Red Hat Enterprise Linux 9

9.1 リリースノート

Red Hat Enterprise Linux 9.1 リリースノート

Last Updated: 2024-06-27

Red Hat Enterprise Linux 9 9.1 リリースノート

Red Hat Enterprise Linux 9.1 リリースノート

法律上の通知

概要

このリリースノートでは、Red Hat Enterprise Linux 9.1 での改良点および実装された追加機能の概要、このリリースにおける既知の問題などを説明します。また、重要なバグ修正、テクニカルプレビュー、非推奨機能などの詳細も説明します。 Red Hat Enterprise Linux をインストールする方法の詳細については、Installation に進んでください。

目次

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに関するご意見やご感想をお寄せください。また、改善点があればお知らせください。

Jira からのフィードバック送信 (アカウントが必要)

1. Jira の Web サイトにログインします。
2. 上部のナビゲーションバーで **Create** をクリックします。
3. **Summary** フィールドにわかりやすいタイトルを入力します。
4. **Description** フィールドに、ドキュメントの改善に関するご意見を記入してください。ドキュメントの該当部分へのリンクも追加してください。
5. ダイアログの下部にある **Create** をクリックします。

第1章 概要

1.1. RHEL 9.1 における主な変更点

インストーラーおよびイメージの作成

以下は、RHEL 9.1-GA の Image Builder に関する重要事項です。

- オンプレミスの Image Builder は以下をサポートするようになりました。
  - GCP へのイメージのアップロード
  - **/boot** パーティションのカスタマイズ
  - コンテナーイメージのレジストリーへの直接プッシュ
  - ユーザーによるイメージ作成プロセス中のブループリントのカスタマイズ

詳細は、「インストーラーおよびイメージの作成」 を参照してください。

RHEL for Edge

RHEL 9.1-GA の RHEL for Edge の主な特色を以下に示します。

- RHEL for Edge は、サービスのインストールをサポートし、**fdo-admin** CLI ユーティリティーを使用してデフォルト設定で実行できるようになりました。

詳細は、「RHEL for Edge」 を参照してください。

セキュリティー

RHEL 9.1 では、トラステッドプラットフォームモジュール (TPM) テクノロジーを使用したリモートマシン構成証明ツールである **Keylime** が導入されています。Keylime を使用すると、リモートマシンのインテグリティーを検証し、継続的にモニタリングできます。

SELinux ユーザー空間パッケージがバージョン 3.4 にアップグレードされました。以下は、主な変更点です。

- ラベル再設定の並列処理によるラベル再設定パフォーマンスの向上
- **semodule** ツールでの SHA-256 のサポート
- **libsepol-utils** パッケージの新しいポリシーユーティリティー

システム設定と **clevis-luks-systemd** サブパッケージの変更により、Clevis 暗号化クライアントは、デプロイメントプロセス中に **systemctl enable clevis-luks-askpass.path** コマンドを使用せずに、起動プロセスの後半にマウントされる LUKS 暗号化ボリュームもロック解除できるようになりました。

詳細は、新機能 - セキュリティー を参照してください。

シェルおよびコマンドラインツール

RHEL 9.1 では、新しいパッケージ **xmlstarlet** が導入されました。**XMLStarlet** を使用すると、XML ファイルを解析、変換、クエリー、検証、および編集できます。

次のコマンドラインツールが RHEL 9.1 で更新されました。

- **opencryptoki** をバージョン 3.18.0 に更新
- **powerpc-utils** をバージョン 1.3.10 に更新

- **libvpd** をバージョン 2.2.9 に更新
- **lsvpd** をバージョン 1.7.14 に更新
- **ppc64-diag** をバージョン 2.7.8 に更新

詳細については、新機能 - シェルおよびコマンドラインツール を参照してください。

インフラストラクチャーサービス

RHEL 9.1 では、次のインフラストラクチャーサービスツールが更新されました。

- **chrony** をバージョン 4.2 に更新
- **unbound** tをバージョン 1.16.2 に更新
- **frr** をバージョン 8.2.2 に更新

詳細については、新機能 - インフラストラクチャーサービス を参照してください。

ネットワーク

NetworkManager は、非推奨の **ifcfg** 形式から keyfile 形式への接続プロファイルの移行をサポートしています。

NetworkManager は、WEP サポートが RHEL 9 で利用できないことを明確に示すようになりました。

カーネルの MultiPath TCP (MPTCP) コードが、アップストリームの Linux 5.19 から更新されました。

詳細については、新機能 - ネットワーキング を参照してください。

動的プログラミング言語、Web サーバー、およびデータベースサーバー

以下のコンポーネントの後続のバージョンが、新しいモジュールストリームとして利用できるようになりました。

- PHP 8.1
- Ruby 3.1
- Node.js 18

さらに、Apache HTTP Server がバージョン 2.4.53 に更新されました。

詳細は、新機能 - 動的プログラミング言語、Web サーバー、およびデータベースサーバー を参照してください。

コンパイラーおよび開発ツール

更新されたシステムツールチェーン

RHEL 9.1 では、以下のシステムツールチェインコンポーネントが更新されました。

- GCC 11.2.1
- glibc 2.34
- binutils 2.35.2

パフォーマンスツールとデバッガーの更新

RHEL 9.1 では、以下のパフォーマンスツールおよびデバッガーが更新されました。

- GDB 10.2

- Valgrind 3.19
- SystemTap 4.7
- Dyninst 12.1.0
- elfutils 0.187

更新されたパフォーマンスモニタリングツール
RHEL 9.1 では、以下のパフォーマンス監視ツールが更新されました。

- PCP 5.3.7
- Grafana 7.5.13

更新されたコンパイラーツールセット
次のコンパイラーツールセットが RHEL 9.1 で更新されました。

- GCC Toolset 12
- LLVM Toolset 14.0.6
- Rust Toolset 1.62
- Go Toolset 1.18

詳細な変更は、「コンパイラーおよび開発ツール」 を参照してください。

RHEL 9 の Java 実装
RHEL 9 AppStream リポジトリーには、以下が含まれます。

- **java-17-openjdk** パッケージ。OpenJDK 17 Java Runtime Environment および OpenJDK 17 Java Software Development Kit を提供します。
- **java-11-openjdk** パッケージ。OpenJDK 11 Java Runtime Environment および OpenJDK 11 Java Software Development Kit を提供します。
- **java-1.8.0-openjdk** パッケージ。OpenJDK 8 Java Runtime Environment および OpenJDK 8 Java Software Development Kit を提供します。

