Red Hat Hardware Certification 2024

Red Hat Hardware Certification テストスイート ユーザーガイド

Red Hat Hardware Certification で使用する場合

Last Updated: 2024-05-14

Red Hat Hardware Certification 2024 Red Hat Hardware Certification テストスイートユーザーガイド

Red Hat Hardware Certification で使用する場合

法律上の通知

概要

Red Hat Hardware Certification テストスイートユーザーガイドでは、Red Hat Enterprise ソフトウェアのハードウェアを認証するために必要な手順を説明しています。認証プロセス全体の概要、認証環境の設定方法、認証されるシステムまたはコンポーネントのテスト方法、検証のために Red Hat に結果を提出する方法について説明しています。また、テスト方法や結果の評価など、背景となる情報も記載されています。 バージョン 8.78 は、2024 年 4 月 16 日に更新されました。

目次

多様性を受け入れるオープンソースの強化

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

第1章 RED HAT HARDWARE CERTIFICATION プログラムの概要

このガイドを使用して、会社のハードウェア製品が 1 つ以上の Red Hat 製品を実行できることを証明してください。

1.1. RED HAT 認定プログラムの概要

Red Hat 認定プログラムは、Red Hat のパートナー企業のハードウェアおよびソフトウェア製品と、Red Hat Enterprise Linux、Red Hat OpenStack Platform、Red Hat Enterprise Linux for Real Time、および ハードウェアプラットフォーム 上のその他の Red Hat ソフトウェア製品との互換性を保証します。このプログラムは、次の 3 つの主な要素で構成されます。

- **テストスイート**: 認定を受けるハードウェアまたはソフトウェアアプリケーションのテストで構成されます。
- **Red Hat Certification Ecosystem** ハードウェア、ソフトウェア、クラウド、およびサービスプロバイダーなどの認定製品を調べて見つけることができます。
- **サポート**: パートナーと Red Hat の間の共同サポート関係。

1.2. 認定ワークフロー

ハードウェア認定は、Red Hat Enterprise Linux、Red Hat OpenStack Platform Compute、および Red Hat Enterprise Linux for Real Time を実行するサーバー、デスクトップ、ワークステーション、ラップトップ、および個々のコンポーネントのテストを対象としています。

前提条件

1. Red Hat との認定関係を確立している。
2. パートナーの製品と、認定を受ける Red Hat 製品の組み合わせで設定されるテスト環境を構築する。
3. この組み合わせが適切に機能することを確認するために、予備テストを実行する。
4. redhat-certification ツールをインストールしている。

手順

1. Red Hat Certification ツールを使用して、特定のソフトウェアまたはハードウェアコンポーネントの認定リクエストを作成します。
2. Red Hat の認定チームは、認定ポリシーをハードウェア仕様に適用して、公式テストプランを作成します。RHEL 8 および RHEL 9 のシステムまたはコンポーネントを認定するためのテスト計画は、Red Hat に提出された特定されたコンポーネントとその仕様に基づいて公開されるテストと機能で構成されています。
3. 公式テストプランで指定されたテストを実行し、Red Hat 認定 ツールを使用して、その結果を分析のために Red Hat 認定チーム に送信します。
4. 認定チームがテスト結果を分析し、再テストが必要な場合はその旨を連絡します。
5. 認証される項目をカバーする代表的なハードウェアサンプルを Red Hat に提供してください。

6. すべてのテストで良好な結果が得られた場合は、認定が完了し、認定された製品が Red Hat Ecosystem Catalog で利用できるようになります。

関連情報

- Red Hat ハードウェア認証の要件とポリシーの詳細は、Red Hat Hardware Certification プログラムポリシーガイド を参照してください。

1.3. サポートの利用とフィードバックの提供

割り当てられたエンジニアリングパートナーマネージャー、エンジニアリングアカウントマネージャー、またはテクニカルアカウントマネージャーである専任のサポートリソースをお持ちのパートナー様は、他の Red Hat 製品のサポートを依頼する際に使用するのと同じツールを使用してサポートケースを開くことができます。

専任のサポートリソースをお持ちでないパートナー様は、Red Hat カスタマーポータル を使用して、以下のインスタンスでサポートケースを開くことができます。

- 問題を報告し、認定プロセスに関するヘルプを取得する
- 認定ツールセットおよびドキュメントのフィードバックおよび機能強化のリクエストを送信する
- 製品/アプリケーションが認定されている Red Hat 製品に関するサポートを受けるRed Hat 製品の支援を受けるには、認定固有の資格およびサブスクリプションとは別の、必要な製品の資格およびサブスクリプションが必要です。

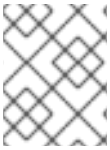
Red Hat カスタマーポータルインターフェイスを使用してサポートケースを作成するには、以下の手順を実施します。

1. Red Hat カスタマーポータル にログインして、Red Hat アカウント認証情報を使用して Red Hat Connect for Technology Partners やソフトウェアサブスクリプションなどの他の Red Hat アセットにアクセスするのにも使用されます。
2. Red Hat カスタマーポータルのホームページで **Open a Support Case** をクリックします。
3. サポートケースフォームには、以下の項目に注意して記入してください。
   - **製品**フィールドから、以下の詳細に基づいて、製品/アプリケーションが認定されている RedHat 製品の名前を選択します。
     - Red Hat OpenStack Platform Certification では、**Red Hat OpenStack Platform**を選択します。
     - CCSP(Certified Cloud and Service Provider) 認証では、**Red Hat Enterprise Linux**を選択します。
     - Red Hat Container Certification には、 **Red Hat Enterprise Linux**を選択します。
     - Red Hat Hardware Certification には、 **Red Hat EnterpriseLinux**を選択します。
   - **Product Version**欄から、製品のバージョンを選択します。
   - **Problem Statement** フィールドに、以下の形式で、問題の説明/問題またはフィードバックを入力します。
     {Partner Certification} (The Issue/Problem or Feedback)

(The Issue/Problem or Feedback) に、認定プロセス/Red Hat 製品の問題や、認定ツールセットまたはドキュメントに関するフィードバックを入力してください。

> For example: {Partner Certification} Error occurred while submitting certification test results using the Red Hat Certification application.

残りのフォームを、こちらの詳細を参考に記入してください。カスタマーポータルでサポートケースを開いて管理するにはどうすればいいですか？



注記

Red Hat では、Red Hat 認定エンジニアまたは同等の経験を有するユーザーが、認定プロセスを開始することを推奨しています。

第2章 認定パートナーのオンボーディング

新しいパートナーの場合は、Red Hat カスタマーポータルを使用して新しいアカウントを作成します。現在のパートナーの場合は、既存の Red Hat アカウントを使用して、製品の認定を受けるための Red Hat のオンボーディング手続きを行います。

2.1. 既存の認定パートナーのオンボーディング

既存のパートナーは、以下を行うことができます。

- 1 対多の EPM プログラムのメンバーで、EPM チームである程度の代表権を持っていますが、ハードウェア認定については支援を受けていません。
  OR
- ハードウェア認定要求に関する質問など、パートナーの管理に割り当てられた専用 EPM チームメンバーを使用して、EPM チームによって完全に管理されるメンバーです。

前提条件

既存の Red Hat アカウントがある。

手順

1. Red Hat カスタマーポータル にアクセスし、**Log in** をクリックします。
2. Red Hat ログインまたはメールアドレスを入力し、**Next** をクリックします。
   次に、以下のオプションのいずれかを使用します。
   a. 企業用シングルサインオンでログイン
   b. Red Hat アカウントへのログイン
3. ヘッダーのメニューバーからアバターをクリックし、アカウントの詳細を表示します。
   a. アカウント番号がアカウントに関連付けられている場合は、certops チーム に問い合わせるか、Red Hat Hardware Certification Program フォーム を送信して、認定プロセスを続行します。
   b. アカウント番号がアカウントに関連付けられていない場合は、まず Red Hat グローバルカスタマーサービスチーム に連絡して、新しいアカウント番号の作成をリクエストしてください。
   その後、certops チーム に問い合わせるか、Red Hat Hardware Certification Program フォーム を送信して、認定プロセスを続行します。

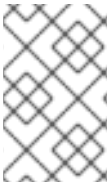
認定後、チームに連絡してください。

2.2. 新しい認定パートナーのオンボーディング

新しい認定パートナーのオンボーディングでは、まず新しい Red Hat アカウントを作成します。

1. Red Hat カスタマーポータル にアクセスし、**Register** をクリックします。
2. 以下の詳細を入力して、新しい Red Hat アカウントを作成します。
   a. **Account Type** フィールドで Corporate を選択します。

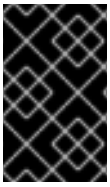
企業タイプのアカウントを作成していて、アカウント番号が必要な場合は、Red Hat グローバルカスタマーサービス チームにお問い合わせください。



注記

個人アカウントではなく、企業アカウントを必ず作成してください。この手順で作成したアカウントは、認定リクエストを使用する際に Red Hat Ecosystem Catalog にサインインするために使用します。

b. Red Hat のログインとパスワードを選択します。



重要

ログイン ID が複数のアカウントに関連付けられている場合は、ログイン時に問題が発生する可能性があるため、連絡先メールアドレスをログイン ID として使用しないでください。また、作成後にログイン ID を変更することはできません。

c. **Personal information** と **Company information** を入力します。

d. **Create My Account** をクリックします。
新しい Red Hat アカウントが作成されます。

3. 利用できる場合は、エコシステムパートナー管理 (EPM) 担当者にお問い合わせください。
OR

認定プロセスを続行するには、Certops チーム にお問い合わせください。

OR

Red Hat Hardware Certification Program フォーム に記入して、送信します。フォームを送信すると、certops チームがユーザーに連絡し、ハードウェア認定プロセスの続行を支援します。

第3章 RED HAT 認定ツールを使用した新規認定ケースの作成

パートナーは、Red Hat 認定ツールで新しい認定ケースを作成し、認定プロセスを開始できます。このタスクを完了すると、Red Hat はユーザーが指定した製品仕様に基づいてテストプランを準備します。

前提条件

- Red Hat と認定関係を確立している。
- ユーザーログイン認証情報がある。
- ユーザーログインにベンダーと製品がリンクされている。

手順

1. Red Hat Certification ポータル にログインします。
2. ホームページで、**Open Certification** をクリックします。
   Open a New Certification Case ダイアログボックスが開きます。
3. **Next** をクリックします。
4. **Partner** および **Product** リストからオプションを選択します。
   該当製品が表示されない場合は、**Product** フィールドに名前を入力して作成します。その後、それを選択します。
5. **What kind of product is this?**セクションで、製品に該当するチェックボックスを選択します。
   製品によっては、複数のエコシステムに適合している可能性があります。
6. **Next** をクリックします。
7. **Make** を入力します。
   Model は、以前に入力した情報に基づいて表示されます。
8. **Which category best describes your product?**で該当するチェックボックスを選択します。
9. オプション: **Product URL**、**Support URL**、および **Specification URL** を入力します。
10. **Next** をクリックします。
    入力に基づいて、**Partner Product** リストに新しい製品が作成されます。
11. **Red Hat Certification** の一覧からオプションを選択し、**Next** をクリックします。
12. 入力した情報を確認し、**Next** をクリックします。

検証

製品の新しい認定ケースが正常に作成されると、メッセージが表示されます。

次のステップ

Red Hat が製品のテスト計画を作成中に、テスト環境をセットアップして、 テスト実行用のシステムを準備してください。

第4章 テスト環境のセットアップ

テストを実行できるテスト環境を設定します。

テスト環境は、少なくとも 2 つのシステムで設定されます。

- システム 1: テスト対象ホスト (HUT) として機能する
- システム 2: テストサーバーとして機能する

4.1. テスト対象ホストのセットアップ

認定を必要とする製品がインストールまたは設定されているシステムは、テスト対象ホスト (HUT) と呼ばれます。

前提条件

- HUT に、RHEL バージョン 8 または 9 がインストールされている。利便性のために、Red Hat は HUT のオペレーティングシステムをインストールするための キックスタートファイル を提供しています。インストールプロセスを開始する前に、お使いのシステムに適したファイルに記載されている指示に従ってください。

注記

Red Hat Hardware Certification を取得するには、ハードウェア認定の対象となる Red Hat Enterprise Linux バージョンの一般公開 (GA) カーネルを使用する必要があります。

RHEL をインストールするときは、ブート ISO イメージではなく、バイナリー DVD オフラインインストールイメージをダウンロードして使用してください。ブート ISO イメージにはネットワーク接続が必要で、必要な GA カーネルの代わりに現在のカーネルが自動的にインストールされます。

インストールプロセス中にシステムを Red Hat Subscription Management (RHSM) に登録しないでください。インストールプロセスが完了した後でのみ、システムを RHSM に登録してください。

手順

1. **Red Hat Certification** リポジトリーを設定します。
   RHN の認証情報を使用して、Red Hat Subscription Management でシステムを登録します。

   ```
   # subscription-manager register
   ```

2. お使いのシステムで利用可能なサブスクリプションのリストを表示します。

   ```
   # subscription-manager list --available*
   ```

3. Red Hat Certification (RHEL Server 用) リポジトリーを提供するサブスクリプションを検索し、サブスクリプションとそのプール ID を書き留めます。

4. システムにサブスクリプションを割り当てます。pool_ID をサブスクリプションのプール ID に置き換えます。

   ```
   # subscription-manager attach --pool=<pool_ID>
   ```

注記

Simple content access for Red Hat Subscription Managementオプションを有効にすると、サブスクリプションをシステムにアタッチする必要はありません。詳細は、How do I enable Simple Content Access for Red Hat Subscription Management? を参照してください。

5. Red Hat Certification チャネルにサブスクライブします。

   - RHEL 8 の場合:

     ```
     # subscription-manager repos --enable=cert-1-for-rhel-8-_<HOSTTYPE>_-rpms
     ```

     HOSTTYPE をシステムアーキテクチャーに置き換えます。システムアーキテクチャーを確認するには、以下を実行します。

     ```
     uname -m
     ```

     以下に例を示します。

     ```
     # subscription-manager repos --enable=cert-1-for-rhel-8-x86_64-rpms
     ```

   - RHEL 9 の場合:

     ```
     # subscription-manager repos --enable=cert-1-for-rhel-9-_<HOSTTYPE>_-rpms
     ```

     HOSTTYPE をシステムアーキテクチャーに置き換えます。システムアーキテクチャーを確認するには、以下を実行します。

     ```
     uname -m
     ```

     以下に例を示します。

     ```
     # subscription-manager repos --enable=cert-1-for-rhel-9-x86_64-rpms
     ```

6. ハードウェアテストスイートパッケージをインストールします。

   ```
   # yum install redhat-certification-hardware
   ```

4.2. テストサーバーのセットアップ

テスト対象ホスト (HUT) で実行している一部のテストは、2 番目のシステムも合格する必要があります。この 2 番目のシステムは、テストサーバーと呼ばれます。

合格するために、帯域幅を確認し、あるシステムから別のシステムにデータを転送するテストがその例です。

前提条件

- テストサーバーに、RHEL バージョン 8 または 9 がインストールされている。Red Hat では HUT のオペレーティングシステムをインストールするための キックスタートファイル を提供

していますが、このファイルを使用してテストサーバーをインストールすることもできます。インストールプロセスを開始する前に、お使いのシステムに適したファイルに記載されている指示に従ってください。

手順

1. **Red Hat Certification** リポジトリーを設定します。
RHN の認証情報を使用して、Red Hat Subscription Management でシステムを登録します。

```
# subscription-manager register
```

2. お使いのシステムで利用可能なサブスクリプションのリストを表示します。

```
# subscription-manager list --available*
```

3. Red Hat Certification (RHEL Server 用) リポジトリーを提供するサブスクリプションを検索し、サブスクリプションとそのプール ID を書き留めます。

4. システムにサブスクリプションを割り当てます。pool_ID をサブスクリプションのプール ID に置き換えます。

```
# subscription-manager attach --pool=<pool_ID>
```

注記

Simple content access for Red Hat Subscription Managementオプションを有効にすると、サブスクリプションをシステムにアタッチする必要はありません。詳細は、How do I enable Simple Content Access for Red Hat Subscription Management? を参照してください。

5. Red Hat Certification チャネルにサブスクライブします。

- RHEL 8 の場合:

```
# subscription-manager repos --enable=cert-1-for-rhel-8-<HOSTTYPE>-rpms
```

HOSTTYPE をシステムアーキテクチャーに置き換えます。システムアーキテクチャーを確認するには、以下を実行します。

```
uname -m
```

以下に例を示します。

```
# subscription-manager repos --enable=cert-1-for-rhel-8-x86_64-rpms
```

- RHEL 9 の場合:

```
# subscription-manager repos --enable=cert-1-for-rhel-9-<HOSTTYPE>-rpms
```

HOSTTYPE をシステムアーキテクチャーに置き換えます。システムアーキテクチャーを確認するには、以下を実行します。

```
uname -m
```

以下に例を示します。

```
# subscription-manager repos --enable=cert-1-for-rhel-9-x86_64-rpms
```

6. ハードウェアテストスイートパッケージをインストールします。

```
#yum install redhat-certification-hardware
```

次のステップ

Downloading the test plan from Red Hat Certification portal を参照してください。

第5章 RED HAT CERTIFICATION ポータルからのテストプランのダウンロード

手順

1. Red Hat Certification ポータル にログインします。
2. 製品認定に関連するケース番号を検索してコピーします。
3. **Cases** をクリックし、製品ケース番号を入力します。
4. オプション: テスト実行中にテストされるコンポーネントをリストするには、**Test Plans** をクリックします。
5. **Download Test Plan** をクリックします。

次のステップ

Cockpit を使用してテストを実行する予定の場合、Configuring the systems and running tests by using Cockpit を参照してください。

CLI を使用してテストを実行する予定の場合は、CLI を使用したシステムの設定とテストの実行 を参照してください。

第6章 COCKPIT を使用したシステムの設定とテストの実行

認定プロセスを完了するには、Cockpit を設定し、テスト対象ホスト (HUT) とテストサーバーを準備し、テストを実行して、テスト結果を取得する必要があります。

6.1. COCKPIT サーバーのセットアップ

Cockpit は、ユーザーフレンドリーな Web ベースのインターフェイスからシステムの設定を変更したり、システムのリソースを監視したりできる RHEL ツールです。

注記

- Cockpit は、テスト対象ホストやテストサーバーとは別の新しいシステムにセットアップする必要があります。
- Cockpit がテスト対象ホストとテストサーバーの両方にアクセスできることを確認してください。

Cockpit のインストールと設定の詳細は、RHEL 8 の RHEL Web コンソールの使用、RHEL 9 の RHEL Web コンソールの使用、および Introducing Cockpit を参照してください。

前提条件

- Cockpit サーバーに RHEL バージョン 8 または 9 がインストールされている。
- システムに Cockpit プラグインがインストールされている。
- Cockpit サービスが有効になっている。

手順

1. Cockpit をインストールしたシステムにログインします。
2. Red Hat 認定チームが指定する Cockpit RPM をインストールします。

```
# yum install redhat-certification-cockpit
```

Cockpit はポート 9090 で実行する必要があります。

6.2. テスト対象ホストとテストサーバーの COCKPIT への追加

テスト対象ホスト (HUT) とテストサーバーを Cockpit に追加すると、2 つのシステムがパスワードレス SSH を使用して通信できるようになります。

この手順を繰り返して、両方のシステムを 1 つずつ追加します。

前提条件

- テスト対象ホストとテストサーバーの IP アドレスまたはホスト名がある。

手順

1. ブラウザーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. ログインしている cockpit ユーザー名の下矢印をクリックし、**Add new host** をクリックします。
   ダイアログボックスが表示されます。
4. **Host** フィールドに、システムの IP アドレスまたはホスト名を入力します。
5. **User name** フィールドに、このシステムに割り当てる名前を入力します。
6. オプション: 追加されたホストに対して、事前定義された色を選択するか、任意の新しい色を選択します。
7. **Add** をクリックします。
8. **Accept key and connect** をクリックして、パスワードなしの SSH で Cockpit がシステムと通信できるようにします。
9. **Password** を入力します。
10. **Authorize SSH Key** チェックボックスを選択します。
11. **Log in** をクリックします。

検証

左側のパネルで **Tools →Red Hat Certification** をクリックし、追加したシステムが右側の Hosts セクションに表示されていることを確認します。

6.3. RED HAT SSO ネットワークでの認証の取得

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools → Red Hat Certification** を選択します。
4. Cockpit ホームページで、**Authorize** をクリックして、Red Hat システムとの接続を確立します。
   Red Hat アカウントにログイン ページが表示されます。
5. 認証情報を入力し、**Next** をクリックします。
   rhcert-cwe へのアクセスの許可 ページが表示されます。
6. **Grant access** をクリックします。確認メッセージに、デバイスへのログインの成功が表示されます。これで、Cockpit Web アプリケーションに接続されました。

6.4. RED HAT 認定ポータルから COCKPIT でのテスト計画をダウンロードする

権限のないユーザーまたはアクセスが制限されているユーザーの場合:

- テスト計画をダウンロードするには、Red Hat 認定ポータルからのテスト計画のダウンロードを参照してください。

許可されたユーザーの場合:

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools → Red Hat Certification**を選択します。
4. **Test Plans** タブをクリックします。**最近の認定サポートケース** のリストが表示されます。
5. **Download Test Plan** をクリックします。テスト計画が正常に追加されたことを確認するメッセージが表示されます。
6. ダウンロードしたテスト計画は、**テスト計画ファイル** セクションの **ファイル名** の下にリストされます。

6.5. テストプランに基づくテスト対象ホストのテスト準備

テスト対象ホストをプロビジョニングすると、Cockpit とのパスワードなしの SSH 通信のセットアップ、認定タイプに基づくシステムへの必要なパッケージのインストール、実行する最終テストプラン (Red Hat が提供するテストプランとシステム要件を検出して生成したテストの両方から取得した一般的なテストのリスト) の作成など、多くの処理が実行されます。

たとえば、テストプランがハードウェア製品の認定用に設計されている場合は、必要なハードウェアパッケージがインストールされます。

前提条件

- Red Hat が提供するテスト計画をダウンロードしている。

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools →Red Hat Certification** を選択します。
4. **Hosts** タブをクリックしてから、テストを実行するテスト対象ホストをクリックします。
5. **Provision** をクリックします。
   ダイアログボックスが表示されます。
   a. **Upload** をクリックして、新しいテストプランの .xml ファイルを選択します。**Next** をクリックします。アップロードに成功したというメッセージが表示されます。
   必要に応じて、以前にアップロードしたテストプランを再利用する場合は、もう一度選択して再アップロードします。

注記

認定プロセス中に、進行中の製品認定のために再設計されたテストプランを受け取った場合は、前のステップに従ってアップロードできます。ただし、続行する前に、Terminal タブで **rhcert-clean all** を実行する必要があります。

b. **Role** フィールドで、**Host under test** を選択し、**Submit** をクリックします。

c. デフォルトでは、ファイルはパス **/var/rhcert/plans/<testplanfile.xml>** にアップロードされます。

6.6. テストプランを使用したテスト用テストサーバーの準備

Provision Host コマンドを実行すると、rhcertd サービスが有効になり、起動します。このサービスは、ネットワークテスト用の iperf や kdump テストで使用される nfs マウントポイントなど、テストサーバーのテストスイートで指定されたサービスを設定します。

前提条件

- Red Hat が提供するテスト計画をダウンロードしている。

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools →Red Hat Certification** を選択します。
4. **Hosts** タブをクリックしてから、テストを実行するテスト対象ホストをクリックします。
5. **Provision** をクリックします。
ダイアログボックスが表示されます。

a. **Upload** をクリックして、新しいテストプランの .xml ファイルを選択します。**Next** をクリックします。アップロードに成功したというメッセージが表示されます。
必要に応じて、以前にアップロードしたテストプランを再利用する場合は、もう一度選択して再アップロードします。

注記

認定プロセス中に、進行中の製品認定のために再設計されたテストプランを受け取った場合は、前のステップに従ってアップロードできます。ただし、続行する前に、Terminal タブで **rhcert-clean all** を実行する必要があります。

b. **Role** フィールドで **Test server** を選択し、**Submit** をクリックします。デフォルトでは、ファイルは **/var/rhcert/plans/<testplanfile.xml>** パスにアップロードされます。

6.7. COCKPIT を使用した認定テストの実行

前提条件

- テスト対象ホストの準備が完了している。
- テストサーバーの準備が完了している。

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools →Red Hat Certification** を選択します。
4. **Hosts** タブをクリックし、テストを実行するホストをクリックします。
5. **Terminal** タブをクリックして、**Run** を選択します。
   アップロードされたテストプランに基づいた推奨テストのリストが表示されます。実行する最終的なテストプランは、Red Hat が提供するテストプランと、システム要件を検出して生成したテストの両方から取得した一般的なテストのリストです。
6. プロンプトが表示されたら、**yes** または **no** を入力して、各テストを実行するかどうかを選択します。
   select を入力して、リストから特定のテストを実行することもできます。

6.8. テスト結果ファイルの確認とダウンロード

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools →Red Hat Certification** を選択します。
4. **Result Files** タブをクリックして、生成されたテスト結果を表示します。
   a. オプション: **Preview** をクリックして、各テストの結果を表示します。
   b. 結果ファイルの横にある **Download** をクリックします。デフォルトでは、結果ファイルは **/var/rhcert/save/hostname-date-time.xml** として保存されます。

6.9. COCKPIT から RED HAT 認定ポータルへのテスト結果の送信

手順

1. ブラウザーのアドレスバーに **http://<Cockpit_system_IP>:9090/** を入力し、Cockpit Web アプリケーションを起動します。
2. ユーザー名とパスワードを入力し、**Login** をクリックします。
3. 左側のパネルで **Tools → Red Hat Certification**を選択します。

4. **Result Files** タブをクリックし、表示されたリストからケース番号を選択します。

    a. 許可されたユーザーの場合は、**Submit** をクリックします。テスト結果ファイルのアップロードが成功したことを確認するメッセージが表示されます。

    b. 権限のないユーザーには、実行されたテスト計画の結果ファイルの Red Hat 認定ポータルへのアップロード を参照してください。

実行されたテスト計画のテスト結果ファイルは、Red Hat 認定ポータルにアップロードされます。

6.10. テスト結果ファイルの RED HAT 認定ポータルへのアップロード

Red Hat 認定ツールを使用して、実行されたテスト計画のテスト結果ファイルを Red Hat Certification チームに送信します。

前提条件

- Cockpit または HUT からテスト結果ファイルをダウンロードしている。

手順

1. Red Hat Certification Tool にログインします。
2. ホームページで、検索バーに製品のケース番号を入力します。
   表示されるリストからケース番号を選択します。
3. **Summary** タブの Files セクションで、 **Upload** をクリックします。

次のステップ

Red Hat は、送信された結果ファイルを確認し、次のステップを提案します。詳細は、Red Hat Certification Tool にアクセスしてください。

