Red Hat Enterprise Linux 9

RHEL の自動インストール

事前定義された設定から 1 つ以上のシステムに RHEL をデプロイする

Last Updated: 2024-09-11

Red Hat Enterprise Linux 9 RHEL の自動インストール

事前定義された設定から 1 つ以上のシステムに RHEL をデプロイする

法律上の通知

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

概要

キックスタートを使用すると、RHEL のインストールを自動化できます。この方法を使用して、同じ RHEL 設定を複数のシステムにデプロイします。キックスタートは、設定ファイルで指定されたパラメーターに基づいて RHEL をインストールします。インストールソースとして、インストールメディア、ISO ファイル、Red Hat コンテンツ配信ネットワーク (CDN)、またはローカルネットワーク内のサーバーを使用できます。

目次

RED HAT ドキュメントへのフィードバック (英語のみ) 5

パート I. RHEL インストールの準備 6

第1章 システム要件とサポート対象のアーキテクチャー 7
1.1. インストール先として対応しているターゲット 7
1.2. ディスクおよびメモリーの要件 7
1.3. グラフィックスディスプレイの解像度要件 8
1.4. UEFI セキュアブートとベータ版リリースの要件 9

第2章 RHEL システムを RED HAT に登録する方法 10

第3章 インストールメディアのカスタマイズ 11

第4章 起動可能な RHEL 用インストールメディアの作成 12
4.1. インストール起動用メディアオプション 12
4.2. 起動可能な DVD の作成 12
4.3. LINUX で起動可能な USB デバイスの作成 13
4.4. WINDOWS で起動可能な USB デバイスの作成 14
4.5. MACOS で起動可能な USB デバイスの作成 15

第5章 ネットワークベースのリポジトリーの準備 18
5.1. ネットワークインストール用のポート 18
5.2. NFS サーバーへのインストールソースの作成 18
5.3. HTTP または HTTPS を使用するインストールソースの作成 20
5.4. FTP を使用するインストールソースの作成 21

第6章 UEFI HTTP インストールソースの準備 24
6.1. ネットワークインストールの概要 24
6.2. ネットワークブート用の DHCPV4 サーバーの設定 25
6.3. ネットワークブート用の DHCPV6 サーバーの設定 26
6.4. HTTP ブート用の HTTP サーバーの設定 27

第7章 PXE インストールソースの準備 30
7.1. ネットワークインストールの概要 30
7.2. ネットワークブート用の DHCPV4 サーバーの設定 31
7.3. ネットワークブート用の DHCPV6 サーバーの設定 32
7.4. BIOS ベースのクライアント用に TFTP サーバーを設定する 33
7.5. UEFI ベースのクライアント用に TFTP サーバーを設定する 35
7.6. IBM POWER システム用のネットワークサーバーの設定 37

第8章 RHEL ベータ版リリースをインストールおよび起動するために UEFI セキュアブートが有効なシステムを準備する 40
8.1. UEFI セキュアブートおよび RHEL ベータ版リリース 40
8.2. UEFI セキュアブートのベータ公開鍵の追加 40
8.3. ベータ版公開鍵の削除 41

第9章 64 ビット IBM Z での RHEL インストールの準備 42
9.1. 64 ビット IBM Z へのインストールの計画 42
9.2. 64 ビット IBM Z サーバーへのインストールプロセスの概要 43
9.3. 64 ビット IBM Z サーバーに RHEL をインストールするためのブートメディア 44
9.4. ブートパラメーターのカスタマイズ 44
9.5. 64 ビット IBM Z のパラメーターおよび設定ファイル 46
9.6. Z/VM ゲスト仮想マシンへのインストールの準備 53

パート II. RHEL の完全自動および半自動インストール 55

第10章 自動インストールのワークフロー 56

第11章 キックスタートファイルの作成 57
11.1. 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 57
11.2. 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 58
11.3. 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 59
11.4. IMAGE BUILDER を使用したカスタムイメージの作成 59

第12章 UEFI HTTP または PXE インストールソースへのキックスタートファイルの追加 60
12.1. ネットワークインストール用のポート 60
12.2. NFS サーバー上でのインストールファイルの共有 60
12.3. HTTP または HTTPS サーバー上でのインストールファイルの共有 61
12.4. FTP サーバー上でのインストールファイルの共有 63

第13章 半自動インストール: RHEL インストーラーへのキックスタートファイルの提供 66
13.1. ローカルボリューム上でのインストールファイルの共有 66
13.2. 自動ロードのためにローカルボリューム上でインストールファイルを共有する 66

第14章 キックスタートインストールの開始 68
14.1. PXE を使用した自動キックスタートインストールの開始 68
14.2. 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 69
14.3. IBM Z でのインストールを起動して LPAR に RHEL をインストールする 70
14.4. IBM Z でインストールを起動して Z/VM に RHEL をインストールする 72
14.5. インストール中のコンソールとロギング 74

パート III. インストール後のタスク 76

第15章 SUBSCRIPTION MANAGER を使用した RHEL の登録 77
15.1. インストーラー GUI を使用した RHEL 9 の登録 77
15.2. REGISTRATION ASSISTANT 77
15.3. コマンドラインを使用したシステムの登録 77

第16章 SUBSCRIPTION-MANAGER コマンドラインツールを使用したシステムの目的の設定 79

第17章 64 ビット IBM Z で LINUX インスタンスの設定 83
17.1. DASD の追加 83
17.2. DASD のオンラインへの動的な設定 83
17.3. ローレベルフォーマットによる新規 DASD の準備 84
17.4. DASD を永続的にオンラインに設定する 85
17.5. ルートファイルシステムの一部である DASD 85
17.6. ルートファイルシステムの一部ではない DASD 87
17.7. ルートファイルシステムの一部である FCP LUN 88
17.8. ルートファイルシステムの一部ではない FCP LUN 90
17.9. QETH デバイスの追加 91
17.10. QETH デバイスの動的な追加 91
17.11. QETH デバイスの永続的な追加 93
17.12. ネットワークの ROOT ファイルシステム用の 64 ビットの IBM Z ネットワークデバイスの設定 96

第18章 システムの保護 97

第19章 コマンドラインを使用した ARM への KERNEL-64K のインストール 98

第20章 サブスクリプションサービスの変更 100
20.1. SUBSCRIPTION MANAGEMENT SERVER からの登録解除 100
20.2. SATELLITE SERVER からの登録解除 101

パート IV. 付録 **102**

付録A キックスタートスクリプトのファイル形式のリファレンス **103**
A.1. キックスタートファイルの形式 103
A.2. キックスタートでのパッケージ選択 104
A.3. キックスタートファイル内のスクリプト 109
A.4. キックスタートでのエラー処理セクション 114
A.5. キックスタートのアドオンセクション 114

付録B キックスタートのコマンドおよびオプションのリファレンス **116**
B.1. キックスタートの変更 116
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 117
B.3. システム設定用キックスタートコマンド 129
B.4. ネットワーク設定用キックスタートコマンド 142
B.5. ストレージを処理するキックスタートコマンド 148
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 176
B.7. ANACONDA で使用されるコマンド 179
B.8. システム復旧用キックスタートコマンド 180

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。

パート I. RHEL インストールの準備

第1章 システム要件とサポート対象のアーキテクチャー

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。RHEL は、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。

インストールする前に、システム、ハードウェア、セキュリティー、メモリー、および RAID に関するガイドラインを確認してください。

システムを仮想ホストとして使用する場合は、仮想化に必要なハードウェア要件 を確認してください。

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

1.1. インストール先として対応しているターゲット

インストールターゲットは、Red Hat Enterprise Linux を格納し、システムを起動するストレージデバイスです。Red Hat Enterprise Linux は、AMD64、Intel 64、および 64 ビット ARM のシステム向けに、以下のインストールターゲットに対応しています。

- SCSI、SATA、SAS などの標準的な内部インターフェイスで接続するストレージ
- BIOS/ファームウェアの RAID デバイス
- **nd_pmem** ドライバーがサポートする、セクターモードに設定された Intel 64 および AMD64 アーキテクチャー上の NVDIMM デバイス
- ファイバーチャネルのホストバスアダプターおよびマルチパスのデバイス。製造元が提供しているドライバーが必要な場合があります。
- Xen 仮想マシンの Intel のプロセッサーの Xen ブロックデバイス
- KVM 仮想マシンの Intel のプロセッサーの VirtIO ブロックデバイス

Red Hat では、USB ドライブや SD メモリーカードへのインストールはサポートしていません。サードパーティーによる仮想化技術のサポートは、Red Hat Hardware Compatibility List を参照してください。

1.2. ディスクおよびメモリーの要件

複数のオペレーティングシステムがインストールされている場合は、割り当てられたディスク領域が Red Hat Enterprise Linux で必要なディスク領域とは異なることを確認することが重要です。場合によっては、特定のパーティションを Red Hat Enterprise Linux 専用にすることが重要になります。たとえば、AMD64、Intel 64、および 64 ビット ARM の場合は、少なくとも 2 つのパーティション (/ およ

び **swap**) を RHEL 専用にする必要があります。IBM Power Systems サーバーの場合は、少なくとも 3 つのパーティション (/、**swap**、および **PReP** ブートパーティション) を RHEL 専用にする必要があります。

さらに、使用可能なディスク容量が最低 10 GiB 必要です。Red Hat Enterprise Linux をインストールするには、パーティションが分割されていないディスク領域か、削除できるパーティション内に、最低 10 GiB の容量が必要です。

詳細は、パーティション設定のリファレンス を参照してください。

表1.1 最小 RAM 要件

インストールタイプ	推奨される最小 RAM
ローカルメディアによるインストール (USB、DVD)	• aarch64、s390x、および x86_64 アーキテクチャー: 1.5 GiB • ppc64le アーキテクチャー: 3 GiB
NFS ネットワークインストール	• aarch64、s390x、および x86_64 アーキテクチャー: 1.5 GiB • ppc64le アーキテクチャー: 3 GiB
HTTP、HTTPS、または FTP ネットワークインストール	• s390x および x86_64 アーキテクチャー: 3 GiB • aarch64 および ppc64le アーキテクチャー: 4 GiB

推奨される最小要件より少ないメモリーでインストールを完了できます。正確な要件は、環境とインストールパスにより異なります。ご使用の環境に必要な最小 RAM を決定するために、さまざまな設定をテストすることが推奨されます。キックスタートファイルを使用して Red Hat Enterprise Linux をインストールするには、標準的なインストールと同じように推奨される最小 RAM 要件があります。ただし、キックスタートファイルに追加のメモリーを必要とするコマンド、または RAM ディスクにデータを書き込むコマンドが含まれている場合は、追加の RAM が必要になることがあります。詳細は、RHEL の自動インストール ドキュメントを参照してください。

1.3. グラフィックスディスプレイの解像度要件

Red Hat Enterprise Linux をスムーズにエラーなしにインストールするには、システムに次の最小解像度が必要です。

表1.2 ディスプレイ解像度

製品バージョン	解決方法
Red Hat Enterprise Linux 9	**最小**: 800 x 600 **推奨**:1026×768

1.4. UEFI セキュアブートとベータ版リリースの要件

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 のベータ版リリースをインストールする予定がある場合は、UEFI セキュアブートオプションを無効にしてから、インストールを開始します。

UEFI セキュアブートでは、オペレーティングシステムのカーネルが、対応する公開鍵を使用してシステムのファームウェアが検証できる、認識済みの秘密鍵で署名されている必要があります。Red Hat Enterprise Linux ベータ版リリースの場合には、カーネルは Red Hat ベータ版固有の公開鍵で署名されていますが、この鍵はデフォルトではシステムで認識できません。その結果、インストールメディアの起動にも失敗します。

関連情報

- IBM への RHEL のインストールについては、IBM のインストールドキュメント を参照してください。
- セキュリティーの強化
- RHEL システムイメージのカスタマイズ
- Red Hat Ecosystem Catalog
- RHEL technology capabilities and limits

第2章 RHEL システムを RED HAT に登録する方法

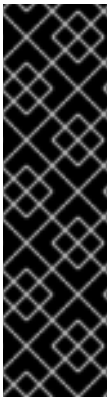
登録することで、システムと Red Hat 間で認可済みの接続が確立されます。Red Hat は、登録されたシステム (物理マシンか仮想マシンかを問わず) に、システムを識別および認証する証明書を発行して、Red Hat から保護されたコンテンツ、ソフトウェア更新、セキュリティーパッチ、サポート、および管理サービスをシステムが受けられるようにします。

有効なサブスクリプションを使用すると、以下の方法で Red Hat Enterprise Linux (RHEL) システムを登録できます。

- インストーラーのグラフィカルユーザーインターフェイス (GUI) またはテキストユーザーインターフェイス (TUI) を使用して、インストールプロセス中に登録する
- インストール後にコマンドラインインターフェイス (CLI) を使用する
- インストール時またはインストール後に自動的にキックスタートスクリプトまたはアクティベーションキーを使用する

システムを登録する特定の手順は、使用している RHEL のバージョンと、選択した登録方法によって異なります。

システムを Red Hat に登録すると、システムの管理とレポートデータに使用できる機能が有効になります。たとえば、登録済みシステムには、Red Hat コンテンツ配信ネットワーク (CDN) または Red Hat Satellite Server を介して、サブスクライブした製品向けの、保護されたコンテンツリポジトリーにアクセスする権限が与えられます。これらのコンテンツリポジトリーには、アクティブなサブスクリプションを持つお客様のみが利用できる Red Hat ソフトウェアパッケージと更新が含まれています。これらのパッケージおよび更新には、RHEL およびその他の Red Hat 製品のセキュリティーパッチ、バグ修正、新機能が含まれます。



重要

エンタイトルメントベースのサブスクリプションモデルは非推奨となり、将来廃止される予定です。Simple Content Access がデフォルトのサブスクリプションモデルになりました。これにより、システムの Red Hat サブスクリプションコンテンツにアクセスする前に、そのシステムにサブスクリプションを割り当てる必要がなくなり、サブスクリプションエクスペリエンスが向上します。お使いの Red Hat アカウントでエンタイトルメントベースのサブスクリプションモデルを使用している場合は、Red Hat アカウントチーム (テクニカルアカウントマネージャー (TAM) やソリューションアーキテクト (SA) など) に問い合わせて、Simple Content Access への移行の準備をしてください。詳細は、Transition of subscription services to the hybrid cloud を参照してください。

第3章 インストールメディアのカスタマイズ

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4章 起動可能な RHEL 用インストールメディアの作成

カスタマーポータル から ISO ファイルをダウンロードして、USB や DVD などの起動可能な物理インストールメディアを準備できます。RHEL 8 以降、Red Hat は **Server** 用と **Workstation** 用の個別のバリアントを提供しなくなりました。**Red Hat Enterprise Linux for x86_64**には、**Server** 機能と **Workstation** 機能の両方が含まれています。**Server** および **Workstation** の区別は、インストールまたは設定プロセス中にシステム目的ロールを通じて管理されます。

カスタマーポータルから ISO ファイルをダウンロードした後、USB や DVD などの起動可能な物理インストールメディアを作成して、インストールプロセスを続行します。

USB ドライブが禁止されているセキュアな環境では、Image Builder を使用して参照イメージを作成し、デプロイすることを検討してください。この方法により、システムの整合性を維持しながらセキュリティーポリシーへの準拠を確保できます。詳細は、Image Builder のドキュメント を参照してください。

4.1. インストール起動用メディアオプション

Red Hat Enterprise Linux インストールプログラムを起動する方法はいくつかあります。

フルインストール用 DVD または USB フラッシュドライブ

DVD ISO イメージを使用して、フルインストールの DVD または USB フラッシュドライブを作成します。ソフトウェアパッケージをインストールする場合は、DVD または USB フラッシュドライブを、ブートデバイスおよびインストールソースとして使用できます。

最小インストール用の DVD、CD、または USB フラッシュドライブ

最小インストール用 CD、DVD、または USB フラッシュドライブは、**Boot ISO** イメージを使用して作成されます。これには、システムを起動し、インストールプログラムを開始するのに最低限必要なファイルのみが含まれます。

重要

コンテンツ配信ネットワーク (CDN) を使用して必要なソフトウェアパッケージをダウンロードする場合は、**Boot ISO** イメージに、必要なソフトウェアパッケージを含むインストールソースが必要です。

4.2. 起動可能な DVD の作成

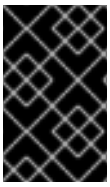
起動可能なインストール DVD は、ディスク書き込みソフトウェアや DVD バーナーを使用して作成できます。ISO イメージファイルから DVD を作成する手順は、オペレーティングシステムや、インストールされているディスク書き込みソフトウェアにより大きく異なります。DVD への ISO イメージファイルの書き込み方法は、お使いの書き込みソフトウェアのドキュメントを参照してください。

警告

起動可能な DVD は、DVD ISO イメージ (フルインストール) または Boot ISO イメージ (最小インストール) のいずれかを使用して作成できます。ただし、DVD ISO イメージが 4.7 GB より大きくなり、1 層または 2 層 DVD に収まらない場合があります。作業を続行する前に、DVD ISO イメージファイルのサイズを確認してください。DVD ISO イメージを使用して起動可能なインストールメディアを作成する場合は、USB フラッシュドライブが推奨されます。USB ドライブが禁止されている環境の場合は、Image Builder のドキュメント を参照してください。

4.3. LINUX で起動可能な USB デバイスの作成

起動可能な USB デバイスを作成し、それを使用して他のマシンに Red Hat Enterprise Linux をインストールできます。



重要

この手順を実行すると、USB ドライブに保存しておいたデータはすべて警告なしに上書きされます。データをバックアップするか、空のフラッシュドライブを使用してください。起動可能な USB ドライブは、データの保存には使用できません。

前提条件

- Product Downloads ページからフルインストール DVD ISO または最小インストールブート ISO イメージをダウンロードした。
- ISO イメージに十分な容量の USB フラッシュドライブがある。必要なサイズはさまざまですが、推奨される USB サイズは 8 GB です。

手順

1. USB フラッシュドライブをシステムに接続します。
2. ターミナルウィンドウを開き、最近のイベントのログを表示します。

```
$ dmesg|tail
```

このログの下部に、接続している USB フラッシュドライブから出力されたメッセージが表示されます。接続したデバイスの名前を記録してください。

3. root ユーザーとしてログインします。

```
$ su -
```

プロンプトに従い root パスワードを入力します。

4. ドライブに割り当てられているデバイスノードを見つけます。この例で使用されているドライブの名前は **sdd** です。

```
# dmesg|tail
```

```
[288954.686557] usb 2-1.8: New USB device strings: Mfr=0, Product=1, SerialNumber=2
[288954.686559] usb 2-1.8: Product: USB Storage
[288954.686562] usb 2-1.8: SerialNumber: 000000009225
[288954.712590] usb-storage 2-1.8:1.0: USB Mass Storage device detected
[288954.712687] scsi host6: usb-storage 2-1.8:1.0
[288954.712809] usbcore: registered new interface driver usb-storage
[288954.716682] usbcore: registered new interface driver uas
[288955.717140] scsi 6:0:0:0: Direct-Access     Generic  STORAGE DEVICE   9228 PQ: 0
ANSI: 0
[288955.717745] sd 6:0:0:0: Attached scsi generic sg4 type 0
[288961.876382] sd 6:0:0:0: sdd Attached SCSI removable disk
```

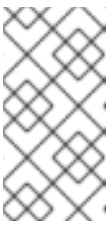
5. 挿入された USB デバイスが自動的にマウントされる場合は、次の手順に進む前にマウントを解除してください。アンマウントするには、**umount** コマンドを使用します。詳細は、umount を使用したファイルシステムのアンマウント を参照してください。

6. ISO イメージを USB デバイスに直接書き込みます。

```
# dd if=/image_directory/image.iso of=/dev/device
```

   - **/image_directory/image.iso** を、ダウンロードした ISO イメージファイルへのフルパスに置き換えます。
   - **device** を、**dmesg** コマンドで取得したデバイス名に置き換えます。
     この例では、ISO イメージのフルパスが **/home/testuser/Downloads/rhel-9-x86_64-boot.iso** で、検出されたデバイス名が **sdd** です。

```
# dd if=/home/testuser/Downloads/rhel-9-x86_64-boot.iso of=/dev/sdd
```