詳細は、OpenJDK のドキュメント を参照してください。

Java ツール
RHEL 9.1 では、新しいモジュールストリームとして **Maven 3.8** が導入されています。

詳細は、「コンパイラーおよび開発ツール」 を参照してください。

Identity Management
RHEL 9.1 の Identity Management (IdM) にはテクノロジープレビュー機能が導入されており、OAuth 2 Device Authorization Grant フローをサポートする外部 ID プロバイダー (IdP) にユーザー認証を委任できます。これらのユーザーが SSSD で認証されると、外部 IdP で認証と許可が完了すると、Kerberos チケットを使用して RHEL IdMSingle Sign-On 機能を受け取ります。

詳細は、テクノロジープレビュー - Identity Management を参照してください。

Red Hat Enterprise Linux システムロール
9.1 RHEL システムロールの主な新機能:

- RHEL システムロールが、ファクト収集が無効になっている Playbook でも利用できるようになりました。
- **ha_cluster** ロールは、SBD フェンシング、Corosync 設定、およびバンドルリソースの設定をサポートするようになりました。
- **network** ロールは、ルーティングルールのネットワークオプションの設定と **nmstate API** を使用したネットワーク設定のサポートするようになり、ユーザーは IPoIB 機能を使用して接続を作成できるようになりました。
- **microsoft.sql.server** ロールには、高可用性クラスターの設定を制御する変数、ファイアウォールポートを自動的に管理するための変数、管理対象ノードで **mssql_tls_cert** および **mssql_tls_private_key** 値を検索する変数など、新しい変数があります。
- **logging** ロールは、さまざまな新しいオプションをサポートします。たとえば、ファイル入力の **startmsg.regex** と **endmsg.regex**、または **template**、**severity**、および **facility** オプションです。
- **storage** ロールには、シンプロビジョニングされたボリュームのサポートが追加され、ロールはデフォルトで詳細レベルが低くなりました。
- **sshd** ロールは、ドロップインディレクトリーの include ディレクティブを検証し、/etc/ssh/sshd_config を通じてロールを管理できるようになりました。
- **metrics** ロールは、postfix パフォーマンスデータをエクスポートできるようになりました。
- **postfix** ロールには、以前の設定を上書きする新しいオプションが追加されました。
- **firewall** ロールは、masquerade または icmp_block_inversion を設定する際に state パラメーターを必要としません。**firewall** ロールでは、absent および present 状態を使用してサービスを追加、更新、または削除できるようになりました。また、このロールは Ansible ファクトを提供し、PCI デバイス ID を使用してインターフェイスの追加や削除も可能になりました。**firewall** ロールには、以前の設定を上書きする新しいオプションがあります。
- **selinux** ロールには、**seuser** および **selevel** パラメーターの設定が含まれるようになりました。

1.2. インプレースアップグレード

RHEL 8 から RHEL 9 へのインプレースアップグレード

現在サポートされているインプレースアップグレードパスは次のとおりです。

- 以下のアーキテクチャーで、RHEL 8.6 から RHEL 9.0 へ:
  - 64 ビット Intel
  - 64 ビット AMD
  - 64-bit ARM
  - IBM POWER 9 (リトルエンディアン)
  - z13 を除く IBM Z アーキテクチャー
- SAP HANA を使用するシステムの RHEL 8.6 から RHEL 9.0 へ

RHEL 9.0 へのアップグレード後もシステムが引き続きサポートされるようにするには、最新の RHEL 9.1 バージョンに更新するか、RHEL 9.0 Extended Update Support (EUS) リポジトリーを有効にします。

インプレースアップグレードの実行方法は、RHEL 8 から RHEL 9 へのアップグレード を参照してください。

SAP 環境があるシステムでインプレースアップグレードを実行する手順については、SAP 環境を RHEL 8 から RHEL 9 にインプレースアップグレードする方法 を参照してください。

主な機能拡張は、次のとおりです。

- Red Hat Update Infrastructure (RHUI) を使用した Microsoft Azure および Google Cloud Platform でのインプレースアップグレードが可能になりました。
- OpenSSH および OpenSSL 設定は、インプレースアップグレード中に移行されるようになりました。

RHEL7 から RHEL 9 へのインプレースアップグレード
RHEL7 から RHEL 9 へのインプレースアップグレードを直接実行することはできません。ただし、RHEL 7 から RHEL 8 へのインプレースアップグレードを実行してから、RHEL 9 への 2 回目のインプレースアップグレードを実行することはできます。詳細は、RHEL7 から RHEL8 へのアップグレードを参照してください。

1.3. RED HAT CUSTOMER PORTAL LABS

Red Hat Customer Portal Labsは、カスタマーポータルのセクションにあるツールセットで、https://access.redhat.com/labs/ から入手できます。Red Hat Customer Portal Labs のアプリケーションは、パフォーマンスの向上、問題の迅速なトラブルシューティング、セキュリティー問題の特定、複雑なアプリケーションの迅速なデプロイメントおよび設定に役立ちます。最も一般的なアプリケーションには、以下のものがあります。

- Registration Assistant
- Kickstart Generator
- Red Hat Product Certificates
- Red Hat CVE Checker
- Kernel Oops Analyzer
- Red Hat Code Browser
- VNC Configurator
- Red Hat OpenShift Container Platform Update Graph
- Red Hat Satellite Upgrade Helper
- JVM Options Configuration Tool
- Load Balancer Configuration Tool
- Red Hat OpenShift Data Foundation サポートおよび相互運用性チェッカー
- Ansible Automation Platform Upgrade Assistant

- Ceph Placement Groups (PGs) per Pool Calculator

1.4. 関連情報

他のバージョンと比較した Red Hat Enterprise Linux 9 の **機能および制限** は、Red Hat ナレッジベースの記事Red Hat Enterprise Linux テクノロジーの機能と制限 を参照してください。

Red Hat Enterprise Linux の **ライフサイクル** に関する情報は Red Hat Enterprise Linux のライフサイクル を参照してください。

パッケージマニフェスト ドキュメントは、ライセンスとアプリケーションの互換性レベルを含む、RHEL 9 の **パッケージリスト** を提供します。

アプリケーションの互換性レベル は、 Red Hat Enterprise Linux 9: アプリケーション互換性ガイド ドキュメントで説明されています。

削除された機能を含む主な**RHEL 8 と RHEL 9 の相違点**は、RHEL 9 の導入における考慮事項 で説明されています。

RHEL 8 から RHEL 9 へのインプレースアップグレード を実行する方法は、Upgrading from RHEL 8 to RHEL 9を参照してください。

すべての RHEL サブスクリプションで、既知の技術問題の特定、検証、および解決をプロアクティブに行う **Red Hat Insights** サービスが利用できます。Red Hat Insights クライアントをインストールし、システムをサービスに登録する方法は、Red Hat Insights を使い始める ページを参照してください。

