



Red Hat Enterprise Linux 7

仮想化入門ガイド

RHEL で利用可能な仮想化テクノロジーの紹介

Red Hat Enterprise Linux 7 仮想化入門ガイド

RHEL で利用可能な仮想化テクノロジーの紹介

Jiri Herrmann

Red Hat Customer Content Services

jherrman@redhat.com

Yehuda Zimmerman

Red Hat Customer Content Services

yzimmerm@redhat.com

Dayle Parker

Red Hat Customer Content Services

Laura Novich

Red Hat Customer Content Services

Jacquelynn East

Red Hat Customer Content Services

Scott Radvan

Red Hat Customer Content Services

法律上の通知

Copyright © 2019 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

Red Hat Enterprise Linux 仮想化入門ガイドでは、仮想化の基本と、Red Hat Enterprise Linux で利用可能な仮想化製品およびテクノロジーについて説明しています。

目次

第1章 仮想化の概要	3
1.1. 仮想化とは	3
1.2. 仮想化ソリューション	3
第2章 仮想化を使用する理由	5
2.1. 仮想化コスト	5
2.2. パフォーマンス	5
2.3. 移行	6
2.4. セキュリティー	7
2.5. 障害復旧	8
第3章 RED HAT VIRTUALIZATION 製品と機能の紹介	9
3.1. RED HAT ENTERPRISE LINUX での KVM と仮想化	9
3.2. LIBVIRT および LIBVIRT ツール	11
3.3. ドメイン XML 設定	12
3.4. 仮想化されたハードウェアデバイス	13
3.5. ストレージ	19
3.6. 仮想ネットワーク	21
第4章 仮想化コマンドラインインターフェイスの使用	23
4.1. 仮想化用の主なコマンドラインユーティリティー	23
4.2. デモ: コマンドラインユーティリティーを使用したゲストの作成と管理	27
第5章 VIRTUAL MACHINE MANAGER の使用	33
5.1. VIRTUAL MACHINE MANAGER の実行	33
5.2. VIRTUAL MACHINE MANAGER のインターフェイス	34
付録A 更新履歴	45

第1章 仮想化の概要

1.1. 仮想化とは

仮想化は、ソフトウェア (通常は複数のオペレーティングシステム) を単一のシステム上で他のプログラムから分離して同時に実行する場合に使用される広範なコンピューティング用語です。仮想化は、ハイパーバイザーを使用して実現されます。これは、ハードウェアを制御し、**仮想マシン (VM)** または **ゲスト** と呼ばれる複数のオペレーティングシステムを単一の (通常は物理) マシンで実行できるようにするソフトウェアレイヤーまたはサブシステムです。オペレーティングシステムを備えたこのようなマシンは、**ホスト** と呼ばれます。詳細は、[Red Hat カスタマーポータル](#) を参照してください。

仮想化にはいくつかの方法があります。

完全仮想化

完全仮想化では、そのままのゲストオペレーティングシステムが使用されます。ゲストは、ハイパーバイザーによって作成されたチャンネルを介してホストの CPU を指定します。ゲストは CPU と直接通信するため、これは最も高速な仮想化方法です。

準仮想化

準仮想化では、変更されたゲストオペレーティングシステムが使用されます。ゲストはハイパーバイザーと通信します。ハイパーバイザーは、ゲストからのそのままの呼び出しを CPU およびその他のインターフェイス (実インターフェイスと仮想インターフェイスの両方) に渡します。呼び出しはハイパーバイザー経由でルーティングされるため、この方法は完全な仮想化よりも遅くなります。

ソフトウェア仮想化 (またはエミュレーション)

ソフトウェア仮想化では、バイナリー変換およびその他のエミュレーション技術を使用して、そのままのオペレーティングシステムを実行します。ハイパーバイザーは、ゲストの呼び出しをホストシステムで使用できる形式に変換します。すべての呼び出しが変換されるため、この方法は仮想化よりも遅くなります。Red Hat は、Red Hat Enterprise Linux でのソフトウェア仮想化をサポートしていないことに注意してください。

コンテナ化

KVM 仮想化では OS カーネルの個別のインスタンスを作成しますが、**コンテナ化** と呼ばれるオペレーティングシステムレベルの仮想化は、既存の OS カーネル上で動作し、**コンテナ** と呼ばれる、ホスト OS の分離されたインスタンスを作成します。コンテナの詳細は、[Red Hat カスタマーポータル](#) を参照してください。

コンテナには KVM 仮想化の多様性はありませんが、より軽量で柔軟な処理が可能です。より詳細な比較は、[Introduction to Linux Containers](#) を参照してください。

Red Hat Enterprise Linux でコンテナを使用するには、[Extras チャンネル](#) から **docker パッケージ** をインストールします。Red Hat は、[Red Hat Enterprise Linux Atomic Host](#) や [Red Hat OpenShift Container Platform](#) など、コンテナを使用するための最適化されたソリューションも提供しています。コンテナのサポートの詳細は、[Red Hat ナレッジベース](#) を参照してください。

1.2. 仮想化ソリューション

Red Hat は、以下の主要な仮想化ソリューションを提供しています。各ソリューションには、さまざまなユーザーフォーカスおよび機能があります。

Red Hat Enterprise Linux

Red Hat Enterprise Linux 7 には、仮想マシンを作成、実行、および管理する機能と、[多数の仮想化ツールおよび機能](#)が含まれています。このソリューションでは、ホストごとに実行できるゲストの数が限られており、ゲストの種類も限られています。そのため、Red Hat Enterprise Linux での仮想化は、複数の環境でのテストが必要な開発者や、厳密なアップタイム要件やサービスレベルアグリーメント (SLA) を持たない複数のサーバーを実行している小規模企業などに役立ちます。



重要

このガイドでは、Red Hat Enterprise Linux での仮想化に関する情報を提供します。他の仮想化ソリューションについては詳しく説明しません。

Red Hat Virtualization

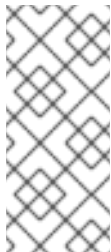
Red Hat Virtualization (RHV) は、Red Hat Enterprise Linux での仮想化と同様に、カーネルベースの仮想マシン (KVM) テクノロジーをベースとしていますが、拡張された一連の機能を提供します。エンタープライズクラスのスケラビリティとパフォーマンスを実現するように設計されており、ホスト、仮想マシン、ネットワーク、ストレージ、ユーザーを含む仮想インフラストラクチャー全体を一元化されたグラフィカルインターフェイスから管理できます。



注記

Red Hat Enterprise Linux と Red Hat Virtualization での仮想化の相違点は、[Red Hat カスタマーポータル](#)を参照してください。

Red Hat Virtualization は、大規模なデプロイメントまたはミッションクリティカルなアプリケーションを実行している企業に使用されることがあります。Red Hat Virtualization に適した大規模デプロイメントの例としては、ダウンタイムなしで継続的に実行する必要がある、データベース、商取引プラットフォーム、メッセージングシステムなどがあります。

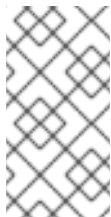


注記

Red Hat Virtualization の詳細について、または完全サポート付きの 60 日間の評価版をダウンロードするには、<http://www.redhat.com/en/technologies/virtualization/enterprise-virtualization>を参照してください。または、[Red Hat Virtualization ドキュメントスイート](#)を参照してください。

Red Hat OpenStack Platform

Red Hat OpenStack Platform は、安全で信頼性の高いパブリックまたはプライベートの [OpenStack](#) クラウドを作成、デプロイ、および拡張するための統合基盤を提供します。



注記

Red Hat OpenStack の詳細について、または 60 日間の評価版をダウンロードするには、<https://www.redhat.com/en/technologies/linux-platforms/openstack-platform>を参照してください。または、[Red Hat OpenStack Platform ドキュメントスイート](#)を参照してください。

第2章 仮想化を使用する理由

仮想化は、サーバーのデプロイメントと個々のデスクトップステーションの両方に役立ちます。デスクトップの仮想化は、コスト効率の高い一元管理と優れた障害復旧を実現します。また、ssh などの接続ツールを使用することで、リモートでデスクトップに接続することができます。

仮想化をサーバーに使用すると、大規模なネットワークだけでなく、複数のサーバーを使用するデプロイメントにもメリットがあります。仮想化は、ライブマイグレーション、高可用性、フォールトトレランス、効率的なバックアップを提供します。

2.1. 仮想化コスト

仮想化の導入には費用がかかる可能性がありますが、長期的には多くの場合、費用を節約できます。メリットとしては以下のものが考えられます。

省電力

仮想化を使用すると、複数の物理プラットフォームを確保する必要性がほとんどなくなります。そのため、マシンの動作と冷却に使用される電力が少なくなり、エネルギーコストが削減されます。複数の物理プラットフォームを購入する初期コストは、マシンの電力消費と必要な冷却と合わせて、仮想化を使用することで大幅に削減されます。

メンテナンスの削減

物理システムを仮想化システムに移行する前に適切な計画が策定されていれば、システムのメンテナンスに必要な時間が短縮されます。これにより、部品にかかる費用と人件費が削減されます。

インストール済みソフトウェアの寿命延長

古いバージョンのソフトウェアは、最新のベアメタルマシンでは直接実行できない場合があります。大規模で高速なシステム上で古いソフトウェアを仮想的に実行することにより、新しいシステムの優れたパフォーマンスを活用しながら、ソフトウェアの寿命を延ばすことができます。

予測可能なコスト

Red Hat Enterprise Linux サブスクリプションは、固定料金で仮想化のサポートを提供するため、コストの予測が容易になります。

省スペース

サーバーを少数のマシンに統合することで、コンピューターシステムに必要な物理スペースが少なくなります。

2.2. パフォーマンス

古い仮想化バージョンでは、単一の CPU しかサポートされていませんでした。その結果、仮想マシンのパフォーマンスには顕著な制限がありました。これにより、仮想化ソリューションは遅いという誤解が長く続いていました。

これは当てはまらなくなりました。最新の仮想化テクノロジーにより、仮想マシンの速度は大幅に向上しました。ベンチマークは、仮想マシンが一般的なサーバーアプリケーションをベアメタルシステムとほぼ同じ効率で実行できることを示しています。

- Red Hat Enterprise Linux 6.4 と KVM は、完全に仮想化された x86 環境で動作する IBM DB2 データベースで [業界最高レベルの TPC-C ベンチマーク](#) を記録し、ベアメタルの 88% のパフォーマンスを達成しました。リソースの都合上、データベースはこれまでベアメタル環境で

のみ利用されてきました。

- 業界標準の SAP Sales and Distribution (SD) Standard Application Benchmark では、Red Hat Enterprise Linux 6.2 と KVM は、同一のハードウェア上で動作するベアメタルシステムと比較して、[85% の仮想化効率で動作](#) することがわかりました。
- Red Hat Enterprise Linux 6.1 と KVM は、Standard Performance Evaluation Corporation (SPEC) によって記録された [SPECvirt_sc2010 ベンチマークで記録的な仮想化パフォーマンス](#) を達成し、公開されている SPECvirt の結果の中で最高の仮想パフォーマンスを記録しています。SPECvirt_sc2010 は、仮想化されたデータセンターサーバーのシステムコンポーネントのエンドツーエンドのパフォーマンスを測定するメトリックです。



注記

仮想化のパフォーマンスチューニングに関する詳細は、[『Red Hat Enterprise Linux 7 仮想化のチューニングと最適化ガイド』](#) を参照してください。

2.3. 移行

移行とは、ゲスト仮想マシンをあるホストから別のホストに移動するプロセスを表します。これが可能なのは、仮想マシンがハードウェア上で直接ではなく、仮想化された環境で実行されているためです。仮想マシンを移行するには、ライブとオフラインの 2 つの方法があります。

移行の種別

オフラインマイグレーション

オフラインマイグレーションでは、ゲスト仮想マシンを一時停止してから、仮想マシンのメモリーのイメージを移行先ホストに移動します。その後、仮想マシンが移行先ホストで再開し、移行元ホストで仮想マシンが使用していたメモリーが解放されます。

ライブマイグレーション

ライブマイグレーションは、アクティブな仮想マシンをある物理ホストから別の物理ホストに移行するプロセスです。これは、すべての Red Hat Enterprise Linux リリース間で可能であるわけではないことに注意してください。詳細は、[『仮想化の導入および管理ガイド』](#) を参照してください。

2.3.1. 仮想マシンの移行の利点

移行は、以下の場合に役立ちます。

負荷分散

ホストマシンが過負荷になったときに、ライブマイグレーションを使用して、そのホストマシンの 1 つ以上の仮想マシンを他のホストに移行できます。同様に、オフラインマイグレーションを使用して、実行していないマシンや過負荷になりがちなマシンを移行できます。

ホストのアップグレードまたは変更

ホスト上のハードウェアデバイスをアップグレード、追加、または削除する必要がある場合、仮想マシンを他のホストに安全に移動できます。これにより、ホストの変更に伴うゲストのダウンタイム発生を回避できます。

エネルギー節約

仮想マシンを他のホストに再配布し、アンロードしたホストシステムの電源をオフにして、低使用期間にエネルギーを節約し、コストを削減できます。

地理的な移行

仮想マシンは、待ち時間を短縮するため、またはその他の理由で、他の物理的な場所に移動できます。

移行プロセスが仮想マシンのメモリーを移動すると、仮想マシンに関連付けられたディスクボリュームも移行されます。このプロセスは、ライブブロックマイグレーションを使用して実行されます。



注記

移行の詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#) を参照してください。

2.3.2. 仮想から仮想への移行 (V2V)

特別なタイプの移行として、Red Hat Enterprise Linux 7 は、仮想マシンを他のタイプのハイパーバイザーから KVM に変換するためのツールを提供します。**virt-v2v** ツールは、Xen、他のバージョンの KVM、および VMware ESX から仮想マシンを変換およびインポートします。



注記

V2V の詳細は、[『V2V に関するナレッジベース記事』](#) を参照してください。

さらに、Red Hat Enterprise Linux 7.3 以降は、**virt-p2v** ツールを使用した物理から仮想 (P2V) への変換をサポートしています。詳細は、[P2V に関するナレッジベースの記事](#) を参照してください。