第7章 RHCERT CLI ツールを使用したシステムの設定とテストの実行

CLI を使用して認証プロセスを完了するには、テスト対象ホスト (HUT) とテストサーバーを準備し、テストを実行して、テスト結果を取得する必要があります。

7.1. テストプランに基づくテスト対象ホストのテスト準備

プロビジョニングコマンドを実行すると、テストサーバーとのパスワードなしの SSH 通信のセットアップ、認定タイプに基づくシステムへの必要なパッケージのインストール、実行する最終テストプラン (Red Hat が提供するテストプランとシステム要件を検出して生成したテストの両方から取得した一般的なテストのリスト) の作成など、多くの処理が実行されます。

たとえば、テストプランがハードウェア製品の認定用に設計されている場合は、必要なハードウェアパッケージがインストールされます。

前提条件

- テストサーバーのホスト名または IP アドレスがある。

手順

1. いずれかの方法でプロビジョニングコマンドを実行します。テスト計画はシステムに自動的にダウンロードされます。
   - すでにテスト計画をダウンロードしている場合:

     ```
     # rhcert-provision <path_to_test_plan_document>
     ```

     <path_to_test_plan_document> を、システムに保存されているテストプランファイルに置き換えます。

     画面上の指示に従ってください。
   - テスト計画をダウンロードしていない場合は、以下を実行します。

     ```
     # rhcert-provision
     ```

     画面上の指示に従い、プロンプトが表示されたら **認定 ID** を入力します。
2. プロンプトが表示されたら、パスワードなしの SSH を設定するために、テストサーバーのホスト名または IP アドレスを指定します。新しいシステムを初めて追加するときにのみプロンプトが表示されます。

7.2. テストプランを使用したテスト用テストサーバーの準備

Provision コマンドを実行すると、**rhcertd** サービスが有効になり、起動します。このサービスは、ネットワークテスト用の iperf や kdump テストで使用される nfs マウントポイントなど、テストサーバーのテストスイートで指定されたサービスを設定します。

前提条件

- テスト対象ホストのホスト名と IP アドレスがある。

手順

1. 追加するシステムに test server ロールを定義して provision コマンドを実行します。
これは、テストサーバーのプロビジョニングにのみ必要です。

```
# rhcert-provision --role test-server <path_to_test_plan_document>
```

<path_to_test_plan_document> を、システムに保存されているテストプランファイルに置き換えます。

7.3. CLI を使用した認定テストの実行

手順

1. 以下のコマンドを実行します。

```
# rhcert-run
```

2. プロンプトが表示されたら、**yes** または **no** を入力して、各テストを実行するかどうかを選択します。
select を入力して、リストから特定のテストを実行することもできます。

注記

テスト再起動後、**rhcert** がバックグラウンドで実行され、イメージが検証されます。**tail -f /var/log/rhcert/RedHatCertDaemon.log** を使用して、検証の現在の進行状況とステータスを確認します。

7.4. テスト結果ファイルの送信

手順

1. ログインして、デバイスを認証します。

注記

テスト結果ファイルを送信するには、ログインが必要です。

```
# rhcert-cli login
```

a. 生成された URL を新しいブラウザーウィンドウまたはタブで開きます。

b. ログインとパスワードを入力し、**Log in** をクリックします。

c. **Grant access** をクリックします。
デバイスのログイン成功メッセージが表示されます。

d. ターミナルに戻り、**Please confirm once you grant access**に **yes** と入力します。

2. 結果ファイルを送信します。

```
# rhcert-submit
```

プロンプトが表示されたら、認定 ID を入力します。

第8章 認定ワークフロー

8.1. 以前に認定されたハードウェアに認定を追加する

このプロセスを使用して、以前の RHEL バージョンのハードウェア認定プロセスをすでに完了しているシステムまたはコンポーネント、または現在認定されているシステムまたはコンポーネントの新しい認定リクエストを作成します。

手順

1. Red Hat Certification ポータル にログインします。
2. **New Certification** をクリックします。
3. 認定する Red Hat 製品、バージョン、およびプラットフォームを選択します。**Next** をクリックします。
4. ベンダー、メーカー、すでに認定された製品の名称をドロップダウンリストから選択します。**Next** をクリックします。

リクエストが作成されたら、レビューチームが公式のテスト計画を作成するときに、レビューチームからの質問のリクエストを監視します。

8.2. 既存の認証の機能またはハードウェアの変更

既存の認定にハードウェアや機能を追加するための補助認定を申請します。

テストが行われなかった、あるいはテストが失敗したなどの理由で、以前に認定されなかった機能については、補助認定を要求することができます。追加機能が認定された後、Red Hat はその機能を認定カタログに追加します。

手順

1. Red Hat Certification ポータル にログインします。
2. 既存のハードウェア認定をクリックします。
3. **製品**をクリックします。
4. **商品詳細**をクリックします。
5. **Attachment**フィールドで**Browse**"をクリックして、新規追加コンポーネントの仕様書ファイルを添付します。**これは仕様ですか**を選択する。
   Red Hat の認定チームは、ハードウェア認定に補助コンポーネントを追加します。
6. 補助認定を作成します。既存のハードウェア認定で補助コンポーネントが表示されるのを待つ必要はありません。
   a. ハードウェア認定ページに移動します。
   b. **Certification** セクションをクリックします。
   c. **Related Certification** タブをクリックし、**Supplemental Certification** セクションに移動します。

d. **New Certification** をクリックし、新しい補助認定を作成します。
Red Hat の認定チームは、補助認定にテストプランを追加します。

7. 認定テストを実行します。新しいテストプランを待つ必要はありません。

8.3. 既存の仕様ファイルを使用したシステムパススルー認定の作成

システムパススルー認定では、認定済みシステムのコピーが作成され、別のベンダー名、別のメーカー、または別のモデルの下にリストされます。

パススルーは、ベンダーが自社のシステムをパートナーに販売し、パートナーがブランドを変更する場合、またはベンダーが 2 つ以上の (1 つのシステムが別のシステムのスーパーセットである) システムを販売する場合に使用されます。

手順

1. Red Hat Certification ポータル にログインします。
2. 既存のハードウェア認定をクリックします。
3. **Related Certification** をクリックします。
4. **Add Related Certification** をクリックし、**Pass-through** を選択します。
5. 適切な製品を選んでください。
   - 製品がすでに作成されている場合は、その製品を選択します。
   - 製品がリストにない場合は、新規の製品として作成します。
6. **New Certification** をクリックして、新しいパススルー認定を作成します。

Red Hat の認定チームがハードウェアの仕様を確認し、新しいシステム.認定を公開します。新しい認定が公開された後、パートナーはそれをパススルー認定として参照できます。

8.3.1. 既存のシステム認証を新しいエントリーにコピーする

手順

1. パススルー認定を作成するには、**Red Hat certification** Web ユーザーインターフェイスに移動し、認定されている既存のハードウェアシステム認定をクリックします。**Certification** セクションをクリックします。**Related Certification** タブで、**Related Certification** セクションに移動し、**New Certification** ボタンをクリックします。
2. **Vendor** フィールドでは、パススルーが必要な製品のベンダーを選択します。**Make** フィールドで、パススルーが必要な Make を選択します。
3. **Create** ボタンをクリックします。これにより、パススルーシステムの仕様書と、生成された仕様書に対するパススルー認証の作成依頼が発生します。

オリジナルシステムの仕様とパススルーシステムの仕様が同一または相違点がない場合は、追加のテストは必要ありません。差異が発見された場合、Red Hat 認定チームは、その差異を説明するために何をすべきかをお客様と話し合います。

8.3.2. 既存の仕様ファイルを使用したシステムパススルー認定の作成

手順

1. **Red Hat certification** Web ユーザーインターフェイスに移動し、認証されている既存のハードウェアシステム認証をクリックします。**Certification** セクションをクリックします。

2. **Related Certification** タブで **Pass through Certification** セクションに移動し、作成されたパススルー仕様ファイルを選択します。

これにより、同じ仕様のエントリーを使用して 2 つ目のパススルー認定が作成されます。

8.4. コンポーネントパススルー認証の作成と公開

コンポーネントパススルー認証は、基本的に、認証されたコンポーネントのコピーを作成し、別のベンダー名、別のメーカー、または別のモデルでリストします。このタイプのパススルーは、システムベンダーが、コンポーネントベンダーによってすでに認定されているコンポーネントを含めたい場合、コンポーネントベンダーがコンポーネントをリブランドするサードパーティーに販売する場合、またはベンダーが 2 つ以上を販売する場合に使用されます。1 つのシステムが他のシステムのスーパーセットであるコンポーネント。

手順

1. システム認定を作成します。Red Hat Certification ポータルを使用した新規認定ケースの作成を参照してください。

2. **Vendor**、**Make**、**Name**を選択します。**New Product**ボタンをクリックします。これにより、**Choose the Certification Program** の Web ページに移動します。

3. **Vendor** とハードウェアとして **Program** を選択します。**Next** ボタンをクリックします。これにより、**Define the Red Hat Hardware Certification Vendor Product**Web ページをに移動します。

4. 関連情報をすべて入力してください。**Category** のドロップダウンリストから、**Component/Peripheral** を選択します。

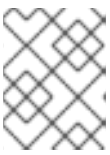
これにより、コンポーネント認証が作成されます。Red Hat 認定チームは、新しく作成された Component 認定を認定し、公開します。証明書が認証され、公開されると、他のパートナーがパススルーコンポーネントとして参照できるようになります。

8.4.1. 既存のコンポーネント認定を新規エントリーへコピー

手順

1. コンポーネント認証をコピーするには、**Red Hat Certification** Web ユーザーインターフェイスに移動し、認証されている既存のハードウェアシステム認証をクリックします。**Certification** セクションをクリックします。**Related Certification** タブで、**Pass through Certification** セクションに移動し、**New Certification** ボタンをクリックします。

2. **Vendor** フィールドでは、パススルーが必要な製品を持つコンポーネントベンダーを選択します。**Make** フィールドで、パススルーが必要な Component Make を選択します。



注記

ここで、Component Vendor と Component Make は、Component Certification の作成と公開 のステップ 1 から 4 を実行する際に生成されるフィールドです。

オリジナルのコンポーネントの仕様とパススルーのコンポーネントの仕様が同じであれば、追加のテストは必要ありません。差異が発見された場合、Red Hat 認定チームは、その差異を考慮して何をすべきかをお客様と話し合います。

8.5. 製品認証への欠落データの追加

正確かつ完全な認証情報を確保するには、認証を公開する前に、不足している属性を追加するためのこの合理化されたプロセスに従ってください。

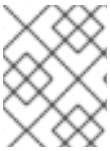
手順

1. Red Hat Certification ポータル にログインします。
2. 既存のハードウェア認定をクリックします。
3. **Certification Status** セクションで、疑問符アイコンをクリックします。**Completion Requirements** 通知バナーには、不足している属性に関する情報が表示されます。
4. 欠落している属性の 1 つをクリックすると、その証明書の **Properties** タブにリダイレクトされます。
5. オプション: **Partner Product** セクションで製品をクリックし、**Properties** タブに移動します。
6. **Properties** タブで、詳細説明、簡単な説明、パートナー製品ロゴ、製品ロゴなどの欠落している詳細を入力します。
7. **Update** をクリックします。

検証

必要なデータがすべて存在するか更新されている場合、疑問符アイコンは表示されなくなります。

認証が完了して公開されると、製品の更新データが Red Hat エコシステムカタログ で利用できるようになります。



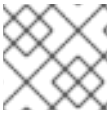
注記

アスタリスク * の付いたフィールドはすべて必須であり、証明書を発行する前に入力する必要があります。

8.6. 64K カーネルの認定

64k ページサイズのカーネルは、ARM プラットフォーム上の大規模なデータセットに便利なオプションです。システム全体のパフォーマンス、特に大規模なデータベース、HPC、および高いネットワークパフォーマンスが大幅に向上するため、メモリーを大量に使用するワークロードに適しています。

RHEL 9.2 以降、ARM アーキテクチャーは 64k ページサイズのカーネルをオプションとして使用し、4k カーネルをデフォルトとして使用します。64k ページサイズのカーネルを認定するには、まずデフォルトの 4k カーネルを使用して RHEL 9 認定を完了する必要があります。その後、64k カーネルを使用して 2 番目の認定を実施できます。2 番目の認定が正常に完了すると、64k ページサイズのサポートと 64k カーネルの使用方法を示す ナレッジベース記事 が 4k サイズのカーネル認定に添付されます。



注記

64k カーネルを認定するには、補足認定を作成する必要があります。

8.7. テストの実行中にゲストイメージをダウンロードする

手順

1. システム上で、ゲストイメージがローカルで使用可能か確認します。使用可能な場合、テストの実行が開始されます。
2. ゲストイメージがローカルで利用できない場合、テストは事前設定されたテストサーバーから、ゲストイメージのダウンロードを試みます。
3. ローカルでの可用性とテストサーバーのダウンロードの両方が失敗した場合、テストは CWE API との接続を確立して、AWS の事前署名された S3 URL を取得します。その後、ゲストイメージは、指定された URL を使用して AWS からダウンロードされます。
4. AWS からのダウンロードでも問題が発生する場合、テストでは CWE API を使用して、ゲストイメージを直接ストリーミングおよびダウンロードします。
5. ゲストイメージを取得するためのこれまでの試行がすべて失敗した場合、テスト全体が FAIL とマークされます。

注記

上記の手順は、rhcert バージョン 8.66 以降に適用されます。

テストの実行中に FV イメージのダウンロードに失敗した場合は、以下の手順に従います。

a. Red Hat Certification ポータル から、ファイルをダウンロードします。

b. ファイルをダウンロードした後、それらをテスト対象のホストの **/var/lib/libvirt/images** ディレクトリーに移動します。

c. ファイルを手動で展開するには、**tar xmvfj <tarred file name>** コマンドを使用します。

d. ファイルを展開した後、**mv <extracted file> <image file name>** コマンドを使用して名前を変更します。例: **mv hwcertData-20211116.img hwcertData.img**
ファイル名については、以下の表を参照してください。

tar が実行されたファイル名	イメージファイル名
hwcertData.img.tar.bz2	hwcertData.img
hwcert-x86_64.img.tar.bz2	hwcert-x86_64.img
rhel-kvm-rt-image.qcow2.tar.bz2	rhel-kvm-rt-image.qcow2

第9章 階層化された製品認定

9.1. 階層化された製品の認定

階層化された製品認定は、RHEL 用に認定済みのシステムに与えられる追加の認定です。

階層化された製品の認定は、次の 2 つの方法で作成できます。

1. 「階層化された認定の自動生成」
2. 「階層化された認定の手動作成」

9.1.1. 階層化された認定の自動生成

階層化された認定は、ハードウェア認定のステータスが **Certified** の場合にのみ自動的に生成できます。Red Hat 認定チームは、Red Hat Certification ポータル にリストされるようにハードウェア認定を **公開** します。

手順

階層化された認定を自動生成するには、次の手順を実行します。

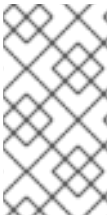
1. Red Hat Certification ポータル にログインします。
2. 認定して公開する必要があるハードウェア認定をクリックします。
3. **Dialog** タブをクリックします。
4. **New Comment** テキストボックスに、Red Hat 認定チームへのコメントを入力し、認定を認定して公開します。
5. **Add Comment** ボタンをクリックします。

要求されたハードウェア認定にコメントを追加すると、Red Hat 認定チームが認定して公開します。認定されて公開されると、メールが届きます。

新たに自動生成された階層化認定は Red Hat Certification ポータル で確認できます。表示されない場合は、更新ボタンをクリックすると、新しい認定証明がダウンロードされます。

9.1.2. 階層化された認定の手動作成

階層化された認定は、ハードウェア認定が通常として分類されている場合、手動で作成できます。



注記

- 階層化された認定は、すべての通常のハードウェア認定でサポートされているわけではありません。
- 階層化された認定を公開する前に、基本の認定を公開する必要があります。

手順

階層化された認定を手動で作成するには、次の手順を実行します。

1. Red Hat Certification ポータル で、Red Hat 認定チームによって認定された通常のハードウェア認定をクリックします。
2. **Related Certification** タブをクリックします。
3. **Add Related Certification** をクリックし、**Certification Types** から **Layered** を選択します。
4. **Red Hat Certification** から **Certification Type** を選択し、**Red Hat Product Version** ドロップダウンから **Product Version** 選択し、**Next** をクリックします。
5. 認定情報を確認し、**Open** をクリックします。

Red Hat Certification ポータル で新しい認定が作成され、新しく作成された認定ページにリダイレクトされます。

9.2. RED HAT ENTERPRISE LINUX FOR REAL-TIME

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定
7、8、9	7、8、9	RHEL

お使いのシステムに対して、前出の表に詳述されている前提条件の認定が授与された後、Red Hat Enterprise Linux for Real Time 認定を申請してください。

Red Hat Hardware Certification テストスイートには、リアルタイム認定を取得するために必要なテストが含まれています。

9.2.1. 関連情報

- Red Hat Enterprise Linux for Real Time Life Cycle

9.3. RED HAT VIRTUALIZATION

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定
7、8	7、8	RHEL

Red Hat Virtualization は RHEL に依存し、RHEL と共同で設計されています。基盤が共通であることから、システムが Red Hat Enterprise Linux 認定の際に、基本的な仮想化テストに合格した場合、Red Hat Virtualization 認定を受けるために追加のテストを実行する必要はありません。システムが高度な仮想化テストにも合格した場合、Red Hat Virtualization 認定には高度な機能も自動的に含まれます。

Red Hat は、すべての 64 ビット Intel および AMD、および RHEL 用に提出されたすべての IBM Power リトルエンディアンサーバー認定について、お客様に代わって Red Hat Virtualization 認定を自動的に開きます。

認定を希望しても自動的に受け取らなかった場合は、Red Hat Virtualization 製品の階層化された新しい認定要求を作成します。

9.3.1. 関連情報

- Red Hat Virtualization
- Red Hat Virtualization のライフサイクル

9.4. RED HAT ENTERPRISE LINUX OPENSTACK PLATFORM COMPUTE

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定
8 および 9	サポートマトリックスを参照してください	RHEL (基本的な仮想化テストに合格している必要があります)

Red Hat OpenStack Platform は RHEL に依存し、RHEL と共同で設計されています。基盤が共通であることから、システムに対して前出の表に詳述されている前提条件の認定を満たしている場合は、Red Hat OpenStack Platform Compute の認定を取得するために追加のテストを実行する必要はありません。また、新しい認定を取得する必要もありません。

ただし、Red Hat OpenStack Platform RHEL 8 の場合、階層化された製品認定は、特に LPAR (論理パーティション) ベースではない POWER システムに適用されます。RHEL 8 で階層化された製品認定を受けるには、テスト中にシステムをベアメタルハイパーバイザーとして機能させる必要があります。

RHEL 9 の場合、Red Hat は POWER システム上の KVM をサポートしません。したがって、仮想化ベースの階層化された製品認定は作成されません。

他のアーキテクチャーまたはカテゴリーの場合、または認定を希望しているけれども、自動的に受け取らなかった場合は、Red Hat OpenStack Platform Compute 製品の新しい階層化認定要求を作成してください。

Red Hat は、ベースボード管理コントローラー (BMC) を備えたシステムを認定するパートナーに対して、Red Hat OpenStack Platform Bare Metal 認定も申請することを推奨しています。

関連情報

- Red Hat OpenStack Platform
- Red Hat OpenStack Platform のライフサイクル
- Red Hat OpenStack Platform ハードウェアベアメタル認定ポリシーガイド
- Red Hat ベアメタルハードウェア認定ワークフローガイド

9.5. RED HAT OPENSTACK PLATFORM FOR REAL-TIME APPLICATIONS

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定
8	サポートマトリックスを参照してください	• RHEL (基本的な仮想化テストに合格している必要があります) • Red Hat Enterprise Linux for Real Time • Red Hat OpenStack Platform Compute

お使いのシステムに対して、前出の表に詳述されている前提条件の認定が授与された後、Red Hat OpenStack Platform for Real-Time Applications 認定を申請してください。Red Hat Hardware Certification テストスイートには、リアルタイム認定を取得するために必要なテストが含まれています。

9.6. RED HAT OPENSHIFT CONTAINER PLATFORM

サポートされている RHEL のバージョン	サポートされる RHEL for Real Time バージョン	前提条件となる認定
9.2 および 9.4 (RHOCP 4.13 または 4.14、4.15)	サポートマトリックスを参照してください	RHEL (基本的な仮想化テストに合格している必要があります)
8 (RHOCP 4.12)	サポートマトリックスを参照してください	RHEL (基本的な仮想化テストに合格している必要があります)

Red Hat OpenShift Container Platform は、RHEL に依存し、RHEL と共同で設計されています。基盤が共通であることから、システムに対して前出の表に詳述されている前提条件の認定を満たしている場合は、Red Hat OpenShift Container Platform の認定を取得するために追加のテストを実行する必要はありません。

ただし、RHEL 8 ベースのサポート対象バージョンである Red Hat OpenShift Container Platform (現在 v4.11 および v4.12) の場合、階層化された製品認定は特に LPAR (論理パーティション) ベースではない POWER システムに適用されます。RHEL 8 で階層化された製品認定を受けるには、テスト中にシステムをベアメタルハイパーバイザーとして機能させる必要があります。

RHEL 9 の場合、Red Hat は POWER システム上の KVM をサポートしません。したがって、仮想化ベースの階層化された製品認定は作成されません。

他のアーキテクチャーまたはカテゴリーの場合、または認定を希望しているけれども、自動的に受け取らなかった場合は、Red Hat OpenShift Container Platform 製品の新しい階層化認定要求を作成してください。

Red Hat OpenShift Platform Bare Metal 認定を申請して、IPI および Assisted installer 機能をカタログの RHOCP エントリーに追加します。

関連情報

- Red Hat OpenShift Container Platform
- Red Hat OpenShift ライフサイクル
- Red Hat OpenStack Platform ハードウェアベアメタル認定ポリシーガイド
- Red Hat ベアメタルハードウェア認定ワークフローガイド

第10章 レバレッジ

レバレッジとは、認定システムのハードウェアから合格したテスト結果を再利用して、新しい認定リクエストで同一のハードウェアのテストをカバーすることです。特定のオプションアイテムにのみ使用でき、これらのアイテムは同一である必要があります。新しいモデルコンポーネントのテスト結果を古いモデルのテスト結果と一緒に活用することはできません。それらがどれほど類似していても。アイテムは完全に一致する必要があります。さらに、レバレッジは、**お客様の組織またはその代理人**が実施したテストに対してのみ使用することができます。

10.1. 同じベンダーのシステム認証を活用するためのルール

以下は、同じベンダーの場合にシステムまたはコンポーネントの認証を活用する際に注意する必要のあるガイドラインです。

1. コンポーネントは同一でなければなりません。
2. 生成される結果は、同じアーキテクチャーのハードウェアからのものである必要があります。
3. コンポーネントの結果を活用するシステムは、同じメジャーリリースを証明する必要があります。
4. システム認証からのレバレッジでは、ベンダー間のレバレッジは実行できません。

以下に例を示します。

- Acme Computers は、そのコンポーネントのいずれかからの合格テスト結果を活用して、別の Acme システムの同一アイテムをカバーできます。

しかし

- Acme Computers は、CloverleafIndustries が実施した認証のテスト結果を参照することはできません。

10.2. さまざまなベンダーのシステム認証を活用するためのルール

1. コンポーネントメーカーが、再販業者によって販売されたコンポーネントのベンダー、メーカー、およびモデル情報を使用して元の認証のパススルーを作成するシナリオでは、次のガイドラインを考慮する必要があります。
    i. **詳細設定**タブで、システムパススルー認証と同様に、元の認証の**パススルーを使用して作成**を選択します。
    ii. 多くのリセラーが同じコンポーネントを使用している場合、コンポーネントの製造元はリセラーごとに 1 つのパススルーを作成する必要があります
    iii. リセラーが同じカードに複数の名前を使用している場合、コンポーネントの製造元は名前ごとに 1 つのパススルーを作成する必要があります
2. Red Hat 認定チームが、使用されているハードウェアが同一であることを確認し、仕様ファイルのドキュメントを使用してパススルー認定が完了した後、認定は公開または非公開になります。
3. リセラーは、このハードウェアを含むシステム認証要求の適切な**レバレッジ**フィールドでパススルーの**認証 ID**を使用する必要があります。

10.3. システム認証から活用するためのテスト結果 ID の生成

以下は、システム認証から活用するためのテスト結果 ID を生成するための手順です。

手順

1. RedHat 認定 Web ユーザーインターフェイスからソースハードウェア製品と認定を作成します。

2. 新しく作成された認定にコンポーネントを追加するには、 **Red Hat Certification** Web ユーザーインターフェイスから、認定済みのハードウェア認定をクリックします。[**製品**]セクションをクリックし、[**製品の詳細**]タブをクリックします。

3. **添付ファイル**セクションで、**ファイルの選択**ボタンをクリックして、仕様ファイルをアップロードします。仕様ファイルは、追加する必要のあるコンポーネントで設定されています。

4. **これは仕様ですか**チェックボックスを選択し、**添付ファイルの説明**テキストボックスに簡単なメモを追加します。たとえば、これは spec.file です。

5. **Red Hat 認定**Web ユーザーインターフェイスから、認定されているハードウェア証明書をクリックします。**Certification** セクションをクリックします。**進行状況**タブで、コンポーネントに関してテスト計画が生成されていることがわかります。
   Red Hat 認定チームは、仕様ファイルに記載されているコンポーネントを確認して追加し、後で追加されたコンポーネントのテストプランを作成します。

6. 表に示されているコンポーネントのテストを実行するには、**実行**ボタンをクリックします。これにより、 **テスト**タブに移動します。

7. **テストシステムの追加**をクリックします。これで**Select Host**の Web ページが表示されます。

8. テストを実行するホストを選択し、**テスト**ボタンをクリックします。

9. **テスト**タブで、**テストの続行**ボタンをクリックします。これにより、コンポーネントのリストが生成されます。

10. テストを実行したいコンポーネントを選択し、**選択したものを実行**ボタンをクリックします。

11. テストの実行が完了すると、**テストの実行が終了しました**というメッセージが表示されます。

12. テストをクリックすると、テストが実行されたコンポーネントには**PASS**という結果が表示されます。テスト結果を RedHat 認定チームに送信するには、**アクション**フィールドのドロップダウンリストから**送信**を選択します。これにより、**ファイル送信**の Web ページが表示されます。

13. **送信**ボタンをクリックします。
    Red Hat 認定チームがテスト結果を承認します。承認されたテスト結果は、コンポーネントに関連付けられた**テスト結果 ID**を生成します。

14. **Red Hat Certification** Web ユーザーインターフェイスから、**Certified** のハードウェア認定をクリックします。**Certification** セクションをクリックします。**進行状況**タブには、**テスト計画のクレジット**が**確認済み**と表示されています。**テスト結果**の欄には、生成された**テスト結果 ID**が表示されます。