第2章 アーキテクチャー

Red Hat Enterprise Linux 9.1 は、カーネルバージョン 5.14.0-162 と共に配布されます。これは、最低限必要なバージョンで次のアーキテクチャーをサポートします。

- AMD および Intel 64 ビットアーキテクチャー (x86-64-v2)
- 64 ビット ARM アーキテクチャー (ARMv8.0-A)
- IBM Power Systems (リトルエンディアン) (POWER9)
- 64 ビット IBM Z (z14)

各アーキテクチャーに適切なサブスクリプションを購入してください。詳細は Get Started with Red Hat Enterprise Linux - additional architectures を参照してください。

第3章 RHEL 9 のコンテンツの配布

3.1. インストール

Red Hat Enterprise Linux 9 は、ISO イメージを使用してインストールします。AMD64、Intel 64 ビット、64 ビット ARM、IBM Power Systems、IBM Z アーキテクチャーで、以下の 2 種類のインストールメディアが利用できます。

- インストール ISO: BaseOS リポジトリーおよび AppStream リポジトリーが含まれ、リポジトリーを追加しなくてもインストールを完了できる完全インストールイメージです。製品のダウンロード ページでは、**インストール ISO** は **バイナリー DVD** と呼ばれます。

注記

インストール用 ISO イメージのサイズは複数 GB であるため、光学メディア形式には適合しない場合があります。インストール ISO イメージを使用して起動可能なインストールメディアを作成する場合は、USB キーまたは USB ハードドライブを使用することが推奨されます。Image Builder ツールを使用すれば、RHEL イメージをカスタマイズできます。Image Builder の詳細は **Composing a customized RHEL system image** を参照してください。

- Boot ISO - インストールプログラムを起動するのに使用する最小限の ISO ブートイメージです。このオプションでは、ソフトウェアパッケージをインストールするのに、BaseOS リポジトリーおよび AppStream リポジトリーにアクセスする必要があります。リポジトリーは、Installation ISO イメージの一部です。インストール中に Red Hat CDN または Satellite に登録して、Red Hat CDN または Satellite から最新の BaseOS および AppStream コンテンツを使用することもできます。

ISO イメージのダウンロード、インストールメディアの作成、RHEL 9 インストールの完了の方法は、標準的な RHEL 9 インストールの実行 を参照してください。自動化したキックスタートインストールなどの高度なトピックは 高度な RHEL 9 インストールの実行 を参照してください。

ベース RHEL インストールで RPM によって作成されるユーザーとグループのリスト、およびこのリストを取得する手順は、ベース RHEL インストールのすべてのユーザーとグループはどのようなものですか? を参照してください。ナレッジベースの記事。

3.2. リポジトリー

Red Hat Enterprise Linux 9 は、2 つのメインリポジトリーで配布されています。

- BaseOS
- AppStream

基本的な RHEL インストールにはどちらのリポジトリーも必要で、すべての RHEL サブスクリプションで利用できます。

BaseOS リポジトリーのコンテンツは、すべてのインストールのベースとなる、基本的な OS 機能のコアセットを提供します。このコンテンツは RPM 形式で提供されており、RHEL の以前のリリースと同様のサポート条件が適用されます。詳細は、対象範囲の詳細 を参照してください。

AppStream リポジトリーには、さまざまなワークロードとユースケースに対応するために、ユーザー空間アプリケーション、ランタイム言語、およびデータベースが同梱されます。

また、CodeReady Linux Builder リポジトリーは、すべての RHEL サブスクリプションで利用できます。このリポジトリーは、開発者向けの追加パッケージを提供します。CodeReady Linux Builder リポジトリーに含まれるパッケージは、サポート対象外です。

RHEL 9 リポジトリーとそれらが提供するパッケージの詳細は、パッケージマニフェスト を参照してください。

3.3. APPLICATION STREAMS (APPSTREAM)

複数のバージョンのユーザー空間コンポーネントが Application Streams として提供され、BaseOS リポジトリーよりも頻繁に更新されます。これにより、プラットフォームや特定のデプロイメントの基盤となる安定性に影響を及ぼさずに、RHEL をより柔軟にカスタマイズできます。

Application Streams は、通常の RPM 形式で、モジュールと呼ばれる RPM 形式への拡張として、Software Collections として、または Flatpak として利用できます。

各 Application Streams コンポーネントには、RHEL 9 と同じか、より短いライフサイクルが指定されています。RHEL のライフサイクル情報は、Red Hat Enterprise Linux のライフサイクル を参照してください。

RHEL 9 では、従来の **dnf install** コマンドを使用して RPM パッケージとしてインストールできる最初の Application Streams バージョンを提供することで、Application Streams エクスペリエンスを向上させています。

注記

RPM 形式を使用する初期 Application Streams の中には、Red Hat Enterprise Linux 9 よりも短いライフサイクルのものがあります。

追加の Application Streams バージョンの中には、将来のマイナー RHEL 9 リリースで、ライフサイクルが短いモジュールとして配布されるものがあります。モジュールは、論理ユニット (アプリケーション、言語スタック、データベース、またはツールセット) を表すパッケージの集まりです。これらのパッケージはまとめてビルドされ、テストされ、そしてリリースされます。

Application Streams のどのバージョンをインストールするかについて決めるには、まず Red Hat Enterprise Linux Application Streams ライフサイクル を確認してください。

代替コンパイラーやコンテナーツールなど、迅速な更新を必要とするコンテンツは、代替バージョンを並行して提供しないローリングストリームで利用できます。ローリングストリームは、RPM またはモジュールとしてパッケージ化されることがあります。

RHEL 9 で使用可能な Application Streams とそのアプリケーション互換性レベルについては、パッケージマニフェスト を参照してください。アプリケーションの互換性レベルは、Red Hat Enterprise Linux 9: アプリケーション互換性ガイド ドキュメントで説明されています。

3.4. YUM/DNF を使用したパッケージ管理

Red Hat Enterprise Linux 9 では、ソフトウェアインストールは **DNF** により保証されます。Red Hat は、以前の RHEL のメジャーバージョンとの整合性を保つため、**yum** コマンドの使用を引き続きサポートします。**yum** の代わりに **dnf** と入力しても、どちらも互換性のためのエイリアスなので、コマンドは期待通りに動作します。