2.4. セキュリティー

KVM 仮想マシンは、次の機能を使用してセキュリティーを向上させます。

SELinux

Security-Enhanced Linux (SELinux) は、すべての Linux システムに強制アクセス制御 (MAC) を提供するため、Linux ゲストにもメリットがあります。SELinux の制御下では、すべてのプロセスとファイルに **タイプ** が与えられ、システム上でのそれらへのアクセスは、さまざまなタイプのきめ細かな制御によって制限されます。SELinux は、攻撃者の能力を制限し、バッファオーバーフロー攻撃や権限昇格などの多くの一般的なセキュリティーエクスプロイトを防ぐように機能します。

sVirt

sVirt は、Red Hat Enterprise Linux 7 に含まれるテクノロジーで、SELinux と仮想化を統合します。仮想マシンを使用する際のセキュリティーを向上させるために強制アクセス制御 (MAC) を適用し、ホストや別の仮想マシンを攻撃するために使用される可能性のあるハイパーバイザーのバグに対してシステムを強化します。



注記

仮想化のセキュリティーの詳細は、[『Red Hat Enterprise Linux 7 仮想化セキュリティーガイド』](#) を参照してください。

2.5. 障害復旧

システムが仮想化されていると、障害復旧がより迅速かつ簡単になります。物理システムで重大な問題が発生した場合、通常はオペレーティングシステムを完全に再インストールする必要があり、復旧に何時間もかかります。しかし、システムが仮想化されていれば、移行が可能なため、復旧がはるかに高速になります。ライブマイグレーションの要件を満たせば、仮想マシンを別のホストで再起動でき、最大の遅延はゲストデータの復元で発生することになります。また、仮想化された各システムは互いに完全に分離されているため、1つのシステムのダウンタイムが他のシステムに影響を与えることはありません。

第3章 RED HAT VIRTUALIZATION 製品と機能の紹介

この章では、Red Hat Enterprise Linux 7 で利用可能な主な仮想化製品と機能を紹介します。

3.1. RED HAT ENTERPRISE LINUX での KVM と仮想化

KVM (カーネルベースの仮想マシン) は、さまざまなアーキテクチャー上の Linux 向けの完全な仮想化ソリューションです。標準の Red Hat Enterprise Linux 7 カーネルに組み込まれ、Quick Emulator (QEMU) と統合されており、複数の ゲストオペレーティングシステムを実行できます。Red Hat Enterprise Linux の KVM ハイパーバイザーは、**libvirt** API、および **libvirt** 用に構築されたツール (**virt-manager** や **virsh** など) で管理されます。仮想マシンは、マルチスレッドの Linux プロセスとして実行され、これらのツールによって制御されます。

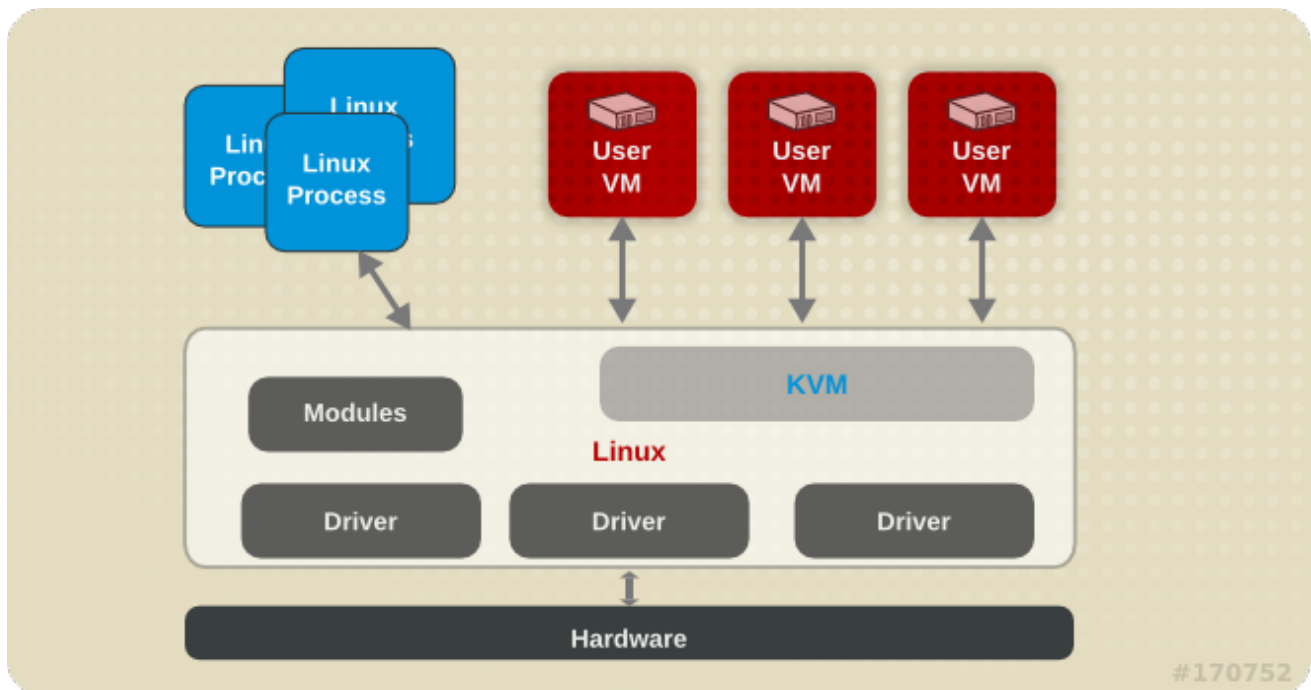


警告

QEMU と **libvirt** は、ハードウェア仮想化のサポートを必要としない QEMU Tiny Code Generator (TCG) を使用した動的変換モードもサポートしています。この設定は、Red Hat ではサポートされていません。

この制限の詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#)を参照してください。

図3.1 KVM アーキテクチャー



Red Hat Enterprise 7 の KVM でサポートされる仮想化機能には、以下が含まれます。

オーバーコミット

KVM ハイパーバイザーは、システムリソースの **オーバーコミット** をサポートしています。オーバーコミットとは、システムで利用可能なリソースよりも多くの仮想化された CPU またはメモリーを割り当てることで、あるゲストが必要とし、別のゲストが使用しない場合に、リソースを動的に

スワップできるようにすることです。これにより、ゲストがホストのリソースを効率的に使用できるようになり、ユーザーが必要とするホストの数を減らすことができます。

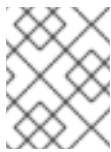


重要

オーバーコミットには、システムの安定性に対する潜在的なリスクが伴います。KVM でのオーバーコミットの詳細と注意事項については、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#)を参照してください。

KSM

KVM ハイパーバイザーが使用する *Kernel Same-page Merging (KSM)* により、KVM ゲストは同じメモリーページを共有できます。これらの共有ページは通常、一般的なライブラリーまたは他の同一の使用頻度の高いデータです。KSM を使用すると、メモリーの重複を避けることで、同一または同様のゲストオペレーティングシステムのゲスト密度が大きくなります。



注記

KSM の詳細は、[『Red Hat Enterprise Linux 7 仮想化のチューニングと最適化ガイド』](#)を参照してください。

QEMU ゲストエージェント

QEMU ゲストエージェントはゲストオペレーティングシステム上で実行され、ホストマシンがゲストオペレーティングシステムにコマンドを発行できるようにします。

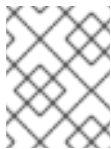


注記

QEMU ゲストエージェントの詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#)を参照してください。

ディスク I/O スロットリング

複数の仮想マシンが同時に実行されている場合、過剰なディスク I/O が使用され、システム全体のパフォーマンスに影響が及ぶ可能性があります。KVM の *ディスク I/O スロットリング* では、個々の仮想マシンからホストマシンに送られるディスク I/O 要求に制限を設定する機能を利用できます。これにより、仮想マシンが共有リソースを過剰に使用して、他の仮想マシンのパフォーマンスに影響を与えるのを防ぐことができます。

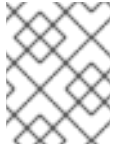


注記

ディスク I/O スロットリングの使用方法是、[『Red Hat Enterprise Linux 7 仮想化のチューニングと最適化ガイド』](#)を参照してください。

自動 NUMA バランシング

自動 Non-Uniform Memory Access (NUMA) バランシング は、タスク (スレッドまたはプロセス) を、アクセスしているメモリーの近くに移動します。これにより、Red Hat Enterprise Linux 7 ゲストを手動で調整しなくても、Non-Uniform Memory Access (NUMA) ハードウェアシステムで実行されているアプリケーションのパフォーマンスが向上します。

**注記**

自動 NUMA バランシングの詳細は、『[Red Hat Enterprise Linux 7 仮想化のチューニングと最適化ガイド](#)』を参照してください。

仮想 CPU ホットアド

仮想 CPU (vCPU) ホットアド機能は、ゲストをシャットダウンすることなく、実行中の仮想マシンの処理能力を必要に応じて向上させる機能を提供します。仮想マシンに割り当てられた vCPU は、ワークロードの要求を満たすため、またはワークロードに関連するサービスレベルアグリーメント (SLA) を維持するために、実行中のゲストに追加できます。

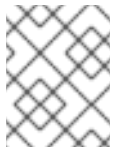
**注記**

仮想 CPU ホットアドの詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

ネストされた仮想化

Red Hat Enterprise Linux 7.2 以降では、テクノロジープレビューとして、ハードウェア支援によるネストされた仮想化を提供しています。この機能により、KVM ゲストはハイパーバイザーとして機能し、独自のゲストを作成できます。

これは、たとえば、仮想マシンでのハイパーバイザーのデバッグや、限られた数の物理マシンでの大規模な仮想デプロイメントのテストに使用できます。

**注記**

ネストされた仮想化の設定と使用に関する詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

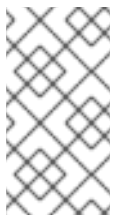
KVM ゲスト仮想マシンの互換性

Red Hat Enterprise Linux 7 サーバーには、特定のサポート制限があります。

以下の URL では、Red Hat Enterprise Linux のプロセッサとメモリの制限を説明します。

- ホストシステムの場合: <https://access.redhat.com/site/articles/rhel-limits>
- KVM ハイパーバイザーの場合: <https://access.redhat.com/site/articles/rhel-kvm-limits>

サポートされているオペレーティングシステムとホストとゲストの組み合わせの完全な表は、[Red Hat カスタマーポータル](#)を参照してください。

**注記**

お使いのプロセッサが仮想化拡張機能をサポートしているかどうかを確認する方法、および仮想化拡張機能が無効になっている場合に仮想化拡張機能を有効にする方法の詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

3.2. LIBVIRT および LIBVIRT ツール

libvirt パッケージは、さまざまなオペレーティングシステムの仮想化機能と対話できる、ハイパーバイザーに依存しない仮想化 API を提供します。これには、以下が含まれます。

- ホスト上の仮想マシンを安全に管理するための仮想化レイヤー。
- ローカルホストとネットワーク接続されたホストを管理するためのインターフェイス。
- 仮想マシンのプロビジョニング、作成、変更、監視、制御、移行、および停止に必要な API。libvirt では複数のホストに同時にアクセスできますが、API は単一ノードの操作に限定されます。



注記

libvirt を使用して実行できるのは、ハイパーバイザーがサポートする操作のみです。

libvirt は単一ホストの管理に重点を置いており、CPU、メモリー、ストレージ、ネットワーク、Non-Uniform Memory Access (NUMA) パーティションなど、管理対象ノードで利用可能なリソースを列挙、監視、および使用するための API を提供します。管理ツールは、ホストが実行されているマシンと同じ物理マシン上にある必要はありません。このようなシナリオでは、管理ツールが実行されているマシンは、ホストが実行されているマシンと安全なプロトコルを使用して通信します。

Red Hat Enterprise Linux 7 は libvirt をサポートし、デフォルトの仮想化管理方法として libvirt ベースのツールを備えています (Red Hat Virtualization Management と同様)。

libvirt パッケージは、GNU Lesser General Public License の下でフリーソフトウェアとして利用できます。libvirt プロジェクトは、長期的に安定した C API を仮想化管理ツールに提供し、さまざまなハイパーバイザーテクノロジー上で実行することを目的としています。libvirt パッケージは、Red Hat Enterprise Linux 5 上の Xen と、Red Hat Enterprise Linux 5、Red Hat Enterprise Linux 6、および Red Hat Enterprise Linux 7 上の KVM をサポートしています。

libvirt は特に、Red Hat Enterprise Linux 7 で仮想化を制御するための 2 つの主要なツールである **virsh** と **virt-manager** も提供しています。

3.3. ドメイン XML 設定

各 KVM ゲスト仮想マシン (ドメインとも呼ばれます) のホストベースの設定は、ゲストの XML 設定 (または *ドメイン XML*) に格納されます。これには、仮想ハードウェア、起動オプション、リソース割り当て、ネットワークインターフェイスなどの設定が含まれます。ゲストプロパティを永続的に変更するアプリケーションまたはユーティリティ (**virsh setmem** など) を使用すると、この変更がゲストの XML 設定に書き込まれます。ハイパーバイザーはゲストの起動時にそれを読み取り、それに応じて仮想マシンを変更します。

特定のゲストの XML 設定を表示するには、**virsh dumpxml *guestname*** コマンドを使用します。

ゲストの XML 設定を編集するには、次のいずれかを使用します。

- **virsh** コマンド - **virsh** コマンドを使用してゲスト仮想マシンに加えられた永続的な変更は、ドメイン XML に記録されます。
- **virt-xml** コマンド - このコマンドは、指定されたオプションに従って、指定されたゲストのドメイン XML ファイルを設定します。
- **Virtual Machine Manager** - **Virtual Machine Manager** でゲスト仮想マシンに加えられた変更は、ドメイン XML に記録されます。



注記

ドメイン XML ファイルを直接編集しなければならないまれなケースでは、**virsh edit guestname** を使用します。このコマンドは、指定されたゲストのドメイン XML 設定を、ルートの bash 設定 (デフォルトでは **vi**) で指定されているテキストエディターで開きます。

virsh edit のエディターを変更するには、目的のユーザーの **.bashrc** ファイルで **EDITOR** 変数を設定または変更します。たとえば、root ユーザーの場合、このファイルは **/root/** ディレクトリーにあります。

virsh edit を使用して実行した変更を有効にするには、編集した XML 設定を保存し、ゲストを再起動します。



警告

gedit /etc/libvirt/qemu/guestname.xml を使用するなどして、ゲストの XML 設定をテキストエディターでファイルとして開いて編集しないでください。この方法で行った変更は有効にならず、自動的に上書きされます。