注記

デバイス上のパーティション名ではなく、正しいデバイス名を使用していることを確認してください。パーティション名は、通常、数字の接尾辞が付いたデバイス名です。たとえば、**sdd** がデバイス名の場合、デバイス **sdd** 上のパーティションの名前は、**sdd1** になります。

7. **dd** コマンドがデバイスへのイメージの書き込みを終了するのを待ちます。**sync** コマンドを実行して、キャッシュされた書き込みをデバイスに同期します。データ転送が完了すると、**#** プロンプトが表示されます。プロンプトが表示されたら、root アカウントからログアウトし、USB ドライブを取り外します。これで、USB ドライブをブートデバイスとして使用できるようになりました。

4.4. WINDOWS で起動可能な USB デバイスの作成

さまざまなツールを使用して、Windows システムに起動可能な USB デバイスを作成できます。Red Hat は、https://github.com/FedoraQt/MediaWriter/releases からダウンロードできる Fedora Media Writer の使用を推奨します。Fedora Media Writer はコミュニティー製品であり、Red Hat のサポート対象外になる点に注意してください。このツールの問題は、https://github.com/FedoraQt/MediaWriter/issues から報告できます。

重要

この手順を実行すると、USB ドライブに保存しておいたデータはすべて警告なしに上書きされます。データをバックアップするか、空のフラッシュドライブを使用してください。起動可能な USB ドライブは、データの保存には使用できません。

前提条件

- Product Downloads ページからフルインストール DVD ISO または最小インストールブート ISO イメージをダウンロードした。
- ISO イメージに十分な容量の USB フラッシュドライブがある。必要なサイズはさまざまですが、推奨される USB サイズは 8 GB です。

手順

1. https://github.com/FedoraQt/MediaWriter/releases から Fedora Media Writer をダウンロードしてインストールします。
2. USB フラッシュドライブをシステムに接続します。
3. Fedora Media Writer を開きます。
4. メイン画面で **Custom Image** をクリックして、ダウンロードしておいた Red Hat Enterprise Linux ISO イメージを選択します。
5. **Write Custom Image** 画面で、使用するドライブを選択します。
6. **Write to disk** をクリックします。起動用メディアの作成プロセスが開始します。プロセスが完了するまでドライブを抜かないでください。ISO イメージのサイズや、USB ドライブの書き込み速度により、この操作には数分かかる場合があります。
7. 操作が完了したら、USB ドライブをアンマウントします。これで USB ドライブを起動デバイスとして使用する準備が整いました。

4.5. MACOS で起動可能な USB デバイスの作成

起動可能な USB デバイスを作成し、それを使用して他のマシンに Red Hat Enterprise Linux をインストールできます。

重要

この手順を実行すると、USB ドライブに保存しておいたデータはすべて警告なしに上書きされます。データをバックアップするか、空のフラッシュドライブを使用してください。起動可能な USB ドライブは、データの保存には使用できません。

前提条件

- Product Downloads ページからフルインストール DVD ISO または最小インストールブート ISO イメージをダウンロードした。
- ISO イメージに十分な容量の USB フラッシュドライブがある。必要なサイズはさまざまですが、推奨される USB サイズは 8 GB です。

手順

1. USB フラッシュドライブをシステムに接続します。

2. **diskutil list** コマンドでデバイスパスを特定します。デバイスパスの形式は **/dev/disknumber** です。**number** はディスクの数になります。ディスク番号は、0 から始まります。通常、**disk0** は OS X リカバリーディスク、**disk1** はメインの OS X インストールになります。以下の例では、**disk2** が USB デバイスです。

```
$ diskutil list
/dev/disk0
#:                       TYPE NAME                    SIZE       IDENTIFIER
0:      GUID_partition_scheme                        *500.3 GB   disk0
1:                        EFI EFI                     209.7 MB   disk0s1
2:          Apple_CoreStorage                         400.0 GB   disk0s2
3:                Apple_Boot Recovery HD              650.0 MB   disk0s3
4:          Apple_CoreStorage                         98.8 GB    disk0s4
5:                Apple_Boot Recovery HD              650.0 MB   disk0s5
/dev/disk1
#:                       TYPE NAME                    SIZE       IDENTIFIER
0:                 Apple_HFS YosemiteHD              *399.6 GB   disk1
Logical Volume on disk0s1
8A142795-8036-48DF-9FC5-84506DFBB7B2
Unlocked Encrypted
/dev/disk2
#:                       TYPE NAME                    SIZE       IDENTIFIER
0:     FDisk_partition_scheme                        *8.1 GB     disk2
1:               Windows_NTFS SanDisk USB            8.1 GB     disk2s1
```

3. NAME、TYPE、および SIZE の列をフラッシュドライブと比較し、USB フラッシュドライブを特定します。たとえば、NAME は、**Finder** ツールのフラッシュドライブアイコンのタイトルになります。この値は、フラッシュドライブの情報パネルの値と比較することもできます。

4. フラッシュドライブのファイルシステムボリュームをアンマウントします。

```
$ diskutil unmountDisk /dev/disknumber
   Unmount of all volumes on disknumber was successful
```

コマンドが完了すると、デスクトップからフラッシュドライブのアイコンが消えます。アイコンが消えない場合は、誤ったディスクを選択した可能性があります。誤ってシステムディスクのマウントを解除しようとすると、**failed to unmount** エラーが返されます。

5. ISO イメージをフラッシュドライブに書き込みます。

```
# sudo dd if=/path/to/image.iso of=/dev/rdisknumber
```

注記

macOS では、ブロック (**/dev/disk***) とキャラクターデバイス (**/dev/rdisk***) の両方のファイルが各ストレージデバイスに提供されます。**/dev/rdisknumber** キャラクターデバイスにイメージを書き込む方が、**/dev/disknumber** ブロックデバイスに書き込むよりも高速です。

たとえば、**/Users/user_name/Downloads/rhel-9-x86_64-boot.iso** ファイルを **/dev/rdisk2** デバイスに書き込むには、以下のコマンドを実行します。

```
# sudo dd if=/Users/user_name/Downloads/rhel-9-x86_64-boot.iso of=/dev/rdisk2
```

6. **dd** コマンドがデバイスへのイメージの書き込みを終了するのを待ちます。データ転送が完了すると、**#** プロンプトが表示されます。プロンプトが表示されたら、root アカウントからログアウトして、USB ドライブを取り外します。これで USB ドライブを起動デバイスとして使用する準備が整いました。

関連情報

- システムの目的の設定
- ISO for RHEL 8/9 Server or Workstation

第5章 ネットワークベースのリポジトリーの準備

ネットワークシステムから RHEL をインストールするには、リポジトリーを準備する必要があります。

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

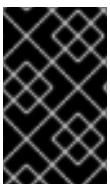
- ネットワークのセキュリティー保護

5.2. NFS サーバーへのインストールソースの作成

このインストール方法を使用すると、物理メディアに接続することなく、単一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。
- Product Downloads ページからフルインストール DVD ISO をダウンロードした。
- 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。
- 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。詳細は、ネットワークインストール用のポートを参照してください。



重要

必ず **inst.ks** と **inst.repo** で異なるパスを使用してください。NFS を使用してインストールソースをホストする場合、同じ NFS 共有を使用してキックスタートをホストすることはできません。

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。

5.3. HTTP または HTTPS を使用するインストールソースの作成

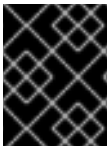
インストールツリー (DVD ISO イメージから抽出したコンテンツと、有効な **.treeinfo** ファイル含むディレクトリー) を使用したネットワークベースのインストール用のインストールソースを作成できます。インストールソースには、HTTP、または HTTPS でアクセスします。

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。
- Product Downloads ページからフルインストール DVD ISO をダウンロードした。
- 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。
- 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。詳細は、ネットワーク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

5.4. 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。
- Product Downloads ページからフルインストール DVD ISO をダウンロードした。
- 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。
- 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。詳細は、ネットワーク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6章 UEFI HTTP インストールソースの準備

ローカルネットワーク上のサーバーの管理者は、ネットワーク上の他のシステムの HTTP ブートとネットワークインストールを有効にするように HTTP サーバーを設定できます。

6.1. ネットワークインストールの概要

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。

サーバー

DHCP サーバー、HTTP、HTTPS、FTP または NFS サーバー、および PXE ブートの場合は TFTP サーバーを実行するシステム。各サーバーを実行する物理システムが同じである必要はありませんが、このセクションの手順では、1 つのシステムですべてのサーバーを実行していることが想定されています。

クライアント

Red Hat Enterprise Linux をインストールしているシステム。インストールが開始すると、クライアントは DHCP サーバーに問い合わせ、HTTP サーバーまたは TFTP サーバーからブートファイルを受け取り、HTTP サーバー、HTTPS サーバー、FTP サーバー、または NFS サーバーからインストールイメージをダウンロードします。その他のインストール方法とは異なり、クライアントはインストールを開始するのに物理的な起動メディアを必要としません。

注記

ネットワークからクライアントを起動するには、ファームウェアまたはクライアントのクイックブートメニューでネットワークブートを有効にします。ハードウェアによっては、ネットワークから起動するオプションが無効になっていたり、利用できない場合があります。

HTTP または PXE を使用してネットワークから Red Hat Enterprise Linux をインストールする準備を行う手順は次のとおりです。

手順

1. インストール ISO イメージまたはインストールツリーを NFS サーバー、HTTPS サーバー、HTTP サーバー、または FTP サーバーにエクスポートします。
2. HTTP または TFTP サーバーと DHCP サーバーを設定し、サーバー上で HTTP または TFTP サービスを起動します。
3. クライアントを起動して、インストールを開始します。

次のネットワークブートプロトコルを選択できます。

HTTP

Red Hat は、クライアント UEFI がサポートしている場合は HTTP ブートを使用することを推奨します。通常、HTTP ブートは信頼性に優れています。

PXE (TFTP)

PXE ブートはクライアントシステムによって広くサポートされています。ただし、このプロトコルを介したブートファイルの送信は低速で、タイムアウトにより失敗する可能性があります。

関連情報

関連情報

- ネットワークベースのリポジトリーの準備
- Red Hat Satellite の製品ドキュメント

6.2. ネットワークブート用の DHCPV4 サーバーの設定

サーバー上で DHCP バージョン 4 (DHCPv4) サービスを有効にし、ネットワークブート機能を提供できるようにします。

前提条件

- IPv4 プロトコルを介したネットワークインストールを準備中である。
  IPv6 の場合は、ネットワークブート用の DHCPv6 サーバーの設定 を参照してください。
- サーバーのネットワークアドレスがわかっている。
  以下の手順の例では、サーバーには次の設定のネットワークカードが搭載されています。

  IPv4 アドレス

  192.168.124.2/24

  IPv4 ゲートウェイ

  192.168.124.1

手順

1. DHCP サーバーをインストールします。

```
dnf install dhcp-server
```

2. DHCPv4 サーバーをセットアップします。**/etc/dhcp/dhcpd.conf** ファイルに次の設定を入力します。アドレスはネットワークカードと一致するように置き換えます。

```
option architecture-type code 93 = unsigned integer 16;

subnet 192.168.124.0 netmask 255.255.255.0 {
  option routers 192.168.124.1;
  option domain-name-servers 192.168.124.1;
  range 192.168.124.100 192.168.124.200;
  class "pxeclients" {
    match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
    next-server 192.168.124.2;
          if option architecture-type = 00:07 {
            filename "redhat/EFI/BOOT/BOOTX64.EFI";
          }
          else {
            filename "pxelinux/pxelinux.0";
          }
  }
  class "httpclients" {
    match if substring (option vendor-class-identifier, 0, 10) = "HTTPClient";
    option vendor-class-identifier "HTTPClient";
```

```
    filename "http://192.168.124.2/redhat/EFI/BOOT/BOOTX64.EFI";
  }
}
```

3. DHCPv4 サービスを起動します。

```
# systemctl enable --now dhcpd
```

6.3. ネットワークブート用の DHCPV6 サーバーの設定

サーバー上で DHCP バージョン 6 (DHCPv4) サービスを有効にし、ネットワークブート機能を提供できるようにします。

前提条件

- IPv6 プロトコルを介したネットワークインストールを準備中である。
  IPv4 の場合は、ネットワークブート用の DHCPv4 サーバーの設定 を参照してください。
- サーバーのネットワークアドレスがわかっている。
  以下の手順の例では、サーバーには次の設定のネットワークカードが搭載されています。

  IPv6 アドレス

  fd33:eb1b:9b36::2/64

  IPv6 ゲートウェイ

  fd33:eb1b:9b36::1

手順

1. DHCP サーバーをインストールします。

```
dnf install dhcp-server
```

2. DHCPv6 サーバーをセットアップします。**/etc/dhcp/dhcpd6.conf** ファイルに次の設定を入力します。アドレスはネットワークカードと一致するように置き換えます。

```
option dhcp6.bootfile-url code 59 = string;
option dhcp6.vendor-class code 16 = {integer 32, integer 16, string};

subnet6 fd33:eb1b:9b36::/64 {
        range6 fd33:eb1b:9b36::64 fd33:eb1b:9b36::c8;

        class "PXEClient" {
                match substring (option dhcp6.vendor-class, 6, 9);
        }

        subclass "PXEClient" "PXEClient" {
                option dhcp6.bootfile-url
"tftp://[fd33:eb1b:9b36::2]/redhat/EFI/BOOT/BOOTX64.EFI";
        }

        class "HTTPClient" {
                match substring (option dhcp6.vendor-class, 6, 10);
        }
```

```
        subclass "HTTPClient" "HTTPClient" {
                option dhcp6.bootfile-url
"http://[fd33:eb1b:9b36::2]/redhat/EFI/BOOT/BOOTX64.EFI";
                option dhcp6.vendor-class 0 10 "HTTPClient";
        }
}
```

3. DHCPv6 サービスを起動します。

```
# systemctl enable --now dhcpd6
```

4. DHCPv6 パケットがファイアウォールの RP フィルターによって破棄されている場合は、そのログを確認してください。ログに **rpfilter_DROP** エントリーが含まれている場合は、**/etc/firewalld/firewalld.conf** ファイルで次の設定を使用してフィルターを無効にします。

```
IPv6_rpfilter=no
```

6.4. HTTP ブート用の HTTP サーバーの設定

サーバーがネットワーク上で HTTP ブートリソースを提供できるように、サーバーに **httpd** サービスをインストールして有効にする必要があります。

前提条件

- サーバーのネットワークアドレスがわかっている。
  次の例では、サーバーには IPv4 アドレス **192.168.124.2** のネットワークカードが搭載されています。

手順

1. HTTP サーバーをインストールします。

```
# dnf install httpd
```

2. **/var/www/html/redhat/** ディレクトリーを作成します。

```
# mkdir -p /var/www/html/redhat/
```

3. RHEL DVD ISO ファイルをダウンロードします。All Red Hat Enterprise Linux Downloads を参照してください。

4. ISO ファイルのマウントポイントを作成します。

```
# mkdir -p /var/www/html/redhat/iso/
```

5. ISO ファイルをマウントします。

```
# mount -o loop,ro -t iso9660 path-to-RHEL-DVD.iso /var/www/html/redhat/iso
```

6. マウントされた ISO ファイルからブートローダー、カーネル、**initramfs** を HTML ディレクトリーにコピーします。

```
# cp -r /var/www/html/redhat/iso/images /var/www/html/redhat/
# cp -r /var/www/html/redhat/iso/EFI /var/www/html/redhat/
```

7. ブートローダー設定を編集可能にします。

```
# chmod 644 /var/www/html/redhat/EFI/BOOT/grub.cfg
```

8. **/var/www/html/redhat/EFI/BOOT/grub.cfg** ファイルを編集し、次のように内容を置き換えます。

```
set default="1"

function load_video {
  insmod efi_gop
  insmod efi_uga
  insmod video_bochs
  insmod video_cirrus
  insmod all_video
}

load_video
set gfxpayload=keep
insmod gzio
insmod part_gpt
insmod ext2

set timeout=60
# END /etc/grub.d/00_header #

search --no-floppy --set=root -l 'RHEL-9-3-0-BaseOS-x86_64'

# BEGIN /etc/grub.d/10_linux #
menuentry 'Install Red Hat Enterprise Linux 9.3' --class fedora --class gnu-linux --class gnu
--class os {
    linuxefi ../../images/pxeboot/vmlinuz inst.repo=http://192.168.124.2/redhat/iso quiet
    initrdefi ../../images/pxeboot/initrd.img
}
menuentry 'Test this media & install Red Hat Enterprise Linux 9.3' --class fedora --class
gnu-linux --class gnu --class os {
    linuxefi ../../images/pxeboot/vmlinuz inst.repo=http://192.168.124.2/redhat/iso quiet
    initrdefi ../../images/pxeboot/initrd.img
}
submenu 'Troubleshooting -->' {
    menuentry 'Install Red Hat Enterprise Linux 9.3 in text mode' --class fedora --class gnu-
linux --class gnu --class os {
     linuxefi ../../images/pxeboot/vmlinuz inst.repo=http://192.168.124.2/redhat/iso inst.text
quiet
     initrdefi ../../images/pxeboot/initrd.img
    }
    menuentry 'Rescue a Red Hat Enterprise Linux system' --class fedora --class gnu-linux --
class gnu --class os {
     linuxefi ../../images/pxeboot/vmlinuz inst.repo=http://192.168.124.2/redhat/iso inst.rescue
quiet
```

```
        initrdefi ../../images/pxeboot/initrd.img
    }
}
```

このファイル内で、次の文字列を置き換えます。

RHEL-9-3-0-BaseOS-x86_64 および Red Hat Enterprise Linux 9.3

ダウンロードした RHEL のバージョンと一致するようにバージョン番号を編集します。

192.168.124.2

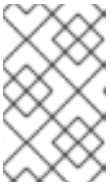
サーバーの IP アドレスに置き換えます。

9. EFI ブートファイルを実行可能にします。

```
# chmod 755 /var/www/html/redhat/EFI/BOOT/BOOTX64.EFI
```

10. ファイアウォールでポートを開いて、HTTP (80)、DHCP (67、68)、および DHCPv6 (546、547) トラフィックを許可します。

```
# firewall-cmd --zone public \
            --add-port={80/tcp,67/udp,68/udp,546/udp,547/udp}
```



注記

このコマンドは、次にサーバーを再起動するまで、一時的にアクセスを有効にします。永続的アクセスを有効にするには、コマンドに **--permanent** オプションを追加します。

11. ファイアウォールルールを再読み込みします。

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

12. HTTP サーバーを起動します。

```
# systemctl enable --now httpd
```

13. **html** ディレクトリーとそのコンテンツを読み取り可能および実行可能にします。

```
# chmod -cR u=rwX,g=rX,o=rX /var/www/html
```

14. **html** ディレクトリーの SELinux コンテキストを復元します。

```
# restorecon -FvvR /var/www/html
```

第7章 PXE インストールソースの準備

PXE ブートとネットワークインストールを有効にするには、PXE サーバーで TFTP と DHCP を設定する必要があります。

7.1. ネットワークインストールの概要

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。

サーバー

DHCP サーバー、HTTP、HTTPS、FTP または NFS サーバー、および PXE ブートの場合は TFTP サーバーを実行するシステム。各サーバーを実行する物理システムが同じである必要はありませんが、このセクションの手順では、1 つのシステムですべてのサーバーを実行していることが想定されています。

クライアント

Red Hat Enterprise Linux をインストールしているシステム。インストールが開始すると、クライアントは DHCP サーバーに問い合わせ、HTTP サーバーまたは TFTP サーバーからブートファイルを受け取り、HTTP サーバー、HTTPS サーバー、FTP サーバー、または NFS サーバーからインストールイメージをダウンロードします。その他のインストール方法とは異なり、クライアントはインストールを開始するのに物理的な起動メディアを必要としません。

