を設定し、インストール環境でネットワークデバイスをアクティブにします。最初の **network** コマンドで指定しているデバイスが自動的にアクティベートされます。**--activate** オプションを使用して、デバイスを明示的にアクティブ化するように要求することもできます。

構文

```
network OPTIONS
```

オプション

- **--activate** - インストール環境でこのデバイスをアクティブにします。
  アクティブしているデバイスに **--activate** オプションを使用すると (たとえば、キックスタートファイルを取得できるように起動オプションで設定したインターフェイスなど)、キックスタートファイルで指定している詳細を使用するようにデバイスが再度アクティブになります。

  デバイスにデフォルトのルートを使用させないようにする場合は **--nodefroute** オプションを使用します。

- **--no-activate** - インストール環境でこのデバイスをアクティブにしません。
  デフォルトでは、**--activate** オプションにかかわらず、Anaconda はキックスタートファイルの 1 番目のネットワークデバイスをアクティブにします。**--no-activate** オプションを使用して、デフォルトの設定を無効にできます。

- **--bootproto=** - **dhcp**、**bootp**、**ibft**、または **static** のいずれかになります。**dhcp** がデフォルトのオプションになります。**dhcp** と **bootp** は同じように処理されます。デバイスの **ipv4** 設定を無効にするには、**--noipv4** オプションを使用します。

  注記

  このオプションは、デバイスの ipv4 設定を行います。ipv6 の設定には、**--ipv6** オプションおよび **--ipv6gateway** オプションを使用します。

  DHCP メソッドでは、DHCP サーバーシステムを使用してネットワーク設定を取得します。BOOTP メソッドも同様で、BOOTP サーバーがネットワーク設定を提供する必要があります。システムが DHCP を使用するようにする場合は、以下のように指定します。

  ```
  network --bootproto=dhcp
  ```

  BOOTP を使用してネットワーク設定を取得する場合は、キックスタートファイルで次の行を使用します。

  ```
  network --bootproto=bootp
  ```

iBFT で指定されている設定を使用する場合は、以下のようにします。

```
network --bootproto=ibft
```

static メソッドの場合は、キックスタートファイルに IP アドレスおよびネットマスクを指定する必要があります。これらの情報は静的となるため、インストール時およびインストール後にも使用されます。

静的なネットワーク設定情報はすべて **一行で** 指定する必要があります。コマンドラインのようにバックスラッシュ (\) を使用して行を折り返すことはできません。

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --
nameserver=10.0.2.1
```

ネームサーバーは同時に複数設定することもできます。以下のように、1つの **--nameserver=** オプションに対して、ネームサーバーの IP アドレスをコンマ区切りで指定します。

```
network --bootproto=static --ip=10.0.2.15 --netmask=255.255.255.0 --gateway=10.0.2.254 --
nameserver=192.168.2.1,192.168.3.1
```

- **--device=** - **network** コマンドで設定する (また最終的に Anaconda でアクティベートさせる) デバイスを指定します。
  最初 に使用される **network** コマンドに **--device=** オプションがない場合は、Anaconda の起動オプション **inst.ks.device=** の値が使用されます (使用可能な場合)。ただし、この動作は廃止が予定されているため注意してください。ほとんどの場合において、すべての **network** コマンドには必ず **--device=** オプションを指定してください。

重要

Red Hat Enterprise Linux 9 では、ネットワークチーミングが非推奨になりました。代わりに、ネットワークボンディングドライバーの使用を検討してください。詳細は、Configuring network bonding を参照してください。

同じキックスタートファイルに記載される 2 番目以降の **network** コマンドの動作は、**--device=** オプションを指定しないと詳細が不明になります。1番目以降の **network** コマンドに、このオプションを指定していることを確認してください。

起動するデバイスは、以下のいずれかの方法で指定します。

  - インターフェイスのデバイス名を使用して指定する (**em1** など)
  - インターフェイスの MAC アドレスを使用して指定する (**01:23:45:67:89:ab** など)
  - **link** キーワードを使用する (リンクが **up** 状態になっている1番目のインターフェイス)。
  - キーワード **bootif** を使用する。これは、pxelinux が **BOOTIF** 変数に設定した MAC アドレスを使用します。pxelinux に **BOOTIF** 変数を設定する場合は、**pxelinux.cfg** ファイルに **IPAPPEND 2** を設定します。

以下に例を示します。

```
network --bootproto=dhcp --device=em1
```

- **--ipv4-dns-search/--ipv6-dns-search** - DNS 検索ドメインを手動で設定します。これらのオプションを **--device** オプションと一緒に使用し、それぞれの NetworkManager プロパティーをミラーリングする必要があります。次に例を示します。

```
network --device ens3 --ipv4-dns-search domain1.example.com,domain2.example.com
```

- **--ipv4-ignore-auto-dns/--ipv6-ignore-auto-dns** - DHCP からの DNS 設定を無視するように設定します。これらのオプションは **--device** オプションと一緒に使用する必要があります。これらのオプションには引数は必要ありません。

- **--ip=** - デバイスの IP アドレスを指定します。

- **--ipv6=** - デバイスの IPv6 アドレスを **address**[/**prefix length**] の形式で指定します (例: **3ffe:ffff:0:1::1/128**)。**prefix** を省略すると、**64** が使用されます。**auto** を使用すると自動設定に、**dhcp** を使用すると DHCPv6 限定の設定 (ルーター広告なし) となります。

- **--gateway=** - 1 つの IPv4 アドレスのデフォルトゲートウェイを指定します。

- **--ipv6gateway=** - 1 つの IPv6 アドレスのデフォルトゲートウェイを指定します。

- **--nodefroute** - インターフェイスがデフォルトのルートとして設定されないようにします。iSCSI ターゲット用に用意した別のサブネットにある NIC など、**--activate=** オプションで追加デバイスを起動させる場合は、このオプションを使用してください。

- **--nameserver=** - IP アドレスに DNS ネームサーバーを指定します。複数のネームサーバーを指定する場合は、1 つの オプションに対して、IP アドレスをコンマ区切りで指定します。

- **--netmask=** - インストール後のシステムのネットワークマスクを指定します。

- **--hostname=** - ターゲットシステムのホスト名を設定するために使用されます。ホスト名は、**hostname.domainname** 形式の完全修飾ドメイン名 (FQDN)、またはドメインなしの短縮ホスト名のいずれかにします。多くのネットワークには、自動的に接続したシステムにドメイン名を提供する DHCP (Dynamic Host Configuration Protocol) サービスがあります。DHCP サービスが、このマシンにドメイン名を割り当てるようにするには、短縮ホスト名のみを指定してください。
  静的 IP およびホスト名の設定を使用する場合、短縮名または FQDN を使用するかどうかは、計画したシステムのユースケースによって異なります。Red Hat Identity Management はプロビジョニング時に FQDN を設定しますが、サードパーティーのソフトウェア製品によっては短縮名が必要になる場合があります。いずれの場合も、すべての状況で両方のフォームの可用性を確保するには、**IP FQDN short-alias** の形式で **/etc/hosts** にホストのエントリーを追加します。

  ホスト名に使用できるのは、英数字と **-** または **.** のみです。ホスト名は 64 文字以下である必要があります。ホスト名は、**-** および **.** で開始したり終了したりできません。DNS に準拠するには、FQDN の各部分は 63 文字以下で、ドットを含む FQDN の合計の長さは 255 文字を超えることができません。

  ターゲットシステムのホスト名のみを設定する場合は、**network** コマンドで **--hostname** オプションを使用し、他のオプションは含めないでください。

  ホスト名の設定時に追加オプションを指定すると、**network** コマンドは指定したオプションを使用してデバイスを設定します。**--device** オプションを使用して設定するデバイスを指定しないと、デフォルトの **--device link** の値が使用されます。また、**--bootproto** オプションを使用してプロトコルを指定しないと、デバイスはデフォルトで DHCP を使用するように設定されます。

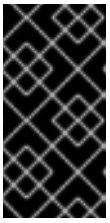
- **--ethtool=** - ethtool プログラムに渡されるネットワークデバイスの低レベルの追加設定を指定します。
- **--onboot=** - システムの起動時にデバイスを有効にするかどうかを指定します。
- **--dhcpclass=** - DHCP クラスを指定します。
- **--mtu=** - デバイスの MTU を指定します。
- **--noipv4** - このデバイスで IPv4 を無効にします。
- **--noipv6** - このデバイスで IPv6 を無効にします。
- **--bondslaves=** - このオプションを使用すると、**--bondslaves=** オプションで定義されたセカンダリーデバイスを使用して、**--device=** オプションで指定したボンディングデバイスが作成されます。以下に例を示します。

```
network --device=bond0 --bondslaves=em1,em2
```

上記のコマンドは、インターフェイスの **em1** および **em2** をセカンダリーデバイスとして使用し、ボンドデバイス **bond0** を作成します。

- オプションの **--bondopts=** - **--bondslaves=** および **--device=** を使用して指定されるボンドインターフェイス用のオプションパラメーターのリストです。このリスト内のオプションは必ずコンマ (,) またはセミコロン (;) で区切ってください。オプション自体にコンマが含まれている場合はセミコロンを使用してください。以下に例を示します。

```
network --bondopts=mode=active-backup,balance-rr;primary=eth1
```



重要

--bondopts=mode= パラメーターは、**balance-rr** や **broadcast** などのフルモード名にしか対応しません。**0** や **3** などの数値による表記には対応していません。利用可能なモードおよびサポートされるモードの一覧は、Configuring and Managing Networking Guide を参照してください。

- **--vlanid=** - **--device=** で指定したデバイスを親として作成する仮想デバイスの仮想 LAN (VLAN) の ID 番号 (802.1q タグ) を指定します。たとえば、**network --device=em1 --vlanid=171** を使用すると仮想 LAN デバイスの **em1.171** が作成されます。
- **--interfacename=** - 仮想 LAN デバイスのカスタムのインターフェイス名を指定します。**--vlanid=** オプションで生成されるデフォルト名が望ましくない場合に使用してください。**--vlanid=** と併用する必要があります。以下に例を示します。

```
network --device=em1 --vlanid=171 --interfacename=vlan171
```

上記のコマンドにより、**em1** デバイスに ID **171** の仮想 LAN インターフェイス **vlan171** が作成されます。

インターフェイスには任意の名前 (**my-vlan** など) を付けることができますが、場合によっては次の命名規則に従う必要があります。

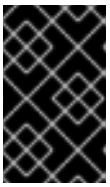
  - 名前にドット (.) が含まれている場合は、**NAME.ID** の形にする必要があります。NAME は任意ですが、**ID** は VLAN ID にする必要があります。たとえば、**em1.171**、**my-vlan.171** などにします。

- **vlan** で開始する名前を付ける場合は、**vlanID** の形式にする必要があります。たとえば、**vlan171** などにします。

- **--teamslaves=** - このオプションで指定したセカンダリーデバイスを使用して、**--device=** オプションで指定したチームデバイスが作成されます。セカンダリーデバイスはコンマで区切ります。各セカンダリーデバイスの後ろにその設定を指定できます。\ 記号でエスケープした二重引用符で、一重引用符の JSON 文字列を囲っている部分が実際の設定になります。以下に例を示します。

```
network --teamslaves="p3p1'{\"prio\": -10, \"sticky\": true}',p3p2'{\"prio\": 100}'"
```

--teamconfig= オプションも参照してください。



重要

Red Hat Enterprise Linux 9 では、ネットワークチーミングが非推奨になりました。代わりに、ネットワークボンディングドライバーの使用を検討してください。詳細は、Configuring network bonding を参照してください。

- **--teamconfig=** - チームデバイスの設定を二重引用符で囲って指定します。これは、二重引用符と \ 記号でエスケープした JSON 文字列になります。デバイス名は、**--device=** オプションで指定し、セカンダリーデバイスとその設定は、**--teamslaves=** オプションで指定します。以下に例を示します。