RHEL 8 と RHEL 9 は**DNF**をベースにしていますが、RHEL 7 で使用していた**YUM**との互換性があります。

詳細は、DNF ツールを使用したソフトウェアの管理 を参照してください。

第4章 新機能

このパートでは、Red Hat Enterprise Linux 9.1 で導入された新機能と主な拡張機能について説明します。

4.1. インストーラーおよびイメージの作成

インストーラーで自動 FCP SCSI LUN スキャンがサポートされました

インストーラーは、IBM Z システムに FCP SCSI LUN をアタッチする際に、自動 LUN スキャンを使用できるようになりました。自動 LUN スキャンは、**zfcp.allow_lun_scan** カーネルモジュールパラメーターで無効にされていない場合、NPIV モードで動作する FCP デバイスで使用できます。これは、デフォルトで有効になっています。これは、指定されたデバイスバス ID を持つ FCP デバイスに接続されたストレージエリアネットワークで見つかったすべての SCSI デバイスへのアクセスを提供します。WWPN と FCP LUN を指定する必要はなくなり、FCP デバイスバス ID のみ指定する必要があります。

(BZ#1937031)

Image Builder オンプレミスが /boot パーティションのカスタマイズをサポートするようになりました

Image Builder オンプレミスバージョンは、カスタムの **/boot** マウントポイントパーティションサイズを使用したイメージのビルドをサポートするようになりました。ブループリントのカスタマイズで **/boot** マウントポイントパーティションのサイズを指定し、デフォルトのブートパーティションサイズが小さすぎる場合に **/boot** パーティションのサイズを増やすことができます。以下に例を示します。

```
[[customizations.filesystem]]
mountpoint = "/boot"
size = "20 GiB"
```

(JIRA:RHELPLAN-130379)

パスワードベースの SSH root ログインを有効にする --allow-ssh キックスタートオプションが追加されました

グラフィカルインストール中に、オプションとしてパスワードベースの SSH ルートログインを有効にできます。この機能は、キックスタートインストールでは利用できませんでした。今回の更新で、オプション **--allow-ssh** が **rootpw** キックスタートコマンドに追加されました。このオプションを使用すると、root ユーザーはパスワード付きの SSH を使用してシステムにログインできます。

(BZ#2083269)

ブートローダーメニューがデフォルトで非表示

GRUB ブートローダーは、デフォルトでブートメニューを非表示にするように設定されるようになりました。これにより、起動がよりスムーズになります。ブートメニューは、次のすべての場合で非表示になります。

- デスクトップ環境またはログイン画面からシステムを再起動した場合。
- インストール後の最初のシステム起動時。
- **greenboot** パッケージがインストールされ、有効になっている場合。

前回のシステムブートが失敗した場合、GRUB は次回のブート時に必ずブートメニューを表示します。

ブートメニューに手動でアクセスするには、次のいずれかのオプションを使用します。

- ブート中に **Esc** を繰り返し押す。
- ブート中に **F8** を繰り返し押す。
- 起動時に**Shift**キーを押したままにする。

この機能を無効にし、ブートローダーメニューをデフォルトで表示するように設定するには、次のコマンドを使用します。

```
# grub2-editenv - unset menu_auto_hide
```

(BZ#2059414)

最小限の RHEL インストールでは、s390utils-core パッケージのみがインストールされるようになりました

RHEL 8.4 以降では、**s390utils-base** パッケージは、**s390utils-core** パッケージと補助 **s390utils-base** パッケージに分割されています。そのため、RHEL インストールを **minimal-environment** に設定すると、必要な **s390utils-core** パッケージのみがインストールされ、補助 **s390utils-base** パッケージはインストールされません。最小限の RHEL インストールで **s390utils-base** パッケージを使用する場合は、RHEL インストールの完了後にパッケージを手動でインストールするか、キックスタートファイルを使用して **s390utils-base** を明示的にインストールする必要があります。

(BZ#1932480)

Image Builder オンプレミスで GCP へのイメージのアップロードがサポートされました

この機能強化により、Image Builder CLI を使用して **gce** イメージを作成し、イメージのアップロードに使用するユーザーまたはサービスアカウントのクレデンシャルを指定できます。その結果、Image Builder はイメージを作成し、指定した GCP 環境に **gce** イメージを直接アップロードします。

(BZ#2049492)

Image Builder オンプレミス CLI で、レジストリーへのコンテナーイメージの直接プッシュがサポートされました

今回の機能強化により、Image Builder CLI を使用して、ビルド後に RHEL for Edge Container イメージをコンテナーレジストリーに直接プッシュできます。Container Image をビルドするには、以下を実行します。

1. アップロードプロバイダーを設定し、必要に応じてクレデンシャルを追加します。
2. コンテナーイメージをビルドし、コンテナーレジストリーとリポジトリーを **composer-cli** に引数として渡します。
   イメージの準備ができたら、設定したコンテナーレジストリーで使用できます。

(JIRA:RHELPLAN-130376)

Image Builder オンプレミスのユーザーは、イメージ作成プロセス中にブループリントをカスタマイズできるようになりました

今回の更新では、**Edit Blueprint** ページが削除され、Image Builder サービスと **cockpit-composer** の Image Builder アプリのユーザーエクスペリエンスが統合されました。ユーザーは、ブループリントを作成し、イメージ作成プロセス中にパッケージの追加やユーザーの追加などのカスタマイズを行うこと

が可能になりました。ブループリントのバージョン管理も削除されたため、ブループリントのバージョンは1つだけ (現在のバージョン) になります。ユーザーは、作成済みのイメージから古いブループリントバージョンにアクセスできます。

(JIRA:RHELPLAN-122735)

4.2. RHEL FOR EDGE

RHEL for Edge が fdo-admin cli ユーティリティーをサポートするようになりました

今回の更新により、CLI を使用して、すべてのデプロイメントシナリオで FDO サービスを直接設定できるようになりました。

次のコマンドを実行して、サービスの証明書とキーを生成します。

注記

この例では、**fdo-admin-cli** RPM パッケージがすでにインストールされていることを考慮しています。ソースコードを使用してコンパイルした場合、ビルドオプションに応じて、正しいパスは **./target/debug/fdo-admin-tool** または **./target/debug/fdo-admin-tool** になります。

```
$ mkdir keys
$ for i in "diun" "manufacturer" "device_ca" "owner"; do fdo-admin-tool generate-key-and-cert $i; done
$ ls keys
device_ca_cert.pem  device_ca_key.der  diun_cert.pem  diun_key.der  manufacturer_cert.pem
manufacturer_key.der  owner_cert.pem  owner_key.der
```

その結果、サービスをインストールして開始すると、デフォルト設定で実行されます。

(JIRA:RHELPLAN-122776)

4.3. サブスクリプションの管理

subscription-manager ユーティリティーは、現在の操作ステータスを表示します

現在の操作を処理している間、**subscription-manager** ユーティリティーは進捗情報を表示するようになりました。これは、**subscription-manager** がサーバー通信に関連する操作 (登録など) を完了するのに通常よりも時間がかかる場合に役立ちます。