10.4. 既存のコンポーネントの活用

同じコンポーネントを使用して新しい認証を作成する場合、そのコンポーネントを 2 つの方法で活用することができます。

手順

1. **結果 ID のコピーによる活用**
   システム認証を活用するためのテスト結果 ID の生成 を参照してください。

   Red Hat Certification チームは、仕様書ファイルに記載されている活用するコンポーネントを承認します。

   a. **Red Hat Certification** Web ユーザーインターフェイスから、**Certified** されており、そのテスト結果 ID を利用するハードウェア認定をクリックします。**Certification** セクションをクリックします。**進行状況**タブの**テスト結果**欄には、テコ入れを行うコンポーネントの生成された**テスト結果 ID**が表示されています。ID を選択してテスト結果の ID をコピーします。

   b. ID のコピーに成功すると、**Copied Leverage Information from System Certification<the_component_name>**というメッセージが表示されます。

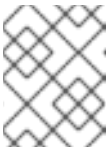
   c. **Red Hat 認定**Web ユーザーインターフェイスから、レバレッジコンポーネントを追加するハードウェアサートをクリックします。

   d. **Certification** セクションをクリックします。**進行状況**タブの**テスト結果**列で、**テスト結果 ID**をクリックすると、コピーしたテスト結果 ID が適用されます。
   レバレッジ ID が正常に適用されると、**システムテストのレバレッジが正常に適用されました**というメッセージが表示されます。

2. **リザルト ID や認証 ID を利用したレバレッジ**

   a. **Certification** セクションをクリックします。**進行状況**タブで、**テスト結果 ID**列に移動し、**レバレッジの結果**をクリックします。

   b. **レバレッジ結果**ウィンドウで、**レバレッジ使用**ドロップダウンから**結果 ID**または**認定 ID**を選択します。

   注記

   テスト結果 ID で活用する場合は**Result ID** を、パススルー認定で活用する場合は **Certification ID**を選択します。

   c. **レバレッジ ID**を入力し、**送信**をクリックします。
   レバレッジが成功した場合は成功メッセージが表示され、提出した ID にエラーがある場合は失敗メッセージが表示されます。

第11章 テスト結果の確認および認定の完了

11.1. レッドハットによるテスト結果のレビュー

結果を提出していただいた後、レビューチームが内容を分析し、テストプランの一部である各合格テストのクレジットを付与します。

合格した各テストを確認することで、チームは各テスト計画項目を、 **確認済** に設定します。これは、カタログの **結果** タブで確認することができます。これにより、どのテストが未解決で、どのテストが合格と確認されたかを一目で確認することができます。

何らかの問題が見つかった場合、審査チームは認証要求を更新して質問を行い、その質問は自動的に認証を提出した人にメールで送られます。

認証の**ダイアログ**タブで、すべての議論を見て、質問に答えたり、質問したりすることができます。

11.2. 認定資格の取得

Red Hat がテストプランの全項目に合格したことを確認した後、認定が完了します。この時点で、認定を閉じて公開するか、認定を閉じて未公開のままにするかを選択することができます。

補助認定は常に未公開のままです。システムおよびコンポーネントの認定は、認定ステータスやシステムまたはコンポーネントの存在をアドバタイズしたくない場合は、未公開のままにすることができます。

公開された認定では、システム情報や、お客様と Red Hat レビューチームとの議論は、公開された認定では一般には見えません。

これらの公開オプションが要件を満たさない場合は、認定がオープンしている間に例外のリクエストを送信するか、すでにケースが閉じられている場合はケースを作成してください。

付録A テスト

ここでは、ハードウェア認証の各テストについて、より詳細な情報をお伝えします。各テストセクションは以下の形式で設定されています。

テストの対象

このセクションでは、この特定のテストが実行されるハードウェアの種類を示します。

RHEL バージョン対応

このセクションでは、テストがサポートされている RHEL のバージョンをリストアップしています。

テストの内容

ここでは、テストスクリプトの機能について説明します。すべてのテストは Python スクリプトであり、テストで実行しているコマンドを正確に知りたい場合は、ディレクトリー **/usr/lib/python2.7/site-packages/rhcert/suites/hwcert/tests** で表示できます。

テストの準備

ここでは、テストの準備に必要な手順について説明します。例えば、USB テストには USB デバイスを、書き換え可能な光学ドライブテストにはブランクディスクを用意しておくことが書かれています。

テストの実行

このセクションでは、テストが対話型か非対話型かを識別し、テストを実行するために必要なコマンドを説明します。

次のどちらかのテスト実行方法を選択できます。

- テストを実行するには CLI を使用した認定テストの実行 に従ってください。次のコマンドを使用して、表示されたリストから適切なテスト名を選択します。

```
rhcert-run
```

- 計画中にハードウェア検出の問題やその他のハードウェア関連の問題が発生した場合は、 テストの手動追加および実行 に従ってください。目的のテスト名を指定して、**rhcert-cli** コマンドを実行します。

```
rhcert-cli run --test=<test name>
```

ランタイム

ここでは、このテストの実行にかかる時間について説明します。**supportable** テストのタイミング情報は、テストスイートのすべての実行に必要なテストであるため、各セクションで言及されています。

A.1. ACPI キー

テストの対象

ACPI キー テストは、システム統合キーボードからさまざまな入力イベントを取得します。

RHEL バージョン対応

- RHEL 8.6 以降

- RHEL 9

テストの内容

テストは以下を取得します。

- 電源、一時停止、スリープなどの ACPI 関連シグナル。
- **<Meta+E>** などのグローバルキーボードショートカットに関連付けられたシグナルを送信するキーを押すと、ファイルブラウザーが開きます。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **ACPI keys** テスト名を選択します。

```
rhcert-run
```

このテストでは、すべての入力イベントを取得する必要があります。テスト中に、デバイスの標準以外のキーおよびマルチメディアキーをすべて押します。

いつでも **Escape キー** を押してテストを終了し、キーのリストを表示できます。テストしたすべてのキーがリストに表示されていればテストは成功です。

ランタイム

テストの所要時間は 5 分未満です。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.2. AUDIO

テストの対象

リムーバブルサウンドカードと統合サウンドデバイスは **audio** テストでテストされます。ハードウェア検出ルーチンが udev データベースに以下の文字列を見つけたときにテストが実行されます。

```
E: SUBSYSTEM=sound
E: SOUND_INITIALIZED=1
```

これらの文字列と、このガイドの他のテストのスケジューリングをトリガーする文字列は、コマンド **udevadm info --export-db** 出力で確認できます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、あらかじめ録音された音 (ギターのコードや録音された声) を再生すると同時に、それをファイルに録音し、録音された音を再生して、その音が聞こえたかどうかを尋ねます。

テストの準備

試運転を始める前に、オーディオテストが予定されていて、システムがサウンドを再生および録音できることを確認する必要があります。オーディオデバイスがインストールされているシステムでテストが表示されない場合は、Red Hat のサポート担当者にお問い合わせください。テストが正しくスケジュールされていれば、続けてサウンドデバイスの再生および録音機能を手動でテストする方法をご紹介します。

内蔵スピーカー、またはヘッドフォン/ラインアウト端子にスピーカー/ヘッドフォンを接続した状態で、再生を確認してからこれらの方法でテストすることができます。

1. **Settings** アプリケーションで **Sound** オプションをクリックします。
2. **Output** タブをクリックし、テストするサウンドカードを選択し、**Output volume** を適切なレベルに調整します。
3. **Test Speakers** をクリックします。
4. **Speaker Testing** ポップアップウィンドウで、**Test** ボタンをクリックしてサウンドを生成します。

音が出ない場合は、スピーカーが正しいポートに差し込まれているか確認してください。任意のライン出力またはヘッドフォンジャックを使用できます (どのポートを使用しなければならないかという要件はありません)。サウンドがミュートになっていないか確認し、スピーカーおよびオペレーティングシステム自体のボリュームを調整してみてください。

オーディオ機器に録音機能がある場合は、テストを実行する前にそれらもテストする必要があります。マイクをシステムのライン入力またはマイクジャックのいずれかに接続します。ノートパソコンをテストする場合は、内蔵マイクを使用することもできます。繰り返しますが、特定の入力ジャックを使用する必要はありません。どちらかが機能すれば、テストは合格になります。

1. **Settings** アプリケーションで **Sound** オプションをクリックします。
2. **Input** タブをクリックし、適切な入力デバイスを選択して、 **Input volume** を 100% に調整します。
3. 入力デバイスに話しかけたり、軽く叩いたりするか、入力デバイスをアクティベートし、**Input level** のグラフィックを確認します。動いていることを確認できれば、入力デバイスは正しく設定されています。動いていない場合は、入力デバイスを接続するための別の入力方法またはマイクポートを試してください。

音が聞こえない、または入力レベル表示が動かない場合は、オーディオテストに失敗したことになりますので、サポート担当者にお問い合わせください。マイクの近くで音を出したときに、正常に音が再生され、入力レベル表示に動きがあることが確認できたら、次のセクションに進み、テストの実行方法を学びます。

テストの実行

オーディオテストはインタラクティブに行われます。オーディオテストを含むテストランを実行する前に、マニュアルテストで使用したマイクを接続し、スピーカーの前に置くか、内蔵マイクに障害物がないことを確認してください。また、ノイズの多い環境でテストする場合には、ラインアウト端子とマイク/ラインイン端子をパッチケーブルで直接接続することもできます。

次のコマンドを実行し、表示されるリストから適切な **Audio** テスト名を選択します。

```
rhcert-run
```

対話の手順は以下の通りです。

1. システムは音を再生し、それが聞こえたかどうかを尋ねます。必要に応じて**y**または**n**と答えてください。スピーカーとマイクではなく、出力と入力を直接接続する場合は、パッチケーブルによってスピーカーがバイパスされるため、答えは関係なく**y**を選択する必要があります。

2. 続いて、記録したファイルを再生します。音が聞こえた場合は、プロンプトが表示されたら**y**と答えてください。それ以外の場合は**n**と答えてください。

ランタイム

オーディオテストの所要時間は、再生と録音を同時に行った後、録音した音を再生するまでに1分以内です。必要な **supportable** テストにより、全体の実行時間が約1分長くなります。

A.3. バックライト

テストの対象

バックライトテストは、システムに接続されたディスプレイを検出して、ソフトウェアによるバックライト制御のサポートが可能な場合に実行されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、ディスプレイの輝度を最小値から最大値に調整することで、バックライトコントロールが期待通りに機能していることが確認できます。

テストの準備

- テスト対象のホストが RHEL 8.0 以降であることを確認します。
- システムがバックライトをサポートしていること、ディスプレイがシステムに接続されていることを確認してください。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **backlight** テスト名を選択します。

```
rhcert-run
```

ディスプレイの明るさが最大から最小値に変わり、元に戻ります。ディスプレイの明るさが期待通りに変化していることを確認すると、テストは合格となります。

ランタイム

このテストの実行には1分もかかりません。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.4. バッテリー

テストの対象

battery テストは、バッテリーを内蔵している有効なシステムでのみ実行できます。このテストは、UPS や BIOS バッテリーなど、システムに一次内部電源を供給しない外部バッテリーには対応していません。テストは、ハードウェア検出ルーチンが **udev** データベース内で次の文字列を検出したときにスケジュールされます。

```
POWER_SUPPLY_TYPE=Battery
```

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、バッテリーが接続されているかどうか、AC アダプターがシステムに接続されているかどうか、バッテリーの充電と放電のステータスが検出されます。

テストの準備

注記

バッテリーが 100% 充電されているときはテストを実行しないでください。テスト失敗の可能性を避けるために、テストを実行する前にバッテリーを放電し、充電レベルを下げてください。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **battery** テスト名を選択します。

```
rhcert-run
```

このテストでは、10 mWh のバッテリーの充放電を検出し、現在の容量とバッテリーの充電状態を表示します。プロンプトが表示されたら、画面上の指示に従って AC アダプターを抜き差しします。

ランタイム

テストの実行時間はバッテリーの充放電速度によって異なります。このテストはラップトップで実行されるため、必須の **supportable** テストが **suspend** テストとともに実行されます。全体で 7 - 10 分かかります。

A.5. BLUETOOTH

テストの対象

bluetooth テストは、bluetooth v3、v4、または v5 コントローラーを搭載したシステムでサポートされています。

RHEL バージョン対応

- RHEL 8

- RHEL 9

テストの内容

このテストは、**rfkill** コマンドを使用して、システム内の bluetooth コントローラーの可用性を確認します。次に、**hciconfig** または **btmgmt** コマンドを実行して bluetooth コントローラーバージョンを取得します。その後、**bluetoothctl** コマンドツールを使用して、コントローラーが他の選択されたアライメントされた bluetooth v3、v4、または v5 デバイスをスキャン、発見、ペアリング、選択、信頼できるかどうかを確認します。HUT に複数の bluetooth コントローラーがある場合は、各 bluetooth コントローラーに対して bluetooth テストが自動的に計画されます。

テストの準備

テストを開始する前に、以下のことを確認してください。

- コントローラーと同じ、またはそれ以上のバージョンの Bluetooth に対応した機器をお持ちください。
- HUT とペアリングデバイスの両方で Bluetooth を有効にする。
- 設定アプリを使用して手動でペアリングし、接続を確認してください。
- デバイスのペアリングを解除します。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **bluetooth** テスト名を選択します。

```
rhcert-run
```

次に、ブルートゥースの機能をテストしたいデバイスを選択します。

ランタイム

テストは 5 分程度で終了します。ただし、ブルートゥースのネットワーク接続状況によって時間が変わることがあります。

A.6. ブルーレイ

テストの対象

Bluray テストは、以下のメディアおよび関連するドライブタイプで実行されます。

- 読み取り専用メディアおよびドライブ (BD-ROM)
- ライトワンスメディアおよびドライブ (BD-R)
- 書き換え可能なメディアおよびドライブ (BD-RE)

udev コマンドからの情報に基づいて、テストスイートはスケジュールする光学ドライブテスト (Blu-ray、DVD、または CD-ROM) と、テストするメディアタイプ (読み取り専用、書き込み可能、および書き換え可能) を決定します。たとえばテストスイートは、書き換え機能を持ち、DVD および CD-ROM ディスクの読み取りも可能な Blu-ray ドライブに以下のテストを計画します。

- Blu-ray メディアの書き換え (消去、書き込み、および読み取り) テスト

- DVD メディアの読み取りテスト
- CD-ROM メディアの読み取りテスト

指定されたドライブに対して 1 回のみ Bluray テストを実行する必要があります。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、ドライブの機能に応じて以下のタスクを実行します。

- **読み取り専用ドライブ**: まずディスクからデータを読み取り、それをハードディスクにコピーします。次に、ディスクのデータをハードディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。
- **書き込み機能搭載ドライブ**: まずハードディスクからデータを読み取り、それを書き込み可能な空のディスクに書き込みます。次に、ハードディスクのデータをディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。
- **書き換え機能搭載ドライブ**: まず書き換え可能なディスクの情報をすべて消去します。次に、ハードディスクからデータを読み取り、それを書き換え可能ディスクに書き込みます。最後に、ハードディスクのデータをディスクのコピーと比較します。消去操作に成功し、すべてのファイルチェックサムが一致していれば、テストに合格します。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **Bluray** テスト名を選択します。

```
rhcert-run
```

画面の手順に従って適切なメディアを挿入し、必要に応じてドライブのトレイを閉じます。

ランタイム

Bluray テストのランタイムは、メディアとドライブの速度によって異なります。2x 25G BD-RE ディスクの場合、テストは約 14 分間で終了します。

A.7. CD ROM

テストの対象

CD ROM テストは、以下のメディアおよび関連するドライブタイプで実行されます。

- 読み取り専用のメディアおよびドライブ (CD-ROM)
- ライトワンスメディアおよびドライブ (CD-R)
- 書き換え可能なメディアおよびドライブ (CD-RW)

udev コマンドからの情報に基づいて、テストスイートはスケジュールする光学ドライブテスト (Blu-ray、DVD、または CD-ROM) と、テストするメディアタイプ (読み取り専用、書き込み可能、および書き換え可能) を決定します。たとえばテストスイートは、書き換え機能を持ち、DVD および CD-ROM ディスクの読み取りも可能な Blu-ray ドライブに以下のテストを計画します。

- Blu-ray メディアの書き換え (消去、書き込み、および読み取り) テスト
- DVD メディアの読み取りテスト
- CD-ROM メディアの読み取りテスト

CD ROM テストは、指定ドライブに対して 1 回のみ実行する必要があります。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、ドライブの機能に応じて以下のタスクを実行します。

- **読み取り専用ドライブ**: まずディスクからデータを読み取り、それをハードディスクにコピーします。次に、ディスクのデータをハードディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。
- **書き込み機能搭載ドライブ**: まずハードディスクからデータを読み取り、それを書き込み可能な空のディスクに書き込みます。次に、ハードディスクのデータをディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。
- **書き換え機能搭載ドライブ**: まず書き換え可能なディスクの情報をすべて消去します。次に、ハードディスクからデータを読み取り、それを書き換え可能ディスクに書き込みます。最後に、ハードディスクのデータをディスクのコピーと比較します。消去操作に成功し、すべてのファイルチェックサムが一致していれば、テストに合格します。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **CD ROM** テスト名を選択します。

```
rhcert-run
```

画面の手順に従って適切なメディアを挿入し、必要に応じてドライブのトレイを閉じます。

ランタイム

CD ROM テストのランタイムは、メディアとドライブの速度に依存します。12x 714MB CD-RW ディスクの場合、テストは約 7 分間で終了します。

A.8. CORE

テストの対象

core テストでは、システムの CPU を検査し、負荷がかかった状態でも適切に機能する能力があるかどうかを確認します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

core のテストは、実際には 2 つの別々のルーチンで設定されています。最初のテストは、クロックジッターを検出するためのものです。ジッターとは、システムのクロックが互いに同期していないときに発生する状態のことです。システムクロックは、CPU クロック速度と同じではありません。これは、CPU が動作している速度を表すもう 1 つの方法です。jitter テストでは、**getimeofday()** 関数を使用して、各論理 CPU で観察された時間を取得し、戻り値を分析します。すべての CPU クロックが互いに 0.2 ナノ秒以内の場合は、テストに合格します。ジッターテストの許容範囲は非常に厳しいものです。良い結果を得るためには、テストの実行時にシステム上で動作している負荷が rhcert テストだけであることが重要です。他の計算負荷が存在すると、タイミングが阻害され、テストが失敗する可能性があります。ジッターテストでは、カーネルがどのクロックソースを使用しているかも確認します。インテルプロセッサーが TSC を使用していない場合、ログに警告が表示されますが、テストの PASS/FAIL ステータスには影響しません。

コアテストで実行された 2 つ目のルーチンは、CPU 負荷テストです。これは、必要な **stress** パッケージにより提供されるテストです。システムのストレステストを行う方法を探している場合に rhcert スイートの外部で使用できるストレスプログラムは、システム上で同時に複数のアクティビティーを起動し、障害を監視します。具体的には、各論理 CPU に平方根を計算するように指示し、**malloc()** ルーチンおよび **free()** ルーチンを使用してシステムにメモリーの負荷をかけ、それぞれメモリーを予約および解放し、**sync()** を呼び出してディスクへの書き込みを強制します。これらのアクティビティーは 10 分間継続し、その時間内に障害が発生しなければ、テストに合格します。ハードウェア認定テスト以外での使用を検討している場合は、man ページの **stress** を参照してください。

テストの準備

コアテストの唯一の準備は、ポリシーガイドに記載されている要件を満たす CPU をインストールすることです。

テストの実行

コアテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **Core** テスト名を選択します。

```
rhcert-run
```

ランタイム、ベアメタル

コアテスト自体は、ベアメタルシステム上で約 12 分で実行できます。テストのジッター部分は 1〜2 分で終わり、ストレス部分はちょうど 10 分間実行されます。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

ランタイム、フルバータイプのゲスト

fv_core テストを KVM ゲストで実行すると、ベアメタル版よりもわずかに時間がかかり、約 14 分かかります。この時間は、ゲストのスタートアップ/シャットダウンのアクティビティーと、ゲスト内で実行される必要な **supportable** テストによるものです。ベアメタルシステムで必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.9. CPU のスケーリング

テストの対象

cpuscaling テストは、CPU が計算需要に応じてクロック速度を増減させる能力を調べるものです。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、異なるスケーリングガバナー (CPU がいつ高クロックまたは低クロックに変更するか、どんな速さでそれを行うかの一連の命令) を使用して、異なる周波数で CPU を動作させ、標準化されたワークロードを完了するのにかかる時間の違いを測定します。ハードウェア検出ルーチンが、/sys 内に複数の cpu 周波数を含む以下のディレクトリーを発見した場合、テストがスケジュールされます。

```
/sys/devices/system/cpu/cpuX/cpufreq
```

cpuscaling テストは、論理 CPU ごとに 1 回記載されるのではなく、パッケージごとに 1 回予定されています。テストが実行されると、 **/sys/devices/system/cpu/cpuX/topology/physical_package_id** を介してトポロジーが決定され、特定のパッケージ内のすべての論理 CPU に対してテストが並行して実行されます。

このテストでは、まず turbostat コマンドを実行して、プロセッサーの統計情報を収集します。サポートされているアーキテクチャーでは、turbostat は、事前の統計カラムが turbostat 出力ファイルに表示されているかどうかをチェックしますが、ファイルにカラムが含まれていない場合は警告を返します。その後、テストは cstate サブテストの実行を試み、失敗した場合は pstate サブテストを実行します。

各 CPU パッケージのテスト手順は以下の通りです。

このテストでは、sysfs ファイルシステムにある値を使用して、CPU の最大および最小周波数を決定します。このコマンドで任意のシステムのこれらの値を見ることができます。

```
# cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies
```

ここには常に最大と最小の 2 つの周波数が表示されますが、プロセッサーによってはより細かい CPU 速度の制御が可能なため、ファイルには 2 つ以上の値が表示されます。最大値と最小値の間にある追加の CPU 速度は、テスト中には特に使用されませんが、CPU が最大値と最小値の間を移行する際に使用されることがあります。テストの手順は以下の通りです。

1. このテストでは、**/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies** ファイルから最大および最小のプロセッサー速度を記録します。
2. ユーザースペースのガバナーを選択し、最大の周波数を選択します。
3. 最大速度は、すべてのプロセッサーの **/sys/devices/system/cpu/cpuX/cpufreq/scaling_cur_freq** の値を読み取ることで確認できます。この値が選択された周波数と一致しない場合、テストは失敗を報告します。

4. パッケージ内のすべてのプロセッサーには、2x10^12 桁までの円周率を計算するというタスクが同時に与えられています。円周率の計算には、意味のある時間 (約 30 秒) がかかることから、この値を選びました。

5. 円周率の計算にかかった時間を CPU ごとに記録し、パッケージの平均値を算出しています。

6. ユーザースペースのガバナーを選択し、最低速度を設定します。

7. 最低速度は sysfs のデータで確認し、要求された速度に達していない CPU があれば失敗となります。

8. 同じ円周率の計算を、パッケージ内のすべてのプロセッサーで行い、その結果を記録します。

9. 作業量に応じて CPU の速度を最小から最大の間で調整するオンデマンドガバナが選択されています。

10. 最低速度は sysfs のデータで確認し、要求された速度に達していない CPU があれば失敗となります。

11. 同じ円周率の計算を、パッケージ内のすべてのプロセッサーで行い、その結果を記録します。

12. CPU を常に最大速度にするパフォーマンスガバナが選択されています。

13. 最大速度は sysfs のデータで確認され、要求された速度に達していない CPU があれば失敗となります。

14. 同じ円周率の計算を、すべてのプロセッサープロセッサーで行い、その結果を記録します。

ここからは、3 つのサブセクションについて分析を行います。ステップ 1 からステップ 8 では、最大および最小の CPU 速度での pi 計算時間を求めます。2 つの速度で円周率を計算するのにかかる時間の差は、CPU 速度の差に比例するはずです。例えば、仮想的なテストシステムの最大周波数が 2GHz、最小周波数が 1GHz で、最大速度で円周率計算を実行するのに 30 秒かかった場合、最小速度では円周率計算に 60 秒かかると予想されます。様々な理由で完璧な結果が得られないことは承知していますので、結果に対して 10% の誤差 (予想よりも速い、または遅い) を許容しています。この例では、最低速度での実行に 54 秒から 66 秒かかっても、合格とみなされることになります (60 の 90%=54、60 の 110%=66)。

ステップ 9〜11 では、オンデマンドガバナを使用したπ計算時間のテストを行います。これにより、仕事をしているときに、システムが CPU の速度を素早く最大にすることができることが確認できました。ステップ 11 で得られた計算時間を、ステップ 5 で得られた最高速度の計算時間と比較します。この 2 つの値の差が 10% 以内であれば合格です。

ステップ 12 から 14 では、パフォーマンスガバナーを使用して、πの計算をテストします。これにより、システムが常に CPU を最大の周波数で保持できることが確認できます。ステップ 14 で得られたπの計算時間を、ステップ 5 で得られた最高速度の計算時間と比較します。繰り返しになりますが、この 2 つの値の差が 10% 以内であれば合格です。

cpuscaling テストの追加部分は、**/proc/cpuinfo** の **ida**CPU フラグの存在により、TurboBoost 機能を持つ Intel プロセッサーが検出された場合に実行されます。このテストでは、各パッケージに搭載されている CPU のうち、ハウスキーピングのために CPU0 を除いた 1 つの CPU を選択し、オンデマンドガバナを最大速度で使用して性能を測定しています。パッケージ内のすべてのコアを並行してテストした前回のテストと比較して、少なくとも 5% の性能向上を期待しています。

テストの準備

テストの準備として、BIOS で CPU の周波数スケーリングが有効になっていることを確認し、ポリシーガイドで説明されている要件を満たす CPU がインストールされていることを確認してください。

テストの実行

cpuscaling テストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **CPU scaling** テスト名を選択します。

```
rhcert-run
```

ランタイム

cpuscaling テストは、Red Hat Enterprise Linux 6.4、AMD64 および Intel 64 が動作する 2013 年当時のシングル CPU、6 コア/12 スレッドの 3.3GHz Intel ベースのワークステーションで、約 42 分かかりました。コア数が多く、ソケットの数が多いシステムでは、より時間がかかります。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.10. DVD

テストの対象

DVD テストは、以下のメディアおよび関連するドライブタイプで実行されます。

- 読み取り専用のメディアおよびドライブ (DVD-ROM)
- ライトワンスメディアおよびドライブ (DVD+R および DVD-R)
- 書き換え可能なメディアおよびドライブ (DVD+RW および DVD-RW)

udev コマンドからの情報に基づいて、テストスイートはスケジュールする光学ドライブテスト (Blu-ray、DVD、または CD-ROM) と、テストするメディアタイプ (読み取り専用、書き込み可能、および書き換え可能) を決定します。たとえばテストスイートは、書き換え機能を持ち、DVD および CD-ROM ディスクの読み取りも可能な Blu-ray ドライブに以下のテストを計画します。

- Blu-ray メディアの書き換え (消去、書き込み、および読み取り) テスト
- DVD メディアの読み取りテスト
- CD-ROM メディアの読み取りテスト

お使いのドライブが DVD-RW と DVD+RW 形式に対応している場合は、テスト時にどちらのタイプのディスクも使用できます。両方の形式をテストする必要はありません。また、指定されたドライブに対して 1 回のみ DVD テストを実行する必要があります。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、ドライブの機能に応じて以下のタスクを実行します。