注記

ネットワークからクライアントを起動するには、ファームウェアまたはクライアントのクイックブートメニューでネットワークブートを有効にします。ハードウェアによっては、ネットワークから起動するオプションが無効になっていたり、利用できない場合があります。

HTTP または PXE を使用してネットワークから Red Hat Enterprise Linux をインストールする準備を行う手順は次のとおりです。

手順

1. インストール ISO イメージまたはインストールツリーを NFS サーバー、HTTPS サーバー、HTTP サーバー、または FTP サーバーにエクスポートします。
2. HTTP または TFTP サーバーと DHCP サーバーを設定し、サーバー上で HTTP または TFTP サービスを起動します。
3. クライアントを起動して、インストールを開始します。

次のネットワークブートプロトコルを選択できます。

HTTP

Red Hat は、クライアント UEFI がサポートしている場合は HTTP ブートを使用することを推奨します。通常、HTTP ブートは信頼性に優れています。

PXE (TFTP)

PXE ブートはクライアントシステムによって広くサポートされています。ただし、このプロトコルを介したブートファイルの送信は低速で、タイムアウトにより失敗する可能性があります。

関連情報

関連情報

- ネットワークベースのリポジトリーの準備
- Red Hat Satellite の製品ドキュメント

7.2. ネットワークブート用の DHCPV4 サーバーの設定

サーバー上で DHCP バージョン 4 (DHCPv4) サービスを有効にし、ネットワークブート機能を提供できるようにします。

前提条件

- IPv4 プロトコルを介したネットワークインストールを準備中である。
  IPv6 の場合は、ネットワークブート用の DHCPv6 サーバーの設定 を参照してください。
- サーバーのネットワークアドレスがわかっている。
  以下の手順の例では、サーバーには次の設定のネットワークカードが搭載されています。

  IPv4 アドレス

  192.168.124.2/24

  IPv4 ゲートウェイ

  192.168.124.1

手順

1. DHCP サーバーをインストールします。

```
dnf install dhcp-server
```

2. DHCPv4 サーバーをセットアップします。**/etc/dhcp/dhcpd.conf** ファイルに次の設定を入力します。アドレスはネットワークカードと一致するように置き換えます。

```
option architecture-type code 93 = unsigned integer 16;

subnet 192.168.124.0 netmask 255.255.255.0 {
  option routers 192.168.124.1;
  option domain-name-servers 192.168.124.1;
  range 192.168.124.100 192.168.124.200;
  class "pxeclients" {
    match if substring (option vendor-class-identifier, 0, 9) = "PXEClient";
    next-server 192.168.124.2;
        if option architecture-type = 00:07 {
          filename "redhat/EFI/BOOT/BOOTX64.EFI";
        }
        else {
          filename "pxelinux/pxelinux.0";
        }
  }
  class "httpclients" {
    match if substring (option vendor-class-identifier, 0, 10) = "HTTPClient";
    option vendor-class-identifier "HTTPClient";
```

```
    filename "http://192.168.124.2/redhat/EFI/BOOT/BOOTX64.EFI";
  }
}
```

3. DHCPv4 サービスを起動します。

```
# systemctl enable --now dhcpd
```

7.3. ネットワークブート用の DHCPV6 サーバーの設定

サーバー上で DHCP バージョン 6 (DHCPv4) サービスを有効にし、ネットワークブート機能を提供できるようにします。

前提条件

- IPv6 プロトコルを介したネットワークインストールを準備中である。
  IPv4 の場合は、ネットワークブート用の DHCPv4 サーバーの設定 を参照してください。
- サーバーのネットワークアドレスがわかっている。
  以下の手順の例では、サーバーには次の設定のネットワークカードが搭載されています。

  IPv6 アドレス

  fd33:eb1b:9b36::2/64

  IPv6 ゲートウェイ

  fd33:eb1b:9b36::1

手順

1. DHCP サーバーをインストールします。

```
dnf install dhcp-server
```

2. DHCPv6 サーバーをセットアップします。**/etc/dhcp/dhcpd6.conf** ファイルに次の設定を入力します。アドレスはネットワークカードと一致するように置き換えます。

```
option dhcp6.bootfile-url code 59 = string;
option dhcp6.vendor-class code 16 = {integer 32, integer 16, string};

subnet6 fd33:eb1b:9b36::/64 {
        range6 fd33:eb1b:9b36::64 fd33:eb1b:9b36::c8;

        class "PXEClient" {
                match substring (option dhcp6.vendor-class, 6, 9);
        }

        subclass "PXEClient" "PXEClient" {
                option dhcp6.bootfile-url
"tftp://[fd33:eb1b:9b36::2]/redhat/EFI/BOOT/BOOTX64.EFI";
        }

        class "HTTPClient" {
                match substring (option dhcp6.vendor-class, 6, 10);
        }
```

```
        subclass "HTTPClient" "HTTPClient" {
                option dhcp6.bootfile-url
"http://[fd33:eb1b:9b36::2]/redhat/EFI/BOOT/BOOTX64.EFI";
                option dhcp6.vendor-class 0 10 "HTTPClient";
        }
}
```

3. DHCPv6 サービスを起動します。

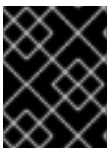
```
# systemctl enable --now dhcpd6
```

4. DHCPv6 パケットがファイアウォールの RP フィルターによって破棄されている場合は、そのログを確認してください。ログに **rpfilter_DROP** エントリーが含まれている場合は、**/etc/firewalld/firewalld.conf** ファイルで次の設定を使用してフィルターを無効にします。

```
IPv6_rpfilter=no
```

7.4. BIOS ベースのクライアント用に TFTP サーバーを設定する

BIOS ベースの AMD および Intel 64 ビットシステムでは、TFTP サーバーと DHCP サーバーを設定し、PXE サーバー上で TFTP サービスを起動する必要があります。



重要

本セクションのすべての設定ファイルは例となります。設定の詳細は、アーキテクチャーや特定の要件によって異なります。

手順

1. root で、次のパッケージをインストールします。

```
# dnf install tftp-server
```

2. ファイアウォールで、**tftp service** サービスへの着信接続を許可します。

```
# firewall-cmd --add-service=tftp
```

注記

- このコマンドは、次にサーバーを再起動するまで、一時的にアクセスを有効にします。永続的アクセスを有効にするには、コマンドに **--permanent** オプションを追加します。
- ISO インストールファイルの場所によっては、HTTP などのサービスの着信接続を許可しないといけない場合があります。

3. DVD ISO イメージファイルの **SYSLINUX** パッケージから **pxelinux.0** ファイルにアクセスします。ここで、**my_local_directory** は、作成するディレクトリーの名前です。

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro
```

```
# cp -pr /mount_point/AppStream/Packages/syslinux-tftpboot-version-architecture.rpm
/my_local_directory
```

```
# umount /mount_point
```

4. パッケージをデプロイメントします。

```
# rpm2cpio syslinux-tftpboot-version-architecture.rpm | cpio -dimv
```

5. **tftpboot**/ に **pxelinux**/ ディレクトリーを作成し、そのディレクトリーから **pxelinux**/ ディレクトリーにすべてのファイルをコピーします。

```
# mkdir /var/lib/tftpboot/pxelinux
```

```
# cp /my_local_directory/tftpboot/* /var/lib/tftpboot/pxelinux
```

6. **pxelinux**/ ディレクトリーに **pxelinux.cfg**/ ディレクトリーを作成します。

```
# mkdir /var/lib/tftpboot/pxelinux/pxelinux.cfg
```

7. **default** という名前の設定ファイルを作成し、以下の例のように **pxelinux.cfg**/ ディレクトリーに追加します。

```
default vesamenu.c32
prompt 1
timeout 600

display boot.msg

label linux
  menu label ^Install system
  menu default
  kernel images/RHEL-9/vmlinuz
  append initrd=images/RHEL-9/initrd.img ip=dhcp inst.repo=http://192.168.124.2/RHEL-
9/x86_64/iso-contents-root/
label vesa
  menu label Install system with ^basic video driver
  kernel images/RHEL-9/vmlinuz
  append initrd=images/RHEL-9/initrd.img ip=dhcp inst.xdriver=vesa nomodeset
inst.repo=http://192.168.124.2/RHEL-9/x86_64/iso-contents-root/
label rescue
  menu label ^Rescue installed system
  kernel images/RHEL-9/vmlinuz
  append initrd=images/RHEL-9/initrd.img inst.rescue
  inst.repo=http:///192.168.124.2/RHEL-8/x86_64/iso-contents-root/
label local
  menu label Boot from ^local drive
  localboot 0xffff
```

注記

- このランタイムイメージなしでは、インストールプログラムは起動できません。**inst.stage2** 起動オプションを使用して、イメージの場所を指定します。または、**inst.repo=** オプションを使用して、イメージおよびインストールソースを指定することも可能です。
- **inst.repo** で使用したインストールソースの場所には、有効な **treeinfo** ファイルが含まれている必要があります。
- インストールソースとして RHEL9 インストール DVD を選択すると、**.treeinfo** ファイルが BaseOS リポジトリーおよび AppStream リポジトリーを指定します。単一の **inst.repo** オプションを使用することで両方のリポジトリーを読み込むことができます。

8. **/var/lib/tftpboot/** ディレクトリーに、ブートイメージファイルを保存するサブディレクトリーを作成し、そのディレクトリーにブートイメージファイルをコピーします。この例のディレクトリーは、**/var/lib/tftpboot/pxelinux/images/RHEL-9/** になります。

```
# mkdir -p /var/lib/tftpboot/pxelinux/images/RHEL-9/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img}
/var/lib/tftpboot/pxelinux/images/RHEL-9/
```

9. **tftp.socket** サービスを開始して有効にします。

```
# systemctl enable --now tftp.socket
```

これにより、PXE 起動サーバーでは、PXE クライアントにサービスを提供する準備が整いました。クライアント (Red Hat Enterprise Linux のインストール先システム) を起動し、起動ソースを指定するように求められたら、**PXE ブート** を選択してネットワークインストールを開始できます。

7.5. UEFI ベースのクライアント用に TFTP サーバーを設定する

UEFI ベースの AMD64、Intel 64、および 64 ビット ARM システムでは、TFTP サーバーと DHCP サーバーを設定し、PXE サーバー上で TFTP サービスを起動する必要があります。

重要

- 本セクションのすべての設定ファイルは例となります。設定の詳細は、アーキテクチャーや特定の要件によって異なります。
- Red Hat Enterprise Linux 9 UEFI PXE ブートは、MAC ベースの grub メニューファイルの小文字のファイル形式に対応します。たとえば、grub2 の MAC アドレスのファイル形式は **grub.cfg-01-aa-bb-cc-dd-ee-ff** です。

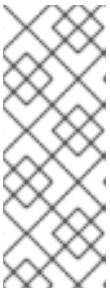
手順

1. root で、次のパッケージをインストールします。

```
# dnf install tftp-server
```

2. ファイアウォールで、**tftp service** サービスへの着信接続を許可します。

```
# firewall-cmd --add-service=tftp
```



注記

- このコマンドは、次にサーバーを再起動するまで、一時的にアクセスを有効にします。永続的アクセスを有効にするには、コマンドに **--permanent** オプションを追加します。
- ISO インストールファイルの場所によっては、HTTP などのサービスの着信接続を許可しないといけない場合があります。

3. DVD ISO イメージから EFI ブートイメージファイルにアクセスします。

```
# mount -t iso9660 /path_to_image/name_of_image.iso /mount_point -o loop,ro
```

4. DVD ISO イメージから EFI ブートイメージをコピーします。

```
# mkdir /var/lib/tftpboot/redhat
# cp -r /mount_point/EFI /var/lib/tftpboot/redhat/
# umount /mount_point
```

5. コピーしたファイルのパーミッションを修正します。

```
# chmod -R 755 /var/lib/tftpboot/redhat/
```

6. **/var/lib/tftpboot/redhat/EFI/BOOT/grub.cfg** の内容を次の例に置き換えます。

```
set timeout=60
menuentry 'RHEL 9' {
  linux images/RHEL-9/vmlinuz ip=dhcp inst.repo=http://192.168.124.2/RHEL-9/x86_64/iso-contents-root/
  initrd images/RHEL-9/initrd.img
}
```

注記

- このランタイムイメージなしでは、インストールプログラムは起動できません。**inst.stage2** 起動オプションを使用して、イメージの場所を指定します。または、**inst.repo=** オプションを使用して、イメージおよびインストールソースを指定することも可能です。
- **inst.repo** で使用したインストールソースの場所には、有効な **treeinfo** ファイルが含まれている必要があります。
- インストールソースとして RHEL9 インストール DVD を選択すると、**.treeinfo** ファイルが BaseOS リポジトリーおよび AppStream リポジトリーを指定します。単一の **inst.repo** オプションを使用することで両方のリポジトリーを読み込むことができます。

7. **/var/lib/tftpboot/** ディレクトリーに、ブートイメージファイルを保存するサブディレクトリーを作成し、そのディレクトリーにブートイメージファイルをコピーします。この例のディレクトリーは、**/var/lib/tftpboot/images/RHEL-9/** になります。

```
# mkdir -p /var/lib/tftpboot/images/RHEL-9/
# cp /path_to_x86_64_images/pxeboot/{vmlinuz,initrd.img}/var/lib/tftpboot/images/RHEL-9/
```

8. **tftp.socket** サービスを開始して有効にします。

```
# systemctl enable --now tftp.socket
```

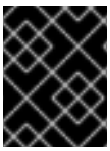
これにより、PXE 起動サーバーでは、PXE クライアントにサービスを提供する準備が整いました。クライアント (Red Hat Enterprise Linux のインストール先システム) を起動し、起動ソースを指定するように求められたら、**PXE ブート** を選択してネットワークインストールを開始できます。

関連情報

- Using the Shim Program

7.6. IBM POWER システム用のネットワークサーバーの設定

GRUB2 を使用して、IBM Power システム用のネットワークブートサーバーを設定できます。



重要

本セクションのすべての設定ファイルは例となります。設定の詳細は、アーキテクチャーや特定の要件によって異なります。

手順

1. root で、次のパッケージをインストールします。

```
# dnf install tftp-server dhcp-server
```

2. **tftp** サービスへの着信接続をファイアウォールで許可します。

```
# firewall-cmd --add-service=tftp
```

注記

- このコマンドは、次にサーバーを再起動するまで、一時的にアクセスを有効にします。永続的アクセスを有効にするには、コマンドに **--permanent** オプションを追加します。
- ISO インストールファイルの場所によっては、HTTP などのサービスの着信接続を許可しないといけない場合があります。

3. TFTP のルート内に GRUB2 ネットワーク起動ディレクトリーを作成します。

```
# grub2-mknetdir --net-directory=/var/lib/tftpboot
Netboot directory for powerpc-ieee1275 created. Configure your DHCP server to point to
/boot/grub2/powerpc-ieee1275/core.elf
```

注記

この手順で説明しているように、コマンドの出力は、DHCP 設定で設定する必要があるファイル名をユーザーに通知します。

a. PXE サーバーを x86 マシンで実行している場合は、tftp root に **GRUB2** ネットワーク起動ディレクトリーを作成する前に、**grub2-ppc64-modules** をインストールする必要があります。

```
# dnf install grub2-ppc64-modules
```

4. 以下の例のように、GRUB2 設定ファイル (**/var/lib/tftpboot/boot/grub2/grub.cfg**) を作成します。

```
set default=0
set timeout=5

echo -e "\nWelcome to the Red Hat Enterprise Linux 9 installer!\n\n"

menuentry 'Red Hat Enterprise Linux 9' {
  linux grub2-ppc64/vmlinuz ro ip=dhcp inst.repo=http://192.168.124.2/RHEL-9/x86_64/iso-contents-root/
  initrd grub2-ppc64/initrd.img
}
```

注記

- このランタイムイメージなしでは、インストールプログラムは起動できません。**inst.stage2** 起動オプションを使用して、イメージの場所を指定します。または、**inst.repo=** オプションを使用して、イメージおよびインストールソースを指定することも可能です。
- **inst.repo** で使用したインストールソースの場所には、有効な **treeinfo** ファイルが含まれている必要があります。
- インストールソースとして RHEL8 インストール DVD を選択すると、**.treeinfo** ファイルが BaseOS リポジトリーおよび AppStream リポジトリーを指定します。単一の **inst.repo** オプションを使用することで両方のリポジトリーを読み込むことができます。

5. このコマンドを使用して DVD ISO イメージをマウントします。

```
# mount -t iso9660 /path_to_image/name_of_iso/ /mount_point -o loop,ro
```

6. ディレクトリーを作成し、DVD ISO イメージから **initrd.img** ファイルおよび **vmlinuz** ファイルをコピーします。以下に例を示します。

```
# cp /mount_point/ppc/ppc64/{initrd.img,vmlinuz} /var/lib/tftpboot/grub2-ppc64/
```

7. 以下の例のように、**GRUB2** に同梱されているブートイメージを使用するように DHCP サーバーを設定します。DHCP サーバーがすでに設定されている場合は、DHCP サーバーでこの手順を実行します。

```
subnet 192.168.0.1 netmask 255.255.255.0 {
  allow bootp;
  option routers 192.168.0.5;
  group { #BOOTP POWER clients
    filename "boot/grub2/powerpc-ieee1275/core.elf";
    host client1 {
    hardware ethernet 01:23:45:67:89:ab;
    fixed-address 192.168.0.112;
    }
  }
}
```

8. ネットワーク設定に合わせて、サンプルパラメーターの **subnet**、**netmask**、**routers**、**fixed-address**、および **hardware ethernet** を変更します。**file name** パラメーターは、この手順のステップで、**grub2-mknetdir** コマンドで出力したファイル名となります。

9. DHCP サーバーで **dhcpd** サービスを開始して有効にします。localhost で DHCP サーバーを設定している場合は、ローカルホストで **dhcpd** サービスを開始して有効にします。