以前の動作に戻すには、次のように入力します。

```
# subscription-manager config --rhsm.progress_messages=0
```

(BZ#2092014)

4.4. ソフトウェア管理

modulesync コマンドを使用して、RHEL 9 の特定のワークフローを置き換えることができるようになりました

RHEL 9 では、モジュラーメタデータがないとモジュラーパッケージをインストールできません。以前は、**dnf** コマンドを使用してパッケージをダウンロードしてから、**createrepo_c** コマンドを使用してそれらのパッケージを再配布できました。

この機能拡張により、**modulesync** コマンドが導入され、モジュラーメタデータの存在が保証され、パッケージのインストールが保証されます。このコマンドは、モジュールから RPM パッケージをダウンロードし、作業ディレクトリーにモジュラーメタデータを含むリポジトリーを作成します。

(BZ#2066646)

4.5. シェルおよびコマンドラインツール

Cronie に、選択した範囲内での時間のランダム化に対するサポートが追加されました

Cronie ユーティリティーは、cronjob の実行で ~(範囲内でのランダム化) Operator をサポートするようになりました。その結果、選択した範囲内のランダムな時間に cronjob を開始できます。

(BZ#2090691)

ReaR は、リカバリー前後にコマンドを実行する新しい変数を追加します

この機能強化により、ReaR にはリカバリー前後に実行されるコマンドの自動化を容易にする 2 つの新しい変数が導入されます。

- **PRE_RECOVERY_COMMANDS** ではコマンド配列を使用できます。これらのコマンドは、リカバリー開始前に実行されます。
- **POST_RECOVERY_COMMANDS** ではコマンド配列を使用できます。これらのコマンドは、リカバリー完了後に実行されます。

これらの変数は、**PRE_RECOVERY_SCRIPT** および **POST_RECOVERY_SCRIPT** に代わるものですが、次の違いがあります。

- 以前の **PRE_RECOVERY_SCRIPT** および **POST_RECOVERY_SCRIPT** 変数では、単一のシェルコマンドを使用できます。これらの変数に複数のコマンドを渡すには、コマンドをセミコロンで区切る必要があります。
- 新しい **PRE_RECOVERY_COMMANDS** および **POST_RECOVERY_COMMANDS** 変数ではコマンド配列が使用でき、配列の各要素が個別のコマンドとして実行されます。

その結果、リカバリー前後にレスキューシステムで実行する複数のコマンドを提供することが容易になり、エラーが発生しにくくなりました。

詳細については、**default.conf** ファイルを参照してください。

(BZ#2111059)

新しいパッケージ: xmlstarlet

XMLStarlet は、XML ファイルの解析、変換、クエリー、検証、および編集を行うための一連のコマンドラインユーティリティーです。新しい **xmlstarlet** パッケージ
は、**grep**、**sed**、**awk**、**diff**、**patch**、**join** などのプレーンテキストファイルに対して UNIX コマンドを使用する場合と同様の方法で使用できる、シェルコマンドの単純なセットを提供します。

(BZ#2069689)

opencryptoki がバージョン 3.18.0 にリベースされました

Public-Key Cryptography Standard (PKCS) #11 の実装である **opencryptoki** パッケージがバージョン 3.18.0 に更新されました。以下は、主な改善点です。

- デフォルトは連邦情報処理標準 (FIPS) 準拠のトークンデータ形式 (tokversion = 3.12) です。
- グローバルポリシーでメカニズムとキーの使用を制限するためのサポートが追加されました。
- メカニズム使用状況の統計カウントのサポートが追加されました。
- **ICA/EP11** トークンが **libica** ライブラリーバージョン 4 をサポートするようになりました。
- **p11sak** ツールを使用すると、公開鍵と秘密鍵に異なる属性を設定できます。
- **C_GetMechanismList** は、EP11 トークンで **CKR_BUFFER_TOO_SMALL** を返しません。

openCryptoki は、2 つの異なるトークンデータ形式をサポートしています。

- FIPS で承認されていないアルゴリズム (DES や SHA1 など) を使用する以前のデータ形式
- FIPS で承認されたアルゴリズムのみを使用する新しいデータ形式

FIPS プロバイダーは FIPS で承認されたアルゴリズムのみ使用を許可しているため、古いデータ形式は機能しなくなりました。

重要

openCryptoki を RHEL 9 で機能させるには、システムで FIPS モードを有効にする前に、トークンを移行して新しいデータ形式を使用します。**openCryptoki 3.17** では以前のデータ形式がデフォルトのままであるため、これが必要です。システムが FIPS 対応に変更されると、以前びトークンデータ形式を使用する既存の **openCryptoki** インストールは機能しなくなります。

openCryptoki で提供される **pkcstok_migrate** ユーティリティーを使用して、トークンを新しいデータ形式に移行できます。移行中は、**pkcstok_migrate** は FIPS で承認されていないアルゴリズムを使用することに注意してください。したがって、システムで FIPS モードを有効にする前に、このツールを使用します。詳細は、FIPS 準拠への移行 -pkcstok_migrate ユーティリティーを 参照してください。

(BZ#2044179)

powerpc-utils がバージョン 1.3.10 にリベースされました

PowerPC プラットフォーム用のさまざまなユーティリティーを提供する **powerpc-utils** パッケージがバージョン 1.3.10 に更新されました。以下は、主な改善点です。

- **ppc64_cpu** ツールのエネルギーと周波数に関する Power Architecture Platform Reference (PAPR) 情報を解析する機能が追加されました。
- 最大設定システムで **lparstat -E** コマンドが失敗した場合に、拡張エラーメッセージを表示するように **lparstat** ユーティリティーを改善しました。**lparstat** コマンドは、論理パーティション関連の情報を報告します。
- 報告された **lparstat** コマンドにおけるレガシー形式のオンラインメモリーが修正されました。
- NX GZIP アクセラレータのサービス品質クレジット (QoS) を動的に変更する **acc** コマンドのサポートが追加されました。
- **printf()** および **sprintf()** 呼び出しの書式指定子が改善されました。