```
network --device team0 --activate --bootproto static --ip=10.34.102.222 --
netmask=255.255.255.0 --gateway=10.34.102.254 --nameserver=10.34.39.2 --
teamslaves="p3p1'{\"prio\": -10, \"sticky\": true}',p3p2'{\"prio\": 100}'" --teamconfig="
{\"runner\": {\"name\": \"activebackup\"}}"
```

重要

Red Hat Enterprise Linux 9 では、ネットワークチーミングが非推奨になりました。代わりに、ネットワークボンディングドライバーの使用を検討してください。詳細は、Configuring network bonding を参照してください。

- **--bridgeslaves=** - このオプションを使用すると、**--device=** オプションで指定したデバイス名でネットワークブリッジが作成され、このネットワークブリッジに、**--bridgeslaves=** オプションで指定したデバイスが追加されます。以下に例を示します。

```
network --device=bridge0 --bridgeslaves=em1
```

- **--bridgeopts=** - オプションでブリッジしたインターフェイス用パラメーターのリストをコンマで区切って指定します。使用できる値は **stp**、**priority**、**forward-delay**、**hello-time**、**max-age**、**ageing-time** などです。これらのパラメーターの詳細は、**nm-settings(5)** man ページまたは ネットワーク設定仕様 にある **ブリッジ設定** の表を参照してください。
ネットワークブリッジの一般情報は、Configuring and managing networking も参照してください。

- **--bindto=mac** - インストールされたシステムのデバイス設定ファイルをインターフェイス名 (**DEVICE**) へのデフォルトのバインドではなく、デバイスの MAC アドレス (**HWADDR**) にバインドします。このオプションは **--device=** オプションとは独立している点に注意してください。同じ **network** コマンドでデバイス名、または **link**、**bootif** が指定されていても、**--bindto=mac** が適用されます。

注記

- 命名方法の変更により、Red Hat Enterprise Linux では **eth0** などの **ethN** デバイス名を使用できなくなりました。デバイスの命名スキームの詳細は、アップストリームドキュメント Predictable Network Interface Names を参照してください。
- キックスタートのオプションまたは起動オプションを使用して、ネットワークにあるインストールリポジトリーを指定したものの、インストール開始時にネットワークが利用できない状態になっている場合は、**インストール概要** ウィンドウが表示される前に、ネットワーク接続の設定を求める **ネットワークの設定** ウィンドウが表示されます。詳細は、**標準的な RHEL 9 インストールの実行** ドキュメントの ネットワークおよびホスト名のオプションの設定 セクションを参照してください。

B.4.2. realm

キックスタートコマンドの **realm** は任意です。Active Directory や IPA ドメインを参加させます。このコマンドの詳細は、man ページ **realm(8)** の **join** のセクションを参照してください。

構文

```
realm join [OPTIONS] domain
```

必須オプション

- **domain** - 参加するドメイン。

オプション

- **--computer-ou=OU=** - コンピューターアカウントを作成するために、組織単位の識別名を指定します。識別名の形式は、クライアントソフトウェアおよびメンバーシップのソフトウェアにより異なります。通常、識別名のルート DSE の部分は省略できます。
- **--no-password** - パスワードの入力なしで自動的に参加します。
- **--one-time-password=** - ワンタイムパスワードを使用して参加します。すべてのレルムで使用できるとは限りません。
- **--client-software=** - ここで指定したクライアントソフトウェアを実行できるレルムにしか参加しません。使用できる値は **sssd** や **winbind** などになります。すべてのレルムがすべての値に対応しているとは限りません。デフォルトでは、クライアントソフトウェアは自動的に選択されます。
- **--server-software=** - ここで指定したサーバーソフトウェアを実行できるレルムにしか参加しません。使用できる値は **active-directory** や **freeipa** などになります。
- **--membership-software=** - レルムに参加する際に、ここに指定したソフトウェアを使用します。使用できる値は **samba** や **adcli** などになります。すべてのレルムがすべての値に対応しているとは限りません。デフォルトでは、メンバーシップソフトウェアは自動的に選択されます。

B.5. ストレージを処理するキックスタートコマンド

本セクションのキックスタートコマンドは、デバイス、ディスク、パーティション、LVM、ファイルシステムなど、ストレージの設定を行います。

重要

sdX (または **/dev/sdX**) 形式では、デバイス名が再起動後に維持される保証がないため、一部のキックスタートコマンドの使用が複雑になる可能性があります。コマンドにデバイスノード名が必要な場合は、**/dev/disk** の項目を代わりに使用できます。以下に例を示します。

part / --fstype=xfs --onpart=sda1

上記のコマンドの代わりに、以下のいずれかを使用します。

part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1

part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1

このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR/dev/disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

B.5.1. autopart

キックスタートコマンドの **autopart** は任意です。自動的にパーティションを作成します。

自動的に作成されるパーティション - ルート (/) パーティション (1 GB 以上)、**swap** パーティション、アーキテクチャーに応じた **/boot** パーティション。容量が十分にあるドライブの場合 (50 GiB 以上)、**/home** パーティションも作成されます。

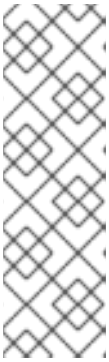
構文

```
autopart OPTIONS
```

オプション

- **--type=** - 事前定義済み自動パーティション設定スキームの中から、使用するスキームを選択します。次の値を取ります。
  - **lvm** - LVM パーティション設定スキーム
  - **plain** - LVM がない普通のパーティション
  - **thinp** - LVM シンプロビジョニングのパーティション設定スキーム
- **--fstype=** - 利用可能なファイルシステムのタイプを選択します。利用可能な値は、**ext2**、**ext3**、**ext4**、**xfs**、および **vfat** です。デフォルトのファイルシステムは **xfs** です。
- **--nohome** - **/home** パーティションの自動作成を無効にします。
- **--nolvm** - 自動パーティション設定に LVM を使用しません。このオプションは **--type=plain** と同じです。
- **--noboot** - **/boot** パーティションを作成しません。
- **--noswap** - swap パーティションを作成しません。

- **--encrypted** - Linux Unified Key Setup (LUKS) ですべてのパーティションを暗号化します。手動によるグラフィカルインストールを行った際の初期パーティション設定ウィンドウで表示される **Encrypt partitions (パーティションの暗号化)** のチェックボックスと同じです。



注記

1 つまたは複数のパーティションを暗号化する際には、安全な暗号化を行うため、Anaconda が 256 ビットのエントロピーを収集しようとします。エントロピーの収集には時間がかかる場合があります。十分なエントロピーが収集されたかどうかにかかわらず、このプロセスは最大 10 分後に終了します。

プロセスは、インストールシステムと対話することにより高速化できます (キーボードで入力またはマウスの移動)。仮想マシンにインストールしている場合は、**virtio-rng** デバイス (仮想乱数ジェネレーター) をゲストに登録できます。

- **--luks-version=LUKS_VERSION** - ファイルシステムの暗号化に使用する LUKS 形式のバージョンを指定します。**--encrypted** と併用しないと有効ではありません。

- **--passphrase=** - 暗号化した全デバイスに、デフォルトのシステムワイドパスフレーズを指定します。

- **--escrowcert=URL_of_X.509_certificate** - 暗号化した全ボリュームのデータ暗号化の鍵を **/root** 配下にファイル形式で格納します。**URL_of_X.509_certificate** で指定した URL の X.509 証明書を使用して暗号化します。鍵は暗号化したボリュームごとに別のファイルとして格納されます。**--encrypted** と併用しないと有効ではありません。

- **--backuppassphrase** - 暗号化されたボリュームにそれぞれランダムに生成されたパスフレーズを追加します。パスフレーズは、**/root** 配下に別々のファイルで格納され、**--escrowcert** で指定した X.509 証明書を使用して暗号化されます。**--escrowcert** と併用しないと有効ではありません。

- **--cipher=** - Anaconda のデフォルトである **aes-xts-plain64** では十分ではない場合に使用する暗号化の種類を指定します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。使用できる暗号化の種類は セキュリティーの強化 に記載されていますが、Red Hat では、**aes-xts-plain64** または **aes-cbc-essiv:sha256** のいずれかの使用を強く推奨しています。

- **--pbkdf=PBKDF** - LUKS 鍵スロット用の PBKDF (Password-Based Key Derivation Function) アルゴリズムを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

- **--pbkdf-memory=PBKDF_MEMORY** - PBKDF のメモリーコストを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

- **--pbkdf-time=PBKDF_TIME** - PBKDF パスフレーズ処理にかかるミリ秒数を設定します。**cryptsetup(8)** の man ページの **--iter-time** も併せて参照してください。このオプションは、**--encrypted** が指定される場合に限り有効になり、**--pbkdf-iterations** と相互に排他的になります。

- **--pbkdf-iterations=PBKDF_ITERATIONS** - 反復の数を直接設定し、PBKDF ベンチマークを回避します。**cryptsetup(8)** の man ページの **--pbkdf-force-iterations** も併せて参照してください。このオプションは、**--encrypted** が指定されている場合に限り有効になり、**--pbkdf-time** と相互に排他的になります。

注記

- **autopart** オプションは、同じキックスタートファイル内で は、**part/partition**、**raid**、**logvol**、**volgroup** などのオプションとは併用できません。
- **autopart** コマンドは必須ではありませんが、キックスタートスクリプトに **part** コマンドまたは **mount** コマンドがない場合は、このコマンドを組み込む必要があります。
- CMS タイプの 1 つの FBA DASD にインストールする場合は、**autopart --nohome** のキックスタートオプションを使用することが推奨されます。これを使用すると、インストールプログラムが別の **/home** パーティションを作成しません。その後、インストールは成功します。
- LUKS パスフレーズが分からなくなると、暗号化されたパーティションと、その上にあるデータには完全にアクセスできなくなります。分からなくなったパスフレーズを復元する方法はありません。ただし、**--escrowcert** を使用して暗号パスフレーズを保存し、**--backuppassphrase** オプションを使用してバックアップの暗号化パスフレーズを作成できます。
- **autopart**、**autopart --type=lvm**、または **autopart=thinp** を使用する場合は、ディスクのセクターサイズに一貫性があることを確認してください。

B.5.2. bootloader (必須)

キックスタートコマンド の **bootloader** が必要です。ブートローダーをインストールする方法を指定します。

構文

```
bootloader [OPTIONS]
```

オプション

- **--append=** - 追加のカーネルパラメーターを指定します。複数のパラメーターを指定する場合は空白で区切ります。以下に例を示します。

  ```
  bootloader --location=mbr --append="hdd=ide-scsi ide=nodma"
  ```

  plymouth パッケージをインストールすると、**rhgb** パラメーターおよび **quiet** パラメーターをここで指定しなくても、もしくは **--append=** コマンドを使用しなくても、自動的に追加されます。この動作を無効にするには、**plymouth** のインストールを明示的に拒否します。

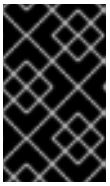
  ```
  %packages
  -plymouth
  %end
  ```

  このオプションは、Meltdown および Spectre に起因する脆弱性の問題を軽減するために実装されたメカニズムを無効にする場合に便利です。投機的実行を悪用するもので、今日のほとんどのプロセッサーで確認されています (CVE-2017-5754、CVE-2017-5753、および CVE-2017-5715)。場合によっては、これらのメカニズムは不要で、有効にしてもセキュリティーは向上せずパフォーマンスが低下する可能性があります。これらのメカニズムを無効にするには、無効にするオプションをキックスタートファイルに追加します (AMD64/Intel 64 システムの例: **bootloader --append="nopti noibrs noibpb"**)。

警告

脆弱性の問題を軽減するメカニズムを無効にする場合は、システムが攻撃の危険にさらされていないことを確認する必要があります。Meltdown および Spectre に起因する脆弱性は、カーネルのサイドチャネル攻撃 - CVE-2017-5754 CVE-2017-5753 CVE-2017-5715 の記事を参照してください。

- **--boot-drive=** - ブートローダーの書き込み先のドライブを指定します。つまり、コンピューターが起動するドライブです。ブートドライブにマルチパスデバイスを使用する場合は、disk/by-id/dm-uuid-mpath-WWID 名を使用してデバイスを指定します。



重要

現在、**zipl** ブートローダーを使用する 64 ビットの IBM Z システムの Red Hat Enterprise Linux インストールでは、**--boot-drive=** オプションが無視されます。**zipl** をインストールすると、そこに起動ドライブがあると判断されます。

- **--leavebootorder** - インストールプログラムは、Red Hat Enterprise Linux 9 を UEFI のインストール済みシステムのリストに追加します。インストールされたシステムを起動順序に追加するわけではありません。既存のすべてのブートエントリーとその順序は保持されます。

重要

このオプションは、電源および UEFI システムに適用されます。

- **--driveorder=** - BIOS の起動順序で最初のドライブを指定します。以下に例を示します。