- **読み取り専用ドライブ**: まずディスクからデータを読み取り、それをハードディスクにコピーします。次に、ディスクのデータをハードディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。

- **書き込み機能搭載ドライブ**: まずハードディスクからデータを読み取り、それを書き込み可能な空のディスクに書き込みます。次に、ハードディスクのデータをディスクのコピーと比較します。すべてのファイルのチェックサムが一致すれば、テストは合格です。
- **書き換え機能搭載ドライブ**: まず書き換え可能なディスクの情報をすべて消去します。次に、ハードディスクからデータを読み取り、それを書き換え可能ディスクに書き込みます。最後に、ハードディスクのデータをディスクのコピーと比較します。消去操作に成功し、すべてのファイルチェックサムが一致していれば、テストに合格します。

テストの実行

次のコマンドを実行し、表示されるリストから適切な **DVD** テスト名を選択します。

```
rhcert-run
```

画面の手順に従って適切なメディアを挿入し、必要に応じてドライブのトレイを閉じます。

ランタイム

DVD テストのランタイムは、メディアとドライブの速度によって異なります。4x 4.7GB DVD-RW ディスクの場合、テストは約 13 分間で終了します。

A.11. イーサネット

テストの対象

Ethernet テストは、ネットワーク機器の速度がテストスイートで認識されない場合にのみ表示されます。これは、ケーブルが接続されていないか、その他の障害によって接続速度が正しく検出されていないことが原因と考えられます。テストスイートを終了して接続を確認し、機器が正しく接続された状態で再度テストスイートを実行してください。それでも問題が解決しない場合は、Red Hat のサポート担当者にご相談ください。

注記

このテストを実行するために、テスト対象の RHEL 7 ホストで RHEL 8 テストサーバーを使用している場合は、テストスイートがこのサービスを自動的に開始しないため、**httpd** サービスを手動で開始する必要があります。

以下の例では、2 つのギガビットイーサネットデバイス (eth0 と eth1) を持つシステムを示しています。デバイスの eth0 はきちんと接続されていますが、eth1 は接続されていません。

ethtool コマンドの出力では、eth0 のギガビットイーサネットの速度は 1000Mb/s と予想されています。

```
# ethtool eth0
Settings for eth0:
 Supported ports: [ TP ]
 Supported link modes:   10baseT/Half 10baseT/Full
                         100baseT/Half 100baseT/Full
                         1000baseT/Full
 Supported pause frame use: No
 Supports auto-negotiation: Yes
 Advertised link modes:  10baseT/Half 10baseT/Full
                         100baseT/Half 100baseT/Full
                         1000baseT/Full
```

```
Advertised pause frame use: No
Advertised auto-negotiation: Yes
Speed: 1000Mb/s
Duplex: Full
Port: Twisted Pair
PHYAD: 2
Transceiver: internal
Auto-negotiation: on
MDI-X: on
Supports Wake-on: pumbg
Wake-on: g
Current message level: 0x00000007 (7)
        drv probe link
Link detected: yes
```

ただし、eth1 では、**ethtool** コマンドは不明な速度を表示します。これにより、**Ethernet** テストが計画されます。

```
# ethtool eth1
Settings for eth1:
 Supported ports: [ TP ]
 Supported link modes:   10baseT/Half 10baseT/Full
                         100baseT/Half 100baseT/Full
                         1000baseT/Full
 Supported pause frame use: No
 Supports auto-negotiation: Yes
 Advertised link modes:  10baseT/Half 10baseT/Full
                         100baseT/Half 100baseT/Full
                         1000baseT/Full
 Advertised pause frame use: No
 Advertised auto-negotiation: Yes
 Speed: Unknown!
 Duplex: Unknown! (255)
 Port: Twisted Pair
 PHYAD: 1
 Transceiver: internal
 Auto-negotiation: on
 MDI-X: Unknown
 Supports Wake-on: pumbg
 Wake-on: g
 Current message level: 0x00000007 (7)
        drv probe link
 Link detected: no
```

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

A.12. エクスプレスカード

テストの対象

expresscard test は、ExpressCard インターフェイスをもつ、USB および PCI Express (PCIe) の両方のタイプのデバイスを探し、システムが両方で通信できることを確認します。ExpressCard スロットの検出は、システム内の他のデバイスの検出ほど簡単ではありません。ExpressCard は、専用のブリッジデバイスを必要としないように設計されています。これは単に、PCIe と USB を組み合わせた新しいフォームファクターのインターフェイスです。このため、udev の出力で確認できる特定の ExpressCard スロットのエントリーはありません。バッテリー、USB、PCIe インターフェイスを搭載したシステムでテストを行うことにしました。これは、ExpressCard を搭載したノート PC 以外、このようなハードウェアの組み合わせを持つ機器を確認していないからです。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストではまず、**lsusb** および **lspci** コマンドを使用して、USB および PCIe バス上のすべてのデバイスのスナップショットを取得します。そして、システムに存在する ExpressCard スロットの数を尋ねてきます。試験者は、スロットの一つにカードを挿入するよう求められます。このシステムは、USB および PCIe バスをスキャンし、その結果を元の lsusb および lspci の出力と比較して、新しいデバイスを検出します。USB デバイスが検出された場合は、カードを取り外して、同じスロットに PCIe インターフェイスのカードを挿入するように求められます。PCIe ベースのカードが検出された場合は、カードを取り外して、同じスロットに USB ベースのカードを挿入するように求められます。カードが両方のインターフェイスで挿入されている場合 (ドッキングステーションカードなど)、そのスロットの両方のテスト要件を一度に満たします。この手順がシステム内のすべてのスロットについて繰り返されます。

テストの準備

USB と PCIe バスを搭載した ExpressCard カードが必要です。これは、2 枚の独立したカードでも、1 枚のカードに両方のインターフェイスを搭載したものでも構いません。テストを実行する前に、すべての ExpressCard カードを取り外してください。

テストの実行

expresscard のテストはインタラクティブです。次のコマンドを実行し、表示されるリストから適切な **Expresscard** テスト名を選択します。

```
rhcert-run
```

すべての ExpressCard を取り外すように促され、PCI Express hotplug モジュール (pciehp) がロードされていない場合は、ロードの許可を求められます。システム稼働中に PCIe ベースの ExpressCard カードを追加または削除するには、PCIe ホットプラグ機能が必要です。次に、システムに搭載されている ExpressCard スロットの数が尋ねられ、続いて、両方のタイプのインターフェイス (USB と PCIe) を持つカードを任意の順番で抜き差しを促すプロンプトが表示されます。

A.13. FINGERPRINTREADER

テストの対象

フィンガープリントリーダーのテストは、システムに内蔵またはプラグインのフィンガープリントリーダーがある場合に計画されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、フィンガープリントリーダーが、フィンガープリントマネージャーに登録された指紋をスキャンし、登録し、確認できることを検証します。

テストの準備

フィンガープリントリーダーがシステムに接続されていることを確認します。

テストの実行

テストはインタラクティブに行われます。以下のコマンドを実行し、表示されるリストから適切な **fingerprintreader** テスト名を選択します。

```
rhcert-run
```

テストは、フィンガープリントリーダーの検出を開始し、登録が完了するまで適切なインデックスフィンガープリントリーダーで複数回配置し、スキャンします。本人確認のために、再度指をスキャンして登録済みの指紋と照合するよう求められます。

ランタイム

このテストは、リーダーのスキャンが終了し、**登録完了**の状態が表示されるまで、約 2 分かかります。

A.14. FIRMWARE

テストの対象

ファームウェアテストがサポートされているのは、Unified Extensible Firmware Interface (UEFI) と EFI System Resource Table (ESRT) を使用する x86_64 アーキテクチャーシステムの RHEL バージョン 8 以降で実行してファームウェアを管理する場合のみです。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

テストでは以下のサブテストが実行されます。

- セキュリティーチェックサブテスト: サブテストでは、システムとデバイスのファームウェアが HSI-1 レベルの標準 を満たしているかどうかを検証することで、テスト対象のホストがセキュリティーのベストプラクティスに従っているかどうかをチェックします。このテスト

は、**fwupdagent security --force** コマンドを使用して HSI-1 セキュリティー属性を確認し、出力を取得します。

- 更新サービスサブテスト: サブテストでは、テスト対象のホストが Linux Vendor Firmware Service (LVFS) を通じてファームウェア更新をダウンロードしてインストールできるかどうかを検証します。

合格の基準

- テストは、すべての HSI-1 属性に合格した場合にのみ合格します。
- システムが LVFS 更新をインストールした場合、テストに合格します。

テストの準備

- テスト対象のホストが RHEL 8.0 以降であることを確認します。
- システムが、レガシー BIOS モードではなく、UEFI モードで起動されていることを確認します。

テストの実行

このテストは、対話型ではありません。次のコマンドを実行し、表示されるリストから適切な **firmware** テスト名を選択します。

```
rhcert-run
```

ランタイム

このテストの実行には 1 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

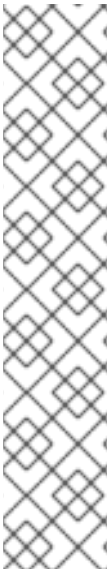
A.15. FV_CORE

fv_core テストは、FV ゲストを起動し、 **core** テストを実行するラッパーです。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9



注記

完全仮想化テストを初めて実行するとき、テストツールは FV ゲストファイルを取得する必要があります。テストツールの実行時間は、FV ゲストファイルの転送速度によって異なります。以下に例を示します。

- FV ゲストファイルがローカルテストサーバーにあり、1GbE 以上のネットワークを使用している場合は、約 300MB のゲストファイルを転送するのに約 1、2 分かかります。
- ファイルが CWE API から取得 (ゲストファイルがローカルテストサーバーにインストールされていないか見つからない場合に自動的に取得) される場合、最初の実行時間は CWE API からの転送速度に依存します。

ゲストファイルがテスト対象ホスト (HUT) で利用できる場合は、その後の fv_* テストの実行すべてで使用されます。

関連情報

- テスト方法や実行時間などの詳細については、core をご覧ください。
- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.16. FV_CPU_PINNING

CPU ピニングとは、システムリソースを特定のプロセスに割り当てる手法のことです。例えば、アプリケーションを特定の論理コアにロックすることで、タスクの切り替えを減らすことができます。

仮想化 (fv) の CPU ピニング方法は、ピニングが KVM ベースの仮想マシン内の仮想 CPU (vCPU) からホストマシン上の物理コアに行われることを除いて、似ています。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの対象

fv_cpu_pinningテストは、ゲスト仮想マシン (VM) の vCPU がホストマシンの専用 CPU に設定され、固定されるかどうかを検証します。このテストはホストマシン上で実行され、RHEL 8 ベースの RHV 4 リリースの機能認定のために RHEL 8 でサポートされています。

テストの内容

fv_cpu_pinningテストでは、3 つのサブテストが実行されます。ゲスト VM の VCPU を設定し、FV CPU Pinning を実行し、FV CPU Pinning を検証します。Setup guest VM VCPU subtest では、ホストマシンの論理コア数をカウントし、その中で最後の番号のコアを分離して VM に割り当てます。Perform FV CPU Pinning サブテストでは、VM の vCPU をホストマシンの CPU にさらにピン接続します。このテストでは、**virt vcpupin**と**vcpuinfo**というコマンドを使い、/proc ディレクトリーの情報をチェックしてピン留めを確認します。最後に、verify FV CPU Pinning では、ロードテストを用いて、ゲスト VM の vCPU のワークロードがピン留めされた CPU のみで処理されているかどうかを検証します。

テストの準備

このテストを実行するための特別な条件はありません。

テストの実行

fv_cpu_pinning テストは非インタラクティブです。次のコマンドを実行し、表示されるリストから適切な **fv_cpu_pinning** テスト名を選択します。

```
rhcert-run
```

ランタイム

fv_cpu_pinning テストは、約 5 分で完了します。その他の必須または選択されたテストは、全体の実行時間に追加されます。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.17. FV_LIVE_MIGRATION

テストの対象

fv_live_migration テストは、実行中の仮想マシンをローカルテストサーバー (LTS) に移行するテスト対象ホスト (HUT) の機能をチェックします。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、複数のサブテストを実行し、実行中の仮想マシンの HUT から LTS への移行を完了させます。テストを成功させるには、すべてのサブテストに合格する必要があります。本試験では、HUT が移行の要件を満たしているかどうかを確認し、仮想マシンを設定して SUT 上で起動する。そして、実行中の仮想マシンを HUT から LTS に移行する。移行後、仮想マシンが HUT 上で実行されておらず、LTS 上で動作していることを確認します。最後に、このテストでは、実行中の仮想マシンを LTS から HUT に移行し、仮想マシンが SUT 上で実行されており、LTS 上では実行されていないことを再度確認します。

テストの準備

LTS および HUT が Red Hat Enterprise Linux 8 を実行しており、かつ LTS および SUT に **redhat-certification-hardware** パッケージがインストールされていることを確認する。

LTS および HUT のそれぞれの**/etc/hosts**ファイルにホスト名を追加し、以下のようにホスト名を完全修飾名のエイリアスにします。

- **<HUT の IP アドレス><HUT のホスト名>**
- **<LTS の IP アドレス><LTS のホスト名>**

テストの実行

このテストは非対話型です。現在、このテストは CLI を使用して手動で計画および実行することしかできません。

RHEL 8 の場合:

```
# rhcert-cli plan --add -t fv_live_migration
```

```
# rhcert-cli run -t fv_live_migration --server=<server name>
```

RHEL 9 の場合:

```
# rhcert-cli plan --add -t fv_live_migration
```

```
# rhcert-cli run --test fv_live_migration --server=<server name>
```

ランタイム

テストは 5 分程度で終了します。ただし、LTS および HUT がそれぞれ同一または異なるラボまたはネットワークに属する場合は、時間が減少または増加する可能性があります。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

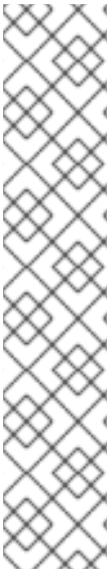
A.18. FV_MEMORY

fv_memory テストは、FV ゲストを起動し、 **memory** テストを実行します。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9



注記

完全仮想化テストを初めて実行するとき、テストツールは FV ゲストファイルを取得する必要があります。テストツールの実行時間は、FV ゲストファイルの転送速度によって異なります。以下に例を示します。

- FV ゲストファイルがローカルテストサーバーにあり、1GbE 以上のネットワークを使用している場合は、約 300MB のゲストファイルを転送するのに約 1、2 分かかります。
- ファイルが CWE API から取得 (ゲストファイルがローカルテストサーバーにインストールされていないか見つからない場合に自動的に取得) される場合、最初の実行時間は CWE API からの転送速度に依存します。

ゲストファイルがテスト対象ホスト (HUT) で利用できる場合は、その後の fv_* テストの実行すべてで使用されます。

関連情報

- テスト方法やランタイムの詳細については、メモリー を参照してください。
- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.19. FV_PCIE_STORAGE_PASSTHROUGH

テストの対象

fv_pcie_storage_passthroughテストは、ホストマシンに搭載された SAS や SATA などの PCIe ベースのストレージデバイスの制御が、仮想マシンに移行できるかどうかを検証するために使用されます。このテストは Red Hat Enterprise Linux 8 でサポートされており、ホストマシン上で実行する必要があります。このテストは、ホストがデバイスパススルーをサポートし、IOMMU が有効になっている場合に自動的に計画されます。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、複数のサブテストを実行して、ホストマシンの HBA デバイスを仮想マシンに割り当て、仮想マシン内でストレージテストを実行します。テストを成功させるには、すべてのサブテストに合格する必要があります。このテストでは、ホストマシンに接続された PCIe デバイスをゲスト仮想マシンにネイティブに表示されるように割り当てることができるかどうかを検証し、パススルー PCIe デバイスを使用するようにゲスト仮想マシンを設定し、仮想マシンを起動して、デバイスが仮想マシン内で期待どおりに機能していることを確認します。

テストの準備

ホストマシンがデバイスパススルーをサポートし、IOMMU が有効になっていることを確認してください。設定するには、Configuring a Host for PCI Passthrough を参照してください。

注記

ホストマシンのルートパーティションがあるストレージデバイスでテストを実行しないでください。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **fv_pcie_storage_passthrough** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストの実行には約 30 分かかります。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.20. FV_USB_NETWORK_PASSTHROUGH

テストの対象

fv_usb_network_passthroughテストは、ホストマシンにある USB 接続のネットワークデバイスの制御が、仮想マシンに移行できるかどうかを検証するために使用します。このテストは、Red Hat Enterprise Linux バージョン 8 以上に対応しており、ホストマシン上で実行する必要があります。このテストは、ホストマシンがデバイスパススルーをサポートし、IOMMU が有効になっている場合に自動的に計画されます。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、複数のサブテストを実行して、ホストマシンの USB デバイスを仮想マシンに取り付け、仮想マシンの内部でネットワークテストを実行します。テストを成功させるには、すべてのサブテストに合格する必要があります。このテストでは、ホストマシンに接続された USB デバイスをゲスト仮想マシンにネイティブに表示されるように割り当てることができるかどうかを検証し、パススルー PCIe デバイスを使用するようにゲスト仮想マシンを設定し、仮想マシンを起動して、デバイスが仮想マシン内で期待どおりに機能していることを確認します。

テストの準備

- USB デバイスが、デバイスパススルーをサポートし、IOMMU が有効になっている HUT に接続されていることを確認します。設定するには、PCI パススルー用のホストを設定する を参照してください。

- HUT には最低 2 つの NIC があり、両方のネットワークが LTS にルーティング可能であることを確認します。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **fv_usb_network_passthrough** テスト名を選択します。

```
rhcert-run
```

注記

ネットワーク帯域幅の問題でテストが失敗した場合は、より高い帯域幅を得るために、仮想マシンに割り当てる CPU や RAM を増やす必要があるかもしれません。

ランタイム

テストの実行には約 90 分かかりますが、USB 機器のサイズや接続速度によって長さは異なります。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.21. FV_USB_STORAGE_PASSTHROUGH

テストの対象

fv_usb_storage_passthroughテストは、ホストマシンの USB 接続ストレージデバイスの制御が、仮想マシンに移行できるかどうかを検証するために使用されます。このテストは Red Hat Enterprise Linux 8 でサポートされており、ホストマシン上で実行する必要があります。このテストは、ホストがデバイスパススルーをサポートし、IOMMU が有効になっている場合に自動的に計画されます。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、ホストマシンの USB デバイスを仮想マシンに取り付け、仮想マシン内でストレージテストを行うという複数のサブテストを行います。テストを成功させるには、すべてのサブテストに合格する必要があります。このテストでは、ホストマシンに接続された USB デバイスをゲスト仮想マシンにネイティブに表示されるように割り当てることができるかどうかを検証し、パススルー PCIe デバイスを使用するようにゲスト仮想マシンを設定し、仮想マシンを起動して、デバイスが仮想マシン内で期待どおりに機能していることを確認します。

テストの準備

USB デバイスが、デバイスパススルーをサポートし、IOMMU が有効になっているホストマシンに接続されていることを確認してください。設定するには、**Red Hat Virtualization Administration Guide**の Configuring a Host for PCI Passthrough セクションを参照してください。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **fv_usb_storage_passthrough** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストの実行には約 90 分かかりますが、USB 機器のサイズや接続速度によって長さは異なります。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.22. FV_PCIE_NETWORK_PASSTHROUGH

テストの対象

fv_pcie_network_passthroughテストは、ホストマシンの NIC、LOM、ALOM などの PCIe ベースのネットワークデバイスの制御が、仮想マシンに移行できるかどうかを検証するために使用されます。このテストは、Red Hat Enterprise Linux バージョン 8 以上に対応しており、ホストマシン上で実行する必要があります。このテストは、ホストマシンがデバイスパススルーをサポートし、IOMMU が有効になっている場合に自動的に計画されます。

RHEL 9.4 以降では、このテストは ARM システムで実行できるようになりました。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、複数のサブテストを実行して、ホストマシンのネットワークデバイスを仮想マシンにアタッチし、仮想マシンの内部でネットワークテストを実行します。テストを成功させるには、すべてのサブテストに合格する必要があります。このテストでは、ホストマシンに接続された PCIe デバイスをゲスト仮想マシンにネイティブに表示されるように割り当てることができるかどうかを検証し、パススルー PCIe デバイスを使用するようにゲスト仮想マシンを設定し、仮想マシンを起動して、デバイスが仮想マシン内で期待どおりに機能していることを確認します。

テストの準備

- テスト対象ホスト (HUT) がデバイスパススルーをサポートし、IOMMU が有効になっていることを確認します。設定するには、**Red Hat Virtualization Administration Guide**の Configuring a Host for PCI Passthrough セクションを参照してください。
- HUT には最低 2 つの NIC があり、両方のネットワークが LTS にルーティング可能であることを確認します。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **fv_pcie_network_passthrough** テスト名を選択します。

```
rhcert-run
```

注記

ネットワーク帯域幅の問題でテストが失敗した場合は、より高い帯域幅を得るために、仮想マシンに割り当てる CPU や RAM を増やす必要があるかもしれません。

ランタイム

テストの実行には約 30 分かかります。

関連情報

- ゲストイメージの詳細は、テスト実行中のゲストイメージのダウンロード を参照してください。

A.23. INFINIBAND 接続

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

Infiniband 接続 テストでは、次のサブテストを実行して、必要に応じて、テストの開始時にドロップダウンから選択した IP アドレスを使用してベースライン機能を確認します。

1. Ping テスト
   HUT でテスト中のデバイスの開始 IP アドレスから、LTS の選択した IP アドレスへ ping を実行します。
2. ping テスト
   選択された LTS の IP アドレスを使用して LTS および HUT 上で rping を実行し、結果を比較して完了まで実行されたことを確認します。
3. Rcopy テスト
   LTS と HUT で rcopy を実行し、ランダムに生成されたファイルを送信し、LTS と SUT で md5sum を比較して、転送が成功したことを確認します。
4. Rdma-ndd サービステスト
   サービスの停止、開始、再起動コマンドが期待通りに機能することを確認します。
5. Opensm サービステスト
   サービスの停止、開始、再起動コマンドが期待通りに機能することを確認します。
6. LID 検証テスト
   デバイスの LID が設定されており、デフォルト値ではないことを確認します。
7. Smpquery テスト

LTS でデバイスとポートを使用して smpquery を実行し、デバイス/ポートがファブリックに登録されているかどうかを確認します。

8. ib_write_bw テスト
   HUT から LTS の選択された IP アドレスに対して ib_write_bw を実行し、InfiniBand の書 き込み帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証する。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

9. ib_read_bw test
   HUT から LTS の選択された IP アドレスに対して ib_read_bw を実行し、InfiniBand の読 み取り帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証する。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

10. ib_send_bw test
    HUT から LTS の選択した IP アドレスに対して ib_send_bw を実行して、InfiniBand 送信帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

テストの準備

- LTS と HUT が、同じファブリック上の別々のマシンであることを確認します。

テストの実行

このテストは、インタラクティブなテストです。次のコマンドを実行し、表示されたリストから適切な **infiniband connection** テスト名を選択します。

```
rhcert-run
```

テストを実行する IP アドレス (LTS の IP アドレス) を選択するためのドロップダウンが表示されます。テストを実行している SUT デバイスと同じファブリック上のデバイスに対応する IP アドレスを選択します。

表A.1 手動でのテストの追加と実行

レートタイプ	infiniband 接続テストを手動で追加するコマンド	infiniband 接続テストを手動で実行するコマンド
Infiniband_QDR	rhcert-cli plan --add --test Infiniband_QDR --device <devicename>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test Infiniband_QDR --server <LTS IP addr>

レートタイプ	infiniband 接続テストを手動で追加するコマンド	infiniband 接続テストを手動で実行するコマンド
Infiniband_FDR	rhcert-cli plan --add --test Infiniband_FDR --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test Infiniband_FDR --server <LTS IP addr>
Infiniband_EDR	rhcert-cli plan --add --test Infiniband_EDR --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test Infiniband_EDR --server <LTS IP addr>
Infiniband_HDR	rhcert-cli plan --add --test Infiniband_HDR --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test Infiniband_HDR --server <LTS IP addr>
Infiniband_NDR	rhcert-cli plan --add --test Infiniband_NDR --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test Infiniband_NDR --server <LTS IP addr>
Infiniband_Socket_Direct	rhcert-cli plan --add --test Infiniband_Socket_Direct	rhcert-cli run --test Infiniband_Socket_Direct --server <LTS IP addr>

<device name>、**<port number>**、**<net device>**、および **<LTS IP addr>** は、適切な値に置き換えます。

ランタイム

このテストの実行には 10 分もかかりません。

関連情報

- InfiniBand と RDMA の詳細については、Understanding InfiniBand and RDMA technologies を参照してください。

A.24. INTEL_SST

テストの対象

intel_sst テストは、Intel の SST (Speed Select Technology) 機能を実行する CPU 周波数スケーリングテストです。この機能を利用して、CPU とワークロードを一致させるために、コアごとのパフォーマンスをカスタマイズし、パフォーマンスごとに割り当てることができます。これにより、対象となるアプリケーションのパフォーマンスをランタイムで向上させることができます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

intel_sst testは SST に対応したシステムでのみ実行され、以下の機能をサポートしています。

- Speed Select Base Freq (SST-BF) - 他のコアのベース周波数 (P1) を下げることで、特定のコアのベース周波数 (P1) を高くすることができます。
- Frequency Prioritization (SST-CP) - 優先度の低いソフトウェアを実行しているコアの周波数を下げることで、特定のコアのクロックを高くすることができます。

このテストでは、上記の機能がシステムでサポートされ、設定されているかどうかを確認します。その結果に基づいて、それぞれの機能のサブテストを 1 つだけ実行します。

テストの準備

このテストは、インテルチップセットアーキテクチャーでのみ実行する必要があります。

- Intel® SST-BF 機能を Red Hat Enterprise Linux(RHEL) ベースのプラットフォームで使用するには、次のようにします。
  前提条件
  1. BIOS で Intel® SST-BF 機能を有効にする
  2. カーネルパラメーターの設定 -**intel_idle.max_cstate=1**
- Intel® SST-CP 機能を Red Hat Enterprise Linux(RHEL) ベースのプラットフォームで使用するには、次のようにします。
  前提条件
  1. BIOS で Intel® SST-CP 機能を有効にする
  2. カーネルパラメーターの設定 -**intel_idle.max_cstate=1 intel_pstate=disable**

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **intel_sst** テスト名を選択します。

```
rhcert-run
```

-

ランタイム

このテストは約 5 分で完了します。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.25. IPXE

iPXE テストの対象

iPXE テストは、x86 Red Hat Enterprise Linux (RHEL) システムで実行されるインタラクティブなテストです。システムは UEFI ブートモードで起動するはずです。

efi directory が存在する場合は、マシンが UEFI ブートモードで実行されています。以下のコマンドを実行して、マシンが UEFI モードで実行しているかどうかを確認します。

```
ls /sys/firmware/efi/
```