```
# systemctl enable --now dhcpd
```

10. **tftp.socket** サービスを開始して有効にします。

```
# systemctl enable --now tftp.socket
```

これにより、PXE 起動サーバーでは、PXE クライアントにサービスを提供する準備が整いました。クライアント (Red Hat Enterprise Linux のインストール先システム) を起動し、起動ソースを指定するように求められたら、**PXE ブート** を選択してネットワークインストールを開始できます。

第8章 RHEL ベータ版リリースをインストールおよび起動するために UEFI セキュアブートが有効なシステムを準備する

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。

8.1. 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。

8.2. 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。

8.3. 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。

第9章 64 ビット IBM Z での RHEL インストールの準備

次のセクションは、64 ビットの IBM Z アーキテクシャーに Red Hat Enterprise Linux をインストールする方法を説明します。

9.1. 64 ビット IBM Z へのインストールの計画

Red Hat Enterprise Linux 9 は、IBM z14 または IBM LinuxONE II システム以降で動作します。

IBM Z へのインストールプロセスでは、ユーザーが 64 ビットの IBM Z の操作に慣れていること、また **論理パーティション** (LPAR) および z/VM ゲスト仮想マシンをセットアップできることを前提としています。

Red Hat Enterprise Linux を 64 ビットの IBM Z にインストールする場合、Red Hat では、FCP (ファイバーチャネルプロトコル) で接続された DASD (Direct Access Storage Device) および SCSI デスクデバイス、**virtio-blk** デバイス、および **virtio-scsi** デバイスをサポートします。FCP デバイスを使用する場合、Red Hat は信頼性を高めるためにマルチパス設定で使用することを推奨します。

重要

DASD は、デバイスごとに最大 3 つのパーティションを許可するディスクです。たとえば、**dasda** には、**dasda1**、**dasda2**、および **dasda3** のパーティションを設定できます。

インストール前に決めること

- オペレーティングシステムを LPAR、KVM 上で稼働するか、z/VM ゲストのオペレーティングシステムとして稼働するか。
- swap 領域が必要かどうか、どのぐらいの大きさが必要か。z/VM のゲスト仮想マシンに十分なメモリーを割り当て、z/VM が必要なスワッピングを行えるようにすることが推奨されますが、必要な RAM サイズの予測が困難な場合もあります。このような場合にはケースバイケースで検討してください。
- ネットワーク設定。64 ビットの IBM Z 向けの Red Hat Enterprise Linux 9 は、以下のネットワークデバイスに対応しています。
  - 物理および仮想の **OSA (オープンシステムアダプター)**
  - 物理および仮想の HiperSockets
  - 物理 OSA 対応の **LCS (LAN チャネルステーション)**
  - **virtio-net** デバイス
- z/VM 仮想マシンのマシンタイプとして **ESA** を必ず選択してください。他のマシンタイプを選択すると、RHEL がインストールされなくなる可能性があります。IBM のドキュメント を参照してください。

ディスク領域

DASD または SCSI のディスクで十分なディスク容量を計算して割り当てる必要があります。

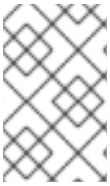
- サーバーのインストールには最低 10 GiB が必要です。すべてのパッケージをインストールする場合は 20 GiB が必要です。

- アプリケーションデータにもディスク領域が必要です。インストール後に、DASD パーティションまたは SCSI パーティションを追加または削除できます。
- 新規インストールの Red Hat Enterprise Linux システム (Linux インスタンス) で使用するディスク領域と、お使いのシステムにインストールされているその他の OS で使用されるディスク領域は、別にしておく必要があります。

RAM

システムに十分な RAM があることを確認します。

- NFS からインストールする場合は最低 1.5 GiB。
- HTTP または FTP のインストールソースからインストールする場合は、最小で 3 GiB。
- テキストモードでインストールする場合は、NFS インストールソースを使用している場合に限り 1 GiB で十分です。
- Red Hat は、インストールされている Linux インスタンス用に 2 GiB を推奨します。ただし、適切にチューニングされたシステムでは、1 GiB で十分です。



注記

- **SWAPGEN** ユーティリティーを使用して FBA (Fixed Block Architecture) DASD 上のスワップ領域を初期化する場合は、**FBAPART** オプションを使用する必要があります。

関連情報

- 64 ビット IBM Z の追加情報は、https://www.ibm.com/it-infrastructure/z を参照してください。
- Linux on IBM Z でセキュアブートを使用する方法の詳細は、Secure boot for Linux on IBM Z を参照してください。

9.2. 64 ビット IBM Z サーバーへのインストールプロセスの概要

Red Hat Enterprise Linux の 64 ビットの IBM Z へのインストールは、対話形式または無人モードで行うことが可能です。64 ビットの IBM Z へのインストールは通常、ローカルメディアからではなく、ネットワーク経由で行われるという点で他のアーキテクチャーと異なります。インストールは次の 3 つのフェーズで構成されます。

1. インストールの起動
   - メインフレームへの接続します。
   - ブートパラメーターのカスタマイズ
   - インストールプログラムを含むメディアから IPL (initial program load)、つまり起動を実行します。
2. インストールシステムへの接続
   - ローカルマシンから SSH でリモートの 64 ビットの IBM Z システムに接続し、Virtual Network Computing (VNC) を使用してインストールプログラムを起動します。
3. RHEL インストールプログラムを使用したインストールの完了

9.3. 64 ビット IBM Z サーバーに RHEL をインストールするためのブートメディア

メインフレームとの接続を確立したら、インストールプログラムを含むメディアから IPL (initial program load)、つまり起動を実行する必要があります。このドキュメントでは、64 ビットの IBM Z に Red Hat Enterprise Linux をインストールする最も一般的な方法を説明します。通常、どの方法も、ユーザー定義のパラメーターで補われる **generic.prm** ファイルのパラメーターと、カーネル (**kernel.img**) および初期 RAM ディスク (**initrd.img**) で構成される Linux インストールシステムを起動するために使用できます。また、initrd、カーネル、**generic.prm** のファイル名およびメモリーアドレスを判断するために、**generic.ins** ファイルがロードされます。

本書では、Linux インストールシステムを **インストールプログラム** とも呼びます。

IPL プロセスを開始できる制御ポイントは、Linux を実行する環境によって異なります。Linux が z/VM ゲストオペレーティングシステムとして実行される場合は、制御ポイントはホストしている z/VM の CP (**コントロールプログラム**) になります。Linux を LPAR モードで実行する場合は、メインフレームの **SE (サポートエレメント)** または接続されている 64 ビットの IBM Z の **HMC (ハードウェア管理コンソール)** が制御ポイントになります。

以下の起動メディアは、Linux を z/VM 環境でゲストのオペレーティングシステムとして実行する場合にのみ使用できます。

- z/VM リーダー

以下の起動用メディアは、Linux を LPAR モードで実行する場合にのみ使用できます。

- リモート FTP サーバー経由の SE または HMC
- SE または HMC DVD

以下の起動用メディアは、z/VM と LPAR の両方に使用できます。

- DASD
- FCP チャネルを介して接続している SCSI デスクデバイス

DASD または FCP 接続の SCSI ディスクデバイスをブートメディアとして使用する場合は、**zipl** ブートローダーを設定する必要があります。

9.4. ブートパラメーターのカスタマイズ

インストールを開始する前に、必須の起動パラメーターをいくつか設定する必要があります。z/VM でインストールする場合は、**generic.prm** ファイルで起動する前にこれらのパラメーターを設定する必要があります。LPAR にインストールする場合は、**rd.cmdline** パラメーターはデフォルトで **ask** するよう設定されています。つまり、これらのブートパラメーターを入力することができるプロンプトが表示されます。いずれの場合も、必須パラメーターは同じです。

注記

すべてのネットワーク設定は、パラメーターファイルまたはプロンプトで指定する必要があります。

インストールソース

インストールソースは常に設定される必要があります。

inst.repo オプションを使用して、インストール用のパッケージソースを指定します。

ネットワークデバイス

インストール中にネットワークアクセスが必要となる場合は、ネットワークを設定する必要があります。ディスクなどのローカルメディアのみを使用して無人 (キックスタートベース) インストールを行う場合は、ネットワーク設定を省略できます。

ip=

必要に応じて、基本的なネットワーク設定には **ip=** オプションなどのオプションを使用します。

rd.znet=

また、**rd.znet=** カーネルオプションも指定します。このオプションは、ネットワークプロトコルタイプ、コンマ区切りのサブチャネルリスト、およびオプションでコンマ区切りの **sysfs** パラメーターと値のペアを取ります。複数のネットワークデバイスをアクティベートするには、このパラメーターを複数回にわたり指定することができます。
以下に例を示します。

```
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1,portname=<name>
```

注記

複数の **rd.znet** ブートオプションを指定すると、最後のオプションだけがインストールされているシステムのカーネルコマンドラインに渡されます。インストール中に設定されたすべてのネットワークデバイスは、起動時に適切にアクティブ化および設定されるため、これはシステムのネットワークには影響しません。

qeth デバイスドライバーは、イーサネットデバイスと Hipersockets デバイスに同じインターフェイス名 (**enc<device number>**) を割り当てます。バス ID は、ドットで区切られたチャネルサブシステム ID、サブチャネルセット ID、およびデバイス番号で構成されます。デバイス番号は、先頭のゼロとドットを除いたバス ID の最後の部分です。たとえば、インターフェイス名は、バス ID が **0.0.0a00** のデバイスに対して **enca00** になります。

ストレージデバイス

テキストモードインストールには、少なくとも 1 つのストレージデバイスが常に設定される必要があります
rd.dasd= オプションは、DASD (Direct Access Storage Device) アダプターデバイスバス識別子を取ります。複数の DASD の場合は、パラメーターを複数回指定するか、バス ID のコンマ区切りリストを使用します。DASD の範囲を指定するには、最初と最後のバス ID を指定します。

以下に例を示します。

```
rd.dasd=0.0.0200 rd.dasd=0.0.0202(ro),0.0.0203(ro:failfast),0.0.0205-0.0.0207
```

rd.zfcp= オプションは、SCSI over FCP(zFCP) アダプターデバイスバス識別子、ターゲット World Wide Port Name (WWPN)、および FCP LUN を取得し、SCSI ディスクへの 1 つのパスをアクティブにします。同じディスクへの複数のパスをアクティブにするには、このパラメーターを少なくとも 2 回指定する必要があります。このパラメーターを複数回指定して、それぞれが複数のパスを持つ複数のディスクをアクティブ化できます。9 以降、ターゲットワールドワイドポート名 (WWPN) と FCP LUN は、**zFCP** デバイスが NPIV モードで設定されていない場合や、**zfcp.allow_lun_scan=0** カーネルモジュールパラメーターにより **auto LUN** スキャンが無効になっている場合のみ提供する

必要があります。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。同じディスクへの複数のパスをアクティブにするには、このパラメーターを少なくとも 2 回指定する必要があります。

```
rd.zfcp=0.0.4000,0x5005076300C213e9,0x5022000000000000
rd.zfcp=0.0.4000
```

Kickstart のオプション

Kickstart ファイルを使用して自動インストールを行う場合は、**inst.ks=** オプションで Kickstart ファイルの場所を常に指定している必要があります。無人の完全自動 Kickstart インストールの場合は、**inst.cmdline** オプションを指定すると便利です。

必須パラメーターすべてを含むカスタマイズした **generic.prm** ファイルの例を以下に示します。

例9.1 カスタマイズ generic.prm ファイル

```
ro ramdisk_size=40000 cio_ignore=all,!condev
inst.repo=http://example.com/path/to/repository
rd.znet=qeth,0.0.0600,0.0.0601,0.0.0602,layer2=1,portno=0,portname=foo
ip=192.168.17.115::192.168.17.254:24:foobar.systemz.example.com:enc600:none
nameserver=192.168.17.1
rd.dasd=0.0.0200 rd.dasd=0.0.0202
rd.zfcp=0.0.4000,0x5005076300c213e9,0x5022000000000000
rd.zfcp=0.0.5000,0x5005076300dab3e9,0x5022000000000000
inst.ks=http://example.com/path/to/kickstart
```

インストール方法によっては、HMC DVD または FTP サーバーのファイルシステムのインストールデータの場所のマッピングがあり、データがコピーされるメモリーの場所を持つファイルが必要です。

このファイルは、通常 **generic.ins** と名前が付けられ、初期 RAM ディスク、カーネルイメージ、パラメーターファイル (**generic.prm**) のファイル名と各ファイルのメモリーの場所が格納されています。**generic.ins** の例は、以下のサンプルのようになります。

例9.2 generic.ins サンプルファイル

```
images/kernel.img 0x00000000
images/initrd.img 0x02000000
images/genericdvd.prm 0x00010480
images/initrd.addrsize 0x00010408
```

有効な **generic.ins** ファイルは、インストーラーの起動に必要なその他すべてのファイルとともに Red Hat から提供されます。このファイルは、たとえば、デフォルト以外のカーネルバージョンをデフォルトからロードする場合にのみ変更します。

関連情報

- インストールソースの起動オプション

9.5. 64 ビット IBM Z のパラメーターおよび設定ファイル

このセクションでは、64 ビットの IBM Z のパラメーターおよび設定ファイルを説明します。

9.5.1. 64 ビット IBM Z で必要な設定ファイルパラメーター

いくつかのパラメーターは必須のパラメーターなので、必ずパラメーターファイルに追加してください。このパラメーターはインストール DVD の **images/** ディレクトリー内にある **generic.prm** ファイルでも提供されています。

- **ro**
  RAM ディスクであり、読み取り専用である root ファイルシステムをマウントします。

- **ramdisk_size=size**
  RAM ディスク用に予約されているメモリーサイズを、Red Hat Enterprise Linux インストールプログラムを格納できるサイズに修正します。たとえば、**ramdisk_size=40000** のようになります。

generic.prm ファイルには、追加のパラメーター **cio_ignore=all,!condev** も含まれます。この設定は、デバイスが多いシステムで、起動とデバイス検出を高速化します。インストールプログラムは、無視するデバイスのアクティベーションを透過的に処理します。

9.5.2. 64 ビットの IBM Z/VM 設定ファイル

z/VM では、CMS でフォーマットしたディスクの設定ファイルを使用できます。CMS 設定ファイルの目的は、パラメーターファイル内の領域を節約することにあります。これは、初期ネットワークや、DASD および FCP 仕様を設定するパラメーターを、パラメーターファイルから移動することにより実行します。

CMS 設定ファイルでは、1 つの変数が 1 行で表されます。 **variable=value** のようなシェルスタイルの構文で値が設定されます。

パラメーターファイルには、**CMSDASD** パラメーターおよび **CMSCONFFILE** のパラメーターも追加する必要があります。このパラメーターは、設定ファイルの場所をインストールプログラムに指定します。

CMSDASD=cmsdasd_address

cmsdasd_address は、設定ファイルを格納している CMS フォーマット済みディスクのデバイス番号です。一般的には、CMS ユーザーの **A** ディスクになります。
たとえば、**CMSDASD=191** となります。

CMSCONFFILE=configuration_file

configuration_file は、設定ファイル名になります。**この値は小文字で指定してください。CMS_file_name.CMS_file_type** などの Linux ファイル名の形式で指定します。
CMS ファイルの **REDHAT CONF** は **redhat.conf** として指定されます。CMS のファイル名およびファイルタイプは、それぞれ CMS 規則に従い 1 文字から 8 文字の長さにします。

たとえば、**CMSCONFFILE=redhat.conf** となります。

9.5.3. 64 ビット IBM Z でのインストールネットワーク、DASD および FCP パラメーター

このようなパラメーターは、準備段階のネットワークを自動的に設定するために使用され、CMS 設定ファイル内で定義できます。このパラメーターは、CMS 設定ファイルでも使用できるパラメーターのみに限定されます。その他のセクションで扱われるその他のパラメーターはすべて、パラメーターファ

イル内で指定する必要があります。

NETTYPE="type"

type は、**qeth**、**lcs**、**ctc** のいずれかにしてください。デフォルトは **qeth** です。
以下を使用する場合は **lcs** を選択します。

- OSA-Express 機能

以下を使用する場合は **qeth** を選択します。

- OSA-Express 機能
- HiperSockets
- z/VM 上の仮想接続 (VSWTICH、Guest LAN など)

SUBCHANNELS="device_bus_IDs"

device_bus_IDs は 、コンマで区切られた 2 つまたは 3 つのデバイスバス ID になります。ID は小文字で指定する必要があります。
各ネットワークインターフェイスに、それぞれ必要なデバイスバス ID を入力します。

```
qeth: SUBCHANNELS="read_device_bus_id,write_device_bus_id,data_device_bus_id"
lcs or ctc: SUBCHANNELS="read_device_bus_id,write_device_bus_id"
```

以下に例を示します (qeth SUBCHANNEL ステートメントの場合)。

```
SUBCHANNELS="0.0.f5f0,0.0.f5f1,0.0.f5f2"
```

PORTNAME="osa_portname" PORTNAME="lcs_portnumber"

この変数は、qdio モードまたは非 qdio モードで動作する OSA デバイスに対応します。
qdio モード (**NETTYPE="qeth"**) を使用する場合、qeth モードで動作している OSA デバイスで指定するポート名は **osa_portname** です。

非 qdio モード (**NETTYPE="lcs"**) を使用する場合は、**lcs_portnumber** を使用して、0 から 15 の整数で適切なポート番号を渡します。

PORTNO="portnumber"

CMS 設定ファイルに **PORTNO="0"** (ポート 0 を使用) または **PORTNO="1"** (各 CHPID にポートが 2 つある OSA 機能のポート 1 を使用) のいずれかを追加すると、モード入力が要求されなくなります。

LAYER2="value"

value は、**0** または **1** です。
レイヤー 3 モード (**NETTYPE="qeth"**) で OSA または HiperSocket を動作させる場合は、**LAYER2="0"** を使用します。レイヤー 2 モードの場合は、**LAYER2="1"** を使用します。z/VM 環境の仮想ネットワークデバイスの場合、この設定はデバイスを接続する GuestLAN または VSWITCH の定義と同じにしてください。

DHCP などのレイヤー 2 (Data Link Layer またはその MAC サブレイヤー) で動作するネットワークサービスを使用する場合は、レイヤー 2 モードを選択することが推奨されます。

OSA デバイス用の qeth デバイスドライバーのデフォルトがレイヤー 2 モードになります。以前のデフォルトであるレイヤー 3 モードを引き続き使用する場合は、**LAYER2="0"** を明示的に設定します。

VSWITCH="value"

value は、**0** または **1** です。
z/VM VSWITCH または GuestLAN に接続する場合は **VSWITCH="1"** を指定します。実際の OSA または実際の HiperSocket を直接接続して使用する場合は **VSWITCH="0"** を指定します (または何も指定しません)。

MACADDR="MAC_address"

LAYER2="1" と **VSWITCH="0"** を指定している場合は、このパラメーターを使用して MAC アドレスを指定することもできます。Linux では、小文字と 16 進数の組み合わせをコロンで区切った、6 つのオクテット形式が必要です (**MACADDR=62:a3:18:e7:bc:5f** など)。z/VM で使用される表記とは異なります。
LAYER2="1" と **VSWITCH="1"** を指定する場合は、**MACADDR** を指定しないでください。レイヤー 2 モードの場合は、z/VM により固有の MAC アドレスが仮想ネットワークデバイスに割り当てられます。

CTCPROT="value"

value は、**0**、**1**、または **3** です。
NETTYPE="ctc" の CTC プロトコルを指定します。デフォルトは **0** です。

HOSTNAME="string"

string は、新たにインストールした Linux インスタンスのホスト名です。

IPADDR="IP"

IP は、新しい Linux インスタンスの IP アドレスです。

NETMASK="netmask"

netmask はネットマスクです。
IPv4 の **CIDR (クラスレス相互ドメインルーティング)** で規定されているように、ネットマスクでは接頭辞の整数 (1 から 32) の構文に対応しています。たとえば、**255.255.255.0** の代わりに **24** を指定したり、**255.255.240.0** の代わりに **20** を指定できます。

GATEWAY="gw"

gw は、このネットワークデバイスのゲートウェイ IP アドレスです。

MTU="mtu"

mtu は、このネットワークデバイスの **Maximum Transmission Unit** (MTU) です。

DNS="server1:server2:additional_server_terms:serverN"

server1:server2:additional_server_terms:serverN は、コロンで区切った DNS サーバーのリストです。以下に例を示します。

```
DNS="10.1.2.3:10.3.2.1"
```

SEARCHDNS="domain1:domain2:additional_dns_terms:domainN"

domain1:domain2:additional_dns_terms:domainN は、コロンで区切った検索ドメインのリストです。以下に例を示します。

```
SEARCHDNS="subdomain.domain:domain"
```

SEARCHDNS= の指定が必要となるのは、**DNS=** パラメーターを使用する場合のみです。

DASD=

DASD または DASD の範囲を定義して、インストールを設定します。
インストールプログラムは、オプション属性である **ro**、**diag**、**erplog**、および **failfast** を持つ、コンマ区切りのデバイスバス ID のリスト、またはデバイスバス ID の範囲のリストをサポートします。必要に応じて、デバイス番号で先行するゼロを除くことでデバイスバス ID を短縮できます。いずれのオプション属性も、コロンで区切り、括弧で囲む必要があります。オプションの属性は、デバイスバス ID、またはデバイスバス ID の範囲の後に続きます。

サポートされている唯一のグローバルオプションは **autodetect** です。ここでは、存在しない DASD の仕様をサポートして、後で追加する DASD 用にカーネルデバイス名を確保するということは行いません。永続性のある DASD デバイス名 (例: **/dev/disk/by-path/name**) を使用して、後で透過的なディスクを追加できるようにします。**probeonly**、**nopav**、**nofcx** などの他のグローバルオプションは、インストールプログラムではサポートしていません。

システムにインストールする必要がある DASD だけを指定します。ここで指定した未フォーマットの DASD はすべて、インストールプログラムで後で確認してからフォーマットする必要があります。

インストール後に、root ファイルシステム、または **/boot** パーティションに必要ではないデータの DASD を追加します。

以下に例を示します。