```
bootloader --driveorder=sda,hda
```

- **--location=** - ブートレコードの書き込み先を指定します。使用できる値は以下のとおりです。
  - **mbr** - デフォルトのオプションです。ドライブが使用しているのが Master Boot Record (MBR) スキームか GUID Partition Table (GPT) スキームかによって、動作が異なります。GPT フォーマット済みディスクの場合は、ブートローダーのステージ 1.5 が BIOS 起動パーティションにインストールされます。

    MBR フォーマット済みディスクの場合は、MBR と 1 番目のパーティションの間にある空白領域にステージ 1.5 がインストールされます。
  - **partition** - カーネルを置くパーティションの 1 番目のセクターに、ブートローダーをインストールします。
  - **none** - ブートローダーをインストールしません。

  ほとんどの場合、このオプションは指定する必要がありません。
- **--nombr** - MBR にブートローダーをインストールしません。

- **--password=** - GRUB2 を使用する場合は、このオプションで指定したパスワードを、ブートローダーのパスワードとして設定します。任意のカーネルオプションが渡される可能性のある GRUB2 シェルへのアクセスを限定する場合に使用してください。
  パスワードを指定すると、GRUB2 ではユーザー名の入力も求められます。ユーザー名は常に **root** です。

- **--iscrypted** - **--password=** オプションを使用してブートローダーのパスワードを指定すると、通常、キックスタートファイルにプレーンテキスト形式で保存されます。このパスワードを暗号化する場合に、このオプションを使用して暗号化パスワードを生成します。
  暗号化したパスワードを生成するには、**grub2-mkpasswd-pbkdf2** コマンドを使用し、使用するパスワードを入力し、コマンドからの出力 (**grub.pbkdf2** で始まるハッシュ) をキックスタートファイルにコピーします。暗号化したパスワードがあるキックスタートエントリーの **bootloader** の例を以下に示します。

  ```
  bootloader --iscrypted --
  password=grub.pbkdf2.sha512.10000.5520C6C9832F3AC3D149AC0B24BE69E2D4FB0DBE
  EDBD29CA1D30A044DE2645C4C7A291E585D4DC43F8A4D82479F8B95CA4BA4381F8550
  510B75E8E0BB2938990.C688B6F0EF935701FF9BD1A8EC7FE5BD2333799C98F28420C5
  CC8F1A2A233DE22C83705BB614EA17F3FDFDF4AC2161CEA3384E56EB38A2E39102F53
  34C47405E
  ```

- **--timeout=** - ブートローダーがデフォルトオプションで起動するまでの待ち時間を指定します (秒単位)。

- **--default=** - ブートローダー設定内のデフォルトのブートイメージを設定します。

- **--extlinux** - GRUB2 の代わりに extlinux ブートローダーを使用します。このオプションが動作するには、extlinux が対応しているシステムのみです。

- **--disabled** - このオプションは、**--location=none** のより強力なバージョンになります。**--location=none** は単にブートローダーのインストールを無効にしますが、**--disabled** だとブートローダーのインストールを無効にするほか、ブートローダーを含むパッケージのインストールを無効にするため、領域が節約できます。

注記

- Red Hat は、全マシンにブートローダーのパスワードを設定することを強く推奨します。ブートローダーが保護されていないと、攻撃者によりシステムの起動オプションが修正され、システムへの不正アクセスが許可されてしまう可能性があります。

- AMD64、Intel 64、および 64 ビット ARM のシステムにブートローダーをインストールするのに、特殊なパーティションが必要になります。このパーティションの種類とサイズは、ブートローダーをインストールしているディスクが、MBR (Master Boot Record) または GPT (GUID Partition Table) スキーマを使用しているかどうかにより異なります。詳細は、**Performing a standard RHEL 9 installation** の Configuring boot loader セクションを参照してください。

- **sdX** (または **/dev/sdX**) 形式では、デバイス名が再起動後に維持される保証がないため、一部のキックスタートコマンドの使用が複雑になる可能性があります。コマンドにデバイスノード名が必要な場合は、**/dev/disk** の項目を代わりに使用できます。以下に例を示します。

  ```
  part / --fstype=xfs --onpart=sda1
  ```

  上記のコマンドの代わりに、以下のいずれかを使用します。

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1

part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR/dev/disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

B.5.3. zipl

キックスタートコマンドの **zipl** は任意です。これは、64 ビットの IBM Z の ZIPL 設定を指定します。

オプション

- **--secure-boot** - インストールシステムで対応しているかどうかを、セキュアな起動を有効にします。

注記

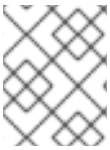
インストールシステムは、IBM z14 以降のシステムにインストールする場合、IBM z14 またはそれ以前のモデルからは起動できません。

- **--force-secure-boot** - セキュアな起動を無条件で有効にします。

注記

IBM z14 以前のモデルでは、インストールに対応していません。

- **--no-secure-boot** - セキュアな起動を無効にします。



注記

Secure Boot は、IBM z14 とそれ以前のモデルでは対応していません。IBM z14 以前のモデルでインストール済みシステムを起動する場合は、**--no-secure-boot** を使用します。

B.5.4. clearpart

キックスタートコマンドの **clearpart** は任意です。新しいパーティションを作成する前に、システムからパーティションを削除します。デフォルトでは、パーティションは削除されません。

構文

```
clearpart OPTIONS
```

オプション

- **--all** - システムにあるすべてのパーティションを消去します。
このオプションを使用すると、接続しているネットワークストレージなど、インストールプログラムでアクセスできるディスクがすべて消去されます。使用する場合は注意が必要です。

clearpart に **--drives=** オプションを使用して消去するドライブのみを指定する、ネットワークストレージは後で接続する (キックスタートファイルの **%post** セクションを利用するなど)、ネットワークストレージのアクセスに使用されるカーネルモジュールを拒否リストに記載するなどの手段を取ると、保持したいストレージが消去されるのを防ぐことができます。

- **--drives=** - ドライブを指定してパーティションを消去します。次の例では、プライマリー IDE コントローラーの 1 番目と 2 番目のドライブにあるパーティションをすべて消去することになります。

```
clearpart --drives=hda,hdb --all
```

マルチパスのデバイスを消去する場合は、**disk/by-id/scsi-WWID** の形式を使用します。**WWID** はデバイスの World-Wide Identifier になります。WWID **58095BEC5510947BE8C0360F604351918** のディスクを消去する場合は以下を使用します。

```
clearpart --drives=disk/by-id/scsi-58095BEC5510947BE8C0360F604351918
```

マルチパスのデバイスを消去する場合はこの形式が適しています。ただし、エラーが発生する場合は、そのマルチパスデバイスが論理ボリューム管理 (LVM) を使用していなければ、**disk/by-id/dm-uuid-mpath-WWID** の形式を使用して消去することもできます。**WWID** はデバイスの World-Wide Identifier です。WWID **2416CD96995134CA5D787F00A5AA11017** のディスクを消去する場合は以下を使用します。

```
clearpart --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
```

mpatha などのデバイス名でマルチパスデバイスを指定しないでください。このようなデバイス名は、特定のディスクに固有の名前ではありません。インストール時に、**/dev/mpatha** という名前のディスクが必ずしも期待したディスクを指すとは限りません。したがって、**clearpart** コマンドを使用する際は、間違ったディスクが対象となる可能性があります。

- **--initlabel** - フォーマット対象の全ディスクで、デフォルトのディスクラベルを作成してディスクを初期化します。たとえば、x86 の場合は msdos になります。**--initlabel** によりすべてのディスクが処理されてしまうため、フォーマット対象のドライブだけを接続することが重要です。**--initlabel** が使用されていない場合でも、**clearpart** によってクリアされたディスクにはラベルが作成されます。

```
clearpart --initlabel --drives=names_of_disks
```

以下に例を示します。

```
clearpart --initlabel --drives=dasda,dasdb,dasdc
```

- **--list=** - 消去するパーティションを指定します。このオプションを使用すると、**--all** および **--linux** のオプションは無効になります。異なるドライブ間で使用できます。以下に例を示します。

```
clearpart --list=sda2,sda3,sdb1
```

- **--disklabel=LABEL** - 使用するデフォルトのディスクラベルを設定します。そのプラットフォームでサポートされるディスクラベルのみが設定できます。たとえば、64 ビットの Intel アーキテクチャーおよび AMD アーキテクチャーでは、**msdos** ディスクラベルおよび **gpt** ディスクラベルが使用できますが、**dasd** は使用できません。

- **--linux** - すべての Linux パーティションを消去します。

- **--none** (デフォルト) - パーティションを消去しません。
- **--cdl** - LDL DASD を CDL 形式に再フォーマットします。

注記

- **sdX** (または **/dev/sdX**) 形式では、デバイス名が再起動後に維持される保証がないため、一部のキックスタートコマンドの使用が複雑になる可能性があります。コマンドにデバイスノード名が必要な場合は、**/dev/disk** の項目を代わりに使用できます。以下に例を示します。

```
part / --fstype=xfs --onpart=sda1
```

上記のコマンドの代わりに、以下のいずれかを使用します。

```
part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1

part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
```

このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR /dev/disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

- **clearpart** コマンドを使用する場合は、論理パーティションには **part --onpart** コマンドは使用できません。

B.5.5. fcoe

キックスタートコマンドの **fcoe** は任意です。Enhanced Disk Drive Services (EDD) で検出されたデバイス以外で、自動的にアクティベートする FCoE デバイスを指定します。

構文

```
fcoe --nic=name [OPTIONS]
```

オプション

- **--nic=** (必須) - アクティベートするデバイス名です。
- **--dcb=** - データセンターブリッジ (DCB) の設定を確立します。
- **--autovlan** - VLAN を自動的に検出します。このオプションはデフォルトで有効になっています。

B.5.6. ignoredisk

キックスタートコマンドの **ignoredisk** は任意です。インストールプログラムが、指定したディスクを無視するようになります。

自動パーティション設定を使用して、特定のディスクを無視したい場合に便利なオプションです。たとえば、**ignoredisk** を使用せずに SAN クラスターに導入しようとすると、インストールプログラムが SAN へのパッシブパスを検出し、パーティションテーブルがないことを示すエラーが返されるため、キックスタートが失敗します。

構文

```
ignoredisk --drives=drive1,drive2,... | --only-use=drive
```

オプション

- **--drives=driveN,...** - Replace **driveN** を、**sda**, **sdb**,..., **hda**, などに置き換えます。
- **--only-use=driveN,...**: インストールプログラムで使用するディスクのリストを指定します。これ以外のディスクはすべて無視されます。たとえば、インストール中に **sda** ディスクを使用し、他はすべて無視する場合は以下のコマンドを使用します。

  ```
  ignoredisk --only-use=sda
  ```

  LVM を使用しないマルチパスのデバイスを指定する場合は、次のコマンドを実行します。

  ```
  ignoredisk --only-use=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
  ```

  LVM を使用するマルチパスのデバイスを指定する場合は、次のコマンドを実行します。