ドメイン XML 設定ファイルの詳細は、[仮想化の導入および管理ガイド](#) を参照してください。

3.4. 仮想化されたハードウェアデバイス

Red Hat Enterprise Linux 7 の仮想化では、仮想マシンがホストの物理ハードウェアを 3 つの異なるタイプのデバイスとして使用できます。

- 仮想化およびエミュレートされたデバイス
- 準仮想化デバイス
- 物理的に共有されているデバイス

これらのハードウェアデバイスはすべて、仮想マシンに物理的に接続されたハードウェアデバイスとして表示されますが、デバイスドライバーはさまざまな方法で機能します。

3.4.1. 仮想化およびエミュレートされたデバイス

KVM は、仮想マシンの多くのコアデバイスをソフトウェアとして実装します。これらのエミュレートされたハードウェアデバイスは、オペレーティングシステムの仮想化に不可欠です。エミュレートされたデバイスは、完全にソフトウェア内に存在する仮想デバイスです。

さらに、KVM はエミュレートされたドライバーを提供します。これらのドライバーは、仮想マシンと (ソースデバイスを管理する) Linux カーネルの間の変換レイヤーを形成します。デバイスレベルの命令は、KVM ハイパーバイザーによって完全に変換されます。Linux カーネルによって認識される同じタイプのデバイス (ストレージ、ネットワーク、キーボード、またはマウス) は、エミュレートされたドライバーのバックリングソースデバイスとして使用できます。

仮想 CPU (vCPU)

Red Hat Enterprise Linux 7.2 以降では、ホストシステムは、ホスト CPU の数に関係なく、ゲストに提示して使用できる仮想 CPU (vCPU) を最大 240 個持つことができます。これは、Red Hat Enterprise Linux 7.0 の 160 から増加しています。

エミュレートされたシステムコンポーネント

基本的なシステム機能を提供するために、次のコアシステムコンポーネントがエミュレートされます。

- Intel i440FX ホスト PCI ブリッジ
- PIIX3 PCI-ISA ブリッジ
- PS/2 マウスとキーボード
- EvTouch USB グラフィックタブレット
- PCI UHCI USB コントローラーと仮想化された USB ハブ
- エミュレートされたシリアルポート
- EHCI コントローラー、仮想化された USB ストレージ、および USB マウス
- USB 3.0 xHCI ホストコントローラー (Red Hat Enterprise Linux 7.3 ではテクノロジープレビュー)

エミュレートされたストレージドライバー

ストレージデバイスとストレージプールは、エミュレートされたドライバーを使用して、ストレージデバイスを仮想マシンに接続できます。ゲストは、エミュレートされたストレージドライバーを使用して、ストレージプールにアクセスします。

すべての仮想デバイスと同様に、ストレージドライバーはストレージデバイスではないことに注意してください。ドライバーは、バックリングストレージデバイス、ファイル、またはストレージプールボリュームを仮想マシンに接続するのに使用されます。バックリングストレージデバイスは、サポートされている任意のタイプのストレージデバイス、ファイル、またはストレージプールボリュームにすることができます。

エミュレートされた IDE ドライバー

KVM は、2 つのエミュレートされた PCI IDE インターフェイスを提供します。エミュレートされた IDE ドライバーは、最大 4 つの仮想化 IDE ハードディスクまたは仮想化 IDE CD-ROM ドライブの組み合わせを各仮想マシンに割り当てるために使用できます。エミュレートされた IDE ドライバーは、仮想化された CD-ROM および DVD-ROM ドライブにも使用されます。

エミュレートされたフロッピーディスクドライブドライバー

エミュレートされたフロッピーディスクドライブドライバーは、仮想化されたフロッピードライブを作成するために使用されます。

エミュレートされたサウンドデバイス

エミュレートされた (Intel) HDA サウンドデバイス **intel-hda** は、次のゲストオペレーティングシステムでサポートされています。

- Red Hat Enterprise Linux 7 (AMD64 および Intel 64 アーキテクチャー向け)

- Red Hat Enterprise Linux 4、5、および 6 (32 ビット AMD および Intel アーキテクチャー向け、AMD64 および Intel 64 アーキテクチャー向け)



注記

次のエミュレートされたサウンドデバイスも利用できますが、特定のゲストオペレーティングシステムとの互換性の問題があるため、推奨されません。

- **ac97**、エミュレートされた Intel 82801AA AC97 Audio 互換サウンドカード

エミュレートされたグラフィックスカード

次のエミュレートされたグラフィックスカードが提供されます。

- Cirrus CLGD 5446 PCI VGA カード
- Bochs VESA 拡張機能を備えた標準 VGA グラフィックスカード (すべての非標準モードを含むハードウェアレベル)

ゲストは、Simple Protocol for Independent Computing Environments (SPICE) プロトコルまたは Virtual Network Computing (VNC) システムを使用してこれらのデバイスに接続できます。

エミュレートされたネットワークデバイス

次の 2 つのエミュレートされたネットワークデバイスが提供されます。

- **e1000** デバイスは、Intel E1000 ネットワークアダプター (Intel 82540EM、82573L、82544GC) をエミュレートします。
- **rtl8139** デバイスは、Realtek 8139 ネットワークアダプターをエミュレートします。

エミュレートされたウォッチドッグデバイス

ウォッチドッグを使用すると、マシンが過負荷になったり応答しなくなったりしたときに、仮想マシンを自動的に再起動できます。

Red Hat Enterprise Linux 7 は、以下のエミュレートされたウォッチドッグデバイスを提供します。

- **i6300esb**、エミュレートされた Intel 6300 ESB PCI ウォッチドッグデバイス。これは、ゲストオペレーティングシステムの Red Hat Enterprise Linux バージョン 6.0 以降でサポートされており、使用が推奨されるデバイスです。
- **ib700**、エミュレートされた iBase 700 ISA ウォッチドッグデバイス。**ib700** ウォッチドッグデバイスは、Red Hat Enterprise Linux 6.2 以降を使用するゲストでのみサポートされます。

どちらのウォッチドッグデバイスも、ゲストオペレーティングシステム Red Hat Enterprise Linux 6.2 以降の 32 ビットおよび 64 ビットの AMD および Intel アーキテクチャーでサポートされています。

3.4.2. 準仮想化デバイス

準仮想化は、ゲストがホストマシン上のデバイスを使用するための高速で効率的な通信手段を提供します。KVM は、ハイパーバイザーとゲストとの間のレイヤーとして virtio API を使用して、準仮想化デバイスを仮想マシンに提供します。

I/O レイテンシーを低減し、ベアメタルに近いレベルまで I/O スループットを向上させる準仮想化デバ

イス もあれば、他の方法では利用できない機能を仮想マシンに追加する準仮想化デバイスもあります。I/O 集約型アプリケーションを実行する仮想マシンには、エミュレートされたデバイスの代わりに準仮想化デバイスを使用することが推奨されます。

すべての virtio デバイスには、ホストデバイスとゲストドライバーの 2 つの部分があります。準仮想化デバイスドライバーにより、ゲストオペレーティングシステムはホストシステム上の物理デバイスにアクセスできます。

このデバイスを使用するには、準仮想化デバイスドライバーをゲストオペレーティングシステムにインストールする必要があります。デフォルトでは、準仮想化デバイスドライバーは Red Hat Enterprise Linux 4.7 以降、Red Hat Enterprise Linux 5.4 以降、および Red Hat Enterprise Linux 6.0 以降に含まれています。



注記

準仮想化デバイスおよびドライバーの使用に関する詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#) を参照してください。

準仮想化ネットワークデバイス (virtio-net)

準仮想化ネットワークデバイスは、I/O パフォーマンスが高くレイテンシーが低いネットワークアクセスを仮想マシンに提供する仮想ネットワークデバイスです。

準仮想化ブロックデバイス (virtio-blk)

準仮想化ブロックデバイスは、I/O パフォーマンスが高くレイテンシーが低いストレージを仮想マシンに提供する高パフォーマンスの仮想ストレージデバイスです。準仮想化ブロックデバイスは、ハイパーバイザーによってサポートされ、仮想マシンに接続されます (エミュレートする必要があるフロッピーディスクドライブを除く)。

準仮想化コントローラーデバイス (virtio-scsi)

準仮想化 SCSI コントローラーデバイスは、virtio-blk に代わる、より柔軟でスケーラブルな代替手段です。virtio-scsi ゲストは、ターゲットデバイスの機能セットを継承することができ、28 個のデバイスしか処理できない virtio-blk と比較して、数百のデバイスを処理できます。

virtio-scsi は、次のゲストオペレーティングシステムで完全にサポートされています。

- Red Hat Enterprise Linux 7
- Red Hat Enterprise Linux 6.4 以降

準仮想化されたクロック

Time Stamp Counter (TSC) をクロックソースとして使用しているゲストには、タイミングの問題が発生する場合があります。KVM は、ゲストに準仮想化クロックを提供することで、一定の Time Stamp Counter を持たないホストに対処します。さらに、準仮想化クロックは、ゲストがスリープ (S3) または RAM へのサスペンド操作を実行した後に必要な時間調整を支援します。

準仮想化されたシリアルデバイス (virtio-serial)

準仮想化シリアルデバイスは、バイトストリーム指向の文字ストリームデバイスであり、ホストのユーザー空間とゲストのユーザー空間の間の単純な通信インターフェイスを提供します。

バルーンデバイス (virtio-balloon)

バルーンデバイスは、仮想マシンの RAM の一部が使用されていないことを指定できます (バルーンの膨張として知られているプロセス)。これにより、ホストや、そのホストのその他の仮想マシン

が使用するメモリーを解放できます。仮想マシンにメモリーが再度必要になると、バルーンが収縮され、ホストは RAM を仮想マシンに分散して戻すことができます。

準仮想化乱数ジェネレーター (virtio-rng)

準仮想化乱数ジェネレーターは、仮想マシンがホストから直接エントロピーまたはランダム性を収集して、暗号化されたデータとセキュリティーに使用できるようにします。仮想マシンでは、通常の入力 (ハードウェアの使用状況など) が利用できないため、エントロピーが枯渇することがよくあります。エントロピーの収集には時間がかかる場合があります。**virtio-rng** は、エントロピーをホストからゲスト仮想マシンに直接注入することで、このプロセスを高速化します。

準仮想化グラフィックカード (QXL)

準仮想化グラフィックカードは QXL ドライバーと連携して、リモートホストから仮想マシンのグラフィックを表示する効率的な方法を提供します。SPICE を使用するには、QXL ドライバーが必要です。

3.4.3. 物理ホストデバイス

特定のハードウェアプラットフォームにより、仮想マシンはさまざまなハードウェアデバイスやコンポーネントに直接アクセスできます。仮想化におけるこのプロセスは、**デバイス割り当て**や **パススルー**と呼ばれます。

VFIO デバイスの割り当て

Virtual Function I/O (VFIO) は、Red Hat Enterprise Linux 7 の新しいカーネルドライバーであり、物理ハードウェアへの高パフォーマンスアクセスを仮想マシンに提供します。

VFIO は、ホストシステム上の PCI デバイスを仮想マシンに直接接続し、さまざまなタスクのために PCI デバイスへの排他的アクセスをゲストに提供します。これにより、PCI デバイスがゲスト仮想マシンに物理的に接続されているかのように表示され、動作するようになります。

VFIO は、KVM ハイパーバイザーからデバイス割り当てを移動し、カーネルレベルでデバイス分離を実施することにより、以前の PCI デバイス割り当てアーキテクチャーを改善します。VFIO はより優れたセキュリティーを提供し、セキュアブートと互換性があります。VFIO は Red Hat Enterprise Linux 7 のデフォルトのデバイス割り当てメカニズムです。

VFIO により、割り当てられるデバイスの数が増加し、Red Hat Enterprise Linux 7 では 32 デバイスとなりました。Red Hat Enterprise Linux 6 では最大 8 デバイスです。VFIO は、NVIDIA GPU の割り当てもサポートしています。

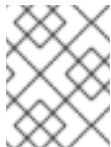


注記

VFIO デバイスの割り当てに関する詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

USB、PCI、および SCSI パススルー

KVM ハイパーバイザーは、ホストシステム上の USB、PCI、および SCSI デバイスの仮想マシンへの接続をサポートしています。USB、PCI、および SCSI デバイスの割り当てにより、デバイスが仮想マシンに物理的に接続されているかのように表示され、動作するようになります。そのため、さまざまなタスクを実行するために、これらのデバイスへの排他的アクセスをゲストに提供できます。

**注記**

USB、PCI、および SCSI パススルーの詳細については、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

SR-IOV

SR-IOV (Single Root I/O Virtualization) は、単一の物理 PCI 機能を拡張してその PCI リソースを個別の **仮想機能 (VF)** として共有する PCI Express (PCI-e) 標準です。各機能は、PCI デバイスの割り当てを介して異なる仮想マシンで使用できます。

SR-IOV 対応の PCI-e デバイスは、シングルルート機能 (単一のイーサネットポートなど) を提供し、複数の個別の仮想デバイスを独自の PCI デバイス機能として提供します。各仮想デバイスには、独自の固有の PCI 設定領域、メモリーにマップされたレジスター、および個々の MSI ベースの割り込みがあります。

**注記**

SR-IOV の詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

NPIV

N_Port ID Virtualization (NPIV) は、一部のファイバーチャネルデバイスで利用できる機能です。NPIV は、単一の物理 N_Port を複数の N_Port ID として共有します。NPIV は、SR-IOV が PCIe インターフェイスに提供するのと同様の機能をファイバーチャネルホストバスアダプター (HBA) に提供します。NPIV を使用すると、ストレージエリアネットワーク (SAN) への仮想ファイバーチャネルイニシエーターを仮想マシンに提供できます。

NPIV は、エンタープライズレベルのストレージソリューションを備えた高密度の仮想化環境を提供できます。

**注記**

NPIV の詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

3.4.4. ゲスト CPU モデル

CPU モデル は、どのホスト CPU 機能をゲストオペレーティングシステムに公開するかを定義します。**KVM** と **libvirt** には多くのプロセッサモデルの定義が含まれており、ユーザーは新しい CPU モデルでのみ利用可能な CPU 機能を有効にすることができます。ゲストに公開できる CPU 機能のセットは、ホスト CPU、カーネル、および **KVM** コードでのサポートに依存します。

CPU 機能のセットが異なるホスト間で仮想マシンを安全に移行できるように、**KVM** は、デフォルトでホスト CPU のすべての機能をゲストオペレーティングシステムに公開しません。そうではなく、選択された CPU モデルに基づいて CPU 機能を公開します。仮想マシンで特定の CPU 機能が有効になっている場合、その機能をゲストに公開することをサポートしていないホストにその仮想マシンを移行することはできません。