- HMC ツールをハイブリッド仮想ネットワークに提供する **hcnmgr** ユーティリティーには、以下の機能拡張が含まれます。
  - ハイブリッドネットワーク仮想化 **HNV FEATURE** リストに **wicked** 機能が追加されました。**hcnmgr** ユーティリティーは、wicked ハイブリッドネットワーク仮想化 (HNV) をサポートし、ボンディングに **wicked** 関数を使用します。
  - **hcnmgr** は、後のクリーンアップのために **hcnid** 状態を維持します。
  - **hcnmgr** は、NetworkManager (NM) **nmcli** コードを除外します。
  - NM HNV **primary slave** 設定が修正されました。
  - **hcnmgr** は、仮想ネットワークインターフェイスコントローラー (vNIC) をバックアップデバイスとしてサポートします。
- **bootlist** の無効な 16 進数のシステムメッセージを修正しました。
- **-l** フラグが、**bootlist** コマンドの **-p** デリミタ値として **kpartx** ユーティリティーに含まれました。
- IO スロットを一覧表示する際のメモリーリークを防ぐために、**sslot** ユーティリティーが修正されました。
- **lsslot** ユーティリティーに、最新の PCIe (Peripheral Component Interconnect Express) スロットタイプの DRC タイプ記述文字列が追加されました。
- **errinjct** ツールの RTAS への無効な設定アドレスが修正されました。
- **ofpathname** ユーティリティーで、non-volatile memory over fabrics (NVMf) デバイスのサポートが追加されました。このユーティリティーは、論理デバイス名からオープンファームウェアデバイスパス、およびその逆に変換するメカニズムを提供します。
- **ofpathname** ユーティリティーの非対称名前空間アクセス (ANA) モードでの不揮発性メモリー (NVMe) に対するサポートが修正されました。
- 設定ファイルとして **smt.state** ファイルをインストールしました。

(BZ#1920964)

Redfish モジュールが redhat.rhel_mgmt Ansible コレクションに含まれるようになりました。

redhat.rhel_mgmt Ansible コレクションには、次のモジュールが含まれるようになりました。

- **redfish_info**
- **redfish_command**
- **redfish_config**

これによりユーザーは、Redfish モジュールを使用してサーバーのヘルスステータスの取得、ハードウェアとファームウェアのインベントリーに関する情報の取得、電源管理、BIOS 設定の変更、帯域外 (OOB) コントローラーの設定、ハードウェア RAID の設定、ファームウェアの更新を実行し、自動化管理の恩恵を受けることができます。

(BZ#2112434)

libvpd がバージョン 2.2.9 にリベースされました

Vital Product Data (VPD) にアクセスするためのクラスを含む **libvpd** パッケージがバージョン 2.2.9 に更新されました。以下は、主な改善点です。

- データベースのロックを修正
- **libtool** ユーティリティーのバージョン情報を更新

(BZ#2051288)

lsvpd がバージョン 1.7.14 にリベースされました

ハードウェアインベントリーシステムを設定するためのコマンドを提供する **lsvpd** パッケージがバージョン 1.7.14 に更新されました。今回の更新により、**lsvpd** ユーティリティーは、**vpdupdate** コマンドの実行時にデータベースファイルが破損するのを防ぎます。

(BZ#2051289)

ppc64-diag がバージョン 2.7.8 にリベースされました

プラットフォーム診断用の **ppc64-diag** パッケージがバージョン 2.7.8 に更新されました。以下は、主な改善点です。

- **libvpd** ユーティリティーバージョン 2.2.9 以降を使用するようにビルド依存関係を更新
- サポート対象外のプラットフォームでの **extract_opal_dump** エラーメッセージを修正
- **GCC-8.5** および **GCC-11** コンパイラーでのビルド警告を修正

(BZ#2051286)

sysctl に、systemd-sysctl と同じ引数構文が導入されました

ランタイムでカーネルパラメーターを変更するために使用できる **procps-ng** パッケージの **sysctl** ユーティリティーは、**systemd-sysctl** ユーティリティーと同じ引数構文を使用するようになりました。今回の更新により、**sysctl** は設定行にハイフン (**-**) またはグロブ (*****) を含む設定ファイルを解析するようになりました。**systemd-sysctl** 構文の詳細については、**sysctl.d (5)** man ページを参照してください。

(BZ#2052536)

更新された systemd-udevd が、InfiniBand インターフェイスに一貫性のあるネットワークデバイス名を割り当てる

RHEL 9 で導入された **systemd** パッケージの新しいバージョンには、更新された **systemd-udevd** デバイスマネージャーが含まれています。デバイスマネージャーは、InfiniBand インターフェイスのデフォルト名を、**systemd-udevd** が選択した一貫性のある名前に変更します。

Renaming IPoIB devices の手順に従って、InfiniBand インターフェイスの名前にカスタム命名ルールを定義できます。

命名スキームの詳細は、**systemd.net-naming-scheme(7)** の man ページを参照してください。

(BZ#2136937)

4.6. インフラストラクチャーサービス

chrony は DHCPv6 NTP サーバーを使用するようになりました

chrony の NetworkManager ディスパッチャースクリプトは、動的ホスト設定プロトコル (DHCP) オプ

ションから渡されたネットワークタイムプロトコル (NTP) ソースを更新します。RHEL 9.1 以降、スクリプトは DHCPv4 に加え、DHCPv6 によって提供される NTP サーバーを使用します。DHCP オプション 56 は DHCPv6 の使用を指定し、DHCP オプション 42 は DHCPv4 固有です。

(BZ#2047415)

chrony がバージョン 4.2 にリベースされました

chrony スイートがバージョン 4.2 に更新されました。バージョン 4.1 からの注目すべき機能強化は次のとおりです。

- サーバーインターリーブモードが改善され、信頼性が向上し、単一のアドレストランスレーター (Network Address Translation - NAT) の背後で複数のクライアントをサポートするようになりました。
- Network Time Protocol Version 4 (NTPv4) 拡張フィールドの実験的サポートが追加され、時刻同期の安定性と推定誤差の精度が向上しました。**extfield F323** オプションを使用して、プロトコル NTPv4 の機能を拡張するこのフィールドを有効にできます。
- Precision Time Protocol (PTP) を介した NTP 転送の実験的サポートが追加され、タイムスタンプが PTP パケットに制限されているネットワークインターフェイスカード (NIC) で完全なハードウェアタイムスタンプが有効になりました。**ptpport 319** ディレクティブを使用して、NTP over PTP を有効にできます。

(BZ#2051441)

unbound がバージョン 1.16.2 にリベースされました

unbound コンポーネントがバージョン 1.16.2 に更新されました。**unbound** は、検証、再帰、およびキャッシング DNS リゾルバーです。以下は、主な改善点です。