  ```
  ignoredisk --only-use==/dev/disk/by-id/dm-uuid-mpath-
  ```

  ```
  bootloader --location=mbr
  ```

--drives または **--only-use** のいずれかのみを指定する必要があります。

注記

- 論理ボリューム管理 (LVM) を使用していないマルチパスデバイスを無視する場合は、**disk**/**by-id**/**dm-uuid-mpath-WWID** の形式を使用します。**WWID** はデバイスの World-Wide Identifier です。たとえば、WWID **2416CD96995134CA5D787F00A5AA11017** のディスクを無視する場合は以下を使用します。

  ```
  ignoredisk --drives=disk/by-id/dm-uuid-mpath-2416CD96995134CA5D787F00A5AA11017
  ```

- **mpatha** などのデバイス名でマルチパスデバイスを指定しないでください。このようなデバイス名は、特定のディスクに固有の名前ではありません。インストール時に、/**dev**/**mpatha** という名前のディスクが必ずしも期待したディスクを指すとは限りません。したがって、**clearpart** コマンドを使用する際は、間違ったディスクが対象となる可能性があります。

- **sdX** (または /**dev**/**sdX**) 形式では、デバイス名が再起動後に維持される保証がないため、一部のキックスタートコマンドの使用が複雑になる可能性があります。コマンドにデバイスノード名が必要な場合は、/**dev**/**disk** の項目を代わりに使用できます。以下に例を示します。

  ```
  part / --fstype=xfs --onpart=sda1
  ```

  上記のコマンドの代わりに、以下のいずれかを使用します。

  ```
  part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1
  ```

  ```
  part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
  ```

このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR /dev/disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

B.5.7. iscsi

キックスタートコマンドの **iscsi** は任意です。インストール時に接続する追加の iSCSI ストレージを指定します。

構文

```
iscsi --ipaddr=address [OPTIONS]
```

必須オプション

- **--ipaddr=** (必須) - 接続先ターゲットの IP アドレスを指定します。

任意のオプション

- **--port=** (必須) - ポート番号を指定します。存在しない場合は、**--port=3260** がデフォルトで自動的に使用されます。
- **--target=** - ターゲットの IQN (iSCSI 修飾名) を指定します。
- **--iface=** - ネットワーク層で確定されるデフォルトのネットワークインターフェイスではなく、特定のネットワークインターフェイスに接続をバインドします。これを一度使用したら、キックスタートファイルには、**iscsi** コマンドのインスタンスをすべて指定する必要があります。
- **--user=** - ターゲットでの認証に必要なユーザー名を指定します。
- **--password=** - ターゲットに指定したユーザー名のパスワードを指定します。
- **--reverse-user=** - 逆 CHAP 認証を使用するターゲットのイニシエーターでの認証に必要なユーザー名を指定します。
- **--reverse-password=** - イニシエーターに指定したユーザー名のパスワードを指定します。

注記

- また、**iscsi** コマンドを使用する場合は、**iscsiname** コマンドで iSCSI ノードに名前を割り当てる必要があります。**iscsiname** コマンドは、キックスタートファイルで、**iscsi** コマンドより先に指定してください。
- iSCSI ストレージは、できる限り **iscsi** コマンドではなくシステムの BIOS またはファームウェア (Intel システムの場合は iBFT) 内で設定してください。BIOS またはファームウェア内で設定されたディスクは Anaconda で自動的に検出されて使用されるため、キックスタートファイルで特に設定する必要がありません。
- **iscsi** コマンドを使用する必要がある場合は、インストールの開始時にネットワークがアクティブであること、**iscsi** コマンドが、キックスタートファイルで **clearpart** や **ignoredisk** などのコマンドによる iSCSI ディスクの参照よりも**前に**指定されていることを確認してください。

B.5.8. iscsiname

キックスタートコマンドの **iscsiname** は任意です。これは、**iscsi** コマンドが指定した iSCSI ノードに名前を割り当てます。

構文

```
iscsiname iqname
```

オプション

- **iqname** - iSCSI ノードに割り当てる名前。

注記

- キックスタートファイルで **iscsi** コマンドを使用する場合は、キックスタートファイルで **iscsiname** earlier を指定する必要があります。

B.5.9. logvol

キックスタートコマンドの **logvol** は任意です。論理ボリューム管理 (LVM) に論理ボリュームを作成します。

構文

```
logvol mntpoint --vgname=name --name=name [OPTIONS]
```

必須オプション

mntpoint

パーティションがマウントされているマウントポイント。次のいずれかの形式になります。

- **/path**
  たとえば /、または **/home**。
- **swap**
  このパーティションは、swap 領域として使用されます。

  自動的に swap パーティションのサイズを確定させる場合は、**--recommended** オプションを使用します。

  ```
  swap --recommended
  ```

  自動的に swap パーティションサイズを確定しながら、ハイバネート用に追加領域も配分するには、**--hibernation** オプションを使用します。

  ```
  swap --hibernation
  ```

  --recommended で割り当てられる swap 領域に加え、システムの RAM 容量が加算されたサイズが割り当てられるようになります。

--vgname=name

ボリュームグループの名前。

--name=name

論理ボリュームの名前。

任意のオプション

--noformat

既存の論理ボリュームを使用し、フォーマットは行いません。

--useexisting

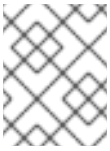
既存の論理ボリュームを使用し、再フォーマットします。

--fstype=

論理ボリュームのファイルシステムのタイプを設定します。**xfs**、**ext2**、**ext3**、**ext4**、**swap**、および **vfat** が使用できる値になります。

--fsoptions=

ファイルシステムをマウントする場合に使用するオプションの文字列を自由形式で指定します。この文字列は、インストール後の **/etc/fstab** ファイルにコピーされるため、引用符で囲む必要があります。



注記

EFI システムパーティション (**/boot/efi**) では、anaconda が値をハードコードし、ユーザー指定の **--fsoptions** 値を無視します。

--mkfsoptions=

このパーティションでファイルシステムを作成するプログラムに渡す追加のパラメーターを指定します。引数のリストでは処理が行われないため、mkfs プログラムに直接渡すことが可能な形式で提供する必要があります。つまり、複数のオプションはコンマ区切りにするか、二重引用符で囲む必要があります (ファイルシステムによって異なります)。以下に例を示します。

```
part /opt/foo1 --size=512 --fstype=ext4 --mkfsoptions="-O
^has_journal,^flex_bg,^metadata_csum"

part /opt/foo2 --size=512 --fstype=xfs --mkfsoptions="-m bigtime=0,finobt=0"
```

詳細は、作成しているファイルシステムの man ページを参照してください。たとえば、**mkfs.ext4** または **mkfs.xfs** です。

--fsprofile=

このパーティションでファイルシステムを作成するプログラムに渡すのに使用するタイプを指定します。ファイルシステムの作成時に使用されるさまざまなチューニングパラメーターは、この使用タイプにより定義されます。ファイルシステム側で使用タイプという概念に対応し、有効なタイプを指定する設定ファイルがないと、このオプションは正しく機能しません。**ext2**、**ext3**、および **ext4** の場合、この設定ファイルは **/etc/mke2fs.conf** になります。

--label=

論理ボリュームのラベルを設定します。

--grow

論理ボリュームを拡張して、利用可能なサイズ (存在する場合) を埋めるか、指定されている場合は最大サイズまで埋めます。このオプションを使用する必要があるのは、ディスクイメージに最小限

のストレージ領域を事前に割り当てており、ボリュームを拡大して使用可能な領域を埋める場合のみです。物理的な環境では、これは 1 回限りのアクションです。ただし、仮想環境では、仮想マシンが仮想ディスクにデータを書き込むとボリュームサイズが増加します。

--size=

論理ボリュームのサイズを MiB 単位で指定します。このオプションを、**--percent=** オプションと併用することはできません。

--percent=

サイズを静的に指定した論理ボリュームを考慮に入れた後のボリュームグループにある空き領域を表すパーセンテージとして、論理ボリュームのサイズを指定します。このオプションは **--size=** オプションと併用することはできません。

重要

論理ボリュームの新規作成時には、**--size=** オプションで静的なサイズを指定するか、**--percent=** オプションで残りの空き領域をパーセンテージとして指定する必要があります。1 つの論理ボリュームで、両方のオプションを使用することはできません。

--maxsize=

論理ボリュームを grow に設定した場合の最大サイズを MiB 単位で指定します。**500** などの整数値を使用してください (単位は不要)。

--recommended

論理ボリュームを作成して、システムのハードウェアに基づいてそのボリュームのサイズを自動的に確定するために、このオプションを使用します。

--resize

論理ボリュームのサイズを変更します。このオプションを使用する場合は、**--useexisting** と **--size** も指定する必要があります。

--encrypted

この論理ボリュームを、**--passphrase=** オプションで入力したパスフレーズを使用する LUKS (Linux Unified Key Setup) で暗号化します。このパスフレーズを指定しない場合は、インストールプログラムが **autopart --passphrase** コマンドで設定されるデフォルトのシステムワイドパスフレーズを使用します。このデフォルトのパスフレーズも設定されていない場合は、インストールプロセスが中断されてパスフレーズの入力が求められます。

注記

1 つまたは複数のパーティションを暗号化する際には、安全な暗号化を行うため、Anaconda が 256 ビットのエントロピーを収集しようとします。エントロピーの収集には時間がかかる場合があります。十分なエントロピーが収集されたかどうかにかかわらず、このプロセスは最大 10 分後に終了します。

プロセスは、インストールシステムと対話することにより高速化できます (キーボードで入力またはマウスの移動)。仮想マシンにインストールしている場合は、**virtio-rng** デバイス (仮想乱数ジェネレーター) をゲストに登録できます。

--passphrase=

この論理ボリュームを暗号化する際に使用するパスフレーズを指定します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。

--cipher=

Anaconda のデフォルトである **aes-xts-plain64** では十分ではない場合に使用する暗号化の種類を指定します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。使用できる暗号化の種類は セキュリティーの強化 に記載されていますが、Red Hat では、**aes-xts-plain64** または **aes-cbc-essiv:sha256** のいずれかの使用を強く推奨しています。

--escrowcert=URL_of_X.509_certificate

暗号化した全ボリュームのデータ暗号化の鍵を **/root** 配下にファイルとして格納します。**URL_of_X.509_certificate** で指定した URL の X.509 証明書を使用して暗号化します。鍵は暗号化したボリュームごとに別のファイルとして格納されます。**--encrypted** と併用しないと有効ではありません。

--luks-version=LUKS_VERSION

ファイルシステムの暗号化に使用する LUKS 形式のバージョンを指定します。**--encrypted** と併用しないと有効ではありません。

--backuppassphrase

暗号化されたボリュームにそれぞれランダムに生成されたパスフレーズを追加します。パスフレーズは、**/root** 配下に別々のファイルで格納され、**--escrowcert** で指定した X.509 証明書を使用して暗号化されます。**--escrowcert** と併用しないと有効ではありません。

--pbkdf=PBKDF

LUKS 鍵スロット用の PBKDF (Password-Based Key Derivation Function) アルゴリズムを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

--pbkdf-memory=PBKDF_MEMORY

PBKDF のメモリーコストを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

--pbkdf-time=PBKDF_TIME

PBKDF パスフレーズ処理にかかるミリ秒数を設定します。**cryptsetup(8)** の man ページの **--iter-time** も併せて参照してください。このオプションは、**--encrypted** が指定される場合に限り有効になり、**--pbkdf-iterations** と相互に排他的になります。

--pbkdf-iterations=PBKDF_ITERATIONS

反復の数を直接設定し、PBKDF ベンチマークを回避します。**cryptsetup(8)** の man ページの **--pbkdf-force-iterations** も併せて参照してください。このオプションは、**--encrypted** が指定されている場合に限り有効になり、**--pbkdf-time** と相互に排他的になります。

--thinpool

シンプール論理ボリュームを作成します。(**none** のマウントポイントの使用)

--metadatasize=size

新しいシンプールデバイスのメタデータ領域のサイズ (MiB 単位) を指定します。

--chunksize=size

新しいシンプールデバイスのチャンクサイズ (KiB) を指定します。

--thin

シン論理ボリュームを作成します。(**--poolname** が必要です。)

--poolname=name

シン論理ボリュームを作成するシンプールの名前を指定します。**--thin** オプションが必要です。

--profile=name

シン論理ボリュームで使用する設定プロファイル名を指定します。これを使用する場合は、この名前は特定の論理ボリュームのメタデータにも含まれることになります。デフォルトで使用できるプロファイルは **default** と **thin-performance** で、**/etc/lvm/profile/** ディレクトリーで定義します。詳細は **lvm(8)** の man ページを参照してください。

--cachepvs=

該当ボリュームのキャッシュとして使用する物理ボリュームをコンマ区切りで記入します。

--cachemode=

当該論理ボリュームのキャッシュに使用するモードを指定します (**writeback** または **writethrough** になります)。

注記

キャッシュ済み論理ボリュームおよびそのモードの詳細は、**lvmcache(7)** の man ページを参照してください。

--cachesize=

論理ボリュームにアタッチするキャッシュのサイズを MiB 単位で指定します。このオプションは、**--cachepvs=** オプションと併用する必要があります。

注記

- キックスタートを使用して Red Hat Enterprise Linux をインストールする場合は、論理ボリューム名およびボリュームグループ名にダッシュ (**-**) 記号を使用しないでください。この文字を使用すると、インストール自体は正常に完了しますが、**/dev/mapper/** ディレクトリー内の論理ボリューム名とボリュームグループ名にダッシュが二重に付いてしまうことになります。たとえば、ボリュームグループ **volgrp-01** に論理ボリューム **logvol-01** が格納されている場合は、**/dev/mapper/volgrp—01-logvol—01** というような表記になります。
この制約が適用されるのは、新規作成の論理ボリュームおよびボリュームグループ名のみです。既存の論理ボリュームまたはボリュームグループを **--noformat** オプションを使用して再利用する場合は、名前が変更されません。

- LUKS パスフレーズが分からなくなると、暗号化されたパーティションと、その上にあるデータには完全にアクセスできなくなります。分からなくなったパスフレーズを復元する方法はありません。ただし、**--escrowcert** を使用して暗号パスフレーズを保存し、**--backuppassphrase** オプションを使用してバックアップの暗号化パスフレーズを作成できます。

例

- まずパーティションを作成します。次に論理ボリュームグループを作成して、論理ボリュームを作成します。