```
DASD="eb1c,0.0.a000-0.0.a003,eb10-eb14(diag),0.0.ab1c(ro:diag)"
```

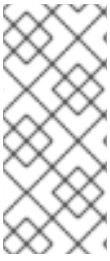
FCP_n="device_bus_ID [WWPN FCP_LUN]"

FCP のみの環境では、DASD が存在しないことを示すために、CMS 設定ファイルから **DASD=** オプションを削除します。

```
FCP_n="device_bus_ID [WWPN FCP_LUN]"
```

詳細は以下のようになります。

- 通常、**n** は整数値になりますが (**FCP_1**、**FCP_2** など)、アルファベット、数字、下線などを使用した文字列でも構いません。
- **device_bus_ID** は、**HBA** (ホストバスアダプター) (例: デバイス fc00 の場合は **0.0.fc00**) を表す FCP デバイスのデバイスバス ID を指定します。
- **WWPN** は、ルーティングに使用される世界共通のポート名です (マルチパスと併用されることが多い)。16 桁の 16 進数の値 (**0x50050763050b073d** など) になります。
- **FCP_LUN** は、ストレージの論理ユニット識別子を指し、16 桁の 16 進数の右側にゼロを加えた値 (**0x4020400100000000** など) で指定します。



注記

zfcp.allow_lun_scan=0 カーネルモジュールパラメーターにより auto LUN スキャンが無効になっているか、RHEL-9.0 以前のリリースをインストールする場合、**zFCP** デバイスが NPIV モードで設定されていないときは、ターゲットのワールドワイドポート名 (WWPN) および FCP_LUN を指定する必要があります。それ以外の場合は、**device_bus_ID** 値のみは必須です。

- この変数は、システムで、FCP デバイスとともに使用して、SCSI ディスクなどの FCP LUN をアクティベートできます。新たな FCP LUN はインストール中に対話式に、またはキックスタートファイルを介してアクティベートできます。サンプル値は以下のようになります。

```
FCP_1="0.0.fc00 0x50050763050b073d 0x4020400100000000"
FCP_2="0.0.4000"
```

重要

FCP パラメーターで使用する各値 (**FCP_1**、**FCP_2** など) はサイト固有となるため、通常は FCP ストレージ管理者から提供されます。

9.5.4. 64 ビット IBM Z へのキックスタートインストールのパラメーター

以下のパラメーターは、パラメーターファイル内で定義できますが、CMS 設定ファイル内では機能しません。

inst.ks=URL

キックスタートファイルを参照します。これは通常、64 ビットの IBM Z 上の Linux インストールのネットワークにあります。**URL** を、キックスタートファイルのファイル名を含む完全なパスに置き換えます。このパラメーターは、キックスタートによる自動インストールを有効にします。

inst.cmdline

インストールプログラムは、cmdline モード内での対話式のユーザー入力をサポートしないため、すべての質問に回答するキックスタートファイルによるインストールが必要になります。キックスタートファイルに必要なパラメーターがすべて含まれていることを確認してから、**inst.cmdline** オプションを使用してください。必要なコマンドがないと、インストールが失敗します。

9.5.5. 64 ビット IBM Z のその他のパラメーター

以下のパラメーターは、パラメーターファイル内で定義できますが、CMS 設定ファイル内では機能しません。

rd.live.check

ISO ベースのインストールソースのテストを起動します。たとえば、ローカルディスク上、または NFS でマウントした ISO で **inst.repo=** を使用する場合などにテストします。

inst.nompath

マルチパスデバイスのサポートを無効にします。

inst.proxy=[protocol://][username[:password]@]host[:port]

HTTP、HTTPS、または FTP を介したインストールで使用するプロキシーを指定します。

inst.rescue

RAM ディスクからレスキューシステムを起動して、インストールされたシステムを修正または復元できます。

inst.stage2=URL

install.img ディレクトリーではなく、**install.img** を含むツリーへのパスを指定します。それ以外は、**inst.repo=** の構文に従います。**inst.stage2** が指定されていると、それが **install.img** を検索する他の方法よりも優先されます。ただし、Anaconda が、ローカルメディア上で **install.img** を検出すると、**inst.stage2** の URL は無視されます。
stage2 が指定されておらず、**install.img** がローカルで見つからない場合、Anaconda は **inst.repo=** または **method=** で指定された場所を検索します。

inst.repo= や **method=** を使用せずに **stage2=** だけが指定されていると、Anaconda は、インストール用にデフォルトで有効にされているインストール済みシステムのリポジトリーを使用します。

複数の HTTP、HTTPS、または FTP ソースを指定する場合は、オプションを複数回使用します。複数の HTTP、HTTPS、または FTP のパスが指定されると、いずれかが成功するまで順番に試行されます。

```
inst.stage2=http://hostname/path_to_install_tree/
inst.stage2=http://hostname/path_to_install_tree/
inst.stage2=http://hostname/path_to_install_tree/
```

inst.syslog=IP/hostname[:port]

ログメッセージをリモートの syslog サーバーに送信します。

ここで説明されているブートパラメーターは、64 ビットの IBM Z へのインストールとトラブルシューティングに非常に便利ですが、インストールプログラムに影響を及ぼすのはこれらのサブセットのみです。

9.5.6. 64 ビット IBM Z のサンプルパラメーターファイルおよび CMS 設定ファイル

パラメーターファイルを変更する場合は、配布されている **generic.prm** ファイルの拡張から始めてください。

generic.prm ファイルの例:

```
ro ramdisk_size=40000 cio_ignore=all,!condev
CMSDASD="191" CMSCONFFILE="redhat.conf"
inst.vnc
inst.repo=http://example.com/path/to/dvd-contents
```

QETH ネットワークデバイスを設定する **redhat.conf** ファイルの例 (**generic.prm** 内の **CMSCONFFILE** により指定されています)

```
NETTYPE="qeth"
SUBCHANNELS="0.0.0600,0.0.0601,0.0.0602"
PORTNAME="FOOBAR"
PORTNO="0"
LAYER2="1"
MACADDR="02:00:be:3a:01:f3"
HOSTNAME="foobar.systemz.example.com"
IPADDR="192.168.17.115"
NETMASK="255.255.255.0"
GATEWAY="192.168.17.254"
DNS="192.168.17.1"
SEARCHDNS="systemz.example.com:example.com"
DASD="200-203"
```

9.5.7. 64 ビット IBM Z でのパラメーターおよび設定ファイルの使用

64 ビットの IBM Z アーキテクチャーでは、カスタマイズされたパラメーターファイルを使用して、カーネルとインストールプログラムに起動パラメーターを渡すことができます。

次を行う場合は、パラメーターを変更する必要があります。

- キックスタートによる無人インストール
- レスキューモードなど、インストールプログラムの対話式ユーザーインターフェイスからはアクセスできない、デフォルト以外のインストール設定を選択します。

パラメーターファイルは、インストールプログラム (**Anaconda**) の開始前に、非対話式にネットワークを設定するために使用できます。

カーネルパラメーターファイルは、895 文字に行末文字を加えた長さに制限されています。パラメーターファイルには、可変長または固定長のレコードフォーマットのいずれかが使用されます。固定長レコードフォーマットは、レコードの長さまで各行を追加してファイルサイズを増やします。インストールプログラムが LPAR 環境内のすべての指定パラメーターを認識しないという問題が生じた場合は、すべてのパラメーターを 1 行に収めるか、各行を空白文字で開始および終了することを試してください。

パラメーターファイルには、**ro** のような カーネルパラメーターと、**vncpassword=test** や **vnc** などのインストールプロセス用のパラメーターが含まれます。

9.6. Z/VM ゲスト仮想マシンへのインストールの準備

端末エミュレーター **x3270** または **c3270** を使用して、その他の Linux システムから z/VM にログインしたり、64 ビットの IBM Z Hardware Management Console (HMC) で IBM 3270 端末エミュレーターを使用します。Microsoft Windows オペレーティングシステムを実行している場合は、インターネットの検索で確認できる複数のオプションが利用できます。**wc3270** と呼ばれる、無料でネイティブの Windows ポート **c3270** もあります。

/VM 仮想マシンのマシンタイプとして **ESA** を必ず選択してください。他のマシンタイプを選択すると、RHEL がインストールされなくなる可能性があります。IBM のドキュメント を参照してください。

手順

1. Linux インストールに選択した z/VM ゲストの仮想マシンにログオンします。

注記

使用中の 3270 接続が割り込みを受け、それまでのセッションがまだアクティブなために再ログインができない場合は、z/VM ログイン画面で以下のコマンドを入力すると、それまでのセッションを新規のセッションに置き換えることができます。

```
logon user here
```

user には z/VM ゲスト仮想マシンの名前を入れてください。RACF などの外部セキュリティーマネージャーが使用されているかどうかによって、ログオンコマンドが異なる場合があります。

2. ゲスト内で **CMS** (z/VM 同梱のシングルユーザー用オペレーティングシステム) を実行していない場合は、以下のコマンドを実行してここで起動します。

```
cp ipl cms
```

3. インストールターゲットには、A ディスク (多くの場合デバイス番号は 0191) などの CMS ディスクを使用しないようにしてください。CMS で使用されているディスクを確認するには、以下のクエリーを使用します。

```
query disk
```

4. 以下の CP (z/VM ハイパーバイザーである z/VM 制御プログラム) の query コマンドを使用すると、z/VM ゲスト仮想マシンのデバイス設定を確認できます。

   a. 利用できるメインメモリーをクエリーします。64 ビットの IBM Z の用語では **ストレージ** と呼ばれています。ゲストには少なくとも 1 GiB のメインメモリーが必要です。

   ```
   cp query virtual storage
   ```

   b. 利用できるネットワークデバイスを以下のタイプ別にクエリーします。

   osa

   OSA - CHPID タイプ OSD、物理または仮想 (VSWITCH または GuestLAN)、いずれも QDIO モード

   hsi

   HiperSockets - CHPID タイプ IQD、物理または仮想 (GuestLAN タイプ Hipers)

   lcs

   LCS - CHPID タイプ OSE

   たとえば、上記のネットワークデバイスタイプをすべて問い合わせる場合は、次を実行します。

   ```
   cp query virtual osa
   ```

   c. 利用できる DASD をクエリーします。インストールターゲットとして使用できるのは、**RW** のフラグが付いた読み書きモードの DASD のみです。

   ```
   cp query virtual dasd
   ```

   d. 使用可能な FCP デバイス (vHBA) のクエリー:

   ```
   cp query virtual fcp
   ```

パート II. RHEL の完全自動および半自動インストール

第10章 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。キックスタートファイルの作成 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。UEFI HTTP または PXE インストールソースへのキックスタートファイルの追加 または RHEL インストーラーへのキックスタートファイルの提供 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。

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。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 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。キックスタートインストールの開始 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11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。

11.1. 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。
6. **pykickstart** パッケージをインストールします。

```
# dnf install pykickstart
```

7. キックスタートファイルに **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。

11.2. 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、インストールメディアからの RHEL の対話型インストール 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。

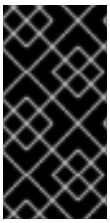
5. **pykickstart** パッケージをインストールします。

```
# dnf install pykickstart
```

6. キックスタートファイルに **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。

11.3. 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 キックスタートファイルを 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 キックスタートファイルを 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。

手順

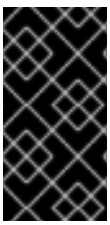
- キックスタートファイルを準備したら、**pykickstart** パッケージをインストールします。

```
# dnf install pykickstart
```

- キックスタートファイルに **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。

11.4. 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12章 UEFI HTTP または PXE インストールソースへのキックスタートファイルの追加

キックスタートファイルの準備ができたら、それをインストール先システムへのインストールに使用できるようになります。

12.1. 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12.1 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

12.2. NFS サーバー上でのインストールファイルの共有

キックスタートスクリプトファイルを NFS サーバーに保存できます。NFS サーバーに保存すると、キックスタートファイル用の物理メディアを使用しなくても、単一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。詳細は、ネットワークインストール用のポート を参照してください。

重要

必ず **inst.ks** と **inst.repo** で異なるパスを使用してください。NFS を使用してキックスタートをホストする場合、同じ NFS 共有を使用してインストールソースをホストすることはできません。

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
```

/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 *
```

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

- VNC を使用したリモートインストールの準備

12.3. HTTP または HTTPS サーバー上でのインストールファイルの共有

キックスタートスクリプトファイルを HTTP または HTTPS サーバーに保存できます。キックスタートファイルを HTTP または HTTPS サーバーに保存すると、キックスタートファイル用の物理メディアを使用しなくても、単一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。詳細は、ネットワークインストール用のポート を参照してください。

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。

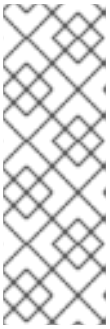
重要

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

12.4. FTP サーバー上でのインストールファイルの共有

キックスタートスクリプトファイルを FTP サーバーに保存できます。スクリプトを FTP サーバーに保存すると、キックスタートファイル用の物理メディアを使用しなくても、単一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。詳細は、ネットワークインストール用のポート を参照してください。

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: 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
   ```

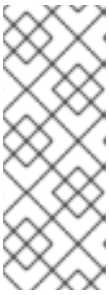
   /etc/vsftpd/vsftpd.conf ファイルを変更する前から、このサービスがすでに実行されていた場合は、サービスを再起動して必ず編集後のファイルを読み込ませてください。

   ```
   # 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。

第13章 半自動インストール: RHEL インストーラーへのキックスタートファイルの提供

キックスタートファイルの準備ができたら、それをインストール先システムへのインストールに使用できるようになります。

13.1. ローカルボリューム上でのインストールファイルの共有

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。

13.2. 自動ロードのためにローカルボリューム上でインストールファイルを共有する

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によってファイルが自動的にロードされます。

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14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。

- 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。

Red Hat コンテンツ配信ネットワーク (CDN) を使用すると、RHEL を登録できます。CDN は地理的に分散された一連の Web サーバーです。これらのサーバーは、たとえば、有効なサブスクリプションを持つ RHEL ホストにパッケージや更新を提供します。

インストール中に、CDN から RHEL を登録してインストールすると、次のような利点があります。

- インストール後すぐに最新のシステムで最新のパッケージを利用できます。
- Red Hat Insights に接続し、システムの目的を有効にするための統合サポートを利用できます。

14.1. 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考資料として提供されています。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。

関連情報

- PXE サーバーの設定については、PXE インストールソースの準備 を参照してください。

14.2. 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。

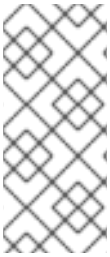
前提条件

- 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の設定と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。UEFI セキュアブートおよび Red Hat Enterprise Linux ベータ版リリースの詳細は、UEFI セキュアブートとベータ版リリースの要件 を参照してください。

14.3. IBM Z でのインストールを起動して LPAR に RHEL をインストールする

14.3.1. FTP サーバーから RHEL インストールを起動して IBM Z LPAR にインストールする

FTP サーバーを使用して Red Hat Enterprise Linux を LPAR にインストールする場合は、この手順に従います。

手順

1. IBM Z Hardware Management Console (HMC)または Support Element (SE)で、LPAR に新しいオペレーティングシステムをインストールするのに十分な権限を持つユーザーにログインします。**SYSPROG** ユーザーが推奨されます。
2. **Systems** タブで、作業するメインフレームを選択してから、**Partitions** タブで、インストールする LPAR を選択します。
3. 画面下部の **Daily**、オペレーティングシステムのメッセージを **確認します**。**Operating System Messages** をダブルクリックして、Linux の起動メッセージが表示されるテキストコンソールを表示します。
4. **Load from Removable Media or Server**をダブルクリックします。
5. 続いて表示されるダイアログボックスで、**FTP Server** を選択し、以下の情報を入力します。
   - **Host Computer** - インストール元となる FTP サーバーのホスト名または IP アドレス (**ftp.redhat.com** など) です。
   - **User ID** - FTP サーバーのユーザー名または、anonymous を指定します。
   - **Password** - パスワード匿名でログインする場合は、メールアドレスを使用します。
   - Red Hat Enterprise Linux for IBM Z を保持する FTP サーバーの **ファイルロケーション（オプション）です(** 例： **/rhel/s390x/**)。
6. **Continue** をクリックします。
7. 続いて表示されるダイアログボックスで、**generic.ins** のデフォルト選択はそのままにして、**Continue** をクリックします。

14.3.2. 設定済み DASD から RHEL インストールを起動し、IBM Z LPAR にインストールする

設定しておいた DASD を使用して、Red Hat Enterprise Linux を LPAR にインストールする場合は、この手順に従います。

手順

1. IBM Z Hardware Management Console (HMC)または Support Element (SE)で、LPAR に新しいオペレーティングシステムをインストールするのに十分な権限を持つユーザーにログインします。**SYSPROG** ユーザーが推奨されます。
2. **Systems** タブで、作業するメインフレームを選択してから、**Partitions** タブで、インストールする LPAR を選択します。
3. 画面下部の **Daily**、オペレーティングシステムのメッセージを **確認します**。**Operating System Messages** をダブルクリックして、Linux の起動メッセージが表示されるテキストコンソールを表示します。
4. **Load** をダブルクリックします。
5. 続いて表示されるダイアログボックスの **Load type** で **Normal** を選択します。
6. Load **address** に、DASD のデバイス番号を入力します。
7. **Load parameter** に、Red Hat Enterprise Linux インストールプログラムを起動するために準備した **zipl** 起動メニューエントリーに対応する数字を入力します。
8. **OK** ボタンをクリックします。

14.3.3. FCP 接続の SCSI ディスクから RHEL インストールの起動による IBM Z LPAR へのインストール

設定済み FCP を接続した SCSI ディスクを使用して、Red Hat Enterprise Linux を LPAR にインストールする場合は、この手順に従います。

手順

1. IBM Z Hardware Management Console (HMC)または Support Element (SE)で、LPAR に新しいオペレーティングシステムをインストールするのに十分な権限を持つユーザーにログインします。**SYSPROG** ユーザーが推奨されます。
2. **Systems** タブで、作業するメインフレームを選択してから、**Partitions** タブで、インストールする LPAR を選択します。
3. 画面下部の **Daily**、オペレーティングシステムのメッセージを **確認します**。**Operating System Messages** をダブルクリックして、Linux の起動メッセージが表示されるテキストコンソールを表示します。
4. **Load** をダブルクリックします。
5. 続いて表示されるダイアログボックスの **Load type**で **SCSI** を選択します。
6. **Load address** には、SCSI ディスクに接続している FCP チャネルのデバイス番号を入力します。
7. **World wide port name** には、ディスクを含むストレージシステムの WWPN を、16 桁の 16 進数で入力します。

8. **Logical unit number** には、ディスクの LUN を、16 桁の 16 進数で入力します。
9. **Boot program selector** には、Red Hat Enterprise Linux インストールプログラムを起動するために準備した **zipl** 起動メニューのエントリーに対応する番号を入力します。
10. **Boot record logical block address** は 0 のままにしておきます。また、**Operating system specific load parameters** は空のままにしておきます。
11. **OK** ボタンをクリックします。

14.4. IBM Z でインストールを起動して Z/VM に RHEL をインストールする

z/VM 環境にインストールする場合は、以下から起動できます。

- z/VM 仮想リーダー
- DASD または FCP 接続の SCSI ディスク (**zipl** ブートローダーを設定済み)

14.4.1. z/VM リーダーを使用した RHEL インストールの起動

以下の手順に従って z/VM リーダーから起動します。

手順

1. 必要に応じて、z/VM の TCP/IP ツールを含むデバイスを CMS ディスクのリストに追加します。以下に例を示します。

```
cp link tcpmaint 592 592
acc 592 fm
```

   fm を **FILEMODE** 文字で置き換えます。

2. コマンドを実行します。