- **RFC 8976** をサポートする ZONEMD ゾーン検証により、受信者はデータのインテグリティーと発信元の信頼性についてゾーンの内容を検証できるようになりました。
- **unbound** を使用すると、永続的な TCP 接続を設定できるようになりました。
- SVCB タイプおよび HTTPS タイプと、DNS **draft-ietf-dnsop-svcb-https** ドキュメントを介したサービスバインディングとパラメーター仕様に基づく処理が追加されました。
- **unbound** は、暗号化ポリシーからデフォルトの TLS 暗号を取得します。
- **RFC8375** に従い、専用ドメイン **home.arpa** を使用できます。このドメインは、住宅ホームネットワークでの非専用の使用に指定されています。
- **unbound** は、スタブゾーンまたはフォワードゾーンの **tcp-upstream** クエリーの選択的有効化をサポートするようになりました。
- **aggressive-nsec** オプションのデフォルトが **yes** になりました。
- **ratelimit** ロジックが更新されました。
- Unbound 応答ポリシーゾーン (RPZ) nxdomain 応答によってクエリーがブロックされた場合に、新しい **rpz-signal-nxdomain-ra** オプションを使用して **RA** フラグを解除できます。
- **RFC8914** に準拠した拡張 DNS エラー (EDE) の基本サポートにより、追加のエラー情報を利用できます。

(BZ#2087120)

whois でパスワード暗号化機能が使えるようになりました

whois パッケージは **/usr/bin/mkpasswd** バイナリーを提供するようになりました。これを使用して、**crypt** C ライブラリーインターフェイスでパスワードを暗号化できます。

(BZ#2054043)

frr はバージョン 8.2.2 にリベースされました

動的ルーティングスタックを管理するための **frr** パッケージがバージョン 8.2.2 に更新されました。バージョン 8.0 からの主な変更点と強化点は次のとおりです。

- イーサネット VPN (EVPN) ルートタイプ 5 ゲートウェイ IP オーバーレイインデックスが追加されました。
- Open-shortest-path-first (OSPFv3) プロトコルに Autonomous System Border Router (ASBR) の要約が追加されました。
- OSPFv3 でのスタブおよび Not-So-Stubby-Area (NSSA) の使用が改善されました。
- OSPFv2 および OSPFv3 にグレースフル再起動機能が追加されました。
- Border Gateway Protocol (BGP) のリンク帯域幅は、IEEE 754 規格に従ってエンコードされるようになりました。以前のエンコーディング方式を使用するには、既存の設定で **neighbor PEER disable-link-bw-encoding-ieee** コマンドを実行します。
- BGP に長寿命のグレースフル再起動機能が追加されました。
- 拡張管理シャットダウン通信 **rfc9003** と、BGP の拡張オプションパラメーター長 **rfc9072** を実装しました。

(BZ#2069563)

Tuned リアルタイムプロファイルは、初期 CPU 分離設定を自動決定するようになりました

Tuned は、システムを監視し、パフォーマンスプロファイルを最適化するためのサービスです。また、**tuned-profiles-realtime** パッケージを使用して中央処理装置 (CPU) を分離し、アプリケーションスレッドの実行時間を可能な限り長くすることもできます。

以前は、**isolated_cores** パラメーターで分離する CPU のリストを指定しないと、リアルタイムカーネルを実行しているシステムのリアルタイムプロファイルが読み込まれませんでした。

この機能強化により、TuneD は、ハウスキーピングおよび分離されたコアのリストを自動的に計算し、計算を **isolated_cores** パラメーターに適用する **calc_isolated_cores** 組み込み関数を導入します。自動プリセットでは、各ソケットの 1 つのコアがハウスキーピング用に予約されており、追加の手順なしでリアルタイムプロファイルの使用を開始できます。プリセットを変更する場合は、分離する CPU のリストを指定して、**isolated_cores** パラメーターをカスタマイズします。

(BZ#2093847)

4.7. セキュリティー

新しいパッケージ: keylime

RHEL 9.1 では、トラステッドプラットフォームモジュール (TPM) テクノロジーを使用するリモートシステムの認証用ツールである Keylime が導入されています。Keylime を使用すると、リモートシステムのインテグリティーを検証し、継続的に監視できます。また、Keylime が監視対象のマシンに配信する

暗号化されたペイロードを指定し、システムがインテグリティーテストに失敗するたびにトリガーされる自動アクションを定義することもできます。

詳細については、RHEL 9 のセキュリティー強化ドキュメントの Keylime によるシステム整合性 の確保を参照してください。

(JIRA:RHELPLAN-92522)

OpenSSH の新しいオプションは、最小 RSA キー長の設定をサポートします

誤って短い RSA キーを使用すると、システムが攻撃に対してより脆弱になります。今回の更新により、OpenSSH サーバーおよびクライアントの最小 RSA キー長を設定できるようになりました。最小 RSA キー長を定義する場合、OpenSSH サーバーでは **/etc/ssh/sshd_config** ファイルで、OpenSSH クライアントでは **/etc/ssh/ssh_config** ファイルで新しい **RSAMinSize** オプションを使用します。

(BZ#2066882)

crypto-policies は、デフォルトで OpenSSH の最小 2048 ビット RSA キー長を強制します

短い RSA キーを使用すると、システムが攻撃に対してより脆弱になります。OpenSSH は最小 RSA キー長の制限をサポートするようになったため、システム全体の暗号化ポリシーは、デフォルトで RSA の最小キー長を 2048 ビットにします。

Invalid key length エラーメッセージで OpenSSH 接続が失敗する場合は、より長い RSA キーの使用を開始してください。

または、セキュリティーを犠牲にしてカスタムサブポリシーを使用し、制限を緩和することもできます。たとえば、**update-crypto-policies --show** コマンドが **DEFAULT** を現在のポリシーとして報告した場合:

1. **/etc/crypto-policies/policies/modules/RSA-OPENSSH-1024.pmod** ファイルに **min_rsa_size@openssh = 1024** パラメーターを挿入して、カスタムサブポリシーを定義します。

2. **update-crypto-policies --set DEFAULT:RSA-OPENSSH-1024** コマンドを使用して、カスタムサブポリシーを適用します。

(BZ#2102774)

OpenSSL の新しいオプションは、署名で SHA-1 をサポートします

RHEL 9 の OpenSSL 3.0.0 は、デフォルトで署名の作成と検証で SHA-1 をサポートしていません (SHA-1 鍵派生関数 (KDF) とハッシュベースのメッセージ認証コード (HMAC) はサポートされています)。ただし、引き続き署名に SHA-1 を使用する RHEL 8 システムとの後方互換性をサポートするために、新しい設定オプション **rh-allow-sha1-signatures** が RHEL 9 に導入されました。このオプションが **openssl.cnf** の **alg_section** で有効になっている場合、SHA-1 署名の作成と検証が許可されます。

LEGACY システム全体の暗号化ポリシー (レガシープロバイダーではない) が設定されている場合、このオプションは自動的に有効になります。

これは、SHA-1 署名を使用した RPM パッケージのインストールにも影響することに注意してください。LEGACY システム全体の暗号化ポリシーへの切り替えが必要になる場合があります。