```
part pv.01 --size 3000
volgroup myvg pv.01
logvol / --vgname=myvg --size=2000 --name=rootvol
```

- 最初にパーティションを作成します。次に論理ボリュームグループを作成して、ボリュームグループに残っている領域の 90 % を占める論理ボリュームを作成します。

```
part pv.01 --size 1 --grow
volgroup myvg pv.01
logvol / --vgname=myvg --name=rootvol --percent=90
```

関連情報

- 論理ボリュームの設定および管理

B.5.10. mount

キックスタートコマンドの **mount** は任意です。これは、既存のブロックデバイスにマウントポイントを割り当て、必要に応じて、指定の形式で再フォーマットします。

構文

```
mount [OPTIONS] device mountpoint
```

必須オプション:

- **device** - マウントするブロックデバイス。
- **mountpoint** - **device** をマウントする場所。/、/**usr** などの有効なマウントポイントを指定する必要があります。デバイスがマウントできない場合 (**swap** など) は **none** と指定します。

任意のオプション:

- **--reformat=** - デバイスを再フォーマットする際の新しいフォーマット (例: **ext4**) を指定します。
- **--mkfsoptions=** - **--reformat=** で指定した新しいファイルシステムを作成するコマンドに渡す追加のオプションを指定します。ここで指定するオプションのリストは処理されません。したがって、直接 **mkfs** プログラムに渡すことのできる形式で指定する必要があります。オプションのリストは、コンマ区切りとするか、二重引用符で囲む必要があります (ファイルシステムによって異なります)。詳細は、作成するファイルシステムの **mkfs** の man ページで確認してください (例: **mkfs.ext4(8)** または **mkfs.xfs(8)**)。
- **--mountoptions=** - ファイルシステムをマウントする場合に使用するオプションを含む文字列を、自由形式で指定します。この文字列はインストールされたシステムの /**etc**/**fstab** ファイルにコピーされるため、二重引用符で囲んでください。マウントオプションの全リストは **mount(8)** の man ページを、概要は **fstab(5)** を参照してください。

注記

- キックスタートの他の多くのストレージ設定コマンドとは異なり、**mount** の場合には、キックスタートファイルにすべてのストレージ設定を記述する必要がありません。確認する必要があるのは、記述されたブロックデバイスがシステムに存在することだけです。ただし、すべてのデバイスがマウントされたストレージスタックを **作成する** 場合には、**part** などの他のコマンドを使用する必要があります。
- 同じキックスタートファイル内で、**mount** を **part**、**logvol**、または **autopart** などの他のストレージ関連コマンドと併用することはできません。

B.5.11. nvdimm

キックスタートコマンドの **nvdimm** は任意です。これは、NVDIMM (Non-Volatile Dual In-line Memory Module) デバイスでアクションを実行します。

構文

```
nvdimm action [OPTIONS]
```

アクション

- **reconfigure** - 指定した NVDIMM デバイスを特定のモードに再設定します。なお、指定したデバイスは暗示的に使用先と識別されるため、同じデバイスに対するこれ以降の **nvdimm use** コマンドは冗長になります。このアクションは以下の形式を使用します。

  ```
  nvdimm reconfigure [--namespace=NAMESPACE] [--mode=MODE] [--
  sectorsize=SECTORSIZE]
  ```

  - **--namespace=** - 名前空間でデバイスを指定します。以下に例を示します。

    ```
    nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=512
    ```

  - **--mode=** - モードを指定します。現在、利用できる値は **sector** だけです。

  - **--sectorsize=** - セクターサイズ (セクターモードの場合)。以下に例を示します。

    ```
    nvdimm reconfigure --namespace=namespace0.0 --mode=sector --sectorsize=512
    ```

    サポートされるセクターサイズは 512 バイトおよび 4096 バイトです。

- **use** - NVDIMM デバイスを、インストールのターゲットに指定します。デバイスは、 **nvdimm reconfigure** コマンドでセクターモードに設定されている必要があります。このアクションは以下の形式を使用します。

  ```
  nvdimm use [--namespace=NAMESPACE|--blockdevs=DEVICES]
  ```

  - **--namespace=** - 名前空間でデバイスを指定します。以下に例を示します。

    ```
    nvdimm use --namespace=namespace0.0
    ```

  - **--blockdevs=** - 使用する NVDIMM デバイスに対応するブロックデバイスをコンマ区切りリストで指定します。ワイルドカードとしてアスタリスク * が使用できます。以下に例を示します。

    ```
    nvdimm use --blockdevs=pmem0s,pmem1s
    nvdimm use --blockdevs=pmem*
    ```

注記

- デフォルトでは、インストールプログラムはすべての NVDIMM デバイスを無視します。このデバイスでのインストールを有効にするには、**nvdimm** コマンドを使用する必要があります。

B.5.12. part または partition

キックスタートコマンド **part** または **partition** が必要です。このコマンドは、システムにパーティションを作成します。

構文

```
part|partition mntpoint [OPTIONS]
```

オプション

- **mntpoint** - パーティションをマウントする場所です。値は次のいずれかの形式になります。

- **/path**
  /、**/usr**、**/home** など。

- **swap**
  このパーティションは、swap 領域として使用されます。

  自動的に swap パーティションのサイズを確定させる場合は、**--recommended** オプションを使用します。

  ```
  swap --recommended
  ```

  有効なサイズが割り当てられますが、システムに対して正確に調整されたサイズではありません。

  自動的に swap パーティションサイズを確定しながら、ハイバネート用に余剰領域も割り当てる場合は、**--hibernation** オプションを使用します。

  ```
  swap --hibernation
  ```

  --recommended で割り当てられる swap 領域に加え、システムの RAM 容量が加算されたサイズが割り当てられるようになります。

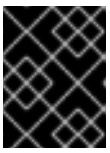
- **raid.id**
  このパーティションはソフトウェア RAID に使用されます (**raid** を参照)。

- **pv.id**
  このパーティションは LVM に使用されます (**logvol** を参照)。

- **biosboot**
  このパーティションは、BIOS 起動パーティションに使用されます。GPT (GUID Partition Table) を使用する BIOS ベースの AMD64 および Intel 64 のシステムには、1 MiB の BIOS 起動パーティションが必要になります。ブートローダーは、このパーティションにインストールされます。UEFI システムには必要ありません。詳細は **bootloader** コマンドも併せてご覧ください。

- **/boot/efi**
  EFI システムパーティションです。UEFI ベースの AMD64、Intel 64、および 64 ビットの ARM には 50 MiB の EFI パーティションが必要になります。推奨サイズは 200 MiB です。BIOS システムには必要ありません。詳細は **bootloader** コマンドも併せてご覧ください。

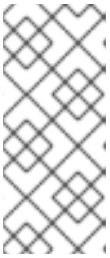
- **--size=** - パーティションの最小サイズを MiB 単位で指定します。**500** などの整数値を使用してください (単位は不要)。

  重要

  --size の値が小さすぎると、インストールに失敗します。**--size** の値は、必要となる領域の最小値として指定します。

- **--grow** - パーティションを拡張して、利用可能なサイズ (存在する場合) を埋めるか、指定されている場合は最大サイズまで埋めます。



注記

swap パーティションに **--maxsize=** を設定せずに **--grow=** を使用すると、swap パーティションの最大サイズは、Anaconda により制限されます。物理メモリーが 2 GiB 未満のシステムの場合は、物理メモリー量の 2 倍に制限されます。物理メモリーが 2 GiB 以上のシステムの場合は、物理メモリー量に 2GiB を足した量に制限されます。

- **--maxsize=** - パーティションが grow に設定されている場合の最大サイズを MiB 単位で指定します。**500** などの整数値を使用してください (単位は不要)。
- **--noformat** - パーティションをフォーマットしない場合に指定します。**--onpart** コマンドと併用してください。
- **--onpart=** または **--usepart=** - パーティションを配置するデバイスを指定します。既存の空のデバイスを使用し、新たに指定したタイプにフォーマットします。以下に例を示します。

```
partition /home --onpart=hda1
```

上記では、**/home** が **/dev/hda1** に配置されます。

このオプションを使用して、パーティションを論理ボリュームに追加することもできます。以下に例を示します。

```
partition pv.1 --onpart=hda2
```

この場合は、デバイスがシステムに存在している必要があります。**--onpart** オプションでデバイスを作成するわけではありません。

パーティションではなく、ドライブ全体を指定することも可能です。その場合、Anaconda はパーティションテーブルを作成せずに、ドライブをフォーマットして使用します。ただし、この方法でフォーマットしたデバイスでは GRUB2 のインストールがサポートされないため、パーティションテーブルのあるドライブに置かれる必要があります。

```
partition pv.1 --onpart=hdb
```

- **--ondisk=** または **--ondrive=** - 既存ディスクに (**part** コマンドで指定した) パーティションを作成します。このコマンドは常にパーティションを作成します。特定のディスクに強制的にパーティションを作成します。たとえば、**--ondisk=sdb** を使用すると、パーティションは 2 番目の SCSI ディスクに作成されます。
論理ボリューム管理 (LVM) を使用しないマルチパスデバイスを指定する場合は、**disk/by-id/dm-uuid-mpath-WWID** の形式を使用します。**WWID** は、デバイスの World-Wide Identifier です。たとえば、WWID **2416CD96995134CA5D787F00A5AA11017** のディスクを指定する場合は以下を使用します。