```
ftp host
```

   host は、ブートイメージ (**kernel.img** および **initrd.img**) をホストする FTP サーバーのホスト名または IP アドレスです。

3. ログインして以下のコマンドを実行します。既存の **kernel.img** ファイル、**initrd.img** ファイル、**generic.prm** ファイル、または **redhat.exec** ファイルを上書きしている場合は、**(repl** オプションを使用します。

```
cd /location/of/install-tree/images/
ascii
get generic.prm (repl
get redhat.exec (repl
locsite fix 80
binary
get kernel.img (repl
get initrd.img (repl
quit
```

4. オプション: CMS コマンド **filelist** を使用して、受信したファイルとその形式を表示し、ファイ

ルが正しく転送されたかどうかを確認します。**kernel.img** と **initrd.img** では、Format 列の固定レコード長の形式が F と示され、Lrecl 列のレコード長が 80 であることが重要です。以下に例を示します。

```
VMUSER FILELIST A0 V 169 Trunc=169 Size=6 Line=1 Col=1 Alt=0
Cmd Filename Filetype Fm Format Lrecl Records Blocks Date Time
REDHAT EXEC  B1 V 22 1  1 4/15/10 9:30:40
GENERIC PRM  B1 V 44 1 1 4/15/10 9:30:32
INITRD IMG  B1 F 80 118545 2316 4/15/10 9:30:25
KERNEL IMG  B1 F 80 74541 912 4/15/10 9:30:17
```

PF3 を押して filelist を終了し、CMS プロンプトに戻ります。

5. 必要に応じて、**generic.prm** 内の起動パラメーターをカスタマイズします。詳細は、ブートパラメーターのカスタマイズ を参照してください。
CMS 設定ファイルを使用して、ストレージデバイスおよびネットワークデバイスを設定する方法もあります。そのような場合は、**CMSDASD=** パラメーターおよび **CMSCONFFILE=** パラメーターを **generic.prm** に追加します。詳細は、IBM Z/VM 設定ファイル を参照してください。

6. 最後に、REXX スクリプト redhat.exec を実行してインストールプログラムを起動します。

```
redhat
```

14.4.2. 設定済み DASD を使用した RHEL インストールの起動

設定済み DASD を使用するには、以下の手順を実行します。

手順

- 準備済みの DASD から起動して、Red Hat Enterprise Linux インストールプログラムを参照する **zipl** ブートメニューエントリーを選択します。コマンドを次の形式で使用します。

```
cp ipl DASD_device_number loadparm boot_entry_number
```

DASD_device_number を、起動デバイスのデバイス番号に置き換え、**boot_entry_number** を、このデバイスの **zipl** 設定メニューに置き換えます。以下に例を示します。

```
cp ipl eb1c loadparm 0
```

14.4.3. 設定済み FCP を接続した SCSI ディスクを使用した RHEL インストールの起動

設定済み FCP を接続した SCSI ディスクから起動するには、以下の手順を実行します。

手順

1. FCP ストレージエリアネットワーク内に準備した SCSI ディスクにアクセスできるように z/VM の SCSI ブートローダーを設定します。Red Hat Enterprise Linux インストールプログラムを参照する設定済み **zipl** ブートメニューエントリーを選択します。コマンドを次の形式で使用します。

```
cp set loaddev portname WWPN lun LUN bootprog boot_entry_number
```

WWPN を、ストレージシステムのワールドワイドポート名に置き換え、**LUN** を、ディスクの論理ユニット番号に置き換えます。16 桁の 16 進数は、それぞれ 8 桁の 2 つのペアに分割する必要があります。以下に例を示します。

```
cp set loaddev portname 50050763 050b073d lun 40204011 00000000 bootprog 0
```

2. オプション: 次のコマンドで設定を確認します。

```
query loaddev
```

3. 以下のコマンドを使用して、ディスクを含むストレージシステムに接続している FCP デバイスを起動します。

```
cp ipl FCP_device
```

以下に例を示します。

```
cp ipl fc00
```

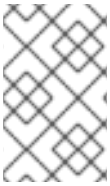
14.5. インストール中のコンソールとロギング

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14.1 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。

パート III. インストール後のタスク

第15章 SUBSCRIPTION MANAGER を使用した RHEL の登録

インストール後、継続的に更新を取得するために、システムを登録する必要があります。

15.1. インストーラー GUI を使用した RHEL 9 の登録

RHEL インストーラーの GUI を使用して Red Hat Enterprise Linux 9 を登録できます。

前提条件

- Red Hat カスタマーポータルに有効なユーザーアカウントがある。Create a Red Hat Login ページを参照してください。
- 有効なアクティベーションキーと組織 ID を持っている。

手順

1. **Installation Summary** 画面の **Software** で、**Connect to Red Hat** をクリックします。
2. **Account** または **Activation Key** オプションを使用して、Red Hat アカウントを認証します。
3. オプション: **Set System Purpose** フィールドで、設定する **Role**、**SLA**、および **Usage** 属性をドロップダウンメニューから選択します。
   この時点で、Red Hat Enterprise Linux 9 システムは正常に登録されています。

15.2. REGISTRATION ASSISTANT

Registration Assistant は、お使いの Red Hat Enterprise Linux 環境に最適な登録オプションの選択をサポートします。

関連情報

- ユーザー名とパスワードを使用して RHEL を Subscription Manager クライアントに登録する方法は、カスタマーポータルの RHEL 登録アシスタント を参照してください。
- RHEL システムを Red Hat Insights に登録する方法は、Hybrid Cloud Console の Insights 登録アシスタント を参照してください。

15.3. コマンドラインを使用したシステムの登録

コマンドラインを使用して、Red Hat Enterprise Linux 9 サブスクリプションを登録できます。

注記

ホストを Red Hat に登録するエクスペリエンスを改善および簡素化するには、リモートホスト設定 (RHC) を使用します。RHC クライアントはシステムを Red Hat に登録し、Insights のデータ収集に備えてシステムの準備を完了し、Insights for Red Hat Enterprise Linux から直接問題を修復できるようにします。詳細は、RHC の登録 を参照してください。

前提条件

- アクティブで、評価版ではない Red Hat Enterprise Linux サブスクリプションを持っている。

- Red Hat のサブスクリプションステータスを確認している。
- Red Hat Enterprise Linux 9 サブスクリプションを受け取ったことがない。
- Red Hat Enterprise Linux 9 システムを正常にインストールし、root としてログインしている。

手順

1. root ユーザーとしてターミナルウィンドウを開きます。
2. アクティベーションキーを使用して Red Hat Enterprise Linux システムを登録します。

```
# subscription-manager register --activationkey=<activation_key_name> --
org=<organization_ID>
```

システムが正常に登録されると、次の例のような出力が表示されます。

```
The system has been registered with id:
62edc0f8-855b-4184-b1b8-72a9dc793b96
```

関連情報

- アクティベーションキーを使用した Red Hat Subscription Manager へのシステムの登録
- RHEL システム登録のスタートガイド

第16章 SUBSCRIPTION-MANAGER コマンドラインツールを使用したシステムの目的の設定

システムの目的は、Red Hat Enterprise Linux インストールの機能の 1 つです。この機能は、Red Hat Hybrid Cloud Console で提供されるサブスクリプションエクスペリエンスとサービスのメリットを RHEL のお客様に提供するためのものです。Red Hat Hybrid Cloud Console は、ダッシュボードベースの Software-as-a-Service (SaaS) アプリケーションであり、これを使用すると、Red Hat アカウントのサブスクリプション使用状況を表示できます。

システム目的属性は、アクティベーションキーまたはサブスクリプションマネージャーツールを使用して設定できます。インストールプロセスでシステムの目的を設定しなかった場合は、インストール後に **subscription-manager syspurpose** コマンドラインツールを使用して必要な属性を設定できます。

前提条件

- Red Hat Enterprise Linux 9 システムをインストールして登録しているが、システムの目的が設定されていない。
- **root** ユーザーとしてログインしている。

注記

エンタイトルメントモードでは、システムが登録されているものの、必要な目的を満たさないサブスクリプションがある場合、**subscription-manager Remove --all** コマンドを実行して、割り当てたサブスクリプションを削除できます。次に、コマンドラインの subscription-manager syspurpose {ロール、使用条件、サービスレベル} ツールを使用して必要な目的属性を設定し、最後に **subscription-manager attach --auto** を実行して、更新した属性を考慮してシステムを再登録できます。一方、SCA が有効なアカウントでは、システム内のサブスクリプションを更新せずに、登録後にシステムの目的の詳細を直接更新できます。

手順

1. 端末で、次のコマンドを実行して、システムの目的のロールを設定します。

```
# subscription-manager syspurpose role --set "VALUE"
```

VALUE を、割り当てるロールに置き換えます。

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

以下に例を示します。

```
# subscription-manager syspurpose role --set "Red Hat Enterprise Linux Server"
```

a. オプション: 値を設定する前に、組織のサブスクリプションがサポートする利用可能なロールを確認します。

```
# subscription-manager syspurpose role --list
```

b. 必要に応じて、次のコマンドを実行してロールの設定を解除します。

```
# subscription-manager syspurpose role --unset
```

2. 次のコマンドを実行して、希望するシステムのサービスレベルアグリーメント (SLA) を設定します。

```
# subscription-manager syspurpose service-level --set "VALUE"
```

VALUE を、割り当てる SLA に置き換えます。

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

以下に例を示します。

```
# subscription-manager syspurpose service-level --set "Standard"
```

a. オプション: 値を設定する前に、組織のサブスクリプションがサポートする利用可能なサービスレベルを確認します。

```
# subscription-manager syspurpose service-level --list
```

b. 必要に応じて、次のコマンドを実行して SLA の設定を解除します。

```
# subscription-manager syspurpose service-level --unset
```

3. 次のコマンドを実行して、希望する使用方法をシステムに設定します。

```
# subscription-manager syspurpose usage --set "VALUE"
```

VALUE を、割り当てる使用方法に置き換えます。

- **Production**
- **Disaster Recovery**
- **Development/Test**

以下に例を示します。

```
# subscription-manager syspurpose usage --set "Production"
```

a. オプション: 値を設定する前に、組織のサブスクリプションがサポートする利用可能な使用条件を確認します。

```
# subscription-manager syspurpose usage --list
```

b. 必要に応じて、次のコマンドを実行して、使用方法の設定を解除します。

```
# subscription-manager syspurpose usage --unset
```

4. 次のコマンドを実行して、現在のシステム目的のプロパティーを表示します。

```
# subscription-manager syspurpose --show
```

   a. オプション: 詳細な構文情報については、以下のコマンドを実行して **subscription-manager** の man ページにアクセスし、SYSPURPOSE OPTIONS を参照します。

```
# man subscription-manager
```

検証

- エンタイトルメントモードが有効になっているアカウントを使用して登録したシステムで、システムのサブスクリプションステータスを確認するには、次の手順を実行します。

```
# subscription-manager status
+-------------------------------------------+
   System Status Details
+-------------------------------------------+
Overall Status: Current

System Purpose Status: Matched
```

  - 全体的なステータス **Current** とは、インストールされている製品がすべて割り当てられたサブスクリプションの対象となり、コンテンツセットリポジトリーにアクセスするためのエンタイトルメントが付与されています。

  - システム目的のステータス **Matched** とは、システムに設定したすべてのシステム目的の属性 (ロール、使用条件、サービスレベル) が、割り当てられたサブスクリプションによって満たされることを意味します。

  - ステータス情報が理想的ではない場合、システム管理者がインストール済みの製品と目的のシステムの目的に対応するために、アタッチされているサブスクリプションに加える修正を決定するのに役立つ追加情報が表示されます。

- SCA モードが有効になっているアカウントを使用して登録したシステムで、システムのサブスクリプションステータスを確認するには、次の手順を実行します。

```
# subscription-manager status
+-------------------------------------------+
   System Status Details
+-------------------------------------------+
Overall Status: Disabled
Content Access Mode is set to Simple Content Access. This host has access to content,
regardless of subscription status.
System Purpose Status: Disabled
```

  - SCA モードでは、サブスクリプションを個々のシステムに割り当てる必要はありません。したがって、全体的なステータスとシステムの目的のステータスの両方が Disabled として表示されます。ただし、システム目的属性によって提供される技術、ビジネス、および運用のユースケースは、サブスクリプションサービスには重要です。これらの属性がないと、サブスクリプションサービスデータの精度が低下します。

関連情報

- サブスクリプションサービスの詳細は、サブスクリプションサービスの使用ガイド を参照してください。

第17章 64 ビット IBM Z で LINUX インスタンスの設定

本セクションでは、64 ビットの IBM Z に Red Hat Enterprise Linux をインストールするための一般的なタスクを説明します (すべてのタスクが記載されているわけではありません)。

17.1. DASD の追加

DASR (Direct Access Storage Devices) は、64 ビットの IBM Z で一般的に使用されるストレージの一種です。詳細は、IBM Knowledge Center のWorking with DASDsを参照してください。次の例では、DASD をオンラインに設定してフォーマットし、変更を永続化します。

z/VM 環境下で実行する場合は、デバイスが Linux システムに接続またはリンクされていることを確認してください。

```
CP ATTACH EB1C TO *
```

アクセスできるミニディスクをリンクするには、次のコマンドを実行します。

```
CP LINK RHEL7X 4B2E 4B2E MR
DASD 4B2E LINKED R/W
```

17.2. DASD のオンラインへの動的な設定

本セクションでは、DASD をオンラインに設定する方法を説明します。

手順

1. **cio_ignore** ユーティリティーを使用して、無視されるデバイスのリストから DASD を削除して、Linux から見えるようにします。

   ```
   # cio_ignore -r device_number
   ```

   device_number を、DASD のデバイス番号に置き換えます。以下に例を示します。

   ```
   # cio_ignore -r 4b2e
   ```

2. デバイスをオンラインに設定します。コマンドを次の形式で使用します。

   ```
   # chccwdev -e device_number
   ```

   device_number を、DASD のデバイス番号に置き換えます。以下に例を示します。

   ```
   # chccwdev -e 4b2e
   ```

   または、sysfs 属性を使用してデバイスをオンラインに設定できます。

   a. **cd** コマンドで、そのボリュームを示す /sys/ ディレクトリーに変更します。

      ```
      # cd /sys/bus/ccw/drivers/dasd-eckd/0.0.4b2e/
      # ls -l
      total 0
      -r--r--r--  1 root root 4096 Aug 25 17:04 availability
      ```

```
-rw-r--r--  1 root root 4096 Aug 25 17:04 cmb_enable
-r--r--r--  1 root root 4096 Aug 25 17:04 cutype
-rw-r--r--  1 root root 4096 Aug 25 17:04 detach_state
-r--r--r--  1 root root 4096 Aug 25 17:04 devtype
-r--r--r--  1 root root 4096 Aug 25 17:04 discipline
-rw-r--r--  1 root root 4096 Aug 25 17:04 online
-rw-r--r--  1 root root 4096 Aug 25 17:04 readonly
-rw-r--r--  1 root root 4096 Aug 25 17:04 use_diag
```

b. デバイスがすでにオンラインになっているかを確認します。

```
# cat online
0
```

c. オンラインになっていない場合は、次のコマンドを実行してオンラインにします。

```
# echo 1 > online
# cat online
1
```

3. どのブロック devnode にアクセスしているかを確認します。

```
# ls -l
total 0
-r--r--r--  1 root root 4096 Aug 25 17:04 availability
lrwxrwxrwx  1 root root    0 Aug 25 17:07 block -> ../../../../block/dasdb
-rw-r--r--  1 root root 4096 Aug 25 17:04 cmb_enable
-r--r--r--  1 root root 4096 Aug 25 17:04 cutype
-rw-r--r--  1 root root 4096 Aug 25 17:04 detach_state
-r--r--r--  1 root root 4096 Aug 25 17:04 devtype
-r--r--r--  1 root root 4096 Aug 25 17:04 discipline
-rw-r--r--  1 root root    0 Aug 25 17:04 online
-rw-r--r--  1 root root 4096 Aug 25 17:04 readonly
-rw-r--r--  1 root root 4096 Aug 25 17:04 use_diag
```

この例では、 /dev/dasdb としてデバイス 4B2E にアクセスしてます。

この命令では、現行セッションに DASD オンラインを設定しましたが、システムが再起動すると元に戻ります。

DASD を永続的にオンラインに設定する方法については、DASD を永続的にオンラインに設定する を参照してください。DASD を使用するときは、**/dev/disk/by-path/** の下にある永続的なデバイスのシンボリックリンクを使用します。

17.3. ローレベルフォーマットによる新規 DASD の準備

ディスクがオンラインになったら、**/root** ディレクトリーに戻り、このデバイスにローレベルフォーマットを行います。DASD の有効期間中に必要なローレベルフォーマットは、この1回のみです。

```
# cd /root
# dasdfmt -b 4096 -d cdl -p /dev/disk/by-path/ccw-0.0.4b2e
Drive Geometry: 10017 Cylinders * 15 Heads =  150255 Tracks

I am going to format the device /dev/disk/by-path/ccw-0.0.4b2e in the following way:
```

```
Device number of device : 0x4b2e
Labelling device        : yes
Disk label              : VOL1
Disk identifier         : 0X4B2E
Extent start (trk no)   : 0
Extent end (trk no)     : 150254
Compatible Disk Layout  : yes
Blocksize               : 4096

--->> ATTENTION! <<---
All data of that device will be lost.
Type "yes" to continue, no will leave the disk untouched: yes
cyl    97 of  3338 |#----------------------------------------------|   2%
```

進捗バーが最後まで到達してフォーマットが完了したら、**dasdfmt** が以下の出力を表示します。

```
Rereading the partition table...
Exiting...
```

ここで、**fdasd** を使用して DASD にパーティションを設定します。DASD には最大 3 つの パーティションを作成できます。この例では、ディスク全体にまたがるパーティションを 1 つ作成します。

```
# fdasd -a /dev/disk/by-path/ccw-0.0.4b2e
reading volume label ..: VOL1
reading vtoc .....: ok

auto-creating one partition for the whole disk...
writing volume label...
writing VTOC...
rereading partition table...
```

(ローレベルフォーマットを行った) DASD をオンラインにすると、Linux 環境下の他のディスクと同様に使用できます。たとえば、ファイルシステム、LVM 物理ボリューム、またはそのパーティション (例: **/dev/disk/by-path/ccw-0.0.4b2e-part1**) にスワップ領域を作成できます。**dasdfmt** コマンドおよび **fdasd** コマンド以外では、絶対に DASD デバイス全体 (**dev/dasdb**) を使用しないでください。DASD 全体を使用する場合は、上述の **fdasd** の例で示すように、ドライブ全体にまたがるパーティションを 1 つ作成します。

たとえば **/etc/fstab** の既存のディスクエントリーの設定を壊さずに新しいディスクを後で追加するには、**/dev/disk/by-path/** 配下で永続的なデバイスシンボリックリンクを使用します。

17.4. DASD を永続的にオンラインに設定する

上記の手順では、実行中のシステムで DASD を動的にアクティベートする手順を説明しています。しかし、そのような変更は永続的ではなく再起動後には維持されません。Linux システム内で DASD 設定の変更を永続的にするには、DASD がルートファイルシステムに属するかどうかによります。root ファイルシステムに必要なこれらの DASD は、ブートプロセスの初期段階で **initramfs** でアクティベートして、root ファイルシステムをマウントできるようにする必要があります。

cio_ignore コマンドは、永続的なデバイス設定に応じて透過的に処理されるため、無視するリストからデバイスを手動で解放する必要はありません。

17.5. ルートファイルシステムの一部である DASD

Red Hat Enterprise Linux 9 では、root ファイルシステムの一部となる DASD を追加するために修正が必要なファイルが変更になりました。**/etc/zipl.conf** ファイルを編集する代わりに、編集する新しいファイルとその場所は、以下のコマンドを実行すると確認できます。