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

iPXE は、主要なオープンソースネットワークブートファームウェアです。HTTP、SAN、ワイヤレスネットワークからのブートなどの追加機能により、完全な PXE 実装が強化されます。このテストは、基礎となる NIC が HTTP ブートを使用して iPXE をサポートしているかどうかを確認します。

iPXE の実行中に、テストサーバーは起動可能なイメージを返しません。ブート画面では **起動できない**エラーが表示されます。これは 予想されるエラーメッセージです。テストサーバーは、RHEL OS の次回のブートローダーを使用して起動します。

テストの準備

- テスト対象ホストが UEFI ブートモードにあることを確認します。iPXE は最初に検出したインターフェイスをテストするため、テスト対象ホストで、テストする必要のあるインターフェイスが接続されていることを確認します。
- このテストの実行中に **httpd** サービスがテストサーバーで実行されていないことを確認します。このテストは、テストサーバーとの通信にポート 80 を使用します。

テストの実行

1. 次のコマンドを実行し、表示されるリストから適切な **iPXE** テスト名を選択します。

   ```
   rhcert-run
   ```

2. ipxe テストはテストプランには表示されないので、以下のコマンドを使用してそれぞれ手動で計画し、手動で実行する必要があります。

   ```
   # rhcert-cli plan --add --test iPXE
   ```

```
# rhcert-cli run --test iPXE
```

3. テストでは、最初に iPXE テストのテスト対象ホスト (HUT) を設定します。HUT の MAC の詳細を保存し、ipxe バイナリーで新しいブートローダーを作成し、ブートローダーを次回のブートとしてマークします。その後、再起動を求めるプロンプトが表示されるので、Yes を押して続行します。テストサーバーは、reboot コマンド の送信後に waiting for a response を表示します。

4. HUT は新しいブートローダーで再起動され、次に iPXE プロンプトがロードされ、GET リクエストが実行されてテストサーバーに到達できるかどうかが確認されます。これは単なる GET リクエストなので、ブートは失敗し、システムは次のブートローダー (RHEL OS) にフォールバックします。

5. テストサーバーは、テスト対象ホストを継続的に監視して、再起動したかどうかを確認します。再起動すると、テストが続行します。このテストは、最初に iPXE に対して行われたブートの変更を元に戻し、次に iPXE ブートが成功したかどうかを確認します。

6. iPXE ブートの GET 要求から受け取った MAC アドレスを、すでに保存されている MAC アドレスと比較します。MAC が iPXE テストと一致すると、テストは成功です。

ランタイム

このテストの実行には 5 分もかかりません。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.26. IWARP 接続

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

IWarp 接続 テストは次のサブテストを実行し、必要に応じてテストの開始時にドロップダウンから選択した IP アドレスを使用してベースライン機能を確保します。

1. Ping テスト - HUT でテストされているデバイスの開始 IP アドレスから、LTS の選択された IP アドレスまで ping を実行します。

2. Rping テスト - 選択した LTS IP アドレスを使用して LTS および HUT で rping を実行し、結果を比較して完了まで実行されたことを確認します。

3. Rcopy test - LTS と HUT で rcopy を実行し、ランダムに生成されたファイルを送信し、LTS と SUT で md5sum を比較して、転送が成功したことを確認します。

4. Ethtool test - roce デバイスの検出されたネットデバイスを渡して、ethtool コマンドを実行します。

5. ib_write_bw テスト
   HUT から LTS の選択した IP アドレスに対して ib_write_bw を実行して、IWrap 書き込み帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証します。

6. ib_read_bw test
   HUT から LTS の選択した IP アドレスに対して ib_read_bw を実行して、IWrap 読み取り帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証します。

7. ib_send_bw test
   HUT から LTS の選択した IP アドレスに対して ib_send_bw を実行して、IWrap 送信帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証します。

テストの準備

- LTS と HUT が、同じファブリック上の別々のマシンであることを確認します。

テストの実行

このテストは、インタラクティブなテストです。次のコマンドを実行し、表示されるリストから適切な **iwrap connection** テスト名を選択します。

```
rhcert-run
```

テストを実行する IP アドレス (LTS の IP アドレス) を選択するためのドロップダウンが表示されます。テストを実行している SUT デバイスと同じファブリック上のデバイスに対応する IP アドレスを選択します。

表A.2 手動でのテストの追加と実行

速度タイプ	IWarpConnection テストを手動で追加するコマンド	IWarpConnection テストを手動で実行するコマンド
10GigiWarp	rhcert-cli plan --add --test 10GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 10Gigiwarp --server <LTS IP addr>
20GigiWarp	rhcert-cli plan --add --test 20GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 20GigiWarp --server <LTS IP addr>
25GigiWarp	rhcert-cli plan --add --test 25GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 25GigiWarp --server <LTS IP addr>

速度タイプ	IWarpConnection テストを手動で追加するコマンド	IWarpConnection テストを手動で実行するコマンド
40GigiWarp	rhcert-cli plan --add --test 40GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 40GigiWarp --server <LTS IP addr>
50GigiWarp	rhcert-cli plan --add --test 50GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 50GigiWarp --server <LTS IP addr>
100GigiWarp	rhcert-cli plan --add --test 100GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 100GigiWarp --server <LTS IP addr>
200GigiWarp	rhcert-cli plan --add --test 200GigiWarp --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 200GigiWarp --server <LTS IP addr>

<device name>、**<port number>**、**<net device>**、および **<LTS IP addr>** は、適切な値に置き換えます。

ランタイム

このテストの実行には 10 分もかかりません。

関連情報

- InfiniBand と RDMA の詳細については、Understanding InfiniBand and RDMA technologies を参照してください。

A.27. KDUMP

テストの対象

kdump テストは、**kdump** サービスを使用して、システムがクラッシュ後に **vmcore** ファイルをキャプチャできること、およびキャプチャされたファイルが有効であることを確認します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストには以下のサブテストが含まれます。

- kdump with local: **kdump** サービスを使用して、このサブテストは次のタスクを実行します。
  - テスト中のホスト (HUT) をクラッシュさせます。
  - **vmcore** ファイルをローカルの **/var/crash** ディレクトリーに書き込みます。
  - **vmcore** ファイルを検証します。
- kdump with NFS: **kdump** サービスを使用して、このサブテストは次のタスクを実行します。
  - **/var/rhcert/export** ファイルシステムを HUT の **/var/crash** ディレクトリーにマウントします。このファイルシステムは、テストサーバーから NFS 経由で共有されます。
  - HUT をクラッシュさせます。
  - **vmcore** ファイルを **/var/crash** ディレクトリーに書き込みます。
  - **vmcore** ファイルを検証します。

テストの準備

- テストを実行する前に、HUT がテストサーバーに接続されていることを確認します。
- **rhcertd** プロセスがテストサーバーで実行されていることを確認します。認定テストスイートは、NFS ファイルシステムを自動的に準備します。スイートが環境をセットアップできない場合、テストは失敗します。

注記

このテストを実行するために、テスト対象の RHEL 7 ホストで RHEL 8 テストサーバーを使用している場合は、テストスイートがこのサービスを自動的に開始しないため、**httpd** サービスを手動で開始する必要があります。

テストの実行

1. HUT にログインします。
2. **kdump** テストを実行します。
   - **rhcert-run** コマンドを使用するには、次の手順を実行します。

i. **rhcert-run** コマンドを実行します。

```
# rhcert-run
```

ii. **kdump** テストを選択します。
テストは、両方のサブテストを順番に実行します。

- **rhcert-cli** コマンドを使用するには、両方のサブテストを順番に実行するか、サブテストを指定するかを選択します。
  - **両方** のサブテストを順番に実行するには、次のコマンドを使用します。

```
# rhcert-cli run -test=kdump –server=<test server's IP>
```

  - **kdump with local** サブテストのみを実行するには、次のコマンドを使用します。

```
# rhcert-cli run -test=kdump -device=local
```

  - **kdump with NFS** サブテストのみを実行するには、次のコマンドを使用します。

```
# rhcert-cli run -test=kdump -device=nfs –server=<test server's IP>
```

さらに、NFS を使用した kdump テストの場合は、テストサーバーで次のコマンドを実行します。

```
# rhcertd start
```

3. クラッシュ後、HUT が再起動するまで待ちます。
kdump サービスは、**vmcore** ファイルを **/var/crash** ディレクトリーに保存する際にいくつかのメッセージを表示します。**vmcore** ファイルが保存されると、HUT が再起動します。

4. 再起動後に HUT にログインすると、rhcert スイートは **vmcore** ファイルが存在するかどうか、およびそれが有効かどうかを確認します。ファイルが存在しないか無効な場合、テストは失敗します。

サブテストを順番に実行している場合、**NFS サブテストを使用した** kdump は、前の **vmcore** ファイルの検証が完了した後に開始されます。

ランタイム

kdump テストの実行時間は、HUT の RAM の量、テストサーバーと HUT のディスク速度、テストサーバーへのネットワーク接続速度、HUT の再起動にかかる時間などの要因によって異なります。

8GB の RAM、7200 回転の 6Gb/s SATA ドライブ、テストサーバーへのギガビットイーサネット接続、1.5 分の再起動時間を備えた 2013 年製のワークステーションの場合、**local kdump** テストは、再起動を含めて約 4 分で完了します。2013 年に発売された同じワークステーションでも、同様の設備を持つネットワークテストサーバーに対して、**NFS kdump** テストを約 5 分で完了させることができます。**supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.28. LID

テストの対象

この **lid** テストは、ディスプレイが内蔵されていて、蓋が開閉できるシステムにのみ有効です。lid は、名前に "lid" が含まれるデバイスを udev データベースで検索することで検出されます。

```
E: NAME="Lid Switch"
```

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、システムが udev のパラメーターによって蓋の閉まっている状態と開いている状態を判断し、蓋が閉まっているときにディスプレイのバックライトをオフにすることができることを確認します。

テストの準備

テストの準備として、電源管理の設定で、蓋を閉めたときにシステムがスリープ状態や休止状態にならないようにしてください。Red Hat Enterprise Linux 7 では、**Tweak Tool** を使用して、蓋を閉じる際のサスペンドまたはハイパーネートを無効にします。蓋が開いていることを確認してから、試運転を開始してください。

テストの実行

蓋のテストは対話式です。次のコマンドを実行し、表示されるリストから適切な **lid** テスト名を選択します。

```
rhcert-run
```

テストを開始する準備ができているかどうかを尋ねられますので、**Yes**と答えて続行してください。バックライトがオフになるのを確認しながら、プロンプトが表示されたら蓋を閉めます。ノートパソコンを閉じた状態で、キーボードと蓋の間のわずかな隙間からバックライトがオフになったことを確認する必要があるかもしれません。バックライトが消灯する場合は**Yes**、バックライトが消灯しない場合は**No**と答えてください。

ランタイム

蓋テストには約 30 秒かかりますが、これはバックライトがオフになる程度に蓋を閉める時間です。このテストはラップトップで実行されるため、サスペンドテストは、実行ごとに必要な **supportable**テストに付随する必要があります。サスペンドテストでは、各テストの実行に約 6 分が追加され、**supportable** ではさらに 1 分追加されます。

A.29. MEMORY

メモリーテストの内容

memory テストは、システム RAM のテストに使用されます。USB フラッシュメモリー、SSD ストレージデバイス、その他のタイプの RAM ベースのハードウェアのテストは行っていません。メインメモリーのみをテストします。

HUT が RHEL の最低要求メモリー基準を満たしているかどうかを確認するために、**CPU コアごとのメモリーチェック**がプランニングプロセスに追加されました。これは、メモリー、コア、リアルタイム、完全仮想化など、いくつかのハードウェア認証テストのプラニング条件となっています。

CPU コアごとのメモリーチェックに合格しなかった場合、上記のテストは自動的に計画されません。しかし、これらのテストは CLI を使用して手動で計画することができます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

このテストが行うこと: このテストでは、**/proc/meminfo** ファイルを使用して、システムに搭載されているメモリーの量を調べます。インストールされている量がわかると、システムのアーキテクチャーが 32 ビットか 64 ビットかを確認します。そして、スワップスペースが利用可能かどうか、またはスワップパーティションがないかどうかを判断します。このテストは、システムにスワップファイルがあるかどうかによって、若干異なる設定で 1 回または 2 回実行されます。

1. スワップが利用可能な場合は、システムに実際に搭載されているよりも多くの RAM をメモリーテストに割り当てます。これにより、実行中にスワップスペースが強制的に使用されます。
2. スワップの有無にかかわらず、メモリーテストには可能な限り多くの RAM を割り当てますが、OOM(Out of Memory) キルが発生する限界を超えないようにします。このバージョンのテストは常に実行されます。

メモリーテストの反復では、**malloc()** は RAM の割り当てに使用され、RAM は任意の 16 進文字列 (0xDEADBEEF) の書き込みでダーティーになり、0xDEADBEEF が想定されるアドレスの RAM に保存されるようにテストが実行されます。テストが完了すると、**free()** を呼び出して RAM を解放します。プロセスサイズが、テストするメモリーの量よりも大きいか小さいかによって、複数のスレッドまたは複数のプロセスが RAM の割り当てに使用されます。

テストの準備

ポリシーガイドのルールに従って、システムに正しい RAM 容量をインストールします。

テストの実行

メモリーテストは非対話的です。次のコマンドを実行し、表示されるリストから適切な **memory** テスト名を選択します。

```
rhcert-run
```

ランタイム、ベアメタル

メモリーテストは、2013 年モデルのシングル CPU、6 コア/12 スレッド 3.3GHz Intel ベースで RAM が 8GB の、AMD64、および Intel 64 の Red Hat Enterprise Linux を実行するワークステーションで、実行するのに約 16 分かかります。テストは、より多くの RAM を持つシステムで時間がかかります。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

ランタイム、フルバータイプのゲスト

fv_memory テストは、ゲストで実行するために、ベアメタルバージョンよりもわずかに長く、18 分程度かかります。この時間は、ゲストのスタートアップ/シャットダウンのアクティビティーと、ゲスト内で実行される必要な **supportable** テストによるものです。ベアメタルシステムで必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。fv_memory テストの実行時間は、ビルド済みゲ

ストに割り当てられた RAM の量が常に同じであるため、ベアメタルメモリーテストほどマシンによる大きな個体差は生じません。基礎となる実際のシステムの速度によって変化しますが、テスト中に使用される RAM の量はマシンごとに変わることはありません。

Creating and Activating Swap for EC2 パートナーは、以下の手順に従って EC2 のスワップを作成およびアクティベートできます。

```
sudo dd if=/dev/zero of=/swapfile bs=1M count=8000
chmod 600 /swapfile
mkswap /swapfile
swapon /swapfile
swapon -s
edit file /etc/fstab and add the following line:
/swapfile swap swap defaults 0 0
write file and quit/exit
```

A.29.1. memory_HBM

Memory_HBM テストの対象となる内容

Memory_HBM テストは、高帯域幅メモリー (HBM) が存在するシステム上でシステムの高帯域幅メモリー (HBM) をテストするために使用されます。3 つの可能なテストのうち 1 つは、HBM 動作モードに基づいて計画されます。システム HBM がサポートされていない場合は、代わりに定期的な **メモリー** テストが計画されています。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

Memory_HBM テストは、HBM が存在するシステム専用のメモリーテストです。

テストの準備

システム HBM が、ポリシーガイドに指定されている要件を満たしていることを確認してください。

テストの実行

1. **rhcert-cli plan** を実行します。HBM 設定が要件と次のいずれかの条件を満たしている場合、memory_HBM テストの 1 つが計画されます。
   - **Memory_HBM_only** : システムに DIMM が設置されていない
   - **Memory_HBM_cache** : HBM が DIMM へのキャッシュとして機能する
   - **Memory_HBM_ flat**: メモリーの合計量として DIMM と HBM が使用可能である
2. テストを実行するには、コマンド **rhcert-cli run --test** を使用します。たとえば、**rhcert-cli run --test hwcert/memory_HBM_cache** は memory_HBM_cache テストを実行します。
3. テストの指示に従ってください。
4. **rhcert-print** を使用して結果を確認します。

5. 結果を保存するには、**rhcert-save** を使用します。

A.30. NETWORK

テストの対象

ネットワークテストは、TCP/IP ネットワーク上でデータを転送するデバイスをチェックします。このテストは、以下の表に記載されているように、デバイスをチェックするために設計された、対応するテストをもとに、有線デバイスと無線デバイスの両方の接続速度と帯域幅を確認できます。

ネットワークテスト内の各種テスト

イーサネットテスト	説明
1GigEthernet	1 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
10GigEthernet	10 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
20GigEthernet	20 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
25GigEthernet	25 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
40GigEthernet	40 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
50GigEthernet	50 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
100GigEthernet	100 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
200GigEthernet	200 ギガビットイーサネット接続向けに速度検出が追加されたネットワークテスト。
イーサネット	Ethernet テストがローカルテスト計画に記載されている場合は、テストスイートで、そのデバイスの速度が認識されなかったことを示します。特定のデバイスをテストする前に、接続を確認します。

ワイヤレステスト	説明
WirelessG	802.11g ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。

ワイヤレステスト	説明
WirelessN	802.11n ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。
WirelessAC	802.11ac ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。
WirelessAX(WiFi6 による Supersed by WiFi6)	802.11ax ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。
WiFi6	802.11ac ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。
WiFi6E	802.11ac ワイヤレスイーサネット接続向けに速度検出が追加されたネットワークテスト。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは以下のサブテストを実行して、すべてのネットワークデバイスに関する情報を収集します。

1. インターフェイスでのバウンステストは、**nmcli conn up** および **nmcli conn down** コマンド (RHEL 7 では **ifup**、**ifdown**) を使用して実行されます。
2. ルートパーティションが NFS または iSCSI マウントされていない場合、バウンステストはインターフェイスで実行されます。さらに、トラフィックがテスト対象のインターフェイスを介してルーティングされるようにするために、テストされない他のインターフェイスはすべてシャットダウンされます。
3. ルートパーティションが NFS または iSCSI マウントされている場合、iSCSI または NFS 接続を担当するインターフェイスでのバウンステストはスキップされ、iSCSI または NFS 接続を処理するインターフェイスを除く他のインターフェイスはすべてシャットダウンされます。
4. テストファイルは **/dev/urandom** の場所に作成され、そのサイズは NIC の速度で調整されます。
5. TCP および UDP テスト - テストは iperf ツールを使用して以下を行います。
    a. テストサーバーと、テスト中のホスト間で TCP レイテンシーをテストします。このテストでは、システムで OS タイムアウトが発生しているかどうかがチェックされ、発生している場合は失敗します。
    b. テストサーバーと、テスト中のホスト間の帯域幅をテストします。有線デバイスの場合、速度を理論上の最大値に近づけることを推奨します。

c. テストサーバーと、テスト中のホスト間で UDP レイテンシーをテストします。このテストでは、システムで OS タイムアウトが発生しているかどうかがチェックされ、発生している場合は失敗します。

6. ファイル転送テスト - テストでは、SCP を使用してファイルをテスト対象のホストからリモートシステムまたはテストサーバーに転送し、その後、そのファイルをテスト対象のホストに戻して、転送が適切に機能するかどうかを確認します。

7. ICMP (ping) テスト: スクリプトは、デフォルトのパケットサイズで ping フラッドを発生させ、システムに障害がないことを確認します (システムの再起動、リセットなど、ping フラッドに耐えられないことを示すものは何もありません)。5000 パケットが送信され、100% の成功率が期待されます。テストは、許容可能な成功率を得るまで 5 回再試行されます。

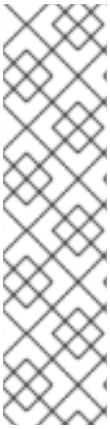
8. 最後に、テストは、テストが実行されたときの元の状態 (アクティブまたは非アクティブ) にすべてのインターフェイスを戻ります。

有線デバイスのテストの準備

各テストの実行に必要な数だけネットワークデバイスをテストできます。

作業を開始する前に:

- 各デバイスをネイティブ (最大) 速度で接続していることを確認してください。そうでない場合にはテストは失敗します。
- テストサーバーが稼働していることを確認します。
- 各ネットワークデバイスに、静的または DHCP 経由で動的に IP アドレスが割り当てられていることを確認します。
- iperf ツールが TCP および UDP サブテストを実行するために、複数のファイアウォールポートが開いていることを確認します。



注記

デフォルトでは、ポート 52001-52101 が開放されています。デフォルトのポートを変更する場合は、**/etc/rhcert.xml** 設定ファイルの **iperf-port** と **total-iperf-ports** の値を更新します。

以下に例を示します。

<server listener-port="8009" iperf-port="52001" total-iperf-ports="100">

ファイアウォールポートが開いていない場合、テストの実行中にファイアウォールポートを開くように求めるプロンプトが表示されます。

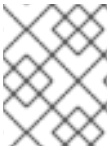
分割可能なネットワーク

このテストでは、フルスピードでのデータ転送とパーテイx本機能をチェックすることにより、いずれかのネットワークデバイスがパーティションをサポートしているかどうかをチェックします。

NIC のパフォーマンスに基づいてテストを実行します。

- パーティション分割中に NIC がフルスピードで実行される場合は、NIC がネイティブ速度で実行されるパーティションを設定し、その設定でネットワークテストを実行します。

- パーティショニング中に NIC がフルスピードで実行されない場合: 1) テストを 2 回実行します。1回目は、パーティションなしで実行してフルスピードの動作を確認し、2 回目はパーティショニングを有効にして実行し、パーティション機能を確認します。



注記

Red Hat は、既存のネットワーク速度テストのいずれかに適合するように、パーティション設定に 1Gb/s または 10Gb/s のいずれかを選択することを推奨します。

ワイヤレスイーサネットデバイスのテストの準備

テスト対象のワイヤレスカードに基づいて、接続するワイヤレスアクセスポイントには、WirelessG、WirelessN、WirelessAC、WirelessAX、WiFi6、および WiFi6E ネットワークテストを実行する機能が必要です。

テストの実行

ネットワークテストは非対話的です。次のコマンドを実行し、表示されるリストから適切な **network** テスト名を選択します。

```
rhcert-run
```

表A.3 手動でのテストの追加と実行

速度タイプ	イーサネットテストを手動で追加するコマンド	イーサネットテストを手動で実行するコマンド
1GigEthernet	`rhcert-cli plan --add --test 1GigEthernet --device <device name>`	`rhcert-cli run --test 1GigEthernet --server <LTS IP addr>`
10GigEthernet	`rhcert-cli plan --add --test 10GigEthernet --device <device name>`	`rhcert-cli run --test 10GigEthernet --server <LTS IP addr>`
20GigEthernet	`rhcert-cli plan --add --test 20GigEthernet --device <device name>`	`rhcert-cli run --test 20GigEthernet --server <LTS IP addr>`
25GigEthernet	`rhcert-cli plan --add --test 25GigEthernet --device <device name>`	`rhcert-cli run --test 25GigEthernet --server <LTS IP addr>`
40GigEthernet	`rhcert-cli plan --add --test 40GigEthernet --device <device name>`	`rhcert-cli run --test 40GigEthernet --server <LTS IP addr>`

速度タイプ	イーサネットテストを手動で追加するコマンド	イーサネットテストを手動で実行するコマンド
50GigEthernet	rhcert-cli plan --add --test 50GigEthernet --device <device name>	rhcert-cli run --test 50GigEthernet --server <LTS IP addr>
100GigEthernet	rhcert-cli plan --add --test 100GigEthernet --device <device name>	rhcert-cli run --test 100GigEthernet --server <LTS IP addr>
200GigEthernet	rhcert-cli plan --add --test 200GigEthernet --device <device name>	rhcert-cli run --test 200GigEthernet --server <LTS IP addr>
400GigEthernet	rhcert-cli plan --add --test 400GigEthernet --device <device name>	rhcert-cli run --test 400GigEthernet --server <LTS IP addr>

<device name> および **<LTS IP addr>** は、適切な値に置き換えます。

ランタイム

ネットワークテストは、各 PCIe ベースのギガビット、有線イーサネットカードのテストに約 2 分かかります。また、必要な supportable テストでは、全体的なランタイムに約 1 分かかります。

関連情報

- 残りのテスト機能の詳細は、イーサネットテスト を参照してください。

A.31. NETWORKMANAGEABLECHECK

テストの対象

NetworkManageableCheck テストは、システムで利用可能なすべてのネットワークインターフェイスに対して実行されます。

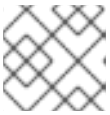
RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

テストは、次のタスクを実行する 2 つのサブテストで設定されます。

1. BIOS デバイス名をチェックして、インターフェイスがファームウェアで設定された用語に従っていることを確認します。



注記

BIOS デバイス名の検証は、x86 システム上でのみ実行されます。

2. 現在のネットワーク管理ステータスを評価するために、ネットワークマネージャーがインターフェイスを管理しているかどうかを確認します。

テストの実行

NetworkManageableCheck テストは必須です。このテストは、ネットワークインターフェースの検証と確認を行うために、**サポート可能な** セルフチェックテストを使用してプランニング、実行されます。

ランタイム

このテストの実行には約 1 分かかります。ただし、テストにかかる時間は、システムの詳細とインターフェイス数によって異なります。

A.32. NVME OVER FABRIC テスト

NVMe over Fabrics(NVMe-oF および non-volatile memory express over fabrics とも呼ばれる) は、NVMe プロトコルを使用して、ネットワークファブリック全体でホストをストレージに接続するように設計されたプロトコル仕様です。

このプロトコルは、ホストコンピューターとターゲットのソリッドステートストレージデバイスまたはシステム間のデータ転送を可能にするように設計されています。NVMe メッセージベースのコマンドで実現されます。データ転送は、Ethernet や InfiniBand などの方法で転送できます。

A.32.1. nvme_infiniband

テストの対象

nvme_infiniband テストは、RDMA ネットワークを介した NVMe SSD ドライブへのアクセスと使用を検証します。テスト対象のホストは NVMe クライアントとして設定され、lab エージェントシステムは NVMe ターゲットとして設定されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

1. 必要なカーネルモジュールが読み込まれており、NVMe クライアントが NVMe ターゲットに接続されていることを確認します。
2. discovery、disconnect および connect コマンドを実行して、NVMe ターゲットとクライアント間の接続を確立して確認します。

3. ターゲットシステムのストレージデバイスへの接続に使用されるネットワークインターフェイスを検出し、さらに各 STORAGE および infiniband 接続 テストから 1 種類のテストを実行します。

NVMe over Fabric ストレージテストでは、テストは NVMe クライアントシステムで実行されますが、NVMe デバイスは物理的に NVMe ターゲットホストに置かれます。NVMe クライアントおよびサーバーの両方が RDMA プロトコルを使用して通信します。

テストの準備

テストを開始する前に、以下のことを確認してください。

- NVMe ターゲットシステムおよび NVMe クライアントシステムが適切に設定され、RDMA ネットワークに属しています。
- NVMe クライアントと NVMe ターゲットは、同じ RHEL バージョンを実行しています。そうでない場合、RHEL 9.0 上の NVMe クライアントと RHEL 8.5 上の NVMe ターゲット間の通信で、**Invalid MNAN value 1024 attempting nvme connect**と同様のエラーが発生します。

テストの実行

このテストは非対話型です。現在、このテストは、CLI からのみ計画し、実行できます。

ランタイム

このテストの実行には約 15 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.32.2. nvme_iwarp