```
part / --fstype=xfs --grow --asprimary --size=8192 --ondisk=disk/by-id/dm-uuid-mpath-
2416CD96995134CA5D787F00A5AA11017
```

警告

mpatha などのデバイス名でマルチパスデバイスを指定しないでください。このようなデバイス名は、特定のディスクに固有の名前ではありません。インストール時に、**/dev/mpatha** という名前のディスクが必ずしも期待したディスクを指すとは限りません。したがって、**part** コマンドを使用する際は、間違ったディスクが対象となる可能性があります。

- **--asprimary** - パーティションが **プライマリー** パーティションとして割り当てられるように強制実行します。(通常、すでに割り当てられているプライマリーパーティションが多すぎるという理由で) パーティションをプライマリーとして割り当てられない場合は、パーティション設定のプロセスが失敗します。このオプションは、Master Boot Record (MBR) をディスクが使用する場合にのみ有効で、GUID Partition Table (GPT) ラベルが付いたディスクでは有効ではありません。

- **--fsprofile=** - このパーティションでファイルシステムを作成するプログラムに渡すのに使用するタイプを指定します。ファイルシステムの作成時に使用されるさまざまなチューニングパラメーターは、この使用タイプにより定義されます。ファイルシステム側で使用タイプという概念に対応し、有効なタイプを指定する設定ファイルがないと、このオプションは正しく機能しません。**ext2**、**ext3**、**ext4** の場合、この設定ファイルは **/etc/mke2fs.conf** になります。

- **--mkfsoptions=** - このパーティションでファイルシステムを作成するプログラムに渡す追加のパラメーターを指定します。これは **--fsprofile** と似ていますが、プロフィールの概念に対応するものだけではなく、すべてのファイルシステムで機能するものです。引数のリストでは処理が行われないため、mkfs プログラムに直接渡すことが可能な形式で提供する必要があります。つまり、複数のオプションはコンマ区切りにするか、二重引用符で囲む必要があります (ファイルシステムによって異なります)。以下に例を示します。

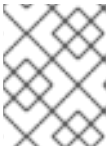
```
part /opt/foo1 --size=512 --fstype=ext4 --mkfsoptions="-O
^has_journal,^flex_bg,^metadata_csum"

part /opt/foo2 --size=512 --fstype=xfs --mkfsoptions="-m bigtime=0,finobt=0"
```

詳細は、作成しているファイルシステムの man ページを参照してください。たとえば、**mkfs.ext4** または **mkfs.xfs** です。

- **--fstype=** - パーティションのファイルシステムタイプを設定します。使用できる値は、**xfs**、**ext2**、**ext3**、**ext4**、**swap**、**vfat**、**efi**、および **biosboot** になります。

- **--fsoptions** - ファイルシステムをマウントする場合に使用するオプション文字列を自由形式で指定します。この文字列は、インストール後の **/etc/fstab** ファイルにコピーされるため、引用符で囲む必要があります。

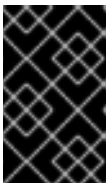


注記

EFI システムパーティション (**/boot/efi**) では、anaconda が値をハードコードし、ユーザー指定の **--fsoptions** 値を無視します。

- **--label=** - 個別パーティションにラベルを割り当てます。

- **--recommended** - パーティションのサイズを自動的に確定します。



重要

このオプションは、**/boot** パーティションや **swap** 領域といったファイルシステムになるパーティションにのみ使用できます。LVM 物理ボリュームや RAID メンバーの作成には使用できません。

- **--onbiosdisk** - BIOS で検出された特定のディスクに強制的にパーティションを作成します。
- **--encrypted** - **--passphrase** オプションで入力したパスフレーズを使用して、LUKS (Linux Unified Key Setup) でこのパーティションを暗号化するように指定します。このパスフレーズを指定しないと、Anaconda は、**autopart --passphrase** コマンドで設定されるデフォルトのシステムワイドパスフレーズを使用します。このデフォルトのパスフレーズも設定されていない場合は、インストールプロセスが中断して、パスフレーズの入力が求められます。

注記

1 つまたは複数のパーティションを暗号化する際には、安全な暗号化を行うため、Anaconda が 256 ビットのエントロピーを収集しようとします。エントロピーの収集には時間がかかる場合があります。十分なエントロピーが収集されたかどうかにかかわらず、このプロセスは最大 10 分後に終了します。

プロセスは、インストールシステムと対話することにより高速化できます (キーボードで入力またはマウスの移動)。仮想マシンにインストールしている場合は、**virtio-rng** デバイス (仮想乱数ジェネレーター) をゲストに登録できます。

- **--luks-version=LUKS_VERSION** - ファイルシステムの暗号化に使用する LUKS 形式のバージョンを指定します。**--encrypted** と併用しないと有効ではありません。
- **--passphrase=** - このパーティションの暗号化を行う際に使用するパスフレーズを入力します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。
- **--cipher=** - Anaconda のデフォルトである **aes-xts-plain64** では十分ではない場合に使用する暗号化の種類を指定します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。使用できる暗号化の種類は セキュリティーの強化 に記載されていますが、Red Hat では、**aes-xts-plain64** または **aes-cbc-essiv:sha256** のいずれかの使用を強く推奨しています。
- **--escrowcert=URL_of_X.509_certificate** - 暗号化した全パーティションのデータ暗号化の鍵を **/root** 配下にファイルとして格納します。**URL_of_X.509_certificate** で指定した URL の X.509 証明書を使用して暗号化します。鍵は、暗号化したパーティションごとに別のファイルとして格納されます。**--encrypted** と併用しないと有効ではありません。
- **--backuppassphrase** - 暗号化されたパーティションにそれぞれランダムに生成されたパスフレーズを追加します。パスフレーズは、**/root** 配下に別々のファイルで格納され、**--escrowcert** で指定した X.509 証明書を使用して暗号化されます。**--escrowcert** と併用しないと有効ではありません。
- **--pbkdf=PBKDF** - LUKS 鍵スロット用の PBKDF (Password-Based Key Derivation Function) アルゴリズムを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

- **--pbkdf-memory=PBKDF_MEMORY** - PBKDF のメモリーコストを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。

- **--pbkdf-time=PBKDF_TIME** - PBKDF パスフレーズ処理にかかるミリ秒数を設定します。**cryptsetup(8)** の man ページの **--iter-time** も併せて参照してください。このオプションは、**--encrypted** が指定される場合に限り有効になり、**--pbkdf-iterations** と相互に排他的になります。

- **--pbkdf-iterations=PBKDF_ITERATIONS** - 反復の数を直接設定し、PBKDF ベンチマークを回避します。**cryptsetup(8)** の man ページの **--pbkdf-force-iterations** も併せて参照してください。このオプションは、**--encrypted** が指定されている場合に限り有効になり、**--pbkdf-time** と相互に排他的になります。

- **--resize=** - 既存パーティションのサイズを変更します。このオプションを使用する際は、**--size=** オプションで目的のサイズ (MiB 単位) を指定し、 **--onpart=** オプションで目的のパーティションを指定します。

注記

- **part** コマンドは必須ではありませんが、キックスタートスクリプトには **part**、**autopart**、または **mount** のいずれかを指定する必要があります。

- 何らかの理由でパーティションの設定ができなかった場合には、診断メッセージが仮想コンソール 3 に表示されます。

- **--noformat** および **--onpart** を使用しないと、作成されたパーティションはすべてインストールプロセスの一部としてフォーマット化されます。

- **sdX** (または /**dev**/**sdX**) 形式では、デバイス名が再起動後に維持される保証がないため、一部のキックスタートコマンドの使用が複雑になる可能性があります。コマンドにデバイスノード名が必要な場合は、/**dev**/**disk** の項目を代わりに使用できます。以下に例を示します。

  ```
  part / --fstype=xfs --onpart=sda1
  ```

  上記のコマンドの代わりに、以下のいずれかを使用します。

  ```
  part / --fstype=xfs --onpart=/dev/disk/by-path/pci-0000:00:05.0-scsi-0:0:0:0-part1

  part / --fstype=xfs --onpart=/dev/disk/by-id/ata-ST3160815AS_6RA0C882-part1
  ```

  このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR**/**dev**/**disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

- LUKS パスフレーズが分からなくなると、暗号化されたパーティションと、その上にあるデータには完全にアクセスできなくなります。分からなくなったパスフレーズを復元する方法はありません。ただし、**--escrowcert** を使用して暗号パスフレーズを保存し、**--backuppassphrase** オプションを使用してバックアップの暗号化パスフレーズを作成できます。

B.5.13. raid

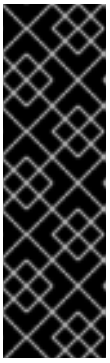
キックスタートコマンドの **raid** は任意です。ソフトウェアの RAID デバイスを組み立てます。

構文

```
raid mntpoint --level=level --device=device-name partitions*
```

オプション

- **mntpoint** - RAID ファイルシステムをマウントする場所です。/ にマウントする場合は、boot パーティション (**/boot**) がなければ RAID レベルを 1 にする必要があります。boot パーティションがある場合は、**/boot** パーティションをレベル 1 にしてください。ルート (/) パーティションのタイプはどれでも構いません。**partitions*** (複数パーティションの指定が可能) には RAID アレイに追加する RAID 識別子を指定します。



重要

  - IBM Power Systems で RAID デバイスの準備は行ったものの、インストール中に再フォーマットを行っていない場合で、この RAID デバイスに **/boot** パーティションおよび PReP パーティションの配置を予定している場合は、RAID メタデータのバージョンが **0.90** または **1.0** になっていることを確認してください。**mdadm** メタデータバージョン **1.1** および **1.2** は、**/boot** および PReP パーティションではサポートされていません。
  - PowerNV システムでは、**PReP** Boot パーティションは必要ありません。

- **--level=** - 使用する RAID レベルを指定します (0、1、4、5、6、10 のいずれか)。
- **--device=** - 使用する RAID デバイス名を指定します (例: **--device=root**)。

重要

mdraid 名を **md0** の形式で使用しないでください。このような名前は永続性が保証されていません。代わりに、**root**、**swap** など意味のある名前にしてください。意味のある名前を使用すると、**/dev/md/name** から、アレイに割り当てられている **/dev/mdX** ノードへのシンボリックリンクが作成されます。

名前を割り当てることができない古い (v0.90 メタデータ) アレイがある場合は、ファイルシステムラベルまたは UUID でアレイを指定できます。たとえば、**--device=LABEL=root** または **--device=UUID=93348e56-4631-d0f0-6f5b-45c47f570b88** です。

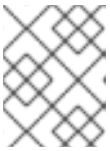
RAID デバイス上のファイルシステムの UUID または RAID デバイス自体の UUID を使用できます。RAID デバイスの UUID は **8-4-4-4-12** 形式である必要があります。mdadm によって報告される UUID は、変更する必要がある **8:8:8:8** 形式です。たとえば、**93348e56:4631d0f0:6f5b45c4:7f570b88** は **93348e56-4631-d0f0-6f5b-45c47f570b88** に変更する必要があります。

- **--chunksize=** - RAID ストレージのチャンクサイズを KiB 単位で設定します。場合によっては、デフォルトのサイズ (**512 Kib**) 以外のチャンクサイズを使用すると、RAID のパフォーマンスが向上することもあります。
- **--spares=** - RAID アレイに割り当てられるスペアドライブの数を指定します。スペアドライブは、ドライブに障害が発生した場合にアレイの再設定に使用されます。
- **--fsprofile=** - このパーティションでファイルシステムを作成するプログラムに渡すのに使用するタイプを指定します。ファイルシステムの作成時に使用されるさまざまなチューニングパラ

メーターは、この使用タイプにより定義されます。ファイルシステム側で使用タイプという概念に対応し、有効なタイプを指定する設定ファイルがないと、このオプションは正しく機能しません。ext2、ext3、ext4 の場合、この設定ファイルは **/etc/mke2fs.conf** になります。

- **--fstype=** - RAID アレイのファイルシステムタイプを設定します。**xfs**、**ext2**、**ext3**、**ext4**、**swap**、および **vfat** が使用できる値になります。

- **--fsoptions=** - ファイルシステムをマウントする場合に使用するオプションの文字列を自由形式で指定します。この文字列は、インストール後の **/etc/fstab** ファイルにコピーされるため、引用符で囲む必要があります。



注記

EFI システムパーティション (**/boot/efi**) では、anaconda が値をハードコードし、ユーザー指定の **--fsoptions** 値を無視します。

- **--mkfsoptions=** - このパーティションでファイルシステムを作成するプログラムに渡す追加のパラメーターを指定します。引数のリストでは処理が行われないため、mkfs プログラムに直接渡すことが可能な形式で提供する必要があります。つまり、複数のオプションはコンマ区切りにするか、二重引用符で囲む必要があります (ファイルシステムによって異なります)。以下に例を示します。