```
# machine_id=$(cat /etc/machine-id)
# kernel_version=$(uname -r)
# ls /boot/loader/entries/$machine_id-$kernel_version.conf
```

ブートプロセスの早い段階で DASD をアクティベートする起動オプションである **rd.dasd=** があります。このオプションは、DASD (Direct Access Storage Device) アダプターデバイスバス識別子を取ります。複数の DASD の場合は、パラメーターを複数回指定するか、バス ID のコンマ区切りリストを使用します。DASD の範囲を指定するには、最初と最後のバス ID を指定します。以下は、LVM ボリュームグループ **vg_devel1** に使用する 2 つの DASD のパーティションで、物理ボリュームを使用するシステムの **/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-4.18.0-80.el8.s390x.conf** ファイルの例です。この LVM ボリュームグループには、root ファイルシステム用の論理ボリューム **lv_root** が含まれています。

```
title Red Hat Enterprise Linux (4.18.0-80.el8.s390x) 8.0 (Ootpa)
version 4.18.0-80.el8.s390x
linux /boot/vmlinuz-4.18.0-80.el8.s390x
initrd /boot/initramfs-4.18.0-80.el8.s390x.img
options root=/dev/mapper/vg_devel1-lv_root crashkernel=auto rd.dasd=0.0.0200 rd.dasd=0.0.0207
rd.lvm.lv=vg_devel1/lv_root rd.lvm.lv=vg_devel1/lv_swap cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0
id rhel-20181027190514-4.18.0-80.el8.s390x
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

デバイスバス ID **0.0.202b** に含まれる 3 番目の DASD のパーティションに、別の物理ボリュームを追加します。これを行うには、**/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-4.18.0-32.el8.s390x.conf** で、ブートカーネルのパラメーター行に **rd.dasd=0.0.202b** を追加します。

```
title Red Hat Enterprise Linux (4.18.0-80.el8.s390x) 8.0 (Ootpa)
version 4.18.0-80.el8.s390x
linux /boot/vmlinuz-4.18.0-80.el8.s390x
initrd /boot/initramfs-4.18.0-80.el8.s390x.img
options root=/dev/mapper/vg_devel1-lv_root crashkernel=auto rd.dasd=0.0.0200 rd.dasd=0.0.0207
rd.dasd=0.0.202b rd.lvm.lv=vg_devel1/lv_root rd.lvm.lv=vg_devel1/lv_swap cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0
id rhel-20181027190514-4.18.0-80.el8.s390x
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

警告

設定ファイルで、カーネルコマンドラインの長さが 896 バイトを超えないようにしてください。これを超えてしまうとブートローダーを保存できず、インストールに失敗します。

zipl を実行して、次回の IPL 用に、設定ファイルの変更を適用します。

```
# zipl -V
Using config file '/etc/zipl.conf'
Using BLS config file '/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-4.18.0-
80.el8.s390x.conf'
Target device information
  Device.....................: 5e:00
  Partition..................: 5e:01
  Device name................: dasda
  Device driver name.........: dasd
  DASD device number.........: 0201
  Type.......................: disk partition
  Disk layout................: ECKD/compatible disk layout
  Geometry - heads...........: 15
  Geometry - sectors.........: 12
  Geometry - cylinders.......: 13356
  Geometry - start...........: 24
  File system block size.....: 4096
  Physical block size........: 4096
  Device size in physical blocks..: 262152
Building bootmap in '/boot'
Building menu 'zipl-automatic-menu'
Adding #1: IPL section '4.18.0-80.el8.s390x' (default)
  initial ramdisk...: /boot/initramfs-4.18.0-80.el8.s390x.img
  kernel image......: /boot/vmlinuz-4.18.0-80.el8.s390x
  kernel parmline...: 'root=/dev/mapper/vg_devel1-lv_root crashkernel=auto rd.dasd=0.0.0200
rd.dasd=0.0.0207 rd.dasd=0.0.202b rd.lvm.lv=vg_devel1/lv_root rd.lvm.lv=vg_devel1/lv_swap
cio_ignore=all,!condev rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0'
  component address:
    kernel image....: 0x00010000-0x0049afff
    parmline........: 0x0049b000-0x0049bfff
    initial ramdisk.: 0x004a0000-0x01a26fff
    internal loader.: 0x0000a000-0x0000cfff
Preparing boot menu
  Interactive prompt......: enabled
  Menu timeout.............: 5 seconds
  Default configuration...: '4.18.0-80.el8.s390x'
Preparing boot device: dasda (0201).
Syncing disks...
Done.
```

17.6. ルートファイルシステムの一部ではない DASD

データディスク など、root ファイルシステムに含まれない DASD (Direct Access Storage Devices) は **/etc/dasd.conf** ファイルで永続設定します。このファイルには、行ごとに DASD が含まれ、各行は DASD のバス ID で始まります。

DASD を **/etc/dasd.conf** ファイルに追加する場合は、キーと値のペアを使用して、各エントリーのオプションを指定します。キーとその値を等号 (=) 記号で区切ります。複数のオプションを追加する場合は、空白またはタブを使用して各オプションを区切ります。

/etc/dasd.conf ファイルの例

```
0.0.0207
0.0.0200 use_diag=1 readonly=1
```

/etc/dasd.conf ファイルへの変更は、システムの再起動後か、システムの I/O 設定を変更して新規の DASD を動的に追加した後 (DASD を z/VM にアタッチ後) に適用されます。

/etc/dasd.conf ファイルに追加した DASD を有効にするには、以下の手順を実行します。

1. **cio_ignore** ユーティリティーを使用して、無視するデバイスのリストから DASD を削除して表示させます。

   ```
   # cio_ignore -r device_number
   ```

   device_number は、DASD デバイス番号に置き換えます。

   たとえば、デバイス番号が **021a** の場合は、次のコマンドを実行します。

   ```
   # cio_ignore -r 021a
   ```

2. デバイスの **uevent** 属性に書き込み、DASD を有効化します。

   ```
   # echo add > /sys/bus/ccw/devices/dasd-bus-ID/uevent
   ```

   dasd-bus-ID は、DASD のバス ID に置き換えます。

   たとえば バス ID が **0.0.021a** の場合には、以下を実行します。

   ```
   # echo add > /sys/bus/ccw/devices/0.0.021a/uevent
   ```

17.7. ルートファイルシステムの一部である FCP LUN

Red Hat Enterprise Linux 9 では、root ファイルシステムの一部である FCP LUN を追加するために必要な唯一のファイルが変更されました。**/etc/zipl.conf** ファイルを編集する代わりに、編集する新しいファイルとその場所は、以下のコマンドを実行すると確認できます。

```
# machine_id=$(cat /etc/machine-id)
# kernel_version=$(uname -r)
# ls /boot/loader/entries/$machine_id-$kernel_version.conf
```

Red Hat Enterprise Linux には、ブートプロセスの早い段階で FCP LUN をアクティブにするパラメーターである **rd.zfcp=** があります。この値は、コンマで区切った FCP デバイスバス ID、 **0x** で始まる 16 進法の 16 桁の数字のターゲットの WWPN、および 0x で始まり 16 進法の 16 桁の数字の右側にゼロが列記される FCP LUN から設定されます。

WWPN および FCP LUN の値は、**zFCP** デバイスが NPIV モードで設定されていない場合にのみ必要です。これは、**zfcp.allow_lun_scan=0** カーネルモジュールパラメーターにより自動 LUN スキャンが無効になっている場合、または RHEL-9.0 以前のリリースをインストールする場合にのみ必要です。それ以外の場合は、**rd.zfcp=0.0.4000** などを省略できます。以下は、ルートファイルシステムの論理ボリューム **lv_root** を含む LVM ボリュームグループ **vg_devel1** の場合 に、FCP 接続された SCSI ディスクのパーティション上の物理ボリュームを使用するシステムの **/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-5.14.0-55.el9.s390x.conf** ファイルの例です。

```
title Red Hat Enterprise Linux (5.14.0-55.el9.s390x) 9.0 (Plow)
version 5.14.0-55.el9.s390x
linux /boot/vmlinuz-5.14.0-55.el9.s390x
initrd /boot/initramfs-5.14.0-55.el9.s390x.img
options root=/dev/mapper/vg_devel1-lv_root crashkernel=auto
rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a000000000
rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a000000000 rd.lvm.lv=vg_devel1/lv_root
rd.lvm.lv=vg_devel1/lv_swap cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0
id rhel-20181027190514-5.14.0-55.el9.s390x
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

1. 既存の物理ボリュームと同じ 2 つのパスを使用して、FCP LUN **0x401040a300000000** を備えた 2 番目の FCP 接続 SCSI ディスクのパーティションに別の物理ボリュームを追加するには、**rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a300000000** および **rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a300000000** を、**/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-5.14.0-55.el9.s390x.conf** のブートカーネルのパラメーター行に追加します。以下に例を示します。

```
title Red Hat Enterprise Linux (5.14.0-55.el9.s390x) 9.0 (Plow)
version 5.14.0-55.el9.s390x
linux /boot/vmlinuz-5.14.0-55.el9.s390x
initrd /boot/initramfs-5.14.0-55.el9.s390x.img
options root=/dev/mapper/vg_devel1-lv_root crashkernel=auto
rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a000000000
rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a000000000
rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a300000000
rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a300000000 rd.lvm.lv=vg_devel1/lv_root
rd.lvm.lv=vg_devel1/lv_swap cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0
id rhel-20181027190514-5.14.0-55.el9.s390x
grub_users $grub_users
grub_arg --unrestricted
grub_class kernel
```

警告

設定ファイルで、カーネルコマンドラインの長さが 896 バイトを超えないようにしてください。これを超えてしまうとブートローダーを保存できず、インストールに失敗します。

- **dracut -f** を実行して、ターゲットカーネルの初期 RAM ディスクを更新します。
- **zipl** を実行して、次回の IPL 用に、設定ファイルの変更を適用します。

```
# zipl -V
Using config file '/etc/zipl.conf'
Using BLS config file '/boot/loader/entries/4ab74e52867b4f998e73e06cf23fd761-5.14.0-
```

```
55.el9.s390x.conf'
Run /lib/s390-tools/zipl_helper.device-mapper /boot
Target device information
Device.....: fd:00
Partition.....: fd:01
Device name.....: dm-0
Device driver name.....: device-mapper
Type.....: disk partition
Disk layout.....: SCSI disk layout
Geometry - start.....: 2048
File system block size.....: 4096
Physical block size.....: 512
Device size in physical blocks..: 10074112
Building bootmap in '/boot/'
Building menu 'zipl-automatic-menu'
Adding #1: IPL section '5.14.0-55.el9.s390x' (default)
kernel image.....: /boot/vmlinuz-5.14.0-55.el9.s390x
kernel parmline...: 'root=/dev/mapper/vg_devel1-lv_root crashkernel=auto
rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a000000000
rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a000000000
rd.zfcp=0.0.fc00,0x5105074308c212e9,0x401040a300000000
rd.zfcp=0.0.fcd0,0x5105074308c2aee9,0x401040a300000000 rd.lvm.lv=vg_devel1/lv_root
rd.lvm.lv=vg_devel1/lv_swap cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0'
initial ramdisk...: /boot/initramfs-5.14.0-55.el9.s390x.img component address:
kernel image....: 0x00010000-0x007a21ff
parmline.....: 0x00001000-0x000011ff
initial ramdisk.: 0x02000000-0x028f63ff
internal loader.: 0x0000a000-0x0000a3ff
Preparing boot device: dm-0.
Detected SCSI PCBIOS disk layout.
Writing SCSI master boot record.
Syncing disks...
Done.
```

17.8. ルートファイルシステムの一部ではない FCP LUN

データディスクなど、root ファイルシステムの一部ではない FCP LUN は、**/etc/zfcp.conf** ファイルで永続的に設定されています。このファイルの各行には FCP LUN が含まれています。各行には、FCP アダプターのデバイスバス ID、**0x** で始まる 16 桁の 16 進数の数字のターゲット WWPN、および **0x** で始まり 16 桁の 16 進数の数字の右側にゼロが列記され、空白またはタブで区切られている FCP LUN から設定されます。

WWPN および FCP LUN の値は、**zFCP** デバイスが NPIV モードで設定されていない場合にのみ必要です。これは、**zfcp.allow_lun_scan=0** カーネルモジュールパラメーターにより **auto LUN** スキャンが無効になっている場合、または RHEL-9.0 以前のリリースをインストールする場合にのみ必要です。それ以外の場合は、省略でき、デバイスバス ID のみが必須となります。

/etc/zfcp.conf 内のエントリーは、FCP アダプターがシステムに追加される際に udev によってアクティベートされ、設定されます。システム起動時に表示される FCP アダプターすべてが追加され、**udev** を開始します。

/etc/zfcp.conf のコンテンツの例:

```
0.0.fc00 0x5105074308c212e9 0x401040a000000000
```

```
0.0.fc00 0x5105074308c212e9 0x401040a100000000
0.0.fc00 0x5105074308c212e9 0x401040a300000000
0.0.fcd0 0x5105074308c2aee9 0x401040a000000000
0.0.fcd0 0x5105074308c2aee9 0x401040a100000000
0.0.fcd0 0x5105074308c2aee9 0x401040a300000000
0.0.4000
0.0.5000
```

/etc/zfcp.conf の変更は、システムの再起動後か、システムの I/O 設定の変更による新規の FCP チャンネルの動的な追加 (たとえば、チャンネルが z/VM 下で接続) の後でのみ反映されます。もしくは、アクティブになっていなかった FCP アダプターに以下のコマンドを実行して、**/etc/zfcp.conf** ファイルでの新しいエントリーのアクティベーションを開始できます。

1. **zfcp_cio_free** ユーティリティーを使用して、無視されたデバイスのリストから FCP アダプターを削除し、Linux で表示できるようにします。

```
# zfcp_cio_free
```

2. **/etc/zfcp.conf** からの追加を実行中のシステムに適用するには、以下を実行します。

```
# zfcpconf.sh
```

17.9. QETH デバイスの追加

qeth ネットワークデバイスドライバーは、64 ビットの IBM Z の OSA-Express 機能を QDIO モード、HiperSockets、z/VM ゲスト LAN および z/VM VSWITCH でサポートします

qeth デバイスドライバーの命名スキームの詳細は、ブートパラメーターのカスタマイズ を参照してください。

17.10. QETH デバイスの動的な追加

このセクションでは、**qeth** デバイスを動的に追加する方法を説明します。

手順

1. **qeth** デバイスドライバーモジュールが読み込まれているかどうかを確認します。以下の例は、読み込み済みの **qeth** モジュールを示しています。

```
# lsmod | grep qeth
qeth_l3                69632  0
qeth_l2                49152  1
qeth                  131072  2 qeth_l3,qeth_l2
qdio                   65536  3 qeth,qeth_l3,qeth_l2
ccwgroup               20480  1 qeth
```

lsmod コマンドの出力で、**qeth** モジュールが読み込まれていないことを示している場合は、**modprobe** コマンドを実行してそのモジュールを読み込みます。

```
# modprobe qeth
```

2. **cio_ignore** ユーティリティーを使用して、無視されるデバイスのリストからネットワークチャネルを削除し、それが Linux から見えるようにします。

```
# cio_ignore -r read_device_bus_id,write_device_bus_id,data_device_bus_id
```

read_device_bus_id、**write_device_bus_id**、および **data_device_bus_id** は、ネットワークデバイスを表す 3 つのデバイスバス ID に置き換えます。たとえば、**read_device_bus_id** が **0.0.f500** で、**write_device_bus_id** が **0.0.f501** で、**data_device_bus_id** が **0.0.f502** の場合は、以下のようになります。

```
# cio_ignore -r 0.0.f500,0.0.f501,0.0.f502
```

3. **znetconf** ユーティリティーを使用して、ネットワークデバイス用の候補設定を識別して、リスト表示します。

```
# znetconf -u
Scanning for network devices...
Device IDs               Type    Card Type      CHPID Drv.
------------------------------------------------------------
0.0.f500,0.0.f501,0.0.f502 1731/01 OSA (QDIO)       00 qeth
0.0.f503,0.0.f504,0.0.f505 1731/01 OSA (QDIO)       01 qeth
0.0.0400,0.0.0401,0.0.0402 1731/05 HiperSockets     02 qeth
```

4. 使用する設定を選択し、**znetconf** を使用して設定を適用し、設定したグループデバイスをネットワークデバイスとしてオンラインにします。

```
# znetconf -a f500
Scanning for network devices...
Successfully configured device 0.0.f500 (encf500)
```

5. オプション: グループデバイスをオンラインに設定する前に、グループデバイスに設定されている引数を渡すこともできます。

```
# znetconf -a f500 -o portname=myname
Scanning for network devices...
Successfully configured device 0.0.f500 (encf500)
```

これで、**encf500** ネットワークインターフェイスの設定を継続できます。

または、**sysfs** 属性を使用して、以下のようにデバイスをオンラインに設定することもできます。

1. **qeth** グループデバイスを作成します。

```
# echo read_device_bus_id,write_device_bus_id,data_device_bus_id >
/sys/bus/ccwgroup/drivers/qeth/group
```

以下に例を示します。

```
# echo 0.0.f500,0.0.f501,0.0.f502 > /sys/bus/ccwgroup/drivers/qeth/group
```

2. 次に、読み込みチャンネルを見つけることで、**qeth** グループデバイスが正しく作成されていることを確認します。

```
# ls /sys/bus/ccwgroup/drivers/qeth/0.0.f500
```

必要なシステムや機能を設定する方法により、オプションで追加のパラメーターや機能を設定できます。以下に例を示します。

- **portno**
- **layer2**
- **portname**

3. オンライン **sysfs** 属性に **1** と書き込んでデバイスをオンラインにします。

```
# echo 1 > /sys/bus/ccwgroup/drivers/qeth/0.0.f500/online
```

4. 次に、デバイスの状態を確認します。

```
# cat /sys/bus/ccwgroup/drivers/qeth/0.0.f500/online
      1
```

戻り値が **1** の場合は、デバイスがオンラインであることを示し、戻り値が **0** の場合は、デバイスがオフラインであることを示します。

5. デバイスに割り当てられたインターフェイス名を見つけます。

```
# cat /sys/bus/ccwgroup/drivers/qeth/0.0.f500/if_name
encf500
```

これで、**encf500** ネットワークインターフェイスの設定を継続できます。

s390utils パッケージの以下のコマンドは、**qeth** デバイスの最も重要な設定を表示します。

```
# lsqeth encf500
Device name                    : encf500
------------------------------------------------
card_type               : OSD_1000
cdev0                   : 0.0.f500
cdev1                   : 0.0.f501
cdev2                   : 0.0.f502
chpid                   : 76
online                  : 1
portname                : OSAPORT
portno                  : 0
state                   : UP (LAN ONLINE)
priority_queueing       : always queue 0
buffer_count            : 16
layer2                  : 1
isolation               : none
```

17.11. QETH デバイスの永続的な追加

新しい **qeth** デバイスを永続的にするには、新しいインターフェイス用に設定ファイルを作成します。ネットワークインターフェイスの設定ファイルは **/etc/NetworkManager/system-connections/** ディレクトリーにあります。

ネットワーク設定ファイルには、命名規則 **device**.nmconnection を使用します。**device** は、以前作成した **qeth** グループデバイスのインターフェイス名ファイルにある値 (**enc9a0** など) です。**cio_ignore**

コマンドは永続的なデバイス設定に応じて透過的に処理されるので、無視する一覧からデバイスを手動で解放する必要はありません。

同じタイプの別のデバイスの設定ファイルがすでに存在する場合は、それを新しい名前でコピーして編集します。

```
# cd /etc/NetworkManager/system-connections/
# cp enc9a0.nmconnection enc600.nmconnection
```

お使いのネットワークデバイスの ID を確認するには、**lsqeth** ユーティリティーを使用します。

```
# lsqeth -p
devices                  CHPID interface        cardtype       port chksum prio-q'ing rtr4 rtr6 lay'2 cnt
------------------------ ----- ---------------- -------------- ---- ------ ---------- ---- ---- ----- -----
0.0.09a0/0.0.09a1/0.0.09a2 x00   enc9a0    Virt.NIC QDIO  0    sw     always_q_2 n/a  n/a  1     64
0.0.0600/0.0.0601/0.0.0602 x00   enc600    Virt.NIC QDIO  0    sw     always_q_2 n/a  n/a  1     64
```