注記

ゲスト CPU モデルの詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

3.5. ストレージ

仮想マシンのストレージは、仮想マシンに割り当てられた物理ストレージから抽象化されます。これは、準仮想化またはエミュレートされたブロックデバイスドライバを使用して仮想マシンに接続されます。

3.5.1. ストレージプール

ストレージプールは、仮想マシンにストレージを提供する目的で `libvirt` によって管理されるファイル、ディレクトリ、またはストレージデバイスです。ストレージプールは、仮想マシンイメージを保存するストレージ ボリュームに分割されるか、追加のストレージとして仮想マシンに割り当てられます。複数のゲストで同じストレージプールを共有できるため、ストレージリソースの割り当てを改善できます。詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

ローカルストレージのプール

ローカルストレージプールは、ホストサーバーに直接割り当てることができます。これには、ローカルデバイスのローカルディレクトリ、直接接続したディスク、物理パーティション、および論理ボリューム管理 (LVM) ボリュームグループが含まれます。ローカルストレージプールは、移行や大量の仮想マシンを必要としない開発、テスト、および小規模なデプロイメントに役立ちます。ローカルストレージプールは、ライブマイグレーションをサポートしていないため、多くの実稼働環境には適していない場合があります。

ネットワーク (共有) ストレージプール

ネットワークストレージプールには、標準プロトコルを使用してネットワーク経由で共有されるストレージデバイスが含まれます。`virt-manager` を使用してホスト間で仮想マシンを移行する場合はネットワークストレージが必要ですが、`virsh` を使用して移行する場合は任意です。ネットワークストレージプールは `libvirt` によって管理されます。

3.5.2. ストレージボリューム

ストレージプールは、ストレージボリュームに分類されます。ストレージボリュームは、`libvirt` が処理する物理パーティション、LVM 論理ボリューム、ファイルベースのディスクイメージ、その他のストレージタイプの抽象化です。ストレージボリュームは、基盤となるハードウェアに関係なく、ローカルストレージデバイスとして仮想マシンに提示されます。

3.5.3. エミュレートされたストレージデバイス

仮想マシンには、ホストによってエミュレートされるさまざまなストレージデバイスを提示できます。各タイプのストレージデバイスは特定のユースケースに適しており、最大限の柔軟性とゲストオペレーティングシステムとの互換性を実現できます。

`virtio-scsi`

`virtio-scsi` は、多数のディスクまたは TRIM などの高度なストレージ機能を使用するゲストに推奨される準仮想化デバイスです。Red Hat Enterprise Linux 7 以外のオペレーティングシステムを使用しているゲストでは、ゲストドライバーのインストールが必要になる場合があります。

virtio-blk

virtio-blk は、イメージファイルをゲストに公開するのに適した準仮想化ストレージデバイスです。**virtio-blk** は最適なディスク I/O パフォーマンスを仮想マシンに提供できますが、**virtio-scsi** より機能が少なくなっています。

IDE

IDE は、**virtio** ドライバーをサポートしていないレガシーゲストに推奨されます。IDE のパフォーマンスは **virtio-scsi** または **virtio-blk** よりも低いものの、IDE はさまざまなシステムと互換性があります。

CD-ROM

ATAPI CD-ROM および **virtio-scsi** CD-ROM が利用可能です。これらにより、ゲストが ISO ファイルまたはホストの CD-ROM ドライブを使用できるようになります。**virtio-scsi** CD-ROM は、**virtio-scsi** ドライバーがインストールされているゲストで使用できます。ATAPI CD-ROM のほうが互換性は高いものの、パフォーマンスは低くなります。

USB 大容量記憶装置とフロッピーディスク

リムーバブルメディアが必要な場合は、エミュレートされた USB 大容量記憶装置とフロッピーディスクを使用できます。USB 大容量記憶装置は、容量が大きいため、フロッピーディスクよりも推奨されます。

3.5.4. ホストストレージ

ディスクイメージは、ホストに接続されたさまざまなローカルおよびリモートストレージテクノロジーに保存できます。

イメージファイル

イメージファイルは、ホストファイルシステムにのみ保存できます。イメージファイルは、**ext4** や **xfs** などのローカルファイルシステム、または **NFS** などのネットワークファイルシステムに保存できます。

libguestfs などのツールで、ファイルの管理、バックアップ、および監視を行うことができます。**KVM** のディスクイメージ形式には次のものがあります。

raw

raw イメージファイルには、追加のメタデータを含まないディスクのコンテンツが含まれます。

raw ファイルには、事前に割り当てられたファイルとスパースファイルがあります (ホストファイルシステムで許可されている場合)。スパースファイルは、オンデマンドでホストディスク領域を割り当てるため、シンプロビジョニングの一種です。事前に割り当てられたファイルは完全にプロビジョニングされますが、スパースファイルよりもパフォーマンスが高くなります。

raw ファイルは、ディスク I/O パフォーマンスが重要で、ネットワーク経由でイメージファイルを転送する必要がほとんどない場合に適しています。

qcow2

qcow2 イメージファイルは、バックアップファイル、スナップショット、圧縮、暗号化など、多くの高度なディスクイメージ機能を提供します。テンプレートイメージから仮想マシンをインスタンス化するために使用できます。

通常、**qcow2** ファイルはネットワーク経由で転送する方が効率的です。これは、仮想マシンによって書き込まれたセクターのみがイメージに割り当てられるためです。

Red Hat Enterprise Linux 7 は、qcow2 バージョン 3 イメージファイル形式をサポートしています。

LVM ボリューム

論理ボリューム (LV) は、ディスクイメージに使用でき、システムの LVM ツールを使用して管理できます。LVM は、ブロックストレージモデルが単純であるため、ファイルシステムよりも高いパフォーマンスを提供します。

LVM シンプロビジョニングは、LVM ボリュームのスナップショットと効率的な領域の使用を提供し、qcow2 への移行の代替手段として使用できます。

ホストデバイス

物理 CD-ROM、raw ディスク、論理ユニット番号 (LUN) などのホストデバイスをゲストに提示できます。これにより、ゲストは SAN または iSCSI LUN とローカル CD-ROM メディアを優れたパフォーマンスで使用できます。

ホストデバイスは、ストレージ管理をホストではなく SANで行う場合に使用できます。

分散ストレージシステム

Gluster ボリュームをディスクイメージとして使用できます。これにより、高パフォーマンスのクラスター化ストレージをネットワーク上で実現できます。

Red Hat Enterprise Linux 7 には、GlusterFS 上のディスクイメージのネイティブサポートが含まれています。これにより、KVM ホストは GlusterFS ボリュームから仮想マシンイメージを起動し、GlusterFS ボリュームのイメージを仮想マシンのデータディスクとして使用できます。GlusterFS FUSE と比較すると、KVM のネイティブサポートはより高いパフォーマンスを提供します。



注記

ストレージと仮想化の詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

3.6. 仮想ネットワーク

仮想ゲストのあらゆるネットワークへの接続は、物理ホストのソフトウェアネットワークコンポーネントを使用します。これらのソフトウェアコンポーネントは、**libvirt** の仮想ネットワーク設定を使用して再配置および再設定できます。したがって、ホストは [仮想ネットワークスイッチ](#) として機能し、ゲストのネットワークニーズに合わせてさまざまな方法で設定できます。

デフォルトでは、1 台のホストにあるすべてのゲストが、同じ libvirt 仮想ネットワーク (**default**) に接続されています。このネットワーク上にあるゲストは、以下の接続を行うことができます。

互いを使用、および仮想化ホストを使用

インバウンドトラフィックとアウトバウンドトラフィックの両方が可能ですが、ゲストオペレーティングシステムのネットワークスタック内のファイアウォールと、ゲストインターフェイスに接続された [libvirt ネットワークフィルタリング](#) ルールの影響を受けます。

仮想化ホスト以外のネットワークにあるその他のホストを使用

アウトバウンドトラフィックのみが可能で、[ネットワークアドレス変換 \(NAT\)](#) ルールおよびホストシステムのファイアウォールの影響を受けます。

ただし、必要な場合は、ゲストインターフェイスを以下のいずれかのモードに設定できます。

分離モード

ゲストは、仮想化ホスト外へのトラフィックが許可されないネットワークに接続されます。

ルーティングモード

ゲストは、NAT を実行せずにゲストと外部ホスト間のトラフィックをルーティングするネットワークに接続されます。これにより、着信接続が有効になりますが、外部ネットワークのシステムに追加のルーティングテーブルエントリが必要になります。

ブリッジモード

ゲストは、ローカルイーサネットに接続された物理イーサネットデバイスに直接接続されているブリッジデバイスに接続されます。これにより、ゲストは物理ネットワーク上で直接見えるようになり、着信接続が可能になりますが、追加のルーティングテーブルエントリは必要ありません。

仮想マシンからの基本的なアウトバウンドのみのネットワークアクセスでは、**default** ネットワークが `libvirt` パッケージと一緒にインストールされ、**libvirtd** サービスの開始時に自動的に開始されるため、通常は追加のネットワーク設定は必要ありません。より高度な機能が必要な場合は、`virsh` または `virt-manager` を使用して追加のネットワークを作成および設定し、[ゲストの XML 設定ファイル](#) を編集して、これらの新しいネットワークのいずれかを使用できます。



注記

高度な仮想ネットワーク設定の詳細は、[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#) を参照してください。

ゲストオペレーティングシステムの視点から見ると、仮想ネットワーク接続は通常の物理ネットワーク接続と同じです。Red Hat Enterprise Linux 7 ゲストでのネットワーク設定の詳細は、[Red Hat Enterprise Linux 7 ネットワークガイド](#) を参照してください。

第4章 仮想化コマンドラインインターフェイスの使用

Red Hat Enterprise Linux 7 で仮想化を操作する標準的な方法は、コマンドラインユーザーインターフェイス (CLI) を使用することです。CLI コマンドを入力すると、ホストシステムで仮想マシンを作成または操作するシステムユーティリティーがアクティブになります。この方法を使用すると、**virt-manager** などのグラフィカルアプリケーションを使用するよりも詳細に制御でき、スクリプト作成や自動化を行うこともできます。

4.1. 仮想化用の主なコマンドラインユーティリティー

以下のサブセクションでは、Red Hat Enterprise Linux 7 で仮想化を設定および管理するために使用できる主なコマンドラインユーティリティーを一覧表示します。これらのコマンドは、他の多数の仮想化ユーティリティーと同様に、Red Hat Enterprise Linux リポジトリが提供するパッケージに含まれており、[Yum パッケージマネージャーを使用](#)してインストールできます。

仮想化パッケージのインストールの詳細は、[仮想化の導入および管理ガイド](#)を参照してください。

4.1.1. virsh

virsh は、ハイパーバイザーとゲスト仮想マシンを管理するための CLI ユーティリティーです。これは、Red Hat Enterprise Linux 7 で仮想化を制御する主要な手段です。その機能は次のとおりです。

- 仮想マシンの作成、設定、一時停止、一覧表示、およびシャットダウン
- 仮想ネットワークの管理
- 仮想マシンの [ディスクイメージ](#) のロード

virsh ユーティリティーは、仮想化管理スクリプトの作成に最適です。root 権限を持たないユーザーも **virsh** を使用できますが、読み取り専用モードで使用するようになります。

virsh の使用

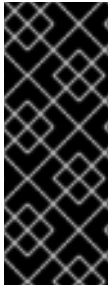
virsh ユーティリティーは、標準のコマンドライン入力で使用できるだけでなく、対話式のシェルとしても使用できます。シェルモードでは、コマンドプレフィックスの **virsh** が不要になり、ユーザーは常に root として登録されます。次の例では、**virsh hostname** コマンドを使用してハイパーバイザーのホスト名を表示します。最初は標準モードで、次に対話モードで使用します。

```
$ virsh hostname
localhost.localdomain

$ virsh
Welcome to virsh, the virtualization interactive terminal.

Type: 'help' for help with commands
      'quit' to quit

virsh # hostname
localhost.localdomain
```



重要

非 root ユーザーとして **virsh** を使用すると、非特権 [libvirt セッション](#) に入ります。この場合、root によって作成されたゲストやその他の仮想化された要素を表示したり操作したりすることはできません。

要素への読み取り専用アクセスを取得するには、**-c qemu:///system** オプションを指定して **virsh** を使用します。

virsh のヘルプの取得

すべての Linux bash コマンドと同様に、**virsh** のヘルプは **man virsh** コマンドまたは **--help** オプションを使用して取得できます。さらに、**virsh help** コマンドを使用して、特定の **virsh** コマンドのヘルプテキストを表示したり、キーワードを使用して、特定のグループに属するすべての **virsh** コマンドを一覧表示したりできます。

virsh コマンドグループとそれぞれのキーワードは次のとおりです。

- ゲスト管理 - キーワード **domain**
- ゲスト監視 - キーワード **monitor**
- ホストとハイパーバイザーの監視と管理 - キーワード **host**
- ホストシステムのネットワークインターフェイス管理 - キーワード **interface**
- 仮想ネットワーク管理 - キーワード **network**
- ネットワークフィルター管理 - キーワード **filter**
- ノードデバイス管理 - キーワード **nodedev**
- パスフレーズや暗号化キーなどのシークレットの管理 - キーワード **secret**
- スナップショット管理 - キーワード **snapshot**
- ストレージプール管理 - キーワード **pool**
- ストレージボリューム管理 - キーワード **volume**
- 一般的な **virsh** の使用法 - キーワード **virsh**

次の例では、ゲスト仮想マシンの名前を変更する方法を知る必要があります。**virsh help** を使用して、まず使用する適切なコマンドを見つけ、次にその構文を理解します。最後に、コマンドを使用して *Fontaine* というゲストの名前を *Atlas* に変更します。

例4.1 キーワードを使用してすべてのコマンドのヘルプを一覧表示する方法

```
# virsh help domain
Domain Management (help keyword 'domain'):
  attach-device      attach device from an XML file
  attach-disk        attach disk device
  [...]
  domname            convert a domain id or UUID to domain name
  domrename          rename a domain
  [...]
# virsh help domrename
```

```

NAME
  domrename - rename a domain

SYNOPSIS
  domrename <domain> <new-name>

DESCRIPTION
  Rename an inactive domain.

OPTIONS
  [--domain] <string> domain name, id or uuid
  [--new-name] <string> new domain name

# virsh domrename --domain Fontaine --new-name Atlas
Domain successfully renamed

```



注記

virsh を使用した仮想マシンの管理に関する詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#) を参照してください。

4.1.2. virt-install

virt-install は、新しい仮想マシンを作成するための CLI ユーティリティです。シリアルコンソール、SPICE、または VNC クライアント/サーバーペアグラフィックを使用したテキストベースインストールとグラフィカルインストールの両方をサポートしています。インストールメディアは、ローカルにあるものか、NFS、HTTP、または FTP サーバーにあるリモートのものを使用できます。このツールは、無人で実行し、キックスタート方式を使用してゲストを準備するように設定することもできるため、インストールを簡単に自動化できます。このツールは `virt-install` パッケージに含まれています。



重要

非 root ユーザーとして **virt-install** を使用すると、非特権 [libvirt セッション](#) に入ります。この場合、作成したゲストは自分だけにのみ表示され、root によって作成されたゲストが持つ特定の機能にはアクセスできません。



注記

virt-install の使用に関する詳細は、[『Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド』](#) を参照してください。

4.1.3. virt-xml

virt-xml は、ドメイン XML ファイルを編集するためのコマンドラインユーティリティです。XML 設定を正常に変更するには、ゲストの名前、XML アクション、および変更内容をコマンドに含める必要があります。

たとえば、次の例では、ゲストのブート設定に関連するサブオプションを一覧表示し、**example_domain** ゲストでブートメニューをオンにします。

```

# virt-xml boot=?
--boot options:
  arch

```

```

cdrom
[...]
menu
network
nvram
nvram_template
os_type
smbios_mode
uefi
useserial
# virt-xml example_domain --edit --boot menu=on
Domain 'example_domain' defined successfully.