```
part /opt/foo1 --size=512 --fstype=ext4 --mkfsoptions="-O
^has_journal,^flex_bg,^metadata_csum"

part /opt/foo2 --size=512 --fstype=xfs --mkfsoptions="-m bigtime=0,finobt=0"
```

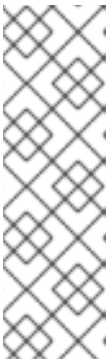
詳細は、作成しているファイルシステムの man ページを参照してください。たとえば、**mkfs.ext4** または **mkfs.xfs** です。

- **--label=** - 作成するファイルシステムのラベルを指定します。指定ラベルが別のファイルシステムですでに使用されている場合は、新しいラベルが作成されます。

- **--noformat** - 既存の RAID デバイスを使用し、RAID アレイのフォーマットが行いません。

- **--useexisting** - 既存の RAID デバイスを使用し、再フォーマット化を行います。

- **--encrypted** - **--passphrase** オプションで入力したパスフレーズを使用して、LUKS (Linux Unified Key Setup) でこの RAID デバイスを暗号化するように指定します。このパスフレーズを指定しないと、Anaconda は、**autopart --passphrase** コマンドで設定されるデフォルトのシステムワイドパスフレーズを使用します。このデフォルトのパスフレーズも設定されていない場合は、インストールプロセスが中断して、パスフレーズの入力が求められます。



注記

1 つまたは複数のパーティションを暗号化する際には、安全な暗号化を行うため、Anaconda が 256 ビットのエントロピーを収集しようとします。エントロピーの収集には時間がかかる場合があります。十分なエントロピーが収集されたかどうかにかかわらず、このプロセスは最大 10 分後に終了します。

プロセスは、インストールシステムと対話することにより高速化できます (キーボードで入力またはマウスの移動)。仮想マシンにインストールしている場合は、**virtio-rng** デバイス (仮想乱数ジェネレーター) をゲストに登録できます。

- **--luks-version=LUKS_VERSION** - ファイルシステムの暗号化に使用する LUKS 形式のバージョンを指定します。**--encrypted** と併用しないと有効ではありません。
- **--cipher=** - Anaconda のデフォルトである **aes-xts-plain64** では十分ではない場合に使用する暗号化の種類を指定します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。使用できる暗号化の種類は セキュリティーの強化 に記載されていますが、Red Hat では、**aes-xts-plain64** または **aes-cbc-essiv:sha256** のいずれかの使用を強く推奨しています。
- **--passphrase=** - この RAID デバイスの暗号化を行う際に使用するパスフレーズを入力します。**--encrypted** オプションと併用してください。単独で使用しても暗号化されません。
- **--escrowcert=URL_of_X.509_certificate** - このデバイス用のデータ暗号化の鍵を **/root** 配下にファイルとして格納します。鍵は、**URL_of_X.509_certificate** で指定した URL の X.509 証明書を使用して暗号化します。**--encrypted** と併用しないと有効ではありません。
- **--backuppassphrase** - このデバイスにランダムに生成されたパスフレーズを追加します。パスフレーズは **/root** 配下にファイルとして格納されます。**--escrowcert** で指定した X.509 証明書を使用して暗号化されます。**--escrowcert** と併用しないと有効ではありません。
- **--pbkdf=PBKDF** - LUKS 鍵スロット用の PBKDF (Password-Based Key Derivation Function) アルゴリズムを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。
- **--pbkdf-memory=PBKDF_MEMORY** - PBKDF のメモリーコストを設定します。**cryptsetup(8)** の man ページも併せて参照してください。**--encrypted** と併用しないと有効ではありません。
- **--pbkdf-time=PBKDF_TIME** - PBKDF パスフレーズ処理にかかるミリ秒数を設定します。**cryptsetup(8)** の man ページの **--iter-time** も併せて参照してください。このオプションは、**--encrypted** が指定される場合に限り有効になり、**--pbkdf-iterations** と相互に排他的になります。
- **--pbkdf-iterations=PBKDF_ITERATIONS** - 反復の数を直接設定し、PBKDF ベンチマークを回避します。**cryptsetup(8)** の man ページの **--pbkdf-force-iterations** も併せて参照してください。このオプションは、**--encrypted** が指定されている場合に限り有効になり、**--pbkdf-time** と相互に排他的になります。

例

以下の例では、/ には RAID レベル 1 のパーティション、 **/home** には RAID レベル 5 のパーティションを作成します。ここでは、システムには SCSI ディスクが 3 つあることが前提です。各ドライブに 1 つずつ、3 つの swap パーティションを作成します。

```
part raid.01 --size=6000 --ondisk=sda
part raid.02 --size=6000 --ondisk=sdb
part raid.03 --size=6000 --ondisk=sdc
part swap --size=512 --ondisk=sda
part swap --size=512 --ondisk=sdb
part swap --size=512 --ondisk=sdc
part raid.11 --size=1 --grow --ondisk=sda
part raid.12 --size=1 --grow --ondisk=sdb
part raid.13 --size=1 --grow --ondisk=sdc
raid / --level=1 --device=rhel8-root --label=rhel8-root raid.01 raid.02 raid.03
raid /home --level=5 --device=rhel8-home --label=rhel8-home raid.11 raid.12 raid.13
```

注記

注記

- LUKS パスフレーズが分からなくなると、暗号化されたパーティションと、その上にあるデータには完全にアクセスできなくなります。分からなくなったパスフレーズを復元する方法はありません。ただし、**--escrowcert** を使用して暗号パスフレーズを保存し、**--backuppassphrase** オプションを使用してバックアップの暗号化パスフレーズを作成できます。

B.5.14. reqpart

キックスタートコマンドの **reqpart** は任意です。使用中のハードウェアプラットホームで必要となるパーティションを自動的に作成します。UEFI ファームウェアのシステム向けに **/boot/efi** パーティション、BIOS ファームウェアおよび GPT のシステム向けに **biosboot** パーティション、IBM Power Systems 向けに **PRePBoot** パーティションが作成されます。

構文

```
reqpart [--add-boot]
```

オプション

- **--add-boot** - ベースコマンドが作成するプラットホーム固有のパーティションとは別に、**/boot** パーティションを作成します。

注記

- このコマンドは、**autopart** と併用することはできません。**autopart** は **reqpart** コマンドの実行内容に加えて、他のパーティションや、/、**swap** といった論理ボリュームも作成するためです。**autopart** とは異なり、このコマンドは、プラットホーム固有のパーティションの作成のみを行い、ドライブの残りは空のままにするため、カスタムレイアウトの作成が可能になります。

B.5.15. snapshot

キックスタートコマンドの **snapshot** は任意です。インストールプロセス時に、このコマンドを使用して LVM のシンボリュームのスナップショットを作成できます。これにより、インストール前後の論理ボリュームのバックアップ作成が可能になります。

複数のスナップショットを作成するには、キックスタートコマンドの **snaphost** を複数回追加します。

構文

```
snapshot vg_name/lv_name --name=snapshot_name --when=pre-install|post-install
```

オプション

- **vg_name/lv_name** - スナップショットの作成元となるボリュームグループや論理ボリュームの名前を設定します。
- **--name=snapshot_name** - スナップショットの名前を設定します。この名前は、ボリュームグループ内で一意のものにする必要があります。
- **--when=pre-install|post-install** - インストール前もしくは完了後にスナップショットを作成することを指定します。

B.5.16. volgroup

キックスタートコマンドの **volgroup** は任意です。論理ボリューム管理 (LVM) グループを作成します。

構文

```
volgroup name [OPTIONS] [partition*]
```

必須オプション

- **name** - 新しいボリュームグループの名前。

オプション

- **partition** - ボリュームグループのバッキングストレージとして使用する物理ボリュームパーティション。
- **--noformat** - 既存のボリュームグループを使用し、フォーマットは行いません。
- **--useexisting** - 既存のボリュームグループを使用し、そのボリュームグループを再フォーマットします。このオプションを使用する場合は **partition** を指定しないでください。以下に例を示します。

  ```
  volgroup rhel00 --useexisting --noformat
  ```

- **--pesize=** - ボリュームグループの物理エクステントのサイズをキビバイト (KiB) 単位で設定します。デフォルト値は 4096 (4 MiB) で、最小値は 1024 (1 MiB) になります。
- **--reserved-space=** - ボリュームグループに未使用で残す領域を MiB 単位で指定します。新規作成のボリュームグループにのみ適用されます。
- **--reserved-percent=** - 未使用で残すボリュームグループ領域全体の割合を指定します。新規作成のボリュームグループにのみ適用されます。

注記

- まずパーティションを作成します。次に論理ボリュームグループを作成して、論理ボリュームを作成します。以下に例を示します。

  ```
  part pv.01 --size 10000
  volgroup my_volgrp pv.01
  logvol / --vgname=my_volgrp --size=2000 --name=root
  ```

- キックスタートを使用して Red Hat Enterprise Linux をインストールする場合は、論理ボリューム名およびボリュームグループ名にダッシュ (**-**) 記号を使用しないでください。この文字を使用すると、インストール自体は正常に完了しますが、**/dev/mapper/** ディレクトリー内の論理ボリューム名とボリュームグループ名にダッシュが二重に付いてしまうことになります。たとえば、ボリュームグループ **volgrp-01** に論理グループ **logvol-01** が格納されている場合は、**/dev/mapper/volgrp--01-logvol--01** のような表記になります。
  この制約が適用されるのは、新規作成の論理ボリュームおよびボリュームグループ名のみです。既存の論理ボリュームまたはボリュームグループを **--noformat** オプションを使用して再利用する場合は、名前が変更されません。

B.5.17. zerombr

キックスタートコマンドの **zerombr** は任意です。**zerombr** は、ディスク上で見つかった無効なパーティションテーブルを初期化し、無効なパーティションテーブルを持つディスクの中身をすべて破棄します。このコマンドは、フォーマットされていない DASD (Direct Access Storage Device) ディスクを備えた 64 ビットの IBM Z システムでインストールを実行する場合に必要です。このコマンドを使用しないと、フォーマットされていないディスクがインストール時にフォーマットされず、使用されません。

構文

```
zerombr
```

注記

- 64 ビットの IBM Z では **zerombr** が指定された場合、インストールプログラムに見えている Direct Access Storage Device (DASD) でまだ低レベルフォーマット処理がなされていないものは、自動的に dasdfmt で低レベルフォーマット処理がなされます。このコマンドでは、対話型インストール中のユーザー選択も行われません。
- **zerombr** が指定されておらず、少なくとも 1 つの未フォーマットの DASD がインストールプログラムに見えている場合、非対話形式のキックスタートを使用したインストールは失敗に終わります。
- **zerombr** が指定されていない場合に、未フォーマットの DASD をインストールプログラムが 1 つ以上認識している場合は、認識されている未フォーマットの DASD のフォーマットにユーザーがすべて同意しなければ、対話形式のインストールが終了します。この状況を回避するには、インストール中に使用する DASD のみをアクティベートします。DASD は、インストール完了後にいつでも追加できます。
- このコマンドにはオプションはありません。

B.5.18. zfcp

キックスタートコマンドの **zfcp** は任意です。Fibre チャンネルデバイスを定義します。

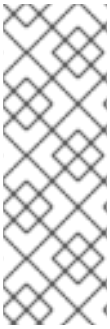
このオプションは、64 ビットの IBM Z にのみ適用されます。

構文