テストの対象

nvme_iwarp テストは、RDMA ネットワークを介した NVMe SSD ドライブへのアクセスと使用を検証します。このテストは、RHEL 8 での実行に対応しています。テスト対象のホストは NVMe クライアントとして設定され、lab エージェントシステムは NVMe ターゲットとして設定されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

1. 必要なカーネルモジュールが読み込まれており、NVMe クライアントが NVMe ターゲットに接続されていることを確認します。
2. discovery、disconnect および connect コマンドを実行して、NVMe ターゲットとクライアント間の接続を確立して確認します。
3. ターゲットシステムのストレージデバイスへの接続に使用されるネットワークインターフェイスを検出し、さらに各 STORAGE および iwarp 接続 テストから 1 種類のテストを実行します。

NVMe over Fabric ストレージテストでは、テストは NVMe クライアントシステムで実行されますが、NVMe デバイスは物理的に NVMe ターゲットホストに置かれます。NVMe クライアントおよびサーバーの両方が RDMA プロトコルを使用して通信します。

テストの準備

テストを開始する前に、以下を確認します。

- NVMe クライアントが RHEL 8.x または 9.x を実行している。
- NVMe ターゲットシステムおよび NVMe クライアントシステムが適切に設定され、RDMA ネットワークに属している。
- NVMe クライアントおよび NVMe ターゲットが同じ RHEL バージョンを実行している。それ以外の場合は、RHEL 9.0 の NVMe クライアントと RHEL 8.5 の NVMe ターゲット間の通信中にエラー (**Invalid MNAN value 1024 attempting nvme connect**) が発生します。

テストの実行

このテストは非対話型です。現在、このテストは、CLI からのみ計画し、実行できます。

ランタイム

このテストの実行には約 15 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.32.3. nvme_omnipath

テストの対象

nvme_omnipath テストは、RDMA ネットワークを介した NVMe SSD ドライブへのアクセスと使用を検証します。このテストは、RHEL 8 での実行に対応しています。テスト対象のホストは NVMe クライアントとして設定され、lab エージェントシステムは NVMe ターゲットとして設定されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

1. 必要なカーネルモジュールが読み込まれており、NVMe クライアントが NVMe ターゲットに接続されていることを確認します。
2. discovery、disconnect および connect コマンドを実行して、NVMe ターゲットとクライアント間の接続を確立して確認します。
3. ターゲットシステムのストレージデバイスへの接続に使用されるネットワークインターフェイスを検出し、さらに各 STORAGE および omnipath 接続 テストから 1 種類のテストを実行します。

NVMe over Fabric ストレージテストでは、テストは NVMe クライアントシステムで実行されますが、NVMe デバイスは物理的に NVMe ターゲットホストに置かれます。NVMe クライアントおよびサーバーの両方が RDMA プロトコルを使用して通信します。

テストの準備

テストを開始する前に、以下を確認します。

- NVMe クライアントが RHEL 8.x または 9.x を実行している。
- NVMe ターゲットシステムおよび NVMe クライアントシステムが適切に設定され、RDMA ネットワークに属している。
- NVMe クライアントおよび NVMe ターゲットが同じ RHEL バージョンを実行している。それ以外の場合は、RHEL 9.0 の NVMe クライアントと RHEL 8.5 の NVMe ターゲット間の通信中にエラー (**Invalid MNAN value 1024 attempting nvme connect**) が発生します。

テストの実行

このテストは非対話型です。現在、このテストは、CLI からのみ計画し、実行できます。

ランタイム

このテストの実行には約 15 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.32.4. nvme_roce

テストの対象

nvme_roce テストは、RDMA ネットワークを介した NVMe SSD ドライブへのアクセスと使用を検証します。このテストは、RHEL 8 での実行に対応しています。テスト対象のホストは NVMe クライアントとして設定され、lab エージェントシステムは NVMe ターゲットとして設定されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

1. 必要なカーネルモジュールが読み込まれており、NVMe クライアントが NVMe ターゲットに接続されていることを確認します。
2. discovery、disconnect および connect コマンドを実行して、NVMe ターゲットとクライアント間の接続を確立して確認します。
3. ターゲットシステムのストレージデバイスへの接続に使用されるネットワークインターフェイスを検出し、さらに各 STORAGE および RoCE 接続 テストから 1 種類のテストを実行します。

NVMe over Fabric ストレージテストでは、テストは NVMe クライアントシステムで実行されますが、NVMe デバイスは物理的に NVMe ターゲットホストに置かれます。NVMe クライアントおよびサーバーの両方が RDMA プロトコルを使用して通信します。

テストの準備

テストを開始する前に、以下を確認します。

- NVMe クライアントが RHEL 8.x または 9.x を実行している。
- NVMe ターゲットシステムおよび NVMe クライアントシステムが適切に設定され、RDMA ネットワークに属している。

- NVMe クライアントおよび NVMe ターゲットが同じ RHEL バージョンを実行している。それ以外の場合は、RHEL 9.0 の NVMe クライアントと RHEL 8.5 の NVMe ターゲット間の通信中にエラー (**Invalid MNAN value 1024 attempting nvme connect**) が発生します。

テストの実行

このテストは非対話型です。現在、このテストは、CLI からのみ計画し、実行できます。

ランタイム

このテストの実行には約 15 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.32.5. nvme_tcp

テストの対象

nvme_tcp テストは、TCP ネットワークを介した NVMe SSD ドライブへのアクセスと使用を検証します。このテストは現在テクノロジープレビューとして提供されており、RHEL 8 での実行に対応しています。テスト対象のホストは NVMe クライアントとして設定され、lab エージェントシステムは NVMe ターゲットとして設定されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

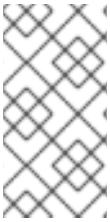
1. 必要なカーネルモジュールが読み込まれており、NVMe クライアントが NVMe ターゲットに接続されていることを確認します。
2. discovery、disconnect および connect コマンドを実行して、NVMe ターゲットとクライアント間の接続を確立して確認します。
3. ターゲットシステムのストレージデバイスへの接続に使用されるネットワークインターフェイスを検出し、さらに各 STORAGE および NETWORK テストから 1 種類のテストを実行します。

NVMe over Fabric ストレージテストでは、テストは NVMe クライアントシステムで実行されますが、NVMe デバイスは物理的に NVMe ターゲットホストに置かれます。NVMe クライアントおよびサーバーの両方が TCP プロトコルを使用して通信します。

テストの準備

テストを開始する前に、以下を確認します。

- NVMe クライアントが RHEL 8.x を実行している。
- NVMe ターゲットシステムおよび NVMe クライアントシステムが適切に設定され、RDMA ネットワークに属している。



注記

NVMe over TCP のデフォルトの TCP ポート番号は 8009 です。NVMe over RDMA のデフォルトの TCP ポート番号は 4420 です。他の現在のアプリケーションと競合しない TCP ポート番号を使用できます。ポートの競合がある場合は、NVMe ポート番号 8009 を別の TCP ポート番号で再設定します。

テストの実行

このテストは非対話型です。現在、このテストは、CLI からのみ計画し、実行できます。

ランタイム

このテストでは、実行に約 10 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.33. OMNIPATH 接続

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

Omnipath 接続 テストでは、次のサブテストを実行して、必要に応じて、テストの開始時にドロップダウンから選択した IP アドレスを使用してベースライン機能を確保します。

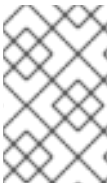
1. Ping テスト - HUT でテストされているデバイスの開始 IP アドレスから、LTS の選択された IP アドレスまで ping を実行します。
2. Rping テスト - 選択した LTS IP アドレスを使用して LTS および HUT で rping を実行し、結果を比較して完了まで実行されたことを確認します。
3. Rcopy test - LTS と HUT で rcopy を実行し、ランダムに生成されたファイルを送信し、LTS と SUT で md5sum を比較して、転送が成功したことを確認します。
4. RDMA-ndd サービステスト - サービスコマンドの停止、起動、および再起動が想定通りに機能することを確認します。
5. Opensm サービステスト - サービスの停止、起動、および再起動が想定通りに機能することを確認します。
6. LID 検証テスト - デバイスの LID が設定され、デフォルト値ではないことを確認します。
7. リンク速度テスト: 検出されたリンク速度が 100Gb であることを確認します。
8. Smpquery テスト - LTS でデバイスとポートを使用して spmquery を実行し、デバイス/ポートがファブリックに登録されているかどうかを確認します。
9. ib_write_bw テスト
   HUT から LTS の選択した IP アドレスに対して ib_write_bw を実行して、Omnipath 書き込み帯域幅をテストし、必要な帯域幅に到達できるかどうかを検証します。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

10. ib_read_bw test
    HUT から LTS の選択した IP アドレスに対して ib_read_bw を実行して、Omnipath 読み取り帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

11. ib_send_bw test
    HUT から LTS の選択した IP アドレスに対して ib_send_bw を実行して、Omnipath の送信帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。キューペアパラメーターは、ラインレートに近いスループットを達成するために、帯域幅テスト中に調整されました。

テストの準備

- LTS と HUT が、同じファブリック上の別々のマシンであることを確認する。Red Hat カスタマーポータルの Web ページの Downloads セクションから LTS に **opa-basic-tools** をインストールする必要があります。



注記

このテストを実行するために、テスト対象の RHEL 7 ホストで RHEL 8 テストサーバーを使用している場合は、テストスイートがこのサービスを自動的に開始しないため、**httpd** サービスを手動で開始する必要があります。

テストの実行

このテストは、インタラクティブなテストです。次のコマンドを実行し、表示される一覧から適切な **omnipath connection** テスト名を選択します。

```
rhcert-run
```

テストを実行する IP アドレス (LTS の IP アドレス) を選択するためのドロップダウンが表示されます。テストを実行している SUT デバイスと同じファブリック上のデバイスに対応する IP アドレスを選択します。

手動でのテストの追加と実行

次のコマンドを使用して、OmnipathConnectionTest を手動で追加します。

```
rhcert-cli plan --add --test Omnipath --device <device name>_devicePort_<port number>
```

次のコマンドを使用して OmnipathConnectionTest を手動で実行します。

```
rhcert-cli run --test Omnipath --server <LTS IP addr>
```

ランタイム

このテストの実行には 10 分もかかりません。

関連情報

- InfiniBand および RDMA の詳細は、 InfiniBand および RDMA テクノロジーについて を参照してください。

A.34. POWER_STOP

テストの対象

Suspend-to-Idle 状態。これを有効にすると、システムが一時停止されている間にプロセッサーがより深いアイドル状態になることを許可します。ユーザー空間をフリーズし、すべての I/O デバイスを低電力状態に入れ、システム上で電力消費を節約できます。

power_stop テストは、これらの Stop(または idle) の状態を有効にすると ppc64le CPU アーキテクチャーマシンで予想通りに機能するかどうか (特に Power9 ベースのシステム) 検証するように設計されています。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、**lsprop**コマンドを使用して、特定のシステムがサポートするすべてのアイドルストップ状態の情報を収集し、**cpupower**コマンドを使用して、これらの状態を有効または無効にします。このテストでは、各 cpu アイドル状態の使用量と持続時間カウンターの増分を観察し、有効かどうかを確認します。

合格の基準:

停止状態を有効にする前と後で、使用量と持続時間のパラメーター値の変化が見られる場合。

- PASS: すべての状態がカウンターパラメーターの値を増やす場合
- WARN: いずれかの状態がカウンターパラメーター値の増加に失敗した場合
- FAIL: どの状態もカウンターを増加しない場合
- REVIEW: その他不明な問題

テストの準備

テスト対象のホスト (HUT) が次の要件を満たしている場合、このテストは自動的に計画されます。

- HUT は、サポートされている RHEL バージョンのいずれかを実行しています。
- 基礎となるアーキテクチャーは ppc64le です。
- CPU モデルは POWER9 です。

注記

このテストは他の RHEL バージョンおよびアーキテクチャーではサポートされておらず、実行されると失敗します。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **power_stop** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストが完了するまで 5 分未満ですが、CPU Idle Stop 状態の数によって異なる場合があります。

A.35. PROFILER

profiler テストは、テスト対象ホストからパフォーマンスメトリックを収集し、メトリックが RHEL カーネルでサポートされているソフトウェアまたはハードウェアのパフォーマンスモニタリングユニット (PMU) から収集されているかどうかを判断します。メトリックがハードウェアベースの場合、テストはさらに、PMU にコアごとのカウンターのみが含まれるのか、パッケージごとのカウンターも含まれるのかを判断します。profiler テストは、**profiler_hardware_core**、**profiler_hardware_uncore**、および**profiler_software**の 3 つのテストに分けられます。

A.35.1. profiler_hardware_core

テストの対象

profiler_hardware_core テストは、サイクルイベントをチェックすることにより、ハードウェアベースのコアごとのカウンターを使用してパフォーマンスメトリックを収集します。コアイベントは、L2 キャッシュなどのプロセッサーコアの機能を測定します。

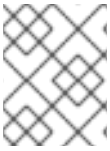
RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストは、コアハードウェアイベントカウンターが見つかった場合に計画され、**find /sys/devices/* -type f -name 'cpu*cycles'** コマンドを実行して **/sys/devices** ディレクトリーで **cpu*cycles** ファイルを見つけます。

このテストでは、複数のコマンドを実行して cycle イベントのサンプルを蓄積し cpu cycle イベントが検出されたかどうかを確認し、サンプルが収集されたかどうかを確認します。



注記

このテストは網羅的なものではなく、特定のプロセッサーが持つ可能性、または持たない可能性のあるすべてのコアカウンターイベントをテストするわけではありません。

テストの準備

このテストを実行するための特別な条件はありません。

テストの実行

このテストは非対話型です。以下のコマンドを実行して、表示されるリストから適切な **profiler_hardware_core** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストの所要時間は約 30 秒です。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.35.2. profiler_hardware_uncore

テストの対象

profiler_hardware_core テストは、ハードウェアベースのパッケージ全体のカウンターを使用してパフォーマンスメトリックを収集します。uncore イベントは、コアの外部にあるがパッケージの内部にあるプロセッサー (メモリーコントローラーなど) の機能を測定します。

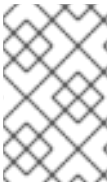
RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

アンコアハードウェアイベントカウンターが見つかった場合、テストが計画されます。アンコアイベントが検出され、いずれか 1 つのイベントの統計情報が収集された場合、テストは合格です。アンコアイベントが見つかったが、それらのイベントがサポートされていないため統計情報を収集しない場合、テストは失敗します。

テストでは、複数のコマンドを実行して、アンコアイベントのリストとアンコアイベントの統計情報を収集します。



注記

このテストは網羅的なものではなく、特定のプロセッサーが持つ可能性、または持たない可能性のあるすべてのアンコアカウンターイベントをテストするわけではありません。

テストの準備

このテストを実行するための特別な条件はありません。

テストの実行

このテストは非対話型です。以下のコマンドを実行して、表示されるリストから適切な **profiler_hardware_uncore** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストの所要時間は約 30 秒です。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.35.3. profiler_software

テストの対象

profiler_software テストは、cpu_clock イベントをチェックすることにより、ソフトウェアベースのカウンターを使用してパフォーマンスメトリックを収集します。

ソフトウェアカウンターは、このテストを使用して認定できます。ただし、高性能要件を持つお客様の場合、このテストは限定的である可能性があります。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

コアハードウェアイベントカウンターが見つからない場合、テストが計画されます。

このテストでは、複数のコマンドを実行して cpu-clock イベントのサンプルを蓄積し cpu-clock イベントが検出されたかどうかを確認し、サンプルが収集されたかどうかを確認します。

テストの準備

このテストを実行するための特別な条件はありません。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **profiler_software** テスト名を選択します。

```
rhcert-run
```

ランタイム

テストの所要時間は約 30 秒です。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.36. リアルタイム

テストの対象

realtime テストは、Red Hat Enterprise Linux for Real Time を実行しているシステムのテストを対象としており、システム管理モードに基づく実行遅延を見つけるテストと、タイマーイベントのサービスの待ち時間を調べるテストの 2 セットがあります。

さらに、RHEL 8 および RHEL 9 の場合、このテストでは、すべてのコアを完全に利用するのではなく、ハウスキーピング用に予約されているコアがあることを確認します。

注記

テストは、Red Hat カーネルを実行しているシステムでのみ計画されています。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストの最初の部分は、**hwlat_detector.ko** という名前の特別なカーネルモジュールを読み込みます。このモジュールは、Timestamp Counter Register (TSC) をポーリングするカーネルスレッドを作成し、指定された閾値を超える連続した読み取りの間隔を探します。TSC の連続した読み取りにギャップがあるということは、読み取りの間にシステムが中断され、他のコード (通常はシステム BIOS で定義されたシステム管理モード (SMM) コード) が実行されたことを意味します。

テストの 2 番目の部分は、cpu ごとの測定スレッドを開始する **cyclictest** という名前のプログラムを起動し、高いリアルタイム優先度で実行します。これらのスレッドには周期 (100 マイクロ秒) があり、以下のような計算を行います。

1. get a timestamp(t1)
2. sleep for period
3. get a second timestamp(t2)
4. latency = t2 -(t1 + period)
5. goto 1

注記

レイテンシーは、理論上のウェイクアップタイム (t1+period) と実際のウェイクアップタイム (t2) の時間差です。各測定スレッドは、最小、最大、および平均の待機時間を追跡し、各データポイントを報告します。

サイクリックテストの実行中、**rteval** は 1 組のシステムロードを開始します。1 つは並列 Linux カーネルコンパイルで、もう 1 つは **hackbench** と呼ばれるスケジューラーベンチマークです。

実行が完了すると、rteval はデータポイントの統計分析を実行し、平均、モード、メディア、差異、および標準偏差を計算します。

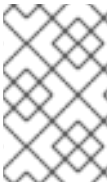
さらに、RHEL 8 および RHEL 9 の場合、このテストでは分離された CPU が **/sys/devices/system/cpu/isolated** に設定されているかどうかがチェックされ、調整されたバージョンには、isolated_cores の初期自動セットアップのサポートが含まれています (バージョン 2.19.0 移行)。また、リアルタイム調整プロファイルがアクティブかどうかもチェックします。いずれかのチェックが失敗した場合、テストは実行を続行する前に警告を出します。

テストの準備

- システムを認定に追加する前に、realtime kernel-rt カーネルをインストールして起動します。このコマンドは、実行中のカーネルがリアルタイムであることを検出し、リアルタイムテストの実行をスケジュールします。
- 調整されたバージョンが 2.19.0 以上の RHEL 8 および RHEL 9 の場合、調整されたプロファイルをリアルタイムとして選択し、システムを再起動します。

注記

リアルタイムのチューニングサポートが必要な場合は、お使いのシステムに Red Hat のアクセス権を設定して、BIOS の変更など、必要な変更を行えるようにする必要があります。



注記

新しくインストールされたカーネルは、以前に設定されたカーネルからカーネルコマンドラインパラメーターを継承します。詳細は、すべてのブートエントリーのカーネルコマンドラインパラメーターの変更 を参照してください。

テストの実行

リアルタイムテストは非対話的です。次のコマンドを実行し、表示されるリストから適切な **realtime** テスト名を選択します。

```
rhcert-run
```

テストは、システムが rt-kernel を実行している場合にのみ表示されます。

ランタイム

システム管理モードは 2 時間実行され、タイマーイベント分析はすべてのマシンで 12 時間実行されます。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.37. REBOOT

テストの対象

reboot テストでは、プロンプトが表示されるとシステムが再起動できるかどうかを確認します。現時点では認定には必要ありません。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストは **shutdown -r 0** コマンドを実行して、遅延なくシステムをすぐに再起動します。

テストの準備

実行中のアプリケーションを終了して、このテストを実行する前にシステムが再起動できるようにしてください。

テストの実行

再起動テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **reboot** テスト名を選択します。

```
rhcert-run
```

テストプログラムの再起動部分に到達すると、**Ready to restart?** と求められます。テストを実行する準備ができたら **y** と答えます。システムが再起動し、再起動後、テストサーバーは再起動が正常に完了したことを確認します。

A.38. ROCE 接続

テストの内容

RoCE 接続 テストは次のサブテストを実行し、必要に応じてテストの開始時にドロップダウンから選択した IP アドレスを使用してベースライン機能を確保します。

1. Ping テスト - HUT でテストされているデバイスの開始 IP アドレスから、LTS の選択された IP アドレスまで ping を実行します。
2. Rping テスト - 選択した LTS IP アドレスを使用して LTS および HUT で rping を実行し、結果を比較して完了まで実行されたことを確認します。
3. Rcopy test - LTS と HUT で rcopy を実行し、ランダムに生成されたファイルを送信し、LTS と SUT で md5sum を比較して、転送が成功したことを確認します。
4. Ethtool test - roce デバイスの検出されたネットデバイスを渡して、ethtool コマンドを実行します。
5. ib_write_bw テスト
   HUT から LTS の選択した IP アドレスに対して ib_write_bw を実行して、RoCE 書き込み帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。
6. ib_read_bw test
   HUT から LTS の選択した IP アドレスに対して ib_read_bw を実行して、RoCE 読み取り帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。
7. ib_send_bw test
   HUT から LTS の選択した IP アドレスに対して ib_send_bw を実行して、RoCE 送信帯域幅をテストし、必要な帯域幅に到達できるかどうかを確認します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの準備

- LTS と HUT が、同じファブリック上の別々のマシンであることを確認します。

テストの実行

このテストは、インタラクティブなテストです。次のコマンドを実行し、表示されるリストから適切な **RoCE connection** テスト名を選択します。

```
rhcert-run
```

テストを実行する IP アドレス (LTS の IP アドレス) を選択するためのドロップダウンが表示されます。テストを実行している SUT デバイスと同じファブリック上のデバイスに対応する IP アドレスを選択します。

表A.4 手動でのテストの追加と実行

速度タイプ	RoCEConnection テストを手動で追加するコマンド	RoCEConnection テストを手動で実行するコマンド
10GigRoCE	rhcert-cli plan --add --test 10GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 10GigRoCE --server <LTS IP addr>
20GigRoCE	rhcert-cli plan --add --test 20GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 20GigRoCE --server <LTS IP addr>
25GigRoCE	rhcert-cli plan --add --test 25GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 25GigRoCE --server <LTS IP addr>
40GigRoCE	rhcert-cli plan --add --test 40GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 40GigRoCE--server <LTS IP addr>
50GigRoCE	rhcert-cli plan --add --test 50GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 50GigRoCE --server <LTS IP addr>
100GigRoCE	rhcert-cli plan --add --test 100GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 100GigRoCE --server <LTS IP addr>

速度タイプ	RoCEConnection テストを手動で追加するコマンド	RoCEConnection テストを手動で実行するコマンド
200GigRoCE	rhcert-cli plan --add --test 200GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 200GigRoCE --server <LTS IP addr>
400GigRoCE	rhcert-cli plan --add --test 400GigRoCE --device <device name>_devicePort_<port number>_netDevice_<net device>	rhcert-cli run --test 400GigRoCE --server <LTS IP addr>

<device name>、**<port number>**、**<net device>**、および **<LTS IP addr>** は、適切な値に置き換えます。

関連情報

- InfiniBand と RDMA の詳細については、Understanding InfiniBand and RDMA technologies を参照してください。

A.39. SATA

SATA テストの対象

現在、いろいろなシステムで、さまざまな種類の永続オンラインストレージデバイスが利用可能です。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

SATA テストは、udev データベース内の disk の **ID_TYPE** を報告するすべてのテストを行うように設計されています。このテストは、**SATA ドライブ** 用です。hwcert/storage/SATA テストは以下の場合に予定されます。

- **SATA** と記載されるディスクのコントローラー名
- ディスクが接続されているホストの lsscsi トランスポートに **SATA**が記載されている

上記の 2 つの基準が満たされない場合、ストレージテストは検出されたデバイスに対して計画されます。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.40. SATA_SSD

SATA_SSD テストの対象

このテストは、対象のストレージユニットが SSD で、そのインターフェイスが SATA であると判断された場合に実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

SATA_SSD テストの内容

このテストは、SCSI ストレージタイプを検出し、**/sys/block/sdap/queue/rotational** の場所に接続されているストレージインターフェイスを識別します。このテストは、ローテーションビットが SSD に対してゼロに設定された場合に予定されます。

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.41. M2_SATA

M2_SATA テストの対象

インターフェイスが SATA で、M2 接続を介して接続された場合には、このテストが実行されます。

RHEL バージョン対応

- RHEL 7

- RHEL 8
- RHEL 9

手動でのテストの追加と実行

次のコマンドを使用して、M2_SATA テストを手動で追加します。

```
rhcert-cli plan --add --test M2_SATA --device host0
```

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.42. U2_SATA

U2_SATA テストの対象

インターフェイスが SATA で、U2 接続を介して接続されていると判断された場合に、このテストが実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

手動でのテストの追加と実行

次のコマンドを使用して、U2_SATA テストを手動で追加します。

```
rhcert-cli plan --add --test U2_SATA --device host0
```

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。

- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.43. SAS

SAS テストの対象

現在、いろいろなシステムで、さまざまな種類の永続オンラインストレージデバイスが利用可能です。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

SAS テストは、udev データベース内の disk の **ID_TYPE** を報告するすべてのテストを行うように設計されています。このテストは、**SAS ドライブ 用です**。hwcert/storage/SAS テストは以下の場合に予定されます。

- ディスクのコントローラー名には **SAS** と認識される必要があり、または、
- ディスクが接続されているホストの lsscsi トランスポートで **SAS**が認識されている必要がある

上記の 2 つの基準が満たされない場合、ストレージテストは検出されたデバイスに対して計画されます。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.44. SAS_SSD

SAS_SSD テストの対象

このテストは、対象のストレージユニットが SSD で、そのインターフェイスが SAS であると判断された場合に実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

SAS_SSD テストの内容

このテストは、SCSI ストレージタイプを検出し、/sys/block/sdap/queue/rotational の場所に接続されているストレージインターフェイスを識別します。このテストは、ローテーションビットが SSD に対してゼロに設定された場合に予定されます。

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.45. PCIE_NVME

PCIe_NVMe テストの対象

インターフェイスが NVMe で、PCIE 接続を介して接続されている場合は、このテストが実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

PCIe_NVMe テストの内容

このテストは、論理デバイスのホスト名文字列に **nvme[0-9]** が含まれる場合に計画されます。

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.46. M2_NVME

M2_NVMe テストの対象

インターフェイスが NVMe で、M2 接続を介して接続されている場合は、このテストが実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

手動でのテストの追加と実行

次のコマンドを使用して、M2_NVMe テストを手動で追加します。

```
rhcert-cli plan --add --test M2_NVMe --device nvme0
```

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.47. U2_NVME

U2_NVMe テストの対象

インターフェイスが NVMe で、U2 接続を介して接続されている場合は、このテストが実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

手動でのテストの追加と実行

次のコマンドを使用して、U2_NVMe テストを手動で追加します。

```
rhcert-cli plan --add --test U2_NVMe --device nvme0
```