同様のデバイスをこれまでに定義していない場合は、新しいファイルを作成します。次の例を使用してください。

```
[connection]
type=ethernet
interface-name=enc600

[ipv4]
address1=10.12.20.136/24,10.12.20.1
dns=10.12.20.53;
method=manual

[ethernet]
mac-address=00:53:00:8f:fa:66
```

新しい **enc600.nmconnection** ファイルを次のように編集します。

1. 新しい接続ファイルが **root:root** によって所有されていることを確認します。

   ```
   # chown root:root /etc/NetworkManager/system-connections/enc600.nmconnection
   ```

2. このファイルに詳細を追加するか、接続要件に基づいてこれらのパラメーターを変更します。
3. ファイルを保存します。
4. 接続プロファイルをリロードします。

   ```
   # nmcli connection reload
   ```

5. 新しく追加した接続の完全な詳細を表示するには、次のように入力します。

   ```
   # nmcli connection show enc600
   ```

enc600.nmconnection ファイルへの変更は、システムの再起動後、システムの I/O 設定の変更による新しいネットワークデバイスチャネルの動的な追加 (たとえば、z/VM での接続) の後、またはネットワーク接続のリロード後に有効になります。あるいは、次のコマンドを実行して、以前はまだアクティ

ブになっていなかったネットワークチャネルの **enc600.nmconnection** のアクティブ化をトリガーできます。

1. **cio_ignore** ユーティリティーを使用して、無視されるデバイスのリストからネットワークチャネルを削除し、それが Linux から見えるようにします。

```
# cio_ignore -r read_device_bus_id,write_device_bus_id,data_device_bus_id
```

read_device_bus_id、**write_device_bus_id**、**data_device_bus_id** は、ネットワークデバイスを表す 3 つのデバイスバス ID で置き換えます。たとえば、**read_device_bus_id** が **0.0.0600** で、**write_device_bus_id** が **0.0.0601** で、**data_device_bus_id** が **0.0.0602** の場合は、以下のようになります。

```
#  cio_ignore -r 0.0.0600,0.0.0601,0.0.0602
```

2. 次に変更をアクティベートする uevent を開始します。

```
# echo add > /sys/bus/ccw/devices/read-channel/uevent
```

以下に例を示します。

```
# echo add > /sys/bus/ccw/devices/0.0.0600/uevent
```

3. ネットワークデバイスのステータスを確認します。

```
# lsqeth
```

4. デフォルトのルート情報が変更された場合は、それに応じて **/etc/NetworkManager/system-connections/**<profile_name>.nmconnection ファイルの **[ipv4]** セクションと **[ipv6]** セクションの両方の **ipaddress1** パラメーターも更新する必要があります。

```
[ipv4]
address1=10.12.20.136/24,10.12.20.1
[ipv6]
address1=2001:db8:1::1,2001:db8:1::fffe
```

5. ここで新しいインターフェイスを開始します。

```
# nmcli connection up enc600
```

6. インターフェイスのステータスを確認します。

```
# ip addr show enc600
3: enc600:  <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP group default qlen 1000
link/ether 3c:97:0e:51:38:17 brd ff:ff:ff:ff:ff:ff
10.12.20.136/24 brd 10.12.20.1 scope global dynamic enc600
valid_lft 81487sec preferred_lft 81487sec
inet6 1574:12:5:1185:3e97:eff:fe51:3817/64 scope global noprefixroute dynamic
valid_lft 2591994sec preferred_lft 604794sec
inet6 fe45::a455:eff:d078:3847/64 scope link
valid_lft forever preferred_lft forever
```

7. 新しいインターフェイスのルーティングを確認します。

```
# ip route
default via 10.12.20.136 dev enc600 proto dhcp src
```

8. **ping** ユーティリティーを使用し、ゲートウェイ、または新規デバイスのサブネットにある別のホストに ping して、変更を確認します。

```
# ping -c 1 10.12.20.136
PING 10.12.20.136 (10.12.20.136) 56(84) bytes of data.
64 bytes from 10.12.20.136: icmp_seq=0 ttl=63 time=8.07 ms
```

9. デフォルトのルート情報を変更した場合は、それに応じて **/etc/sysconfig/network** も更新する必要があります。

関連情報

- **nm-settings-keyfile** man ページ

17.12. ネットワークの ROOT ファイルシステム用の 64 ビットの IBM Z ネットワークデバイスの設定

root ファイルシステムへのアクセスに必要なネットワークデバイスを追加するには、起動オプションの変更だけが必要です。起動オプションはパラメーターファイルに追加できますが、**/etc/zipl.conf** ファイルには、起動レコードの指定が含まれなくなります。修正が必要なファイルは、以下のコマンドを使用して配置できます。

```
# machine_id=$(cat /etc/machine-id)
# kernel_version=$(uname -r)
# ls /boot/loader/entries/$machine_id-$kernel_version.conf
```

Dracut (**mkinitrd** の後継であり、**initrd** の代わりとなる initramfs 内で機能を提供する) は、起動プロセスの早い段階で 64 ビットの IBM Z 上のネットワークデバイスをアクティベートする起動パラメーター **rd.znet=** を提供します。

このパラメーターには、**NETTYPE** (qeth、lcs、ctc) のリスト (2 つ (lcs、ctc) または 3 つ (qeth) のデバイスバス ID) をコンマ区切りで指定します。また、任意で、ネットワークデバイスの sysfs 属性に相当するキー値ペアで構成される追加パラメーターを指定します。このパラメーターは、64 ビットの IBM Z のネットワークハードウェアを設定し、アクティベートします。IP アドレスとその他のネットワーク仕様の設定は、他のプラットフォームと同様に機能します。詳細は **dracut** のドキュメントを参照してください。

ネットワークチャンネルに対する **cio_ignore** コマンドは、起動時に透過的に処理されます。

NFS 経由のネットワークでアクセスした root ファイルシステムの起動オプションの例:

```
root=10.16.105.196:/nfs/nfs_root cio_ignore=all,!condev
rd.znet=qeth,0.0.0a00,0.0.0a01,0.0.0a02,layer2=1,portno=0,portname=OSAPORT
ip=10.16.105.197:10.16.105.196:10.16.111.254:255.255.248.0:nfs-server.subdomain.domain:enc9a0:none rd_NO_LUKS rd_NO_LVM rd_NO_MD rd_NO_DM LANG=en_US.UTF-8
SYSFONT=latarcyrheb-sun16 KEYTABLE=us
```

第18章 システムの保護

インストールプロセスを完了した後、Red Hat Enterprise Linux システムを保護する必要があります。

前提条件

- グラフィカルインストールを完了している。

手順

1. root で以下のコマンドを実行して、システムを更新します。

```
# dnf update
```

2. ファイアウォールサービスの **firewalld** は、Red Hat Enterprise Linux のインストールで自動的に有効になっていますが、キックスタート設定などで明示的に無効となっている場合もあります。このような場合は、ファイアウォールを再度有効にすることが推奨されます。
firewalld を開始するには、root で次のコマンドを実行します。

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

3. セキュリティーを強化するために、不要なサービスは無効にしてください。たとえば、コンピューターにプリンターがインストールされていなければ、次のコマンドを実行して cups サービスを無効にします。

```
# systemctl mask cups
```

アクティブなサービスを確認するには、次のコマンドを実行します。

```
$ systemctl list-units | grep service
```

第19章 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。

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。

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。

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20章 サブスクリプションサービスの変更

サブスクリプションを管理するには、Red Hat Subscription Management Server または Red Hat Satellite Server に RHEL システムを登録します。必要に応じて、後でサブスクリプションサービスを変更できます。登録しているサブスクリプションサービスを変更するには、現在のサービスからシステムの登録を解除し、新しいサービスに登録します。

システムの更新を受け取るには、いずれかの管理サーバーでシステムを登録します。

このセクションは、Red Hat Subscription Management Server および Red Hat Satellite Server から RHEL システムの登録を解除する方法を説明します。

前提条件

以下のいずれかでシステムを登録している。

- Red Hat Subscription Management Server
- Red Hat Satellite Server version 6.11

注記

システムの更新を受け取るには、いずれかの管理サーバーでシステムを登録します。

20.1. SUBSCRIPTION MANAGEMENT SERVER からの登録解除

このセクションでは、コマンドラインと Subscription Manager ユーザーインターフェイスを使用して、Red Hat Subscription Management Server から RHEL システムの登録を解除する方法を説明します。

20.1.1. コマンドラインでの登録解除

unregister コマンドを使用して、Red Hat Subscription Management Server から RHEL システムの登録を解除します。

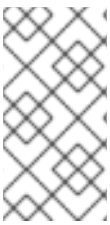
手順

1. root ユーザーで unregister コマンドにパラメーターを付けずに実行します。

```
# subscription-manager unregister
```

2. プロンプトが表示されたら、root パスワードを入力します。

システムが Subscription Management Server から登録解除され、ステータス The system is currently not registered が表示され、**登録** ボタンが有効になります。



注記

中断しなかったサービスを続けるには、いずれかの管理サービスでシステムの再登録を行います。管理サービスでシステムを登録しないと、システムの更新を受け取らないことがあります。システムの登録の詳細は、コマンドラインを使用したシステムの登録 を参照してください。

関連情報

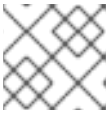
- Using and Configuring Red Hat Subscription Manager

20.1.2. Subscription Manager ユーザーインターフェイスを使用した登録解除

このセクションでは、Subscription Manager ユーザーインターフェイスを使用して、Red Hat Subscription Management Server から RHEL システムの登録を解除する方法を説明します。

手順

1. システムにログインします。
2. 画面左上で、**アクティビティー** をクリックします。
3. メニューオプションから、**アプリケーションを表示する** アイコンをクリックします。
4. **Red Hat Subscription Manager** アイコンをクリックするか、検索に **Red Hat Subscription Manager** と入力します。
5. **認証が必要です** ダイアログボックスで管理者パスワードを入力します。**サブスクリプション** 画面が開き、サブスクリプションの現在のステータス、システムの目的、インストール済み製品が表示されます。未登録の製品には、赤い X 印が表示されます。



注記

システムで特権タスクを実行するには、認証が必要です。

6. **登録解除** ボタンをクリックします。

システムが Subscription Management Server から登録解除され、ステータス The system is currently not registered が表示され、**登録** ボタンが有効になります。

注記

中断しなかったサービスを続けるには、いずれかの管理サービスでシステムの再登録を行います。管理サービスでシステムを登録しないと、システムの更新を受け取らないことがあります。システムの登録の詳細は、Subscription Manager ユーザーインターフェイスを使用したシステム登録 を参照してください。

関連情報

- Using and Configuring Red Hat Subscription Manager

20.2. SATELLITE SERVER からの登録解除

Satellite Server から Red Hat Enterprise Linux システムの登録を解除するには、Satellite Server からシステムを削除します。

詳細は、Red Hat Satellite からのホストの削除 を参照してください。

パート IV. 付録

付録A キックスタートスクリプトのファイル形式のリファレンス

このリファレンスでは、キックスタートファイルの形式について詳しく説明します。

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

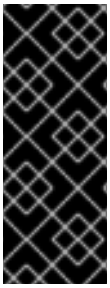
%packages コマンドを使用して、インストールするソフトウェアパッケージを説明するキックスタートセクションを開始します。**%packages** セクションは、**%end** コマンドで終了する必要があります。

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。

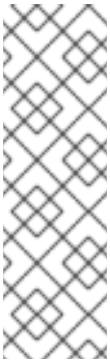
1. 以下の行でキックスタートファイルを設定します。

```
%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。

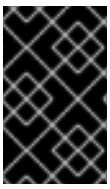
```
%packages
@Graphical Administration Tools --optional
%end
```

--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.2.5. 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

- 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。

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 以降、キックスタートインストールでは、インストールプログラムで致命的なエラーが発生するとカスタムスクリプトが実行されます。このような状況の例としては、インストールで要求されたパッケージのエラー、VNC の起動失敗 (設定で指定されている場合)、ストレージデバイスのスキャン中に発生したエラーなどが挙げられます。このようなイベントが発生した場合、インストールが中断します。このようなイベントを分析するために、インストールプログラムは、キックスタートファイルで指定されているすべての **%onerror** スクリプトを時系列順に実行します。トレースバックが発生した場合は、**%onerror** スクリプトを実行できます。

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。

inst.cmdline を使用してすべてのエラーを致命的なエラーにすることで、あらゆるエラーに対してエラーハンドラーを強制できます。

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リファレンス

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
- **nvdimm**

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。このコマンドは1回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。これはデフォルトになります。このコマンドは 1 回だけ使用してください。

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は、完了方法の指定がない場合、このオプションがデフォルトとして使用されます。このコマンドは 1 回だけ使用してください。

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**)。このコマンドは 1 回だけ使用してください。

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。このコマンドは1回だけ使用してください。

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

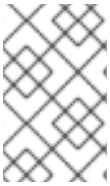
- イメージは、ライブ ISO イメージの **squashfs.img** ファイル、圧縮 tar ファイル (**.tar**、**.tbz**、**.tgz**、**.txz**、**.tar.bz2**、**.tar.gz**、または **.tar.xz**)、もしくはインストールメディアでマウントできるファイルシステムであればどれでも構いません。**ext2**、**ext3**、**ext4**、**vfat**、**xfs** などが対応ファイルシステムになります。

- 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。このコマンドは 1 回だけ使用してください。



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。このコマンドでは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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

--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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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、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。このコマンドは 1 回だけ使用してください。

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、最初のスペースまでのサービスだけが有効化または無効化されます。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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。

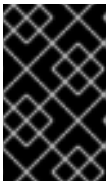
静的なネットワーク設定情報はすべて **一行で** 指定する必要があります。コマンドラインのようにバックスラッシュ (\) を使用して行を折り返すことはできません。

```
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

NIC チーミングは Red Hat Enterprise Linux 9 では非推奨です。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、ネットワークボンディングの設定 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には、このオプションを 1 回使用し、各 IP アドレスをコンマで区切ります。
- **--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。

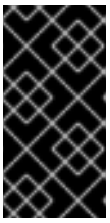
- **--ethtool=** - ethtool プログラムに渡されるネットワークデバイスの低レベルの追加設定を指定します。
- **--onboot=** - システムの起動時にデバイスを有効にするかどうかを指定します。
- **--dhcpclass=** - DHCP クラスを指定します。
- **--mtu=** - デバイスの MTU を指定します。
- **--noipv4** - このデバイスで IPv4 を無効にします。
- **--noipv6** - このデバイスで IPv6 を無効にします。
- **--bondslaves=** - このオプションを使用すると、**--bondslaves=** 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、コンマ (“,”) またはセミコロン (“;”) で区切る必要があります。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。利用可能なモードとサポートされているモードのリストについては、ネットワークの設定および管理ガイド 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。

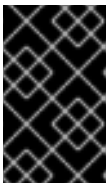
  - 名前にドット (**.**) が含まれている場合は、**NAME.ID** の形にする必要があります。**NAME** は任意ですが、**ID** は VLAN ID にする必要があります。たとえば、**em1.171**、**my-vlan.171** などにします。

  - **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。

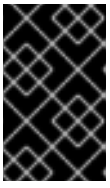


重要

NIC チーミングは Red Hat Enterprise Linux 9 では非推奨です。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、ネットワークボンディングの設定 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

NIC チーミングは Red Hat Enterprise Linux 9 では非推奨です。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、ネットワークボンディングの設定 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、ネットワークおよびホスト名のオプションの設定 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。

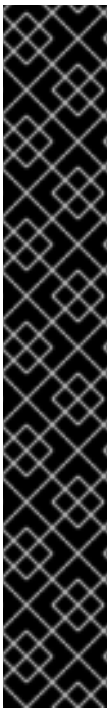
- **--client-software=** - ここで指定したクライアントソフトウェアを実行できるレルムにしか参加しません。使用できる値は **sssd** や **winbind** などになります。すべてのレルムがすべての値に対応しているとは限りません。デフォルトでは、クライアントソフトウェアは自動的に選択されます。

- **--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は、ルート (/) パーティション (1 GiB 以上)、 **swap** パーティション、およびアーキテクチャーに応じた適切な **/boot** パーティションです。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。このコマンドは 1 回だけ使用してください。

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** は必須です。ブートローダーをインストールする方法を指定します。このコマンドは 1 回だけ使用してください。

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、ブートローダーの設定 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
```

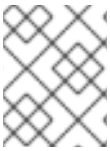
このアプローチを使用すると、コマンドは常に同じストレージデバイスをターゲットとします。これは、大規模なストレージ環境で特に役立ちます。システム上で使用可能なデバイス名を調べるには、対話型インストール中に **ls -lR/dev/disk** コマンドを使用できます。ストレージデバイスを一貫して参照するさまざまな方法の詳細は、永続的な命名属性の概要 を参照してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

構文

```
ignoredisk --drives=drive1,drive2,... | --only-use=drive
```

オプション

- **--drives=driveN,...** - driveN は、**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。このコマンドは 1 回だけ使用してください。

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。これらのコマンドによって割り当てられるスワップサイズについては、AMD64、Intel 64、および 64 ビット ARM システムの 推奨されるパーティション設定スキーム を参照してください。

--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

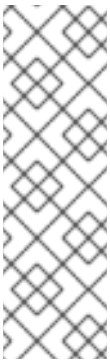
論理ボリュームを作成して、システムのハードウェアに基づいてそのボリュームのサイズを自動的に確定するために、このオプションを使用します。推奨されるスキームの詳細は、AMD64、Intel 64、および 64 ビット ARM システムの 推奨されるパーティション設定スキーム を参照してください。

--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 (非推奨)

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。

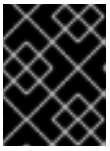
    自動的に swap パーティションサイズを確定しながら、ハイバネート用に余剰領域も割り当てる場合は、**--hibernation** オプションを使用します。

    ```
    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。これらのコマンドによって割り当てられるスワップサイズについては、AMD64、Intel 64、および 64 ビット ARM システムの 推奨されるパーティション設定スキーム を参照してください。
  - **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)。

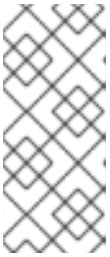


重要

--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。

サイズに関する推奨事項については、推奨されるパーティション設定スキーム を参照してください。

- **--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。

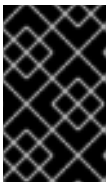
- **--fstype=** - パーティションのファイルシステムタイプを設定します。使用できる値は、**xfs**、**ext2**、**ext3**、**ext4**、**swap**、**vfat**、**efi**、および **biosboot** になります。

- **--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。推奨されるスキームの詳細は、AMD64、Intel 64、および 64 ビット ARM の 推奨されるパーティション設定スキーム を参照してください。



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*
```

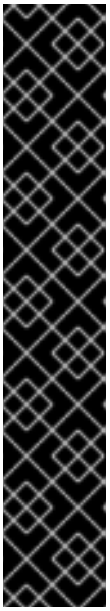
オプション

- **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。

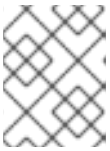
- **--chunksize=** - RAID ストレージのチャンクサイズを KiB 単位で設定します。場合によっては、デフォルトのサイズ (**512 Kib**) 以外のチャンクサイズを使用すると、RAID のパフォーマンスが向上することもあります。

- **--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

- 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。このコマンドは 1 回だけ使用してください。

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。このコマンドは 1 回だけ使用してください。

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。

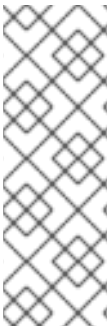
このオプションは、64 ビットの IBM Z にのみ適用されます。

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

この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付属するアドオン (Kdump および OpenSCAP) 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 キックスタートコマンドは任意です。

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。

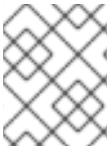
B.8. システム復旧用キックスタートコマンド

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。このコマンドは 1 回だけ使用してください。

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。