```

コマンドの呼び出しごとに、1つのドメイン XML ファイルに対して1つのアクションを実行できることに注意してください。



注記

このツールは `virt-install` パッケージに含まれています。**virt-xml** の使用に関する詳細は、**virt-xml** の man ページを参照してください。

4.1.4. guestfish

guestfish は、仮想マシンのディスクイメージを調査および変更するためのコマンドラインユーティリティです。libguestfs ライブラリーを使用し、**libguestfs** API によって提供されるすべての機能を公開します。

guestfish の使用

guestfish ユーティリティは、標準のコマンドライン入力モードで使用できるだけでなく、対話式のシェルとしても使用できます。シェルモードでは、コマンドプレフィックスの **guestfish** が不要になり、ユーザーは常に `root` として登録されます。次の例では、**guestfish** を使用して `testquest` 仮想マシン上のファイルシステムを表示します。最初は標準モードで、次に対話モードで使します。

```

# guestfish domain testquest : run : list-file systems
/dev/sda1: xfs
/dev/rhel/root: xfs
/dev/rhel/swap: swap
# guestfish

```

Welcome to guestfish, the guest filesystem shell for
editing virtual machine filesystems and disk images.

```

Type: 'help' for help on commands
      'man' to read the manual
      'quit' to quit the shell

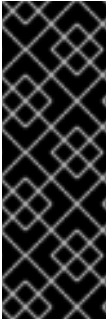
```

```

><fs> domain testquest
><fs> run
><fs> list-file systems
/dev/sda1: xfs
/dev/rhel/root: xfs
/dev/rhel/swap: swap

```

さらに、**guestfish** は、自動化のために [bash スクリプト](#) で使用することもできます。



重要

非 root ユーザーとして **guestfish** を使用すると、非特権 **libvirt セッション** に入ります。この場合、root によって作成されたゲストのディスクイメージを表示したり操作したりすることはできません。

これらのディスクイメージへの読み取り専用アクセスを取得するには、**-ro -c qemu:///system** オプションを指定して **guestfish** を使用します。さらに、ディスクイメージファイルの読み取り権限が必要です。

guestfish のヘルプの取得

すべての Linux bash コマンドと同様に、**guestfish** のヘルプは **man guestfish** コマンドまたは **--help** オプションを使用して取得できます。さらに、**guestfish help** コマンドを使用して、特定の **guestfish** コマンドに関する詳細情報を表示できます。次の例では、**guestfish add** コマンドに関する情報を表示します。

```
$ guestfish help add
NAME
    add-drive - add an image to examine or modify

SYNOPSIS
    add-drive filename [readonly:true|false] [format:...] [iface:...] [name:...] [label:...] [protocol:...] [server:...]
    [username:...] [secret:...] [cachemode:...] [discard:...] [copyonread:true|false]

DESCRIPTION
    This function adds a disk image called filename to the handle. filename
    may be a regular host file or a host device.
    [...]
```



注記

guestfish の詳細は、『[Red Hat Enterprise Linux 7 仮想化の導入および管理ガイド](#)』を参照してください。

4.2. デモ: コマンドラインユーティリティーを使用したゲストの作成と管理

このセクションでは、CLI でどのように仮想化タスクを実行するかを示すために、新しいゲスト仮想マシンを作成し、そこに OS をインストールし、その後 CLI コマンドを使用してゲストを操作および管理するデモを提供します。

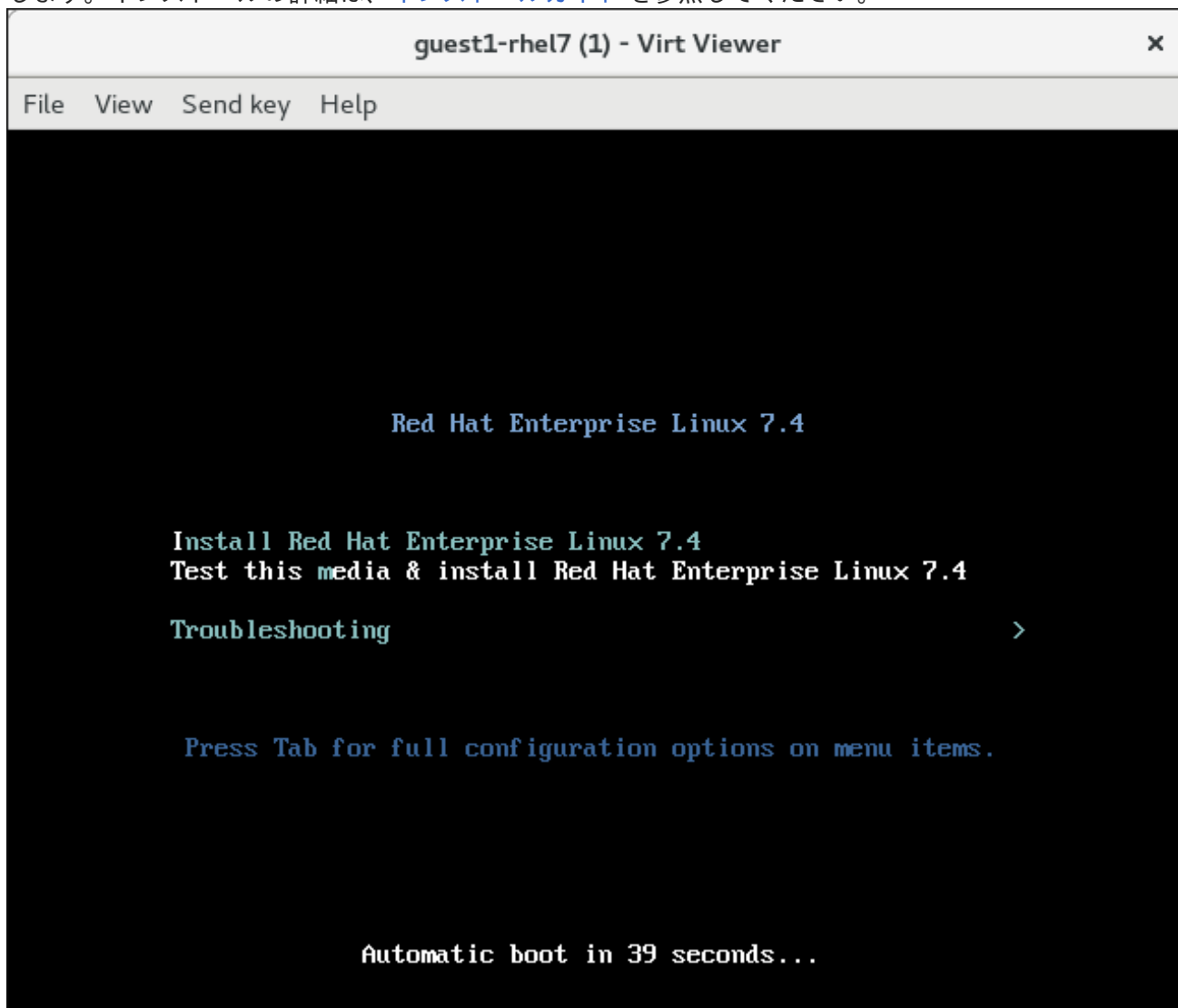
4.2.1. インストール

ここでは、**guest1-rhel7** という名前の新しいゲストを作成し、Red Hat Enterprise Linux 7 Workstation の ISO イメージから OS のインストールを開始します。このイメージは、[カスタマーポータル](#) で入手できます。この例では、現在、**~/Downloads/** フォルダにあります。ゲストには、2 つの仮想 CPU、2048 MB の RAM、および 8 GB のディスク領域を割り当てます。

```
# virt-install --name guest1-rhel7 --memory 2048 --vcpus 2 --disk size=8 --cdrom
/home/username/Downloads/rhel-workstation-7.4-x86_64-dvd.iso --os-variant rhel7

Starting install...
Allocating 'guest1-rhel7.qcow2' | 8.0 GB 00:00:00
```


これを実行すると、**virt-viewer** アプリケーションでグラフィカルな Anaconda インストーラーが起動します。インストールの詳細は、[インストールガイド](#) を参照してください。



注記

グラフィカルインターフェイスにアクセスできないホストシステムでは、次のような **virt-install** コマンドを使用して、[テキストベースの Anaconda](#) を使用してゲスト OS をインストールできます。

```
# virt-install -name rhel7anaconda-guest -r 1024 --
location=/home/jherrman/Downloads/rhel-workstation-7.4-x86_64-dvd.iso --disk
size=8 --nographics --extra-args="console=tty0 console=ttyS0,115200n8"
```

インストールが正常に完了すると、コマンドラインに次のように表示されます。

```
Domain creation completed.
Restarting guest.
```

これで、ゲストに望ましい設定を行えるようになりました。ただし、ゲストの設定を安全に管理するために、まずゲストをシャットダウンすることをお勧めします。

```
# virsh shutdown guest1-rhel7
Domain guest1-rhel7 is being shutdown
```


4.2.2. デバイスの接続

ホストに接続された USB デバイス (この例では Samsung 携帯電話) をゲストが検出して使用できるようにするには、まずホストで **lsusb** コマンドを使用してデバイスの ID を取得します。

```
# lsusb

[...]
Bus 003 Device 007: ID 04e8:6860 Samsung Electronics Co., Ltd Galaxy (MTP)
```

次に、ホストで任意のテキストエディターを使用して、デバイスの XML ファイル (この例では **samsung_USB_device.xml**) を作成し、ベンダー ID と製品 ID を入力します。

```
# vim samsung_USB_device.xml

<hostdev mode='subsystem' type='usb' managed='yes'>
  <source>
    <vendor id='0x04e8'>
    <product id='0x6860'>
  </source>
</hostdev>
```

最後に、**virsh attach-device** コマンドを使用してデバイスをゲストに接続します。

```
# virsh attach-device guest1-rhel7 --file samsung_USB_device.xml --config
Device attached successfully
```



注記

実行中のゲストにデバイスを接続することもできます。これを行うには、**--live** オプションを使用します。

4.2.3. ゲストの操作

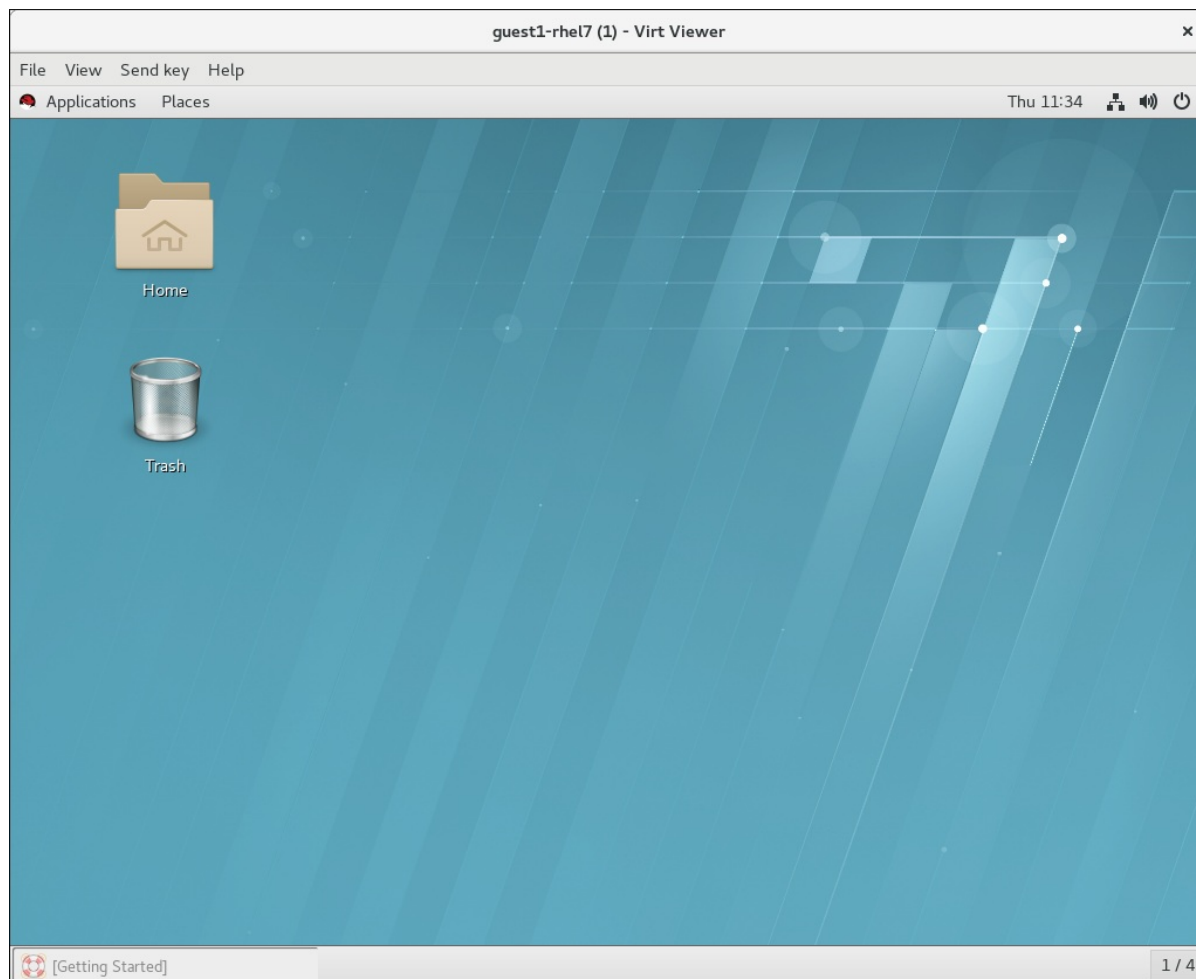
guest1-rhel7 ゲストの使用を開始するには、まずゲストを起動します。

```
# virsh start guest1-rhel7
Domain guest1-rhel7 started
```