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.48. NVDIMM

NVDIMM テストの対象

このテストは、他の SSD 非ローテートストレージテストと同様に動作し、NVDIMM ストレージデバイスを識別します

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、次の場合にストレージデバイスに対して予定されます。

- ndctl list が報告するディスクデバイスの名前空間 (不揮発性メモリーデバイス) が存在します。
- これは、sda の DEVTYPE が disk と等しいと報告されます。

テストの一部として出力される device パラメーターの値は、以下のとおりです。

- logical_block_size - デバイス上の場所のアドレス指定に使用します。
- physical_block_size - デバイスが動作する最小単位
- minimum_io_size - デバイスのランダムな入出力に優先される最小ユニット
- optimal_io_size - 入力または出力ストリーミングに推奨されるデバイスの単位です。
- alignment_offset - 基となる物理アライメントからのオフセット値です。

関連情報

- **テスト内容**および**テストの準備** に関する詳細は、 STORAGE を参照してください。

A.49. SR-IOV

テストの内容

SR-IOV テストは、カードで SR-IOV 機能がサポートされているかどうかを確認することにより、テスト対象ホスト (HUT) およびテストサーバーにインストールされている NIC カードを認証します。

このテストは、単一の物理ハードウェアデバイスを複数の仮想マシンまたはコンテナー間で共有できるようにするシングルルート I/O 仮想化 (SR-IOV) テクノロジーに基づいており、ネットワークパフォーマンスと I/O 操作の効率を向上させます。

RHEL バージョン対応

- RHEL 9

テストの対象

このテストでは、x86_64 システムに取り付けられている NIC カードが SR-IOV テクノロジーをサポートしているかどうかを確認します。

テストの準備

テストを実行する前に:

- HUT とテストサーバーの両方に NIC カードを取り付けます。
- テスト中の NIC カードにそれぞれ 2 つのポートがあることを確認してください。
- テストが正常に完了するには、テストサーバーと HUT が 2 本の直接イーサネットケーブルで背中合わせに接続されていることを確認してください。
- システムの BIOS で Intel VT-d または AMD IOMMU および SR-IOV Global Enable パラメーターを必ず有効にしてください。詳細は、システムの BIOS 設定マニュアル、または製造元が提供するその他の方法を参照してください。
- デバイス設定で NIC カードの SRIOV を必ず有効にしてください。詳細は、NIC カード設定マニュアルまたは製造元が提供するその他の方法を参照してください。
- 次の RPM を、前述と同じ順序で HUT とテストサーバーの両方にインストールします。
  1. epel リポジトリーを有効にして、beakerlib をインストールします。

     ```
     # yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
     ```

     ```
     # yum install beakerlib
     ```

  2. SR-IOV
- ヒュージページHugepage の設定

  ```
  # grubby --args='intel_iommu=on iommu=pt default_hugepagesz=1G hugepagesz=1G
  hugepages=32' --update-kernel=$(grubby --default-kernel)
  ```

  設定が完了したら、必ずシステムを再起動してください。

- HUT とテストサーバーを必ずプロビジョニングしてください。cockpit を使用したシステムの設定とテストの実行 または CLI を使用したシステムの設定とテストの実行 を参照してください。

テストの実行

HUT で以下を実行します。

1. システムの設定に従って、このパスに生成された設定ファイル **/etc/redhat-certification/sriov/nic_cert.conf** を編集します。

注記

プロビジョニングコマンドを実行した後は、毎回設定ファイルを更新してバックアップを保存する必要があります。

2. テストを実行します。

```
# rhcert-run
```

テストが HUT 上で実行されている間、対応するテスト実行もテストサーバー上で自動モードで実行されます。テストは非対話型であり、バックグラウンドで実行します。

ランタイム

テストの実行には約 2 時間かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.50. ストレージ

ストレージテストの対象内容

現在、いろいろなシステムで、さまざまな種類の永続オンラインストレージデバイスが利用可能です。**STORAGE** テストは、udev データベース内の disk の **ID_TYPE** を報告するすべてのテストを行うように設計されています。これには、IDE ドライブ、SCSI ドライブ、SATA ドライブ、SAS ドライブと、SSD ドライブ、PCIe SSD ブロックストレージデバイスと、SD メディア、xD メディア、MemoryStick および MM C カードが含まれます。テスト計画スクリプトは udev データベースを読み込み、上記の基準を満たすストレージデバイスを探します。デバイスを見つけると、デバイスとその親を記録し、記録された他のデバイスの親と比較します。これは、一意の親を持つデバイスのみがテストされるようにするためです。親が以前確認されていない場合は、デバイスがテスト計画に追加されます。これにより、ポリシーガイドに従って、コントローラーごとに 1 つのデバイスのみがテストされるため、テストが高速化されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

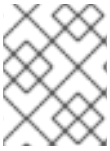
STORAGE テストは、一意の親を持つすべてのストレージデバイスで以下のアクションを実行します。

1. このスクリプトはパーティションテーブルを検索して、LVM またはソフトウェアの RAID デバイスにない swap パーティションを見つけます。見つかった場合は **swapoff** で無効にし、テストにその領域を使用します。swap がない場合、システムはドライブが完全にブランク (パーティションなし) であればテストできます。これを機能させるには、swap デバイスがアクティブである必要があります (テストは **/proc/swaps** を読み取って swap パーティションを読み取り)、swap パーティションをソフトウェアベースのコンテナー (LVM またはソフトウェア RAID なし) 内にはならないことに注意してください。ただし、ハードウェア RAID は、システムに見えないため機能します。

2. このツールは、空のドライブの swap パーティション内のデバイスにファイルシステムを作成します。

3. ファイルシステムがマウントされ、**fio** または **dt** コマンドを使用してデバイスをテストします。**fio** または **dt** コマンドは、I/O テストプログラムで、デバイスへのテスト、読み取り、書き込みが可能な汎用テストツールです。テストパターンの複数セットでは、ストレージデバイスの機能を検証します。

4. マウントされたファイルシステムのテスト後、ファイルシステムはアンマウントされ、ファイルシステムを無視してブロックデバイスに対して dt テストが実行されます。dt テストは、direct パラメーターを使用してこれを処理します。



注記

ストレージテストは、RHEL 7.4 以降のバージョンの **dt** パッケージと、RHEL 7.5 以降のバージョンの **fio** パッケージを使用します。

テストの準備

公式のテストプランにリストされているドライブおよびストレージコントローラーをすべてインストールする必要があります。ストレージオプションが複数になる場合は、一度にシステムに収まるものだけを 1 回の実行でテストできます。または、各ストレージデバイスを個別にインストールして、独自のストレージテストを実行することもできます。テストの順序と、各テストに存在するコントローラーの数を決定できます。システムに接続されている各論理ドライブには、他のパーティションに加えて swap パーティションが含まれているか、完全に空白である必要があります。これは、ファイルシステムを作成し、テストを実行する場所をテストに指定するためです。空白のままにしたデバイスは完全にテストされるため、swap パーティションを使用すると、はるかに迅速なテストが可能になります。それらはほとんどの場合、ドライブに配置された swap パーティションよりもはるかに大きくなります。

注記

SD メディアカードをテストする場合は、可能な限り最速のカードを使用してください。クラス 4 の SD カードはテストの実行に 8 時間以上かかる場合がありますが、クラス 10 または UHS 1/2 カードは 30 分以下でテスト実行を完了できます。

公式のテストプランのストレージデバイスを選択する場合は、レビューチームが操作するルールはコードパスごとに 1 つのテストとなります。つまり、コントローラーが使用できるすべてのドライバーを使用してストレージテストを実行します。同じコントローラーの複数のドライバーのシナリオには、通常、なんらかのタイプの RAID ストレージが必要です。ストレージコントローラーは、通常のディスクモードの場合は 1 つのドライバーを使用し、RAID モードの場合は別のドライバーを使用するのが一般的です。RAID モードによっては、複数のドライバーを使用するものもあります。レビューチームは、すべてのストレージハードウェアを分析して、すべてのテスト要件を満たすために使用する必要があるドライバーを決定します。このため、公式のテスト計画で同じストレージデバイスが複数回記載される場合があります。ストレージデバイスのテストに関する完全な情報は、ポリシーガイドを参照してください。

テストの実行

ストレージテストは非対話的です。次のコマンドを実行し、表示されるリストから適切な **STORAGE** テスト名を選択します。

```
rhcert-run
```

ランタイム、ベアメタル

ストレージテストは、2013 年版ワークステーションシステムにインストールされている 6Gb/s SATA ハードドライブで約 22 分かかります。同じテストは、2013 年版ワークステーションシステムにインストールされている 6Gb/s SATA ソリッドステートドライブで約 3 分かかります。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

関連情報

- 適切なスワップファイルサイズの詳細は、 Red Hat プラットフォームで推奨されるスワップサイズは？を参照してください。

A.51. 特殊キー

テストの対象

特殊キー テストは、システム統合キーボードからさまざまな入力イベントを取得します。このテストは、バッテリー搭載システムでのみ実行します。

RHEL バージョン対応

- RHEL 8.6 以降
- RHEL 9

テストの内容

テストは以下を取得します。

- ボリュームのアップ/ダウン、ボリュームミュート、ディスプレイのバックライト輝度のアップ/ダウンなど、ACPI 関連シグナル以外のシグナル。
- **<Meta+E>** などのグローバルキーボードショートカットに関連付けられたシグナルを送信するキーを押すと、ファイルブラウザーが開きます。

テストの実行

テストはインタラクティブに行われます。次のコマンドを実行し、表示されるリストから適切な **Special keys** テスト名を選択します。

```
rhcert-run
```

このテストでは、すべての入力イベントを取得する必要があります。テスト中に、デバイスの標準以外のキーおよびマルチメディアキーをすべて押します。

いつでも Escape キーを押してテストを終了し、キーのリストを表示できます。テストしたすべてのキーがリストに表示されていればテストは成功です。

ランタイム

テストの所要時間は 5 分未満です。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.52. SUPPORTABLE

テストの対象

supportable テストでは、テスト対象のホスト (HUT) に関する基本情報が収集されます。Red Hat はこの情報を使用して、システムが認定要件に準拠していることを確認します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストには、次のタスクを実行するいくつかのサブテストがあります。

1. **/proc/sys/kernel/tainted** ファイルにゼロ (**0**) が含まれていることを確認します。これは、カーネルがテイントされていないことを示します。
2. **rpm -V** コマンドを使用したパッケージ検証で、ファイルが変更されていないことが示されていることを確認します。
3. **rpm -qa kernel** コマンドが、カーネルパッケージのビルドホストが Red Hat サーバーであることを示していることを確認します。
4. **/proc/cmdline** ファイルからブートパラメーターを記録します。
5. rpm -V redhat-certification コマンドで、どの認証テストスイートファイルにも変更がないことを確認します。
6. **lsmod** コマンドによって表示されるすべてのモジュールが、**rpm -ql カーネル** コマンドを使用してカーネルファイルのリストに表示されることを確認します。
7. すべてのモジュールが Kernel Application Binary Interface (kABI) の **stablelist** にあることを確認します。
8. モジュールベンダーとビルドホストが適切な Red Hat のエントリーであることを確認します。
9. カーネルが Red Hat マイナーリリースの GA カーネルであることを確認します。
   サブテストは、**redhat-certification** パッケージのデータを使用してカーネルを検証しようとします。カーネルが存在しない場合、サブテストはインターネット接続を使用してカーネルを検証しようとします。

   インターネット接続を使用してカーネルを検証するには、インターネットにアクセスするように HUT のルーティングと DNS 解決を設定するか、**ftp_proxy=http://proxy.domain:80** 環境変数を設定する必要があります。
10. カーネルから報告された既知のハードウェアの脆弱性をチェックします。サブテストは **/sys/devices/system/cpu/vulnerabilities/** ディレクトリー内のファイルを読み取り、ファイルに Vulnerable という単語が含まれている場合は警告を表示して終了します。

11. **lscpu** コマンドの出力を調べて、システムにオフラインの CPU があるかどうかを確認します。

12. 同時マルチスレッド (SMT) がシステムで利用可能か、有効か、アクティブかどうかを確認します。

13. RHEL 8 以降を実行しているシステムに、保守されていないハードウェアまたはドライバーがないかどうかを確認します。
保守されていないハードウェアとドライバーは、定期的にテストまたは更新されなくなりました。Red Hat は、セキュリティーの問題を含む重大な問題を修正する可能性がありますが、計画された頻度での更新は期待できません。

    保守されていないハードウェアまたはドライバーは、できるだけ早く交換または削除してください。

14. RHEL 8 以降を実行しているシステムに非推奨のハードウェアまたはドライバーがあるかどうかを確認します。
非推奨のハードウェアとドライバーは引き続きテストおよび保守されていますが、保守されなくなり、最終的には将来のリリースで無効になる予定です。

    非推奨のデバイスまたはハードウェアをできるだけ早く交換または削除してください。

15. RHEL 8 以降を実行しているシステムで無効になっているハードウェアがあるかどうかを確認します。
RHEL は無効なハードウェアを使用できません。テストを再度実行する前に、システムから無効なハードウェアを交換または取り外します。

16. ソフトウェア RPM パッケージで次のチェックを実行します。

    - RPM ビルドホスト情報を確認して、Red Hat 以外のパッケージを分離します。
    このテストでは、Red Hat 以外のパッケージを含める理由を説明するよう求められます。Red Hat は理由を確認し、各パッケージを個別に承認または拒否します。

    - インストールされている RPM パッケージが、オファリングで使用可能な Red Hat 製品のものであり、変更されていないことを確認してください。
    Red Hat は **rpm_verification_report.log** ファイルで検証の失敗を確認します。失敗したパッケージを再インストールして、テストを再実行する必要があります。

17. システム内に Red Hat ファームウェアファイルと Red Hat 以外のファームウェアファイルの両方が存在することを確認します。Red Hat 以外のファイルが存在する場合はそれをリストし、REVIEW ステータスで終了します。

18. **getconf PAGESIZE** コマンドでシステムのページサイズを確認します。

これらのタスクを実行した後、テストは **sosreport** と **dmidecode** コマンドの出力を収集します。

テストの実行

rhcert ツールは、テストスイートのすべての実行の一部として、**supportable** テストを自動的に実行します。**supportable** テストは、他のどのテストよりも前に実行されます。

supportable テストの出力は、テストスイートログの一部として必要です。Red Hat は、 **supportable** テストの出力を含まないテストログを拒否します。

必要に応じて、次のコマンドを使用してテストを手動で実行します。

```
$ rhcert-cli run -test supportable
```

ランタイム

supportable テストは、このガイドのキックスタートファイルを使用してインストールした Red Hat Enterprise Linux 6.4、AMD64、および Intel 64 を実行する 8 GB の RAM を搭載した 2013 年版のシングル CPU、3.3、3.3、6-core/12-thread Intel ワークステーションで約 1 分かかります。マシンの速度やインストールする RPM ファイルの数によって時間は異なります。

A.53. SUSPEND

テストの対象

(ノートパソコンのみ) **suspend** テストは、S3 スリープ状態からの一時停止/再開 (RAM への一時停止) と、S4 ハイバネーション (ディスクへの一時停止) からの一時停止/再開に対応します。このテストでは、より多くのエネルギーを節約できるフリーズ (アイドル状態へのサスペンド - s2idle) 状態も対象とします。このテストは、ノートパソコンなどのバッテリーを内蔵しているシステムでのみ予定されています。

重要

RAM へのサスペンドおよびディスクへのサスペンドの機能は、ラップトップの重要な特性です。したがって、ラップトップでは、すべての認定テストの実行開始時に自動サスペンドテストをスケジュールします。これにより、すべてのハードウェア機能が、通常は再開後に実施されます。テストは、スケジュールされているテストに関係なく、**supportable** テストと同様に常にラップトップで実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは **/sys/power/state** ファイルをクエリーし、ハードウェアでサポートされる状態を判断します。ファイルに mem が表示されると、S3 スリープテストがスケジュールされます。ファイルに disk が表示されると、S4 ハイバネートテストがスケジュールされます。両方が表示される場合には、両方をスケジュールします。以下は、S3 と S4 の状態の両方をサポートするシステムの手順を説明します。システムが両方のタイプに対応していない場合は、サポートされるタイプに関連するテストのみを実行します。

RHEL 8 および RHEL 9 のサスペンド状態は、**/sys/power/state** ファイルに書き込まれます。RHEL 7 では、代わりに **pm-utils** コマンドを使用します。

- S3 スリープに対応している場合、スクリプトは **pm-suspend** コマンドを使用して RAM にサスペンドします。テスト担当者はスリープ後にシステムを起動し、スクリプトが **pm-suspend** の終了コードをチェックして、システムが正常に起動していることを確認します。テストは、テストサーバーインターフェイスで続行されます。
- S4 ハイバネートがサポートされている場合、スクリプトは **pm-suspend** コマンドを使用してディスクにサスペンドします。テスターはシステムがハイバネーションした後に起動し、スクリプトは **pm-suspend** の終了コードをチェックして、システムが正しく起動したことを確認します。テストは、テストサーバーインターフェイスで続行されます。
- S3 スリープに対応している場合、テスト担当者は手動で呼び出す鍵を押すように求められます

(kbd:[Fn]+kbd:[F-key] combination または dedicated kbd:[Sleep] キー)。テスト担当者はスリープ後にシステムを起動し、スクリプトが **pm-suspend** の終了コードをチェックして、システムが正常に起動していることを確認します。テストは、テストサーバーインターフェイスで続行されます。システムにサスペンドキーがない場合は、このセクションを省略できます。

- S4 ハイバネートに対応している場合、テスト担当者は手動で呼び出すキー (kbd:[Fn]+kbd:[F-key] の組み合わせまたは専用の kbd:[Hibernate] キー) を押すように求められます。テスターはシステムがハイバネーションした後に起動し、スクリプトは **pm-suspend** の終了コードをチェックして、システムが正しく起動したことを確認します。テストは、テストサーバーインターフェイスで続行されます。システムにサスペンドキーがない場合は、このセクションを省略できます。

テストの準備

システムのインストール時に RAM の内容を保持するのに十分な大きさのスワップファイルを作成してください。テスト対象ホストをサスペンドおよび休止状態から復帰させるには、このテストの実施中に誰かが立ち会う必要があります。

テストの実行

一時停止テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **suspend** テスト名を選択します。

```
rhcert-run
```

このテストは、**suspend?** とプロンプトを表示します。**Yes** と回答すると、ラップトップを一時停止します。テストサーバーは、suspend コマンド の送信後に **waiting for response** を表示します。ラップトップを確認して、一時停止が完了したことを確認し、電源ボタンまたはサスペンドからウェイクするその他のキーを押します。テストサーバーは、テスト中のシステムを継続的に監視して、起動したかどうかを確認します。起動すると、テストサーバーの GUI に質問 **完全に起動しましたか？** が表示されます。**Yes** または **No** ボタンを押して、何が発生したかテストサーバーに指示します。

その後、サーバーはハイバネートテストを継続します。ここでも、**suspend?** の下にある **Yes** ボタンをクリックして、ラップトップをハイバネートモードにします。

テストサーバーは、hibernate コマンド の送信後に **waiting for response** を表示します。ラップトップを確認して、休止状態が完了したことを確認してから、電源ボタンまたは他のキーを押して、休止状態から復帰させます。テストサーバーは、テスト対象ホストを継続的に監視して、再起動したかどうかを確認します。起動すると、テストサーバーの GUI に質問 **完全に復帰しましたか？** が表示されます。**Yes** または **No** ボタンを押して、何が発生したかテストサーバーに指示します。

次に、テストサーバーは、テスト対象ホストを一時停止させるキーボードキーがシステムにあるかどうかを尋ねます。その場合は、**Does this system have a function key(Fn)to suspend the system to mem?** の質問の下にある **Yes** ボタンをクリックします。上記の手順に従って、一時停止を確認し、システムを起動してテストを続行します。

最後に、テストサーバーは、テスト対象のホストを休止状態にするキーボードキーがシステムにあるかどうかを尋ねます。その場合は、**Does this system have a function key(Fn)to suspend the system to disk?**の質問の下にある **Yes** ボタンをクリックします。上記の手順に従って休止状態を確認し、システムを起動して、スケジュールした追加のテストを続行します。

ランタイム

サスペンドテストでは、4GB の RAM と SSD 以外のハードドライブがある 2012 年版のラップトップで約 6 分かかります。これは、pm-suspend ベースおよび function-key ベースのサスペンドとハイバーネートの実行の両方を含む、完全な一連のテストの時間です。時間は、ラップトップがディスクに書き

込める速度、インストールした RAM の容量と速度、および機能キーを使用してラップトップがサスペンドおよびハイバネートの状態に入る機能によって異なります。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

関連情報

- 適切なスワップファイルサイズの詳細は、 Red Hat プラットフォームで推奨されるスワップサイズは？を参照してください。

A.54. テープ

テストの対象

tape テストは、すべての種類のテープドライブを対象としています。ドライブに関連付けられているロボットは、このテストではテストされません。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、**mt** コマンドを使用してテープを巻き戻し、**/usr** ディレクトリーの tar を実行し、テープに保存します。**tar** 比較は、テープ上のデータがディスクのデータと一致するかどうかを確認するために使用されます。データが一致すると、テストに合格します。

テストの準備

適切なサイズのテープをドライブに挿入します。

テストの実行

テープテストは非対話的です。次のコマンドを実行し、表示されるリストから適切な **tape** テスト名を選択します。

```
rhcert-run
```

A.55. THUNDERBOLT3

テストの対象

Thunderbolt3 テストでは、ホットプラグと基本的な機能の観点から Thunderbolt 3 ポートに対応し、OS がすべてのポートにアクセスできるようにし、ポートに接続されているデバイスが適切に追加および削除されるようにします。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

テストの目的は、システム内のすべての Thunderbolt 3 ポートが予想通りに機能することを確認することです。これにより、利用可能な Thunderbolt3 ポートの数が尋ねられ、次にテスト担当者に Thunderbolt 3 デバイスを各ポートに接続したり取り外したりするように要求されます。テストは Thunderbolt 3 デバイスの割り当ておよび割り当て解除イベントを監視し、それらを記録します。テスト担当者が入力した一意のポートの数に対してプラグインとアンプラグイベントの両方を検出すると、テストに合格します。Thunderbolt 3 デバイスは USB C デバイスと同じ物理コネクターを使用しますが、USB C デバイスは Thunderbolt 3 デバイスではありません。Thunderbolt 3 ポートとの互換性を要求する USB C デバイスを含む USB C デバイスを使用している場合は、テストに合格しません。このテストには、Thunderbolt 3 デバイスのみを使用できます。

テストの準備

利用可能な Thunderbolt3 ポートをカウントし、テスト時に使用する Thunderbolt3 デバイスが利用可能です。

テストの実行

Thunderbolt3 テストは対話的です。以下のコマンドを実行してから、表示されるリストから適切な **Thunderbolt3** テスト名を選択します。

```
rhcert-run
```

システムに求められたら、システムに存在する利用可能な Thunderbolt3 ポートの数を入力します。システムは Thunderbolt3 デバイスをポートに接続するように要求し、テスト担当者が **y** を押して続行するまで一時停止します。次に、システムはデバイスを外すように要求し、テスト担当者が **y** を押して続行するまで再び一時停止します。これらのステップは、入力したポート数に対して繰り返し行います。ポートをテストするための正しい順序や誤った順序はありませんが、各ポートは一度だけテストする必要があります。

ランタイム

Thunderbolt3 テストは、Thunderbolt3 ポートごとに約 15 秒かかります。これには、デバイスを手動で接続し、ポートのスキャン、デバイスのアンプラグ、およびポートの再スキャンを行う時間が含まれます。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.56. THUNDERBOLT4

テストの対象

Thunderbolt4 テストでは、ホットプラグと基本的な機能の観点から Thunderbolt 4 ポートに対応し、OS がすべてのポートにアクセスできるようにし、ポートに接続されているデバイスが適切に追加および削除されるようにします。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

テストの目的は、システム内のすべての Thunderbolt 4 ポートが予想通りに機能することを確認することです。これにより、利用可能な Thunderbolt4 ポートの数が要求され、その後テスト担当者に Thunderbolt 4 デバイスを各ポートに接続したり取り外したりするように要求されます。テストは Thunderbolt 4 デバイスの割り当ておよび割り当て解除イベントを監視し、それらを記録します。テス

ト担当者が入力した一意のポートの数に対してプラグインとアンプラグイベントの両方を検出すると、テストに合格します。Thunderbolt 4 デバイスは USB C デバイスと同じ物理コネクターを使用しますが、USB C デバイスは Thunderbolt 4 デバイスではありません。Thunderbolt 4 ポートとの互換性を要求する USB C デバイスを含む USB C デバイスを使用している場合は、テストに合格しません。このテストには、Thunderbolt 4 デバイスのみを使用できます。このテストは、ホストと接続されたデバイス間の接続の生成が Thunderbolt 4 であることも検証します。

テストの準備

利用可能な Thunderbolt4 ポートをカウントし、テスト時に使用する Thunderbolt4 デバイスが利用可能です。

テストの実行

Thunderbolt4 テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **Thunderbolt4** テスト名を選択します。

```
rhcert-run
```

システムに求められたら、システムに存在する利用可能な Thunderbolt4 ポートの数を入力します。システムは Thunderbolt4 デバイスをポートに接続するように要求し、テスト担当者が **y** を押して続行するまで一時停止します。次に、システムはデバイスを外すように要求し、テスト担当者が **y** を押して続行するまで再び一時停止します。これらのステップは、入力したポート数に対して繰り返し行います。ポートをテストするための正しい順序や誤った順序はありませんが、各ポートは一度だけテストする必要があります。

ランタイム

Thunderbolt4 テストは、Thunderbolt4 ポートごとに約 15 秒かかります。これには、デバイスを手動で接続し、ポートのスキャン、デバイスのアンプラグ、およびポートの再スキャンを行う時間が含まれます。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.57. USB_STORAGE

テストの対象

usb_storage テストは、既存のストレージテストに速度検出機能を追加したものです。usb_storage テストは次の内容で構成されています。

- 接続された USB デバイスのバージョンを検出する **USB2_storage** テスト
- **USB3_storage** テストは、接続された USB デバイスのバージョンとインターフェイス速度を検出し、複数の速度 (5Gbps、10Gbps、20Gbps、40Gbps) をサポートしています。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、システムに接続されている USB デバイスのインターフェイス速度とバージョンが検出され、それに応じて対応するテストが計画されます。たとえば、インターフェイス速度が 10 Gbps (サポート対象) の USB 3.0 デバイスが検出された場合、USB3_10Gbps_Storage サブテストが計画、実行されます。

テストの準備

USB ストレージデバイスがシステムに接続されていることを確認します。

テストの実行

次のどちらかのテスト実行方法を選択できます。

- 次のコマンドを実行し、表示されるリストから適切な USB テスト名を選択します。

```
rhcert-run
```

- 目的のテスト名を指定して、**rhcert-cli** コマンドを実行します。以下に例を示します。

```
rhcert-cli run --test=USB3_10Gbps_Storage
```