```
zfcp --devnum=devnum [--wwpn=wwpn --fcplun=lun]
```

オプション

- **--devnum=** - デバイス番号 (zFCP アダプターデバイスバス ID)。
- **--wwpn=** - デバイスの WWPN (ワールドワイドポートネーム)。**0x** で始まる 16 桁の番号になります。
- **--fcplun=** - デバイスの論理ユニット番号 (LUN)。**0x** で始まる 16 桁の番号になります。



注記

自動 LUN スキャンが利用できる場合や、9 以降のリリースをインストールする場合は、FCP デバイスバス ID を指定するだけで十分です。それ以外の場合は、3 つのパラメーターがすべて必要になります。自動 LUN スキャンは、**zfcp.allow_lun_scan** モジュールパラメーターで無効にされていない場合 (デフォルトでは有効)、NPIV モードで動作する FCP デバイスで使用できます。これは、指定されたバス ID を持つ FCP デバイスに接続されたストレージエリアネットワークで見つかったすべての SCSI デバイスへのアクセスを提供します。

例

```
zfcp --devnum=0.0.4000 --wwpn=0x5005076300C213e9 --fcplun=0x5022000000000000
zfcp --devnum=0.0.4000
```

B.6. RHEL インストールプログラムで提供されるアドオン向けキックスタートコマンド

本セクションのキックスタートコマンドは、Red Hat Enterprise Linux インストールプログラムにデフォルトで提供されるアドオンに関連しています。

B.6.1. %addon com_redhat_kdump

キックスタートコマンドの **%addon com_redhat_kdump** は任意です。このコマンドは、kdump カーネルクラッシュのダンプメカニズムを設定します。

構文

```
%addon com_redhat_kdump [OPTIONS]
%end
```

注記

このコマンドは、ビルトインのキックスタートコマンドではなくアドオンであることから、構文は通常のものとは異なります。

注記

Kdump とは、システムのメモリーの内容を保存して後で分析できるように、カーネルのクラッシュをダンプするメカニズムを指します。これは、**kexec** に依存し、別のカーネルのコンテキストから、システムを再起動することなく Linux カーネルを起動し、通常は失われてしまう 1 番目のカーネルメモリーの内容を維持できます。

システムクラッシュが発生すると、**kexec** は 2 番目のカーネルで起動します (キャプチャーカーネル)。このキャプチャーカーネルは、システムメモリーの予約部分に収納されています。このため、Kdump は、クラッシュしたカーネルメモリーの内容 (クラッシュダンプ) をキャプチャーして、指定した場所に保存します。このキックスタートコマンドを使用して設定することはできません。 インストール後に **/etc/kdump.conf** 設定ファイルを編集して設定する必要があります。

Kdump の詳細は、kdump のインストール を参照してください。

オプション

- **--enable** - インストール済みのシステムで kdump を有効にします。
- **--disable** - インストール済みのシステムで kdump を無効にします。
- **--reserve-mb=** - kdump 用に予約するメモリーの量 (MiB 単位)。以下に例を示します。

```
%addon com_redhat_kdump --enable --reserve-mb=128
%end
```

数値の代わりに **auto** と指定することもできます。その場合は、インストールプログラムは、**Managing, monitoring and updating the kernel**の Memory requirements for kdump のセクションに記載の基準に基づいて、メモリーの量を自動的に決定します。

kdump を有効にして、**--reserve-mb=** オプションを指定しないと、**auto** の値が使用されます。

- **--enablefadump** - 対応するシステム (特に IBM Power Systems サーバー) へのファームウェア補助によるダンピングを有効にします。

B.6.2. %addon com_redhat_oscap

%addon com_redhat_oscap Kickstart コマンドはオプションです。

OpenSCAP インストールプログラムのアドオンは、インストールシステム上で SCAP (Security Content Automation Protocol) のコンテンツ (セキュリティーポリシー) を適用するために使用されます。Red Hat Enterprise Linux 7.2 以降、このアドオンがデフォルトで有効になりました。有効にすると、この機能の提供に必要なパッケージが自動的にインストールされます。ただし、デフォルトではポリシーが強制されることがなく、明確に設定されている場合を除いて、インストール時およびインストール後にチェックが行われません。

重要

セキュリティーポリシーの適用はすべてのシステムで必要なわけではありません。この画面は、特定のポリシーが業務規定や法令で義務付けられている場合に限り使用してください。

多くのコマンドとは異なり、このアドオンは通常のオプションを受け付けず、**%addon** 定義の本文で鍵と値のペアを使用します。空白は無視されます。値は一重引用符 (') または二重引用符 (") で囲みます。

構文

```
%addon com_redhat_oscap
key = value
%end
```

鍵

アドオンは以下の鍵を認識します。

content-type

セキュリティーコンテンツのタイプ。値は、**datastream**、**archive**、**rpm**、または **scap-security-guide** になります。

content-type を **scap-security-guide** にすると、アドオンは scap-security-guide パッケージが提供するコンテンツを使用します。このパッケージは起動用メディアにあります。つまり、**profile** を除く他のすべての鍵の影響がなくなります。

content-url

セキュリティーコンテンツの場所。コンテンツは、HTTP、HTTPS、FTP のいずれかを使用してアクセスできるようにする必要があります。ローカルストレージは現在、サポートされていません。リモートの場所にあるコンテンツ定義に到達するネットワーク接続が必要になります。

datastream-id

content-url で参照されているデータストリームの ID。**content-type** が **datastream** の場合にのみ使用します。

xccdf-id

使用するベンチマークの ID。

content-path

使用するデータストリームまたは XCCDF ファイルのパスを、アーカイブ内の相対パスで指定します。

profile

適用するプロファイルの ID。デフォルトのプロファイルを使用する場合は **default** を使用してください。

フィンガープリント (fingerprint)

content-url で参照されるコンテンツの MD5、SHA1、または SHA2 のチェックサム。

tailoring-path

使用するテーラリングファイルのパスを、アーカイブの相対パスで指定します。

例

- インストールメディアの scap-security-guide のコンテンツを使用する **%addon com_redhat_oscap** セクションの例は、以下のようになります。

例B.1 SCAP Security Guide を使用した OpenSCAP アドオン定義の例

```
%addon com_redhat_oscap
content-type = scap-security-guide
profile = xccdf_org.ssgproject.content_profile_pci-dss
%end
```

- Web サーバーからカスタムプロファイルを読み込むより複雑な例は、以下のようになります。

例B.2 データストリームを使用した OpenSCAP アドオン定義の例

```
%addon com_redhat_oscap
content-type = datastream
content-url = http://www.example.com/scap/testing_ds.xml
datastream-id = scap_example.com_datastream_testing
xccdf-id = scap_example.com_cref_xccdf.xml
profile =  xccdf_example.com_profile_my_profile
fingerprint = 240f2f18222faa98856c3b4fc50c4195
%end
```

関連情報

- セキュリティーの強化
- OpenSCAP installation program add-on
- OpenSCAP Portal

B.7. ANACONDA で使用されるコマンド

pwpolicy コマンドは、キックスタートファイルの **%anaconda** セクションでのみ使用できる Anaconda UI 固有のコマンドです。

B.7.1. pwpolicy(非推奨)

キックスタートコマンドの **pwpolicy** は任意です。このコマンドを使用して、インストール中にカスタムパスワードポリシーを適用します。このポリシーでは、ユーザーアカウントの root、ユーザー、または luks ユーザーのパスワードを作成する必要があります。パスワードの長さや強度などの要因により、パスワードの有効性が決まります。

構文

```
pwpolicy name [--minlen=length] [--minquality=quality] [--strict|--notstrict] [--emptyok|--notempty] [--changesok|--nochanges]
```

必須オプション

- name - **root**、**user**、または **luks** に置き換え、それぞれ **root** パスワード、ユーザーパスワード、もしくは LUKS パスフレーズのポリシーを強制します。

任意のオプション

- **--minlen=** - パスワードの最低文字数を設定します。デフォルト値は **6** です。
- **--minquality=** - **libpwquality** ライブラリーで定義されるパスワードの最低限の質を設定します。デフォルト値は **1** です。
- **--strict** - 厳密なパスワード強制を有効にします。**--minquality=** と **--minlen=** で指定された要件を満たさないパスワードは拒否されます。このオプションはデフォルトで無効になっています。
- **--notstrict** - **--minquality=** オプションおよび **-minlen=** オプションで指定した最小要件を **満たさない** パスワードは、GUI で **完了** を 2 回クリックすると可能になります。テキストモードインターフェイスでは、同様のメカニズムが使用されます。
- **--emptyok** - 空のパスワードの使用を許可します。デフォルトでユーザーパスワードに有効となっています。
- **--notempty** - 空のパスワードの使用を許可しません。root パスワードと LUKS パスフレーズについて、デフォルトで有効になっています。
- **--changesok** - キックスタートファイルでパスワードが設定されていても、ユーザーインターフェイスでのパスワード変更を許可します。デフォルトでは無効です。

- **--nochanges** - キックスタートファイルで設定されているパスワードの変更を許可しません。デフォルトでは有効です。

注記

- **pwpolicy** コマンドは、キックスタートファイルの **%anaconda** セクションでのみ使用できる Anaconda UI 固有のコマンドです。
- **libpwquality** ライブラリーは、パスワードの最低要件 (長さおよび質) の確認に使用されます。**libpwquality** パッケージが提供する **pwscore** コマンドおよび **pwmake** コマンドを使用してパスワードの質のスコアを確認するか、特定スコアのパスワードをランダムに作成できます。これらのコマンドの詳細は、**pwscore(1)** および **pwmake(1)** の man ページを参照してください。

B.8. システム復旧用キックスタートコマンド

このセクションのキックスタートコマンドは、インストールされたシステムを修復します。

B.8.1. rescue

キックスタートコマンドの **rescue** は任意です。これは、root 権限を備えたシェル環境と、インストールを修復して次のような問題のトラブルシューティングを行うための一連のシステム管理ツールを提供します。

- ファイルシステムを読み取り専用としてマウントする
- ドライバーディスクで提供されているドライバーを拒否リスト登録または追加する
- システムパッケージをインストールまたはアップグレードする
- パーティションを管理する

注記

キックスタートレスキューモードは、systemd およびサービスマネージャーの一部として提供されるレスキューモードおよび緊急モードとは異なります。

rescue コマンドは、システム自体を変更することはありません。読み取り/書き込みモードでシステムを /mnt/sysimage の下にマウントすることにより、レスキュー環境を設定するだけです。システムをマウントしないか、読み取り専用モードでマウントするかを選択できます。

構文

```
rescue [--nomount|--romount]
```

オプション

- **--nomount** または **--romount** - インストールを完了したシステムをレスキュー環境でマウントする方法を制御します。デフォルトでは、インストールプログラムによりシステムの検出が行われてから、読み取りと書き込みのモードでシステムのマウントが行われ、マウントされた場所が通知されます。オプションでマウントを行わない (**--nomount** オプション)、または読み取り専用モードでマウントする (**--romount** オプション) のいずれかを選択できます。指定できるのはどちらか一方です。

注記

レスキューモードを実行するには、キックスタートファイルのコピーを作成し、それに **rescue** コマンドを含めます。

rescue コマンドを使用すると、インストーラーは次の手順を実行します。

1. **%pre** スクリプトを実行します。
2. レスキューモードの環境をセットアップします。
   以下のキックスタートコマンドが有効になります。
   a. updates
   b. sshpw
   c. logging
   d. lang
   e. network
3. 高度なストレージ環境を設定します。
   以下のキックスタートコマンドが有効になります。
   a. fcoe
   b. iscsi
   c. iscsiname
   d. nvdimm
   e. zfcp
4. システムをマウントします。

```
rescue [--nomount|--romount]
```

5. %post スクリプトを実行します。
   この手順は、インストールされたシステムが読み取り/書き込みモードでマウントされている場合にのみ実行されます。
6. シェルを開始します。
7. システムを再起動します。