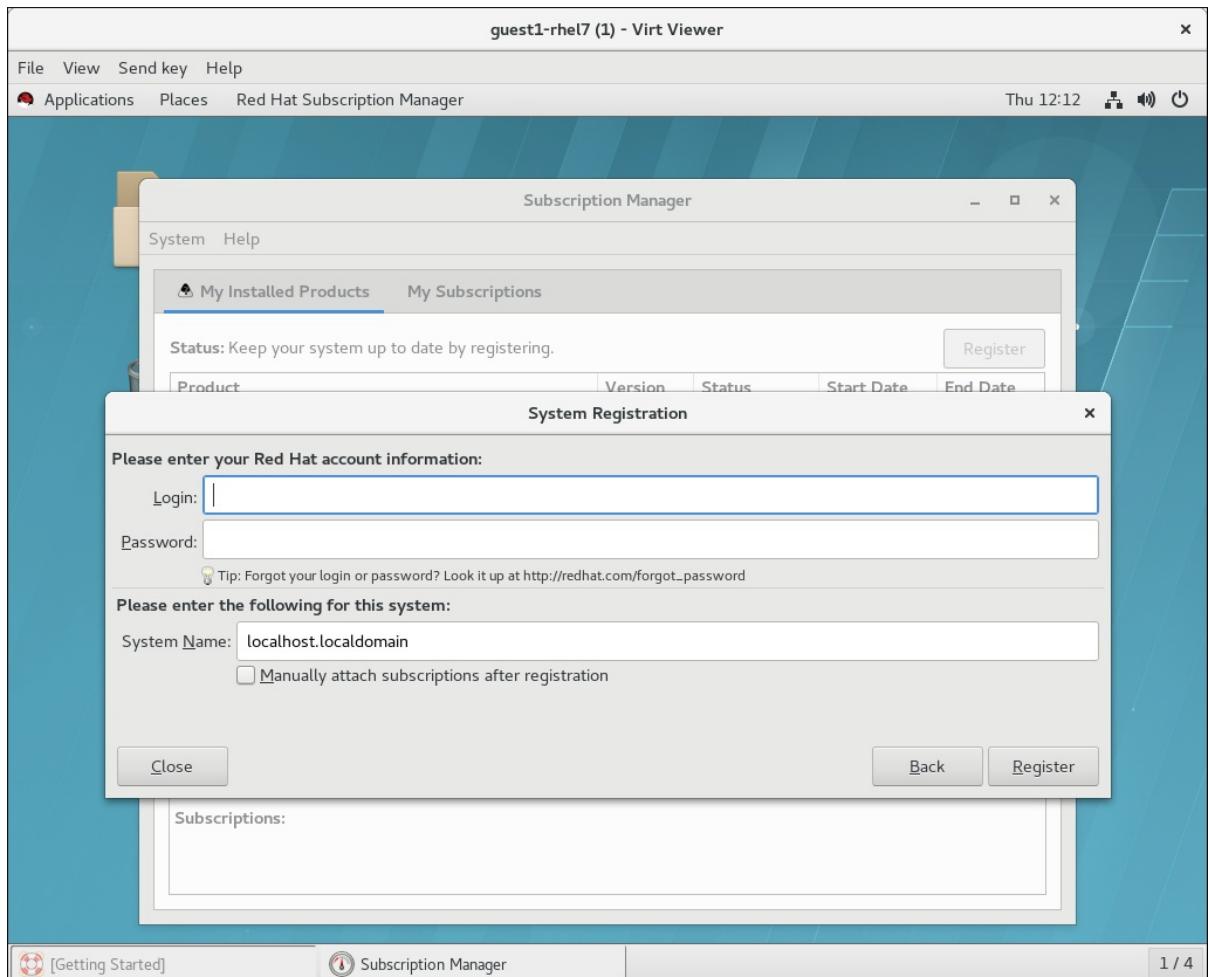
ホストシステムにグラフィカルディスプレイがあるかどうかに応じて、**virt-viewer** アプリケーションまたは SSH シェルを使用してゲストを操作できます。

- グラフィカルディスプレイを備えたシステムでは、**virt-viewer** を使用します。

```
# virt-viewer guest1-rhel7
```



その後、物理マシンの OS GUI と同様に、**virt-viewer** ウィンドウで画面出力を操作できます。たとえば、[Subscription Manager](#) アプリケーションを使用して、Red Hat Enterprise Linux ゲスト OS を登録できます。



- ホストまたはゲストがテキストのみのインターフェースを備えている場合は、SSH を使用します。これには、ゲストの IP アドレスを知っている必要があります。IP アドレスがわからない場合は、**virsh domifaddr** コマンドを使用して取得できます。

```
# virsh domifaddr guest1-rhel7
Name      MAC address      Protocol  Address
-----
vnet0     52:54:00:65:29:21    ipv4      10.34.3.125/24
# ssh root@10.34.3.125
root@10.34.3.125's password:
Last login: Wed Jul 19 18:27:10 2017 from 192.168.122.1
[root@localhost ~]#
```



注記

virsh domifaddr が機能するには、ゲストが実行されていて、ネットワーク上でアクセス可能である必要があります。また、[QEMU ゲストエージェント](#) をアクティブにする必要がある場合があります。

その後、ゲストマシン上のターミナルを使用するときと同じように、ホストターミナルを操作できます。たとえば、[subscription-manager](#) ユーティリティを使用して、Red Hat Enterprise Linux ゲスト OS を登録できます。

```
[root@localhost ~]# subscription-manager register
Registering to: subscription.rhsm.redhat.com:443/subscription
Username: username@sample.com
```

Password:

The system has been registered with ID: 30b5e666-67f9-53bb-4b90-c2a88e5be789

4.2.4. 診断

ゲストの状態に関する一般的な情報を表示します。

```
# virsh dominfo guest1-rhel7
Id:          1
Name:        guest1-rhel7
UUID:        ec0c0122-fb63-4a54-b602-5cf84f5e2dfd
OS Type:     hvm
State:       running
CPU(s):      2
CPU time:    33.4s
Max memory:  2097152 KiB
Used memory: 2097152 KiB
Persistent:  yes
Autostart:   disable
Managed save: no
Security model: selinux
Security DOI: 0
Security label: unconfined_u:unconfined_r:svirt_t:s0:c102,c792 (enforcing)
```

4.2.5. スナップショットの作成

ゲストの状態をバックアップするには、**virsh snapshot-create** コマンドを使用できます。

```
# virsh snapshot-create guest1-rhel7
Domain snapshot 1500563241 created
```

現在のスナップショットと各ゲストの XML 設定を表示できます。

```
# virsh snapshot-list guest1-rhel7
Name                Creation Time          State
-----
1500563241          2017-07-20 17:07:21 +0200 shutoff

# virsh snapshot-dumpxml guest1-rhel7 1500563241
<domainsnapshot>
  <name>1500563241</name>
  <state>shutoff</state>
  <creationTime>1500563241</creationTime>
  <memory snapshot='no'>
  <disks>
    <disk name='vda' snapshot='internal'>
  [...]
```

このスナップショットを後でロードして、ゲストをスナップショットに保存された状態に戻すことができます。

```
# virsh snapshot-revert guest1-rhel7 --snapshotname 150056324
```

第5章 VIRTUAL MACHINE MANAGER の使用

Virtual Machine Manager (**virt-manager** と呼ばれる) は、ゲスト仮想マシンを作成および管理するグラフィカルツールです。この章では、Virtual Machine Manager とその実行方法について説明します。



注記

Virtual Machine Manager は、グラフィカルインターフェイスを備えたシステムでのみ実行できます。

Virtual Machine Manager の使用に関する詳細は、他の [Red Hat Enterprise Linux 仮想化ガイド](#) を参照してください。

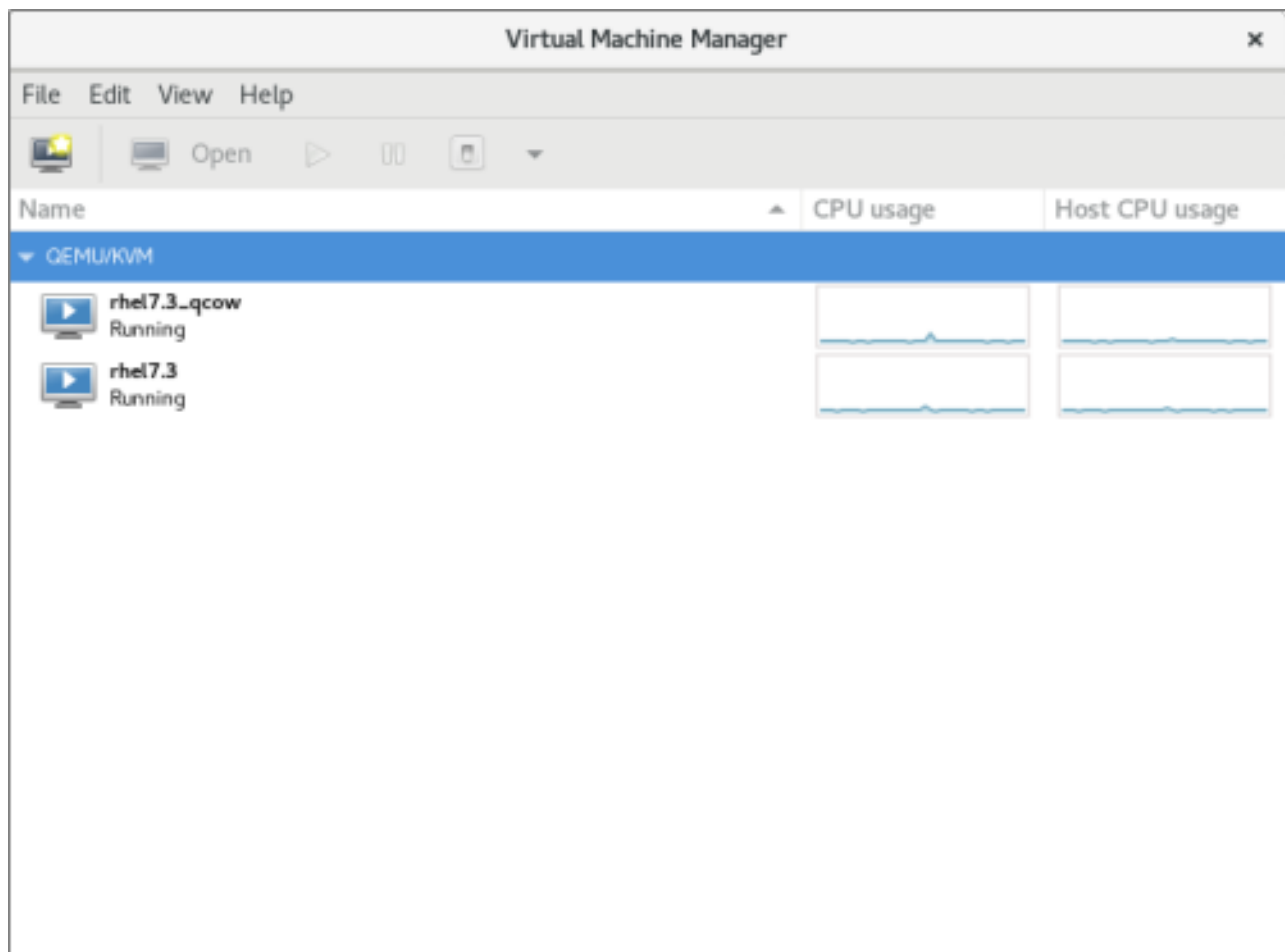
5.1. VIRTUAL MACHINE MANAGER の実行

Virtual Machine Manager を実行するには、アプリケーションの一覧でそれを選択するか、次のコマンドを使用します。

```
# virt-manager
```

Virtual Machine Manager がメインウィンドウに開きます。

図5.1 Virtual Machine Manager





注記

virt-manager の実行に失敗した場合は、virt-manager パッケージがインストールされていることを確認してください。virt-manager パッケージのインストールに関する詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの [仮想化パッケージのインストール](#) を参照してください。