関連情報

- 残りのテスト機能については、ストレージ をご覧ください。

A.58. USB2

テストの対象

USB2 テストは、基本的な機能の観点から USB2 ポートに対応し、OS がすべてのポートにアクセスできるようにします。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストの目的は、システムに存在するすべての USB2 ポートが期待どおりに機能することを確認することです。これにより、利用可能な USB2 ポートの数 (キーボード/マウスなどで使用されているものを引いたもの) が要求され、その後テスト担当者に USB2 デバイスを各ポートに接続したり取り外したりするように要求されます。テストは、イベントの割り当てと割り当て解除を監視し、イベントを記録します。テスト担当者が入力した一意のポートの数に対してプラグインとアンプラグイベントの両方を検出すると、テストに合格します。

テストの準備

利用可能な USB2 ポートをカウントし、テスト中に使用するスペア USB2 デバイスを用意します。USB2 ポートと USB3 ポートを区別するには、マザーボードヘッダーから USB ポートを追跡しないといけない場合があります。

テストの実行

USB2 テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **USB2** テスト名を選択します。

```
rhcert-run
```

-

システムに求められたら、システムに存在する利用可能な USB2 ポートの数を入力します。キーボードまたはマウスで現在使用されているものは数に入れないでください。システムは、テスト USB2 デバイスをポートにプラグインするよう要求し、テスト担当者が **y**を押すまで一時停止します。次に、システムはデバイスを外すように要求し、テスト担当者が **y** を押して続行するまで再び一時停止します。これらのステップは、入力したポート数に対して繰り返し行います。ポートをテストするための正しい順序や誤った順序はありませんが、各ポートは一度だけテストする必要があります。

ランタイム

USB2 テストは、USB2 ポートごとに約 15 秒かかります。これには、デバイスを手動で接続し、ポートのスキャン、デバイスのアンプラグ、およびポートの再スキャンを行う時間が含まれます。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.59. USB3

テストの対象

USB3 テストは、基本的な機能の観点から USB3 ポートに対応し、OS がすべてのポートを列挙、アクセス、およびホットプラグできるようにします。USB3 テストは、5Gbps、10Gbps、20Gbps ごとに 3 つの異なる速度ベースのテストをサポートします。システムが USB3 に対応している場合は、3 つのテストすべてが予定されています。各テストにクレジットに成功すると、認定用に Red Hat Ecosystem Catalog に含まれる該当する機能が提供されます。テストとそれらの成功基準は次のとおりです。

成功基準:

- USB3_5Gbps - デバイス転送速度が 5Gbps の場合、テストは合格します。
- USB3_10Gbps - デバイス転送速度が 10Gbps の場合、テストは合格します。
- USB3_20Gbps - デバイス転送速度が 20Gbps の場合、テストは合格します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストの目的は、システムに存在するすべての USB3 ポートが期待どおりに機能することを確認することです。これにより、利用可能な USB3 ポートの数 (キーボード/マウスなどで使用されているものを引いたもの) が要求され、その後テスト担当者に USB3 デバイスを各ポートに接続したり取り外したりするように要求されます。テストは、イベントの割り当てと割り当て解除を監視し、イベントを記録します。テスト担当者が入力した一意のポートの数に対してプラグインとアンプラグイベントの両方を検出すると、テストに合格します。

テストの準備

利用可能な USB3 ポートをカウントし、テスト時に使用する利用可能な USB3 デバイスを持つ。USB2 ポートと USB3 ポートを区別するには、マザーボードヘッダーから USB ポートを追跡しないといけない場合があります。デバイスの行の速度が、テストの予想される速度 (5Gbps、10 Gbps、または 20Gbps) と一致していることを確認します。

テストの実行

USB3 テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **USB3** テスト名を選択します。

```
rhcert-run
```

システムに求められたら、システムに存在する利用可能な USB3 ポートの数を入力します。キーボードまたはマウスで現在使用されているものは数に入れないでください。システムは、テストの USB3 デバイスをポートに接続するように要求し、テスト担当者が **y** を押して続行するまで一時停止します。次に、システムはデバイスを外すように要求し、テスト担当者が **y** を押して続行するまで再び一時停止します。これらのステップは、入力したポート数に対して繰り返し行います。ポートをテストするための正しい順序や誤った順序はありませんが、各ポートは一度だけテストする必要があります。

ランタイム

USB3 テストは、USB3 ポートごとに約 15 秒かかります。これには、デバイスを手動で接続し、ポートのスキャン、デバイスのアンプラグ、およびポートの再スキャンを行う時間が含まれます。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.60. USB4

テストの対象

USB4 テストは、基本的な機能の観点から USB4 ポートに対応し、すべてのポートを OS が列挙、アクセス、およびホットプラグできるようにします。USB4 テストは、2 つの異なる速度ベースのテストをサポートします。1 つは 20Gbps 用で、40Gbps 用です。システムが USB4 に対応していると、両方のテストが予定されます。各テストにクレジットに成功すると、認定用に Red Hat Ecosystem Catalog に含まれる該当する機能が提供されます。テストとそれらの成功基準は次のとおりです。

合格の基準

- USB4_20Gbps: デバイス転送速度が 20Gbps の場合、テストは合格します。
- USB4_40Gbps: デバイス転送速度が 40Gbps の場合、テストは合格します。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

テストの目的は、システムに存在するすべての USB4 ポートが期待どおりに機能することを確認することです。これにより、利用可能な USB4 ポートの数 (キーボード/マウスなどで使用されているものを引いたもの) が要求され、その後テスト担当者に USB4 デバイスを各ポートに接続したり取り外したりするように要求されます。テストは、イベントの割り当てと割り当て解除を監視し、イベントを記録します。テスト担当者が入力した一意のポートの数に対してプラグインとアンプラグイベントの両方を検出すると、テストに合格します。

テストの準備:

利用可能な USB4 ポートをカウントし、テスト時に使用する利用可能な USB4 デバイスがある。

USB2、USB3、および USB4 ポートを区別するには、マザーボードヘッダーから USB ポートを追跡しないといけない場合があります。デバイスの行速度が、テストの予想される速度と一致していることを確認します (20Gbps または 40Gbps)。

テストの実行

USB4 テストは対話的です。次のコマンドを実行し、表示されるリストから適切な **USB4** テスト名を選択します。

```
rhcert-run
```

システムに求められたら、システムに存在する利用可能な USB4 ポートの数を入力します。キーボードまたはマウスで現在使用されているものは数に入れないでください。システムは、テストの USB4 デバイスをポートに接続するように要求し、テスト担当者が **y** を押して続行するまで一時停止します。次に、システムはデバイスを外すように要求し、テスト担当者が **y** を押して続行するまで再び一時停止します。これらのステップは、入力したポート数に対して繰り返し行います。ポートをテストするための正しい順序や誤った順序はありませんが、各ポートは一度だけテストする必要があります。

ランタイム

USB4 テストは、USB4 ポートごとに約 15 秒かかります。これには、デバイスを手動で接続し、ポートのスキャン、デバイスのアンプラグ、およびポートの再スキャンを行う時間が含まれます。必要な **supportable** テストにより、全体の実行時間が約 1 分長くなります。

A.61. VIDEO

テストの対象

RHEL 7 および RHEL 8 の場合、**VIDEO** テストはマザーボード上のすべてのリムーバブルまたは統合ビデオハードウェアをチェックします。デバイスは、PCI クラス ID によるテストに選択されます。具体的には、テストでは、**udev** コマンド出力でディスプレイコントローラーとして PCI クラスを持つデバイスをチェックします。

RHEL 9 の場合、**VIDEO** テストは同じままです。ただし、フレームバッファーグラフィックソリューションの場合、テストは、ディスプレイカーネルドライバーがフレームバッファーとして使用されているかどうか、および **glxinfo** 直接レンダリングがサポートされていないかどうかを識別した後に計画されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストは、複数のサブテストを実行します。

1. Check Connections: **xrandr** コマンドの出力をログに記録します。このサブテストはオプションであり、失敗しても全体的なテスト結果に影響を与えません。
2. Set Configuration: 次のサブテストの表示深度、フラグ、設定など、必要な設定の前提条件を確認します。

3. X Server Test: 新しい設定ファイルを使用して別のディスプレイサーバーを起動し、軽 量 MESA OpenGL デモンストレーションプログラムである **glxgears** を実行してパフォーマンスを確認します。

4. Log Module and Drivers: **xdpyinfo** を実行して、画面の解像度と色深度を決定します。それに加えて、テストの開始時に作成された設定ファイルにより、システムを最大の解像度で実行できるようになります。

最後に、テストでは grep を使用して **/var/log/Xorg.0.log** ログファイルを検索し、使用中のモジュールとドライバーを特定します。

テストの準備

- システムのモニターとビデオカードが 1024x768 の解像度で実行され、24 ビット別のピクセル (bpp) の色深度を実行できることを確認します。より高い解像度または色深度も使用できます。**xrandr** コマンドの出力で 1024x768 (24 bpp 以上) を確認してください。

- カードまたはモニターの組み合わせで生成できるすべての解像度が表示されない場合は、モニターとビデオカードの間の KVM スイッチをすべて削除してください。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **VIDEO** テスト名を選択します。

```
rhcert-run
```

まず、テストシステム画面が空白になり、x11perf テストプログラムからの一連のテストパターンが表示されます。テストが完了すると、デスクトップまたは仮想ターミナルの画面に戻ります。

ランタイム

このテストの実行には約 1 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.62. VIDEO_PORTS

テストの対象

VIDEO_PORTS テストは、システム内の各グラフィックプロセッサーのポートをすべて出力するかどうかを確認します。

テストは、グラフィック出力ポートが 1 つ以上あるマシンで実行されます。一体型パネルに配線されたポートを備えたラップトップを含む、1 つ以上の組み込みまたはアドオングラフィックプロセッサーを搭載したマシンもサポートされています。

そのポートに接続されているディスプレイが検出されない場合は、テストがポートで実行されません。

RHEL バージョン対応

- RHEL 9

テストの内容

このテストでは、次のアクションが実行されます。

1. テストは、モニターが接続されていると検出する各ポートで実行されます。
2. 次に、テストは glmark2 ウィンドウを起動し、接続されている各ディスプレイにテストウィンドウをドラッグするように求めます。
3. テストされていないポートを新たに検出すると、対話モードになります。これにより、テストされていない各ポートにディスプレイをアタッチし、テストを繰り返すように求められます。
4. テストは、検出されたすべてのポートをテストするまで、またはテストされていないポートを顧客が使用できないことを示すまで、このループで実行され続けます。使用できないポートがある場合、テストは説明を求めるプロンプトを表示します。
5. ループが終了すると、すべてのポートがテストされた場合は PASS の結果を表示し、一部のポートが使用できないと識別された場合は REVIEW の結果を表示します。

テストの準備

- システムに適切なコネクターを備えた一連のモニターを準備します。これには、内蔵モニターと少なくとも 1 つの外部モニターが含まれます。
- ポートよりもモニターが少ない場合、テストはループで実行され、ディスプレイをポートにバッチで接続できます。内蔵モニターは、接続されている各外部モニターに加えて、引き続き動作する必要があります。
- 物理ポートよりも電子ポートの方が多い場合があります。つまり、システムがユーザーに提供する数よりも多くのディスプレイをハードウェアがサポートします。テスト開始時に画面に表示されるポートの一覧は、テストとは関係ありません。
- システム内には、一度に使用できる物理ポートが複数存在する場合があります。サービスポートや USB などの ghost ポートが存在する場合もあります。別のポートとの互換性がないため、または ghost ポートであるために機能していないポートと、まったく機能していないポートを区別できる必要があります。

テストの実行

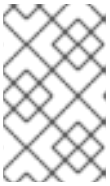
VIDEO_PORTS テストは対話的です。テストを実行する前に、モニターを少なくとも 1 つのグラフィック出力ポートに接続します。

1. システムをプロビジョニングします。
   a. このコマンドを実行します。

   ```
   # rhcert-provision
   ```

   b. プロンプトが表示されたら、システムに保存されているテストプランのパスを入力します。
   c. プロンプトが表示されたら、パスワードレス SSH を設定するには、テストサーバーのホスト名または IP アドレスを指定します。新しいシステムの初回追加時にのみプロンプトが表示されます。
2. テストを開始します。

   ```
   # rhcert-cli run --test VIDEO_PORTS
   ```



注記

このテストは、接続と切断の両方の内部ディスプレイのセットを一覧表示することから始まります。これらは、テストされている物理ポートを表すものではありません。

3. 接続されたグラフィック出力ポートごとに、以下の手順に従います。

   a. テストがポートを識別するまで待機します。プロンプトが表示されたら、任意のキーを押して続行します。

   b. **glmark2** ウィンドウが開きます。アクティブなモニターと異なる場合は、このウィンドウをポートに接続されているモニターに移動します。
   glmark2 ベンチマークは、識別されたディスプレイでの OpenGL (ES) 2.0 パフォーマンスのさまざまな側面を測定します。ベンチマークは、表面、角度、色、および光のさまざまな組み合わせをテストする一連のイメージを呼び出します。

   c. **glmark2** ウィンドウが閉じられるまで待ちます。成功したテストごとに、glmark2 スコアと **Test passed** メッセージが表示されます。

4. 接続されていないグラフィック出力ポートごとに、以下の手順に従います。

   a. プロンプトが表示されたら、モニターをグラフィックポートに接続し、**yes** を入力して続行します。
   最初のプロンプトで **no** を入力して GRAPHICS_PORTS テストを終了します。追加のプロンプトごとにタイマーが表示され、モニターへの接続に 20 秒猶予が与えられます。タイマーは、タイムアウトする前に 3 回繰り返されます。

   b. テストがポートを識別するまで待機します。プロンプトが表示されたら、任意のキーを押して続行します。

   c. ポートに接続されているモニターに **glmark2** ウィンドウを移動します。ウィンドウが閉じられるまで待ちます。

5. 接続する追加のポートがない場合は、タイムアウトになるまでタイマーを 60 秒間実行します。すべてのポートが正常にテストされると、テストは PASS 結果で終了します。

6. 必要に応じて、テスト結果をログファイルに保存します。

```
# rhcert-save
```

7. ログファイルの場所に移動して、ブラウザーからログファイルにアクセスします。

ランタイム

テスト時間は、テストしているポートの数によって異なります。各ポートは、テストに約 2 ~ 3 分かかります。

テスト時間に影響を与えるその他の要因には、メモリー周波数や CPU などのシステムパフォーマンスが含まれます。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.63. VIDEO_DRM

テストの対象

VIDEO_DRM テストは、基本的なグラフィックスをサポートするネイティブ DRM カーネルドライバーを利用するグラフィックコントローラーを検証します。

以下の場合に、テストが計画されます。

- 使用中のディスプレイドライバーが、カーネルモード設定ドライバーとして識別される。
- ディスプレイドライバーがフレームバッファーではない。
- **glxinfo** コマンドで識別されるように、直接レンダリングがサポートされておらず、OpenGL レンダラー文字列が **llvmpipe** である。

RHEL バージョン対応

- RHEL 9

テストの内容

このテストでは、VIDEO テストと同様にグラフィックコントローラーの機能を検証します。

テストの準備

- システムのモニターとビデオカードが 1024x768 の解像度で実行され、24 ビット別のピクセル (bpp) の色深度を実行できることを確認します。より高い解像度または色深度も使用できます。**xrandr** コマンドの出力で 1024x768 (24 bpp 以上) を確認してください。
- カードまたはモニターの組み合わせで生成できるすべての解像度が表示されない場合は、モニターとビデオカードの間の KVM スイッチをすべて削除してください。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されるリストから適切な **VIDEO_DRM** テスト名を選択します。

```
rhcert-run
```

まず、テストシステム画面が空白になり、x11perf テストプログラムからの一連のテストパターンが表示されます。テストが完了すると、デスクトップまたは仮想ターミナルの画面に戻ります。

ランタイム

このテストの実行には約 1 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.64. VIDEO_DRM_3D

テストの対象

VIDEO_DRM_3D テストは、高速化されたグラフィックスをサポートするネイティブ DRM カーネルドライバーを利用するグラフィックコントローラーを検証します。

以下の場合に、テストが計画されます。

- 使用中のディスプレイドライバーが、カーネルモード設定ドライバーとして識別される。
- ディスプレイドライバーがフレームバッファーではない。

- **glxinfo** コマンドで識別されるように、直接レンダリングがサポートされており、OpenGL レンダラー文字列が **llvmpipe** ではない。

このテストでは、Prime GPU オフロードテクノロジーを使用して、すべてのビデオテストサブテストを実行します。

RHEL バージョン対応

- RHEL 9

テストの内容

このテストでは、VIDEO テストと同様にグラフィックコントローラーの機能を検証します。さらに、テストは以下のサブテストを実行します。

1. Vulkaninfo テスト: **vulkaninfo** コマンド出力をログに記録して、識別された GPU のデバイスプロパティー、各 GPU でサポートされている Vulkan 拡張機能、認識されているレイヤー、サポートされているイメージ形式、形式プロパティーなどの Vulkan 情報を収集します。

2. Glmark2 benchmarking テスト: **glmark2** コマンドを実行して、OpenGL 2.0 および ES 2.0 ベンチマークテストセットに基づいてスコアを生成し、3D 機能を確認します。サブテストは、異なるパラメーターセットを使用してユーティリティーを 2 回実行します。最初はハードウェアレンダラーを使用し、後でソフトウェアレンダラーを使用します。ハードウェアレンダラーコマンドの実行結果がソフトウェアよりも優れている場合、テストは合格し、ディスプレイコントローラーの 3D 機能が優れていることが確認されます。それ以外の場合は失敗します。

テストの準備

- システムのモニターとビデオカードが 1024x768 の解像度で実行され、24 ビット別のピクセル (bpp) の色深度を実行できることを確認します。より高い解像度または色深度も使用できます。**xrandr** コマンドの出力で 1024x768 (24 bpp 以上) を確認してください。

- カードまたはモニターの組み合わせで生成できるすべての解像度が表示されない場合は、モニターとビデオカードの間の KVM スイッチをすべて削除してください。

テストの実行

このテストは非対話型です。次のコマンドを実行し、表示されたリストから適切な **VIDEO_DRM_3D** テスト名を選択します。

```
rhcert-run
```

まず、テストシステム画面が空白になり、x11perf テストプログラムからの一連のテストパターンが表示されます。テストが完了すると、デスクトップまたは仮想ターミナルの画面に戻ります。

ランタイム

このテストの実行には約 1 分かかります。その他の必須または選択されたテストは、全体の実行時間に追加されます。

A.65. WIRELESSG

テストの対象

WirelessG テストは、802.11g の最大接続速度を持つすべてのワイヤレスイーサネット接続で実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

これは、既存の **wlan** テストと **network** テストを組み合わせた新しいテストです。このテストでは、既存のネットワークテスト項目すべてに合格することに加えて、**iw** で報告された g リンクタイプを検出し、合格するためには最低 22Mb/s のスループットを実証する必要があります。

関連情報

- 残りのテスト機能については、ネットワーク を参照してください。

A.66. WIRELESSN

テストの対象

WirelessN テストは、802.11n の最大接続速度を持つすべてのワイヤレスイーサネット接続で実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

これは、既存の **wlan** テストと **network** テストを組み合わせた新しいテストです。このテストでは、既存のネットワークテスト項目すべてに合格することに加えて、**iw** で報告された g リンクタイプを検出し、合格するためには最低 100Mb/s のスループットを実証する必要があります。

関連情報

- 残りのテスト機能については、ネットワーク を参照してください。

A.67. WIRELESSAC

テストの対象

WirelessAC テストは、802.11ac の最大接続速度を持つすべてのワイヤレスイーサネット接続で実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8

- RHEL 9

テストの内容

これは、既存の **wlan** テストと **network** テストを組み合わせた新しいテストです。このテストでは、既存のネットワークテスト項目すべてに合格することに加えて、**iw** で報告された g リンクタイプを検出し、合格するためには最低 300Mb/s のスループットを実証する必要があります。

関連情報

- 残りのテスト機能の詳細については、ネットワーク を参照してください。

A.68. WIRELESSAX(WIFI6 による SUPERSED BY WIFI6)

テストの対象

WirelessAX テストは、802.11ax の最大接続速度を持つすべてのワイヤレスイーサネット接続で実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、**iw** で報告された "ax "リンクタイプを検出し、製品名に "wifi 6 "または "AX "が含まれているかどうかを照合して、デバイスが AX サポートを備えているかどうかを判断します。デバイスがワイヤレス AC テストに合格し、合格するための 1200 Mb/s の最小スループットを示す場合は、ワイヤレス AX のテストも計画されます。このテストは自動的に予定されていませんが、CLI を使用して手動で計画できます。代わりに、WiFi6 テストが自動的に計画されます。

A.69. WIFI6

テストの対象

WiFi6 テストは、802.11ax の最大接続速度を持つすべてのワイヤレスイーサネット接続で実行されます。

RHEL バージョン対応

- RHEL 7
- RHEL 8
- RHEL 9

テストの内容

このテストでは、**iw** によって報告された ax リンクタイプを検出し、wifi 6 または AX を含む製品名と照合して、デバイスが AX サポートを備えているかどうかを判断します。デバイスがワイヤレス AC テストを渡し、合格するために 1200 Mb/s の最小スループットを示す場合は、WiFi6 のテストも計画され

ます。

A.70. WIFI6E

テストの対象

WiFi6E テストは、6GHz 周波数帯域を利用した 802.11ax の最大接続速度を備えたすべてのワイヤレス接続で実行されます。

RHEL バージョン対応

- RHEL 8
- RHEL 9

テストの内容

このテストでは、**iw** によって報告された ax リンクタイプを検出し、wifi 6E または AX を含む製品名と照合して、デバイスが AX サポートを備えているかどうかを判断します。デバイスが WiFi6E テストに合格し、合格するために6000 Mb/s の最小スループットを示す場合は、ワイヤレス AX のテストも計画されます。

A.71. 手動でのテストの追加と実行

まれに、ハードウェア検出の問題や、ハードウェア、OS、またはテストスクリプトの他の問題により、テストが計画に失敗する場合があります。この場合は、Red Hat サポートにお問い合わせください。問題のサポートチケットを作成し、HUT の **rhcert-cli** コマンドを使用してテストをローカルテスト計画に手動で追加する方法を説明ように求められます。ローカルテスト計画に加えた変更は LTS に送信されるので、引き続き LTS で Web インターフェイスを使用してテストを実行できます。このコマンドは、以下のように実行します。

```
# rhcert-cli plan --add --test=<testname> --device=<devicename> --udi-<udi>
```

ここでは、**rhcert-cli** コマンドのオプションは次のとおりです。

- **plan** - テストプランの変更
- **--add** - テストプランに項目を追加
- **--test=<testname>** - 追加するテスト。テスト名は以下のとおりです。
  - hwcert/suspend
  - hwcert/audio
  - hwcert/battery
  - hwcert/lid
  - hwcert/usbbase/expresscard
  - hwcert/usbbase/usbbase/usb2
  - hwcert/usbbase/usbbase/usb3

- hwcert/kdump
- hwcert/network/Ethernet/100MegEthernet
- hwcert/network/Ethernet/1GigEthernet
- hwcert/network/Ethernet/10GigEthernet
- hwcert/network/Ethernet/40GigEthernet
- hwcert/network/wlan/WirelessG
- hwcert/network/wlan/WirelessN
- hwcert/network/wlan/WirelessAC(Red Hat Enterprise Linux 7 でのみ利用可能)
- hwcert/memory
- hwcert/core
- hwcert/cpuscaling
- hwcert/fvtest/fv_core
- hwcert/fvtest/fv_live_migration
- hwcert/fvtest/fv_memory
- hwcert/fvtest/fv_network
- hwcert/fvtest/fv_storage
- hwcert/fvtest/fv_pcie_storage_passthrough
- hwcert/fvtest/fv_pcie_network_passthrough
- hwcert/fvtest/fv_usb_storage_passthrough
- hwcert/fvtest/fv_usb_network_passthrough
- hwcert/fvtest/fv_cpu_pinning
- hwcert/profiler
- hwcert/storage
- hwcert/video
- hwcert/supportable
- hwcert/optical/bluray
- hwcert/optical/dvd
- hwcert/optical/cdrom
- hwcert/fencing

- hwcert/realtime
- hwcert/reboot
- hwcert/tape
- hwcert/rdma/Infiniband_QDR
- hwcert/rdma/Infiniband_FDR
- hwcert/rdma/Infiniband_EDR
- hwcert/rdma/Infiniband_HDR
- hwcert/rdma/Infiniband_Socket_Direct
- hwcert/rdma/10GigRoCE
- hwcert/rdma/20GigRoCE
- hwcert/rdma/25GigRoCE
- hwcert/rdma/40GigRoCE
- hwcert/rdma/50GigRoCE
- hwcert/rdma/100GigRoCE
- hwcert/rdma/200GigRoCE
- hwcert/rdma/10GigiWarp
- hwcert/rdma/20GigiWarp
- hwcert/rdma/25GigiWarp
- hwcert/rdma/40GigiWarp
- hwcert/rdma/50GigiWarp
- hwcert/rdma/100GigiWarp
- hwcert/rdma/200GigiWarp
- hwcert/rdma/Omnipath
- hwcert/network/Ethernet/2_5GigEthernet
- hwcert/network/Ethernet/5GigEthernet
- hwcert/network/Ethernet/20GigEthernet
- hwcert/network/Ethernet/25GigEthernet-
- hwcert/network/Ethernet/50GigEthernet
- hwcert/network/Ethernet/100GigEthernet

- hwcert/network/Ethernet/200GigEthernet
- rhcert/self-check
- hwcert/sosreport
- hwcert/storage/U2 SATA
- hwcert/storage/M2 SATA
- hwcert/storage/SATA_SSD
- hwcert/storage/SATA
- hwcert/storage/SAS_SSD
- hwcert/storage/SAS
- hwcert/storage/U2_NVME
- hwcert/storage/M2_NVME
- hwcert/storage/PCIE_NVME
- hwcert/storage/NVDIMM
- hwcert/storage/STORAGE

- 他のオプションは、デバイスを指定する必要がある場合にのみ必要です。たとえば、ネットワークテストおよびストレージテストで、どのデバイスで実行するかを指示する必要があります。ここで使用されるデバイス名または UDI を決定するために、探す必要のある場所が様々にあります。サポートは、適切な名前または UDI の特定に役立ちます。見つかったら、次の 2 つのオプションのいずれかを使用してデバイスを指定します。
  - **--device=<devicename>** - "enp0s25" や "host0" などのデバイス名で識別される必要のあるデバイスです。
  - **--udi=<UDI>** - テストされるデバイスの一意のデバイス ID。UDI 文字列で識別されます。
- テスト名を指定して rhcert-cli コマンドを実行します。

```
rhcert-cli run --test=<test_name>
```

  以下に例を示します。

```
rhcert-cli run --test=audio
```

- **--device** を指定して、特定のデバイスを実行できます。

```
rhcert-cli run --test=<test name> --device=<device name>
```

  以下に例を示します。

```
rhcert-cli run --test=kdump --device=nfs
```

注記

rhcert-cli または **rhcert-run** を個別に使用し、結果を保存することを推奨します。**rhcert-cli** と **rhcert-run** の両方を混合して使用し、結果を一緒に保存すると、結果を正しく処理できなくなる可能性があります。

改訂日時: 2024-05-11