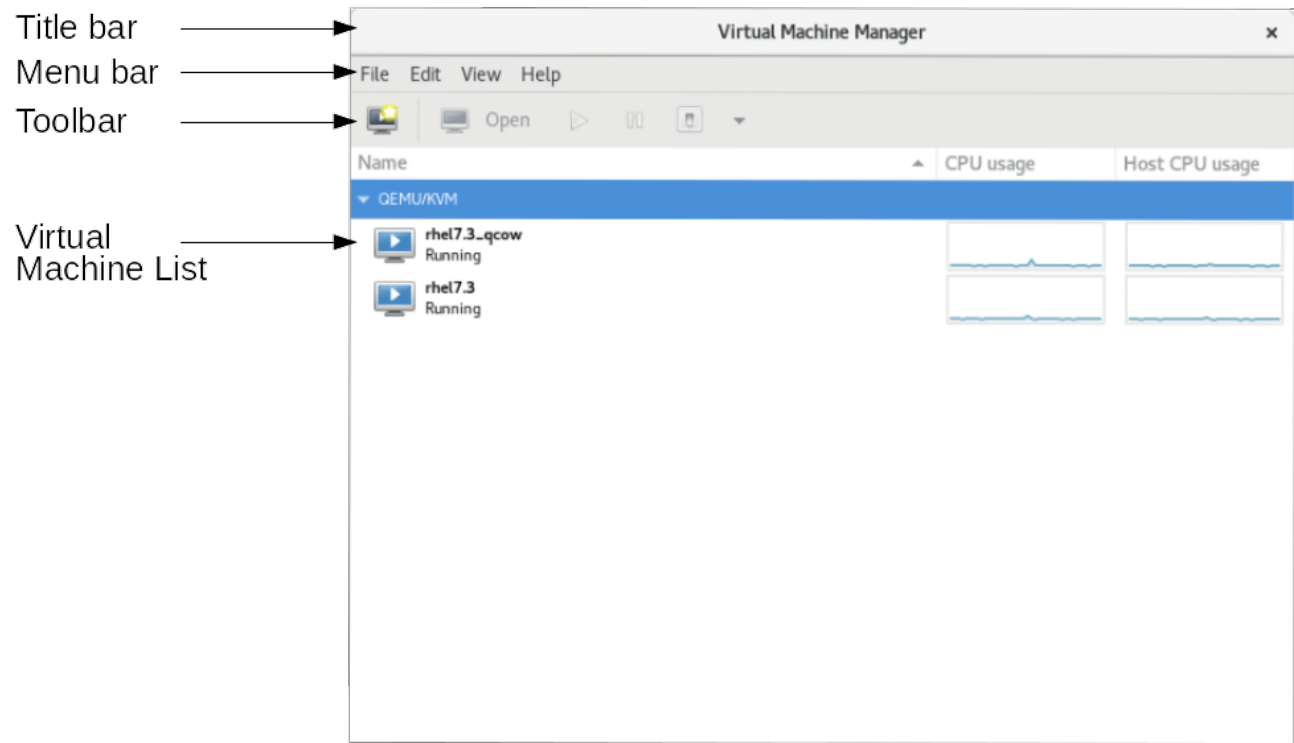
5.2. VIRTUAL MACHINE MANAGER のインターフェイス

次のセクションでは、Virtual Machine Manager のユーザーインターフェイスに関する情報を提供します。ユーザーインターフェイスには、[Virtual Machine Manager のメインウィンドウ](#) と [仮想マシンウィンドウ](#) が含まれます。

5.2.1. Virtual Machine Manager のメインウィンドウ

次の図は、Virtual Machine Manager のメインウィンドウインターフェイスを示しています。

図5.2 Virtual Machine Manager ウィンドウ



Virtual Machine Manager のメインウィンドウのタイトルバーには **Virtual Machine Manager** と表示されます。

5.2.1.1. メインウィンドウのメニューバー

次の表に、Virtual Machine Manager のメインウィンドウメニューのエントリーを示します。

表5.1 Virtual Machine Manager のメインウィンドウのメニュー

メニュー名	メニュー項目	Description
-------	--------	-------------

メニュー名	メニュー項目	Description
ファイル	Add Connection	Add Connection ダイアログを開き、ローカルまたはリモートのハイパーバイザーに接続します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの リモート接続の追加 を参照してください。
	New Virtual Machine	New VM ウィザードを開き、新しいゲスト仮想マシンを作成します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの virt-manager を使用したゲストの作成 を参照してください。
	Close	仮想マシンウィンドウを閉じずに、Virtual Machine Manager ウィンドウを閉じます。実行中の仮想マシンは停止されません。
	Exit	Virtual Machine Manager とすべての仮想マシンウィンドウを閉じます。実行中の仮想マシンは停止されません。
Edit	Connection Details	選択した接続の Connection Details ウィンドウを開きます。
	Virtual Machine Details	選択した仮想マシンの仮想マシンウィンドウを開きます。詳細は、 仮想マシンペイン を参照してください。
	Delete	選択した接続または仮想マシンを削除します。
	Preferences	Virtual Machine Manager オプションを設定するための Preferences ダイアログボックスを開きます。
View	Graph <ul style="list-style-type: none"> ● Guest CPU Usage ● Host CPU Usage ● Memory Usage ● Disk I/O ● Network I/O 	Virtual Machine Manager のメインウィンドウで、仮想マシンの選択したパラメーターの表示を切り替えます。
Help	About	Virtual Machine Manager に関する情報を含む About ウィンドウを表示します。

5.2.1.2. メインウィンドウのツールバー

次の表に、Virtual Machine Manager のメインウィンドウのアイコンを示します。







表5.2 Virtual Machine Manager のメインウィンドウのツールバー

アイコン	Description
	New VM ウィザードを開き、新しいゲスト仮想マシンを作成します。
	選択した仮想マシンの仮想マシンウィンドウを開きます。
	選択した仮想マシンを起動します。
	選択した仮想マシンを一時停止します。
	選択した仮想マシンを停止します。
	<p>選択した仮想マシンで実行する次のいずれかのアクションを選択するためのメニューを開きます。</p> <ul style="list-style-type: none"> ● Reboot - 選択した仮想マシンを再起動します。 ● Shut Down - 選択した仮想マシンをシャットダウンします。 ● Force Reset - 選択した仮想マシンを強制的にシャットダウンして再起動します。 ● Force Off - 選択した仮想マシンを強制的にシャットダウンします。 ● Save - 選択した仮想マシンの状態をファイルに保存します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの ゲスト仮想マシンの設定の保存 を参照してください。

5.2.1.3. 仮想マシンリスト

仮想マシンリストには、Virtual Machine Manager が接続されている仮想マシンのリストが表示されます。リスト内の仮想マシンは、接続ごとにグループ化されています。表の列のヘッダーをクリックすると、リストを並べ替えることができます。

図5.3 仮想マシンリスト

Name	CPU usage	Host CPU usage
▼ QEMU/KVM		
 rhel7.3 Running		
 rhel7.3_qcow Running		

仮想マシンリストには、各仮想マシンで使用されているリソースに関する情報がグラフで表示されます。**Edit** メニューの **Preferences** ダイアログの **Polling** タブから、リソースを表示できます。以下は、仮想マシンリストに表示できるリソースのリストです。

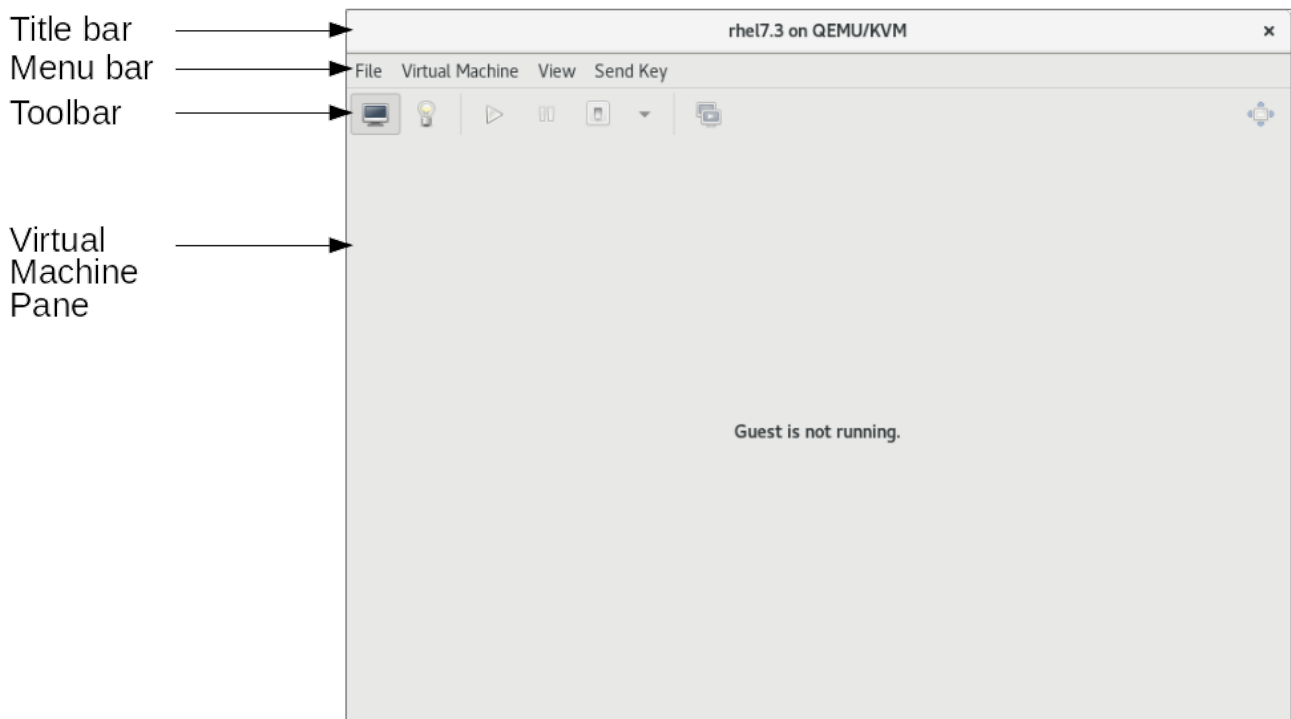
- CPU の使用率
- ホストの CPU 使用率
- メモリー使用率
- ディスク I/O
- ネットワーク I/O

View メニューの **Graph** メニュー項目を使用して、表示するリソースを選択できます。

5.2.2. 仮想マシンウィンドウ

このセクションでは、仮想マシンウィンドウのインターフェイスに関する情報を提供します。

図5.4 仮想マシンウィンドウ



タイトルバーには、仮想マシンの名前とそれが使用する接続が表示されます。

5.2.2.1. 仮想マシンウィンドウのメニューバー

次の表に、仮想マシンウィンドウメニューのエントリーを示します。

表5.3 仮想マシンウィンドウのメニュー

メニュー名	メニュー項目	Description
ファイル	View Manager	Virtual Machine Manager のメインウィンドウを開きます。

メニュー名	メニュー項目	Description
	Close	仮想マシンを停止せずに、仮想マシンウィンドウのみを閉じます。
	Exit	すべての Virtual Machine Manager ウィンドウを閉じます。実行中の仮想マシンは停止されません。
Virtual Machine	Run	仮想マシンを実行します。このオプションは、仮想マシンが実行されていない場合にのみ使用できます。
	Pause	仮想マシンを一時停止します。このオプションは、仮想マシンがすでに実行されている場合にのみ使用できます。
	Shut Down	仮想マシンで実行する次のいずれかのアクションを選択するためのメニューを開きます。 <ul style="list-style-type: none"> ● Reboot - 仮想マシンを再起動します。 ● Shut Down - 仮想マシンをシャットダウンします。 ● Force Reset - 仮想マシンを強制的にシャットダウンして再起動します。 ● Force Off - 仮想マシンを強制的にシャットダウンします。 ● Save - 仮想マシンの状態をファイルに保存します。
	Clone	仮想マシンのクローンを作成します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの virt-manager を使用したゲストのクローン作成 を参照してください。
	Migrate	Migrate the virtual machine ダイアログを開き、仮想マシンを別のホストに移行します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの virt-manager を使用した移行 を参照してください。
	Delete	仮想マシンを削除します。
	Take Screenshot	仮想マシンコンソールのスクリーンショットを撮ります。
	Redirect USB Device	Select USB devices for redirection ダイアログを開き、リダイレクトする USB デバイスを選択します。詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの USB リダイレクト を参照してください。
View	Console	仮想マシンペインでコンソール画面を開きます。
	Details	仮想マシンペインで詳細画面を開きます。詳細は、 仮想マシンの詳細ウィンドウ を参照してください。

メニュー名	メニュー項目	Description
	Snapshots	仮想マシンペインでスナップショット画面を開きます。詳細は、 スナップショットウィンドウ を参照してください。
	Fullscreen	仮想マシンコンソールをフルスクリーンモードで表示します。
	Resize to VM	仮想マシンに設定されているサイズと解像度に合わせて全画面表示のサイズを変更します。
	Scale Display	<p>次のサブメニュー項目の選択に基づいて、仮想マシンのディスプレイをスケーリングします。</p> <ul style="list-style-type: none"> ● Always - 仮想マシンのディスプレイは、常に仮想マシンウィンドウに合わせてスケーリングされます。 ● Only when Fullscreen - 仮想マシンウィンドウがフルスクリーンモードの場合、仮想マシンのディスプレイは仮想マシンウィンドウに合わせてスケーリングされます。 ● Never - 仮想マシンのディスプレイは、仮想マシンウィンドウに合わせてスケーリングされません。 ● Auto resize VM with window - 仮想マシンウィンドウのサイズが変更されると、仮想マシンのディスプレイのサイズが自動的に変更されます。
	Text Consoles	リストで選択した仮想マシンのディスプレイを表示します。仮想マシンのディスプレイの例には、 Serial 1 や Graphical Console Spice などがあります。
	Toolbar	仮想マシンウィンドウのツールバーの表示を切り替えます。

メニュー名	メニュー項目	Description
Send Key	Ctrl+Alt+Backspace	選択したキーを仮想マシンに送信します。
	Ctrl+Alt+Delete	
	Ctrl+Alt+F1	
	Ctrl+Alt+F2	
	Ctrl+Alt+F3	
	Ctrl+Alt+F4	
	Ctrl+Alt+F5	
	Ctrl+Alt+F6	
	Ctrl+Alt+F7	
	Ctrl+Alt+F8	
	Ctrl+Alt+F9	
	Ctrl+Alt+F10	
	Ctrl+Alt+F11	
	Ctrl+Alt+F12	
	Ctrl+Alt+Printscreen	

5.2.2.2. 仮想マシンウィンドウのツールバー

次の表に、仮想マシンウィンドウのアイコンを示します。

表5.4 仮想マシンウィンドウのツールバー

アイコン	Description
	仮想マシンのグラフィカルコンソールを表示します。
	仮想マシンの詳細ペインを表示します。
	選択した仮想マシンを起動します。
	選択した仮想マシンを一時停止します。

アイコン	Description
	選択した仮想マシンを停止します。
	選択した仮想マシンで実行する次のいずれかのアクションを選択するためのメニューを開きます。 <ul style="list-style-type: none">● Reboot - 選択した仮想マシンを再起動します。● Shut Down - 選択した仮想マシンをシャットダウンします。● Force Reset - 選択した仮想マシンを強制的にシャットダウンして再起動します。● Force Off - 選択した仮想マシンを強制的にシャットダウンします。● Save - 選択した仮想マシンの状態をファイルに保存します。
	仮想マシンペインでスナップショット画面を開きます。
	仮想マシンコンソールをフルスクリーンモードで表示します。

5.2.2.3. 仮想マシンペイン

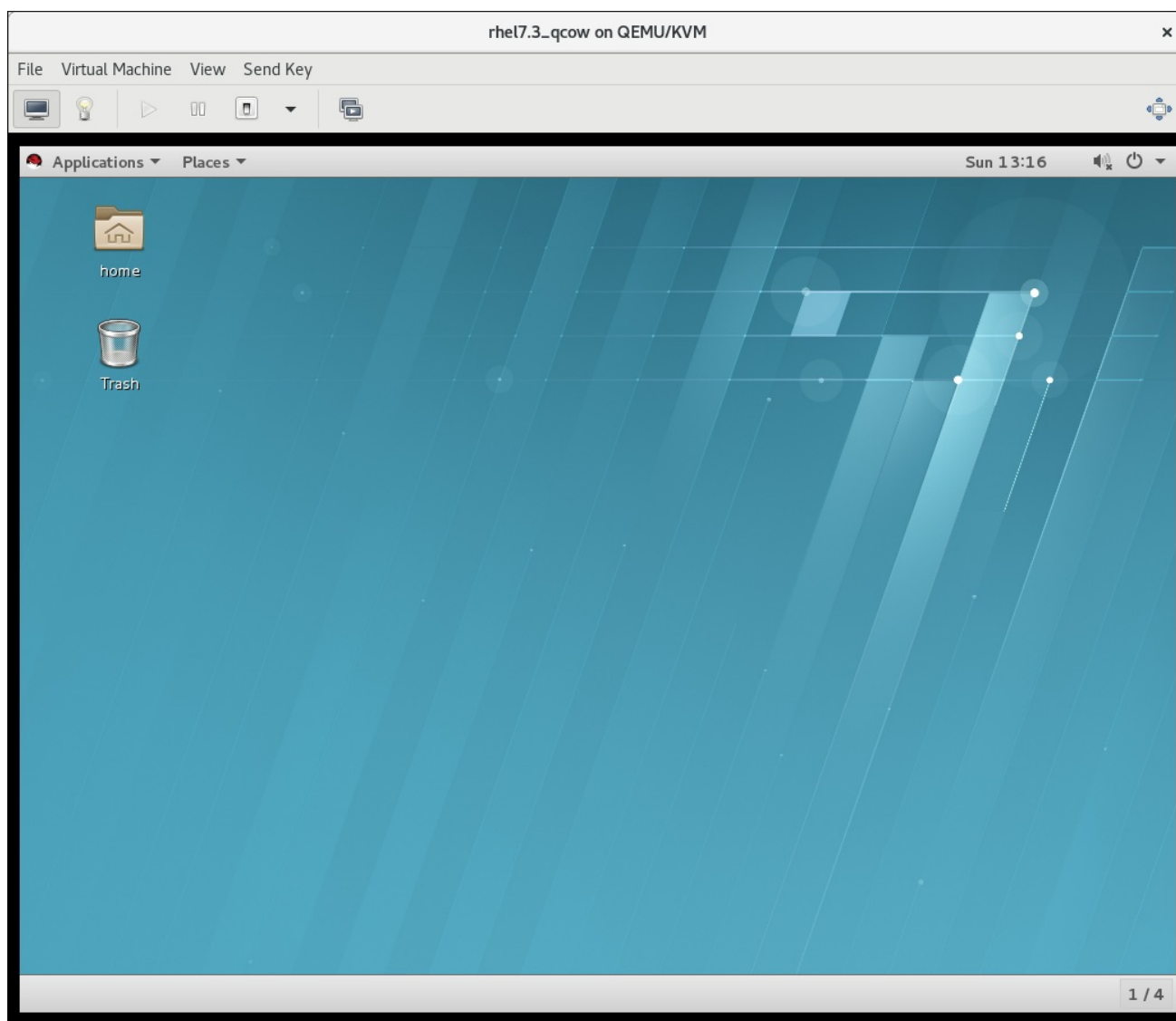
仮想マシンペインには、次のいずれかが表示されます。

- [仮想マシンコンソール](#)
- [仮想マシンの詳細ウィンドウ](#)
- [スナップショットウィンドウ](#)

仮想マシンコンソール

仮想マシンのコンソールには仮想マシンのグラフィカル出力が表示されます。

図5.5 仮想マシンコンソール

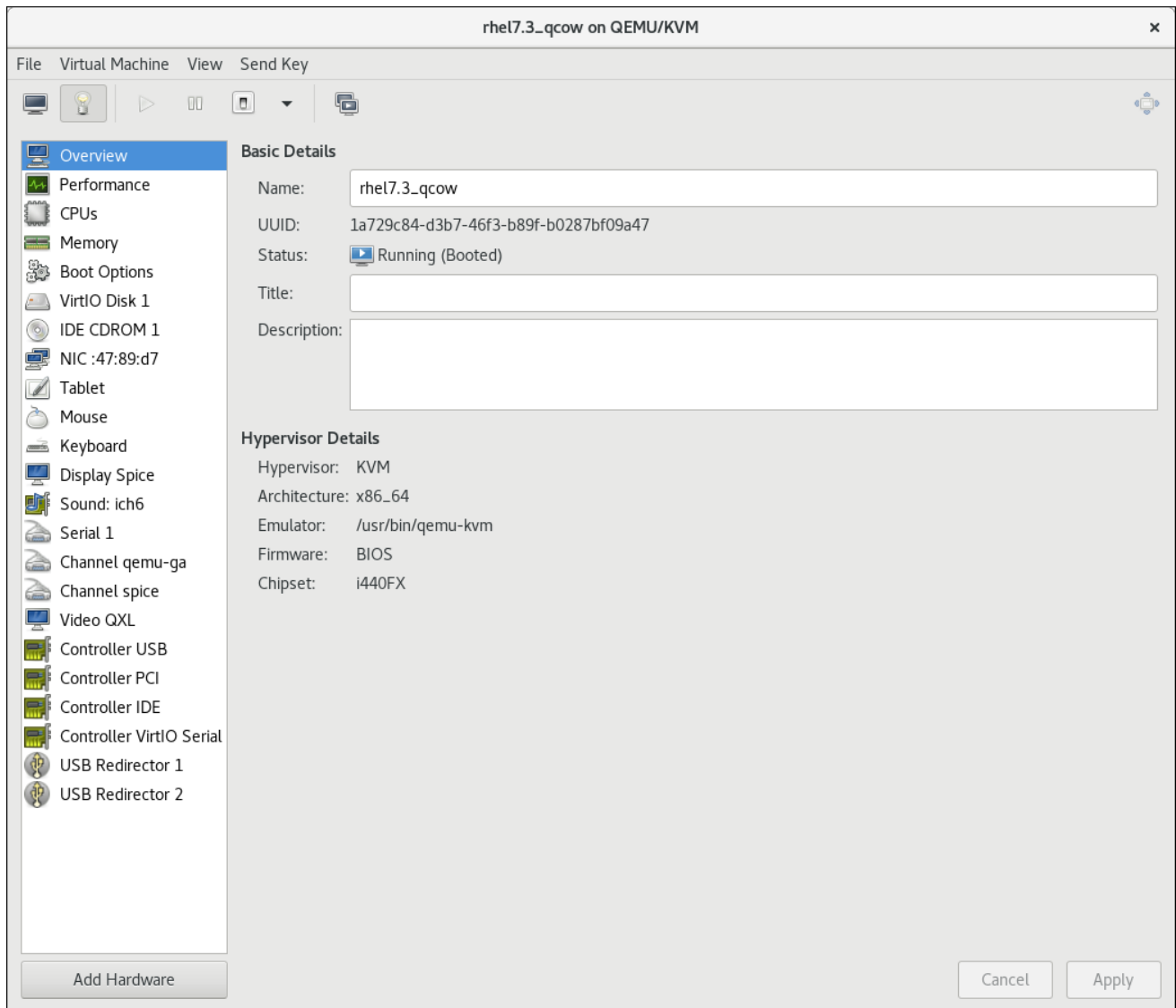


実際のマシンの場合と同じように、マウスとキーボードを使用して仮想マシンのコンソールと相互作用できます。仮想マシンのコンソールには、仮想マシンで実行されているアクティビティが表示されます。

仮想マシンの詳細ウィンドウ

仮想マシンの詳細ウィンドウには、仮想マシン、そのハードウェア、および設定に関する詳細情報が表示されます。

図5.6 仮想マシンの詳細ウィンドウ



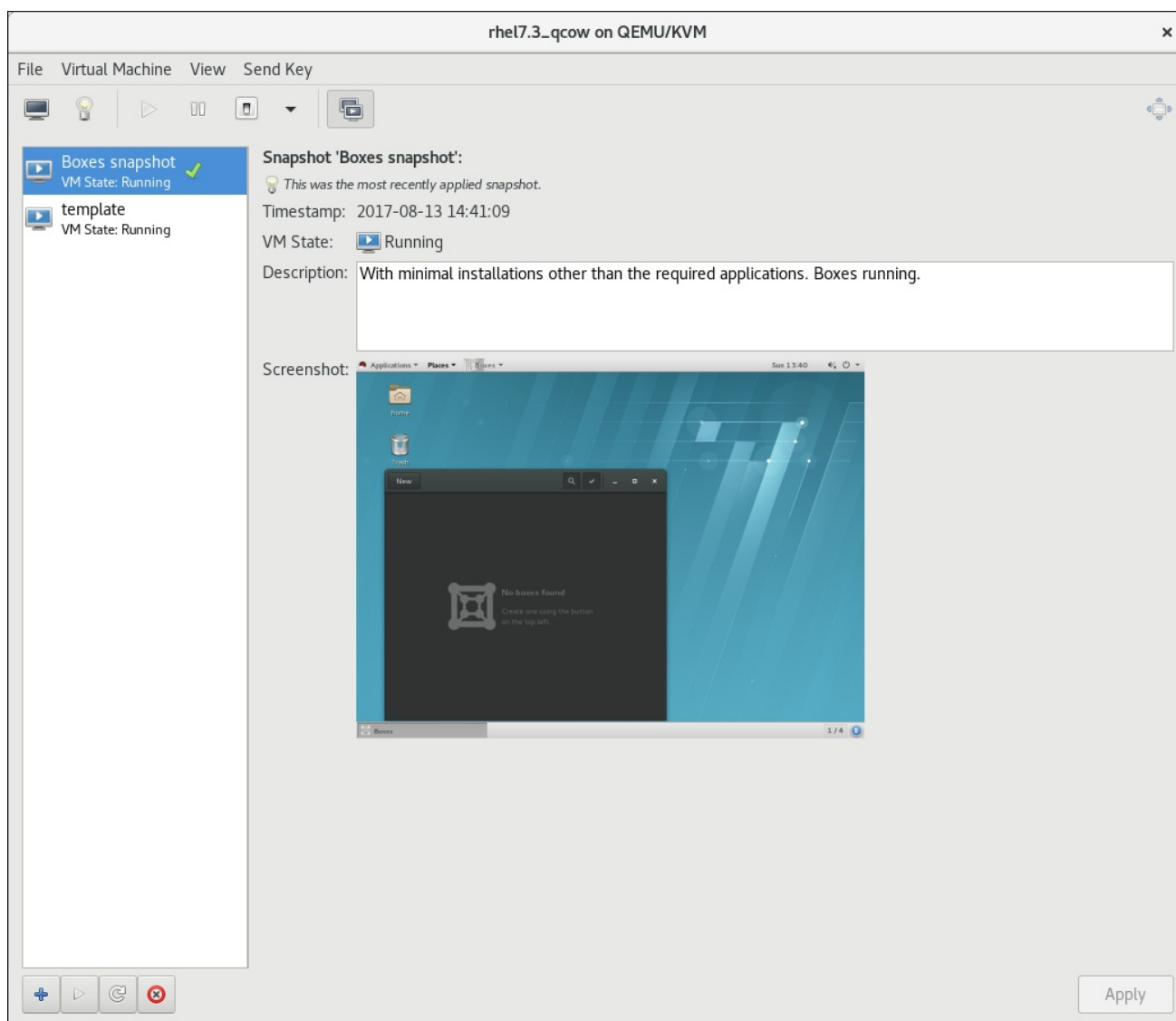
仮想マシンの詳細ウィンドウには、仮想マシンパラメーターのリストが含まれます。リスト内のパラメーターを選択すると、選択したパラメーターに関する情報が仮想マシンの詳細ウィンドウの右側に表示されます。仮想マシンの詳細ウィンドウを使用して、ハードウェアを追加および設定することもできます。

仮想マシンの詳細ウィンドウの詳細は、Red Hat Enterprise Linux 仮想化の導入および管理ガイドの [仮想ハードウェアの詳細ウィンドウ](#) を参照してください。

スナップショットウィンドウ

仮想マシンのスナップショットウィンドウには、仮想マシン用に作成されたスナップショットのリストが表示されます。

図5.7 仮想マシンのスナップショットウィンドウ



仮想マシンのスナップショットウィンドウには、仮想マシン用に保存されたスナップショットのリストが含まれます。リスト内のスナップショットを選択すると、状態、説明、スクリーンショットなど、選択したスナップショットの詳細が仮想マシンのスナップショットウィンドウの右側に表示されます。仮想マシンのスナップショットウィンドウを使用して、スナップショットを追加、削除、および実行できます。

スナップショットの管理に関する詳細は、Red Hat Enterprise Linux 7 仮想化の導入および管理ガイドの [スナップショットの管理](#) を参照してください。

付録A 更新履歴

改訂 1.0-56 7.6 GA リリースのバージョン	Thu May 23 2019	Jiri Herrmann
改訂 1.0-55 7.6 GA リリースのバージョン	Thu Oct 25 2018	Jiri Herrmann
改訂 1.0-53 7.6 ベータ版公開用バージョン	Thu Aug 5 2018	Jiri Herrmann
改訂 1.0-52 7.5 GA 公開用バージョン	Thu Apr 5 2018	Jiri Herrmann
改訂 1.0-49 7.4 GA 公開用バージョン	Thu Jul 27 2017	Jiri Herrmann
改訂 1.0-46 7.3 GA 公開用バージョン	Mon Oct 17 2016	Jiri Herrmann
改訂 1.0-44 いくつかのバグ修正に対応し、ガイドを再公開	Mon Dec 21 2015	Laura Novich
改訂 1.0-43 改訂履歴の整理	Thu Oct 08 2015	Jiri Herrmann
改訂 1.0-42 7.2 ベータリリースに伴う更新	Sun Jun 28 2015	Jiri Herrmann