(BZ#2060510、BZ#2055796)

crypto-policies が sntrup761x25519-sha512@openssh.com をサポートするようになりました

システム全体の暗号化ポリシーの更新により、**sntrup761x25519-sha512@openssh.com** キー交換

(KEX) メソッドのサポートが追加されました。ポスト量子 **sntrup761** アルゴリズムは OpenSSH スイートですでに利用可能であり、この方法は量子コンピューターからの攻撃に対してより優れたセキュリティーを提供します。**sntrup761x25519-sha512@openssh.com** を有効にするには、サブポリシーを作成して適用します。次に例を示します。

```
# echo 'key_exchange = +SNTRUP' > /etc/crypto-policies/policies/modules/SNTRUP.pmod
# update-crypto-policies --set DEFAULT:SNTRUP
```

詳細については、RHEL 9 セキュリティー強化ドキュメントの Customizing system-wide cryptographic policies with subpolicies セクションを参照してください。

(BZ#2070604)

NSS が 1023 ビット未満の RSA 鍵に対応しなくなる

Network Security Services (NSS) ライブラリーの更新により、すべての RSA 操作の最小鍵サイズが 128 から 1023 ビットに変更されます。つまり、NSS は以下の機能を実行しなくなります。

- RSA 鍵の生成は 1023 ビット未満です。
- 1023 ビット未満の RSA 鍵で RSA に署名するか、署名を検証します。
- 1023 ビットより短い RSA キーで値を暗号化または復号化します。

(BZ#2091905)

SELinux ポリシーは追加のサービスを制限します

selinux-policy パッケージが更新されたため、次のサービスが SELinux によって制限されるようになりました。

- **ksm**
- **nm-priv-helper**
- **rhcd**
- **stalld**
- **systemd-network-generator**
- **targetclid**
- **wg-quick**

(BZ#1965013, BZ#1964862, BZ#2020169、BZ#2021131、BZ#2042614、BZ#2053639、BZ#2111069)

SELinux は型遷移で self キーワードをサポートします

SELinux ツールは、ポリシーソースで **self** キーワードを使用した型遷移ルールをサポートするようになりました。**self** キーワードによる型遷移のサポートにより、SELinux ポリシーで匿名 i ノードのラベル付けが準備されます。

(BZ#2069718)

SELinux ユーザー空間パッケージが更新されました

SELinux ユーザー空間パッケージ
libsepol、**libselinux**、**libsemanage**、**policycoreutils**、**checkpolicy**、および **mcstrans** は、最新のアップストリームリリース 3.4 に更新されました。主な変更点は以下のとおりです。

- **setfiles**、**restorecon**、および **fixfiles** ツールの **-T** オプションによる並列ラベル再設定のサポートが追加されました。
  - このオプションでプロセススレッドの数を指定するか、**-T 0** を使用して使用可能なプロセッサーコアの最大数を使用できます。これにより、ラベルの再設定に必要な時間が大幅に短縮されます。
- モジュールの SHA-256 ハッシュを出力する新しい **--checksum** オプションが追加されました。
- **libsepol-utils** パッケージに新しいポリシーユーティリティーが追加されました。

(BZ#2079276)

SELinux の自動ラベル再設定がデフォルトで並列化されるようになりました

新しく導入された並列ラベル再設定オプションにより、マルチコアシステムでの SELinux ラベル再設定に必要な時間が大幅に短縮されるため、自動ラベル再設定スクリプトには、**fixfiles** コマンドラインに **-T 0** オプションが含まれるようになりました。**-T 0** オプションを指定すると、**setfiles** プログラムは、デフォルトでラベルの再設定に使用可能なプロセッサーコアの最大数を使用するようになります。

以前の RHEL バージョンと同じようにラベルの再設定に 1 つのプロセススレッドのみを使用するには、**fixfiles onboot** の代わりに **fixfiles -T 1 onboot** コマンドを入力するか、**touch /.autorelabel** の代わりに **echo "-T 1" >/.autorelabel** コマンドを入力して、この設定をオーバーライドします。

(BZ#2115242)

SCAP セキュリティーガイドが 0.1.63 にリベースされました

SCAP セキュリティーガイド (SSG) パッケージは、アップストリームバージョン 0.1.63 にリベースされました。このバージョンは、さまざまな拡張機能とバグ修正を提供します。特に、次のようなものがあります。

- **sysctl**、**grub2**、**pam_pwquality**、およびビルド時のカーネル設定の新しいコンプライアンスルールが追加されました。
- PAM スタックを強化するルールは、設定ツールとして **authselect** を使用するようになりました。注: この変更により、PAM スタックが他の方法で編集された場合、PAM スタックを強化するルールは適用されません。

(BZ#2070563)

Rsyslog エラーファイルの最大サイズオプションを追加しました

新しい **action.errorfile.maxsize** オプションを使用すると、Rsyslog ログ処理システムのエラーファイルの最大バイト数を指定できます。エラーファイルが指定されたサイズに達すると、Rsyslog は追加のエラーやその他のデータを書き込むことができなくなります。これにより、エラーファイルが原因でファイルシステムがいっぱいになり、ホストが使用できなくなるのを防ぐことができます。

(BZ#2064318)

clevis-luks-askpass がデフォルトで有効になりました

/lib/systemd/system-preset/90-default.preset ファイルには **enable clevis-luks-askpass.path** 設定オプションが含まれます。**clevis-systemd** サブパッケージをインストールすると、必ず **clevis-luks-**

askpass.path ユニットファイルが有効になります。これにより、Clevis 暗号化クライアントは、起動プロセスの後半でマウントされる LUKS 暗号化ボリュームもロック解除できます。この更新の前に、管理者は **systemctl enable clevis-luks-askpass.path** コマンドを使用して、Clevis がそのようなボリュームのロックを解除できるようにする必要があります。

(BZ#2107078)

fapolicyd が 1.1.3 にリベースされました。

fapolicyd パッケージがバージョン 1.1.3 にアップグレードされました。主な改善点とバグ修正は次のとおりです。

- ルールに、サブジェクトの親 PID (プロセス ID) と一致する新しいサブジェクト PPID 属性を含めることができるようになりました。
- OpenSSL ライブラリーは、ハッシュ計算用の暗号化エンジンとして Libgcrypt ライブラリーに取って代わりました。
- **fagenrules --load** コマンドが正しく機能するようになりました。

(BZ#2100041)

4.8. ネットワーク

act_ctinfo カーネルモジュールが追加されました

この機能強化により、**act_ctinfo** カーネルモジュールが RHEL に追加されました。管理者は、**tc** ユーティリティーの **ctinfo** アクションを使用して、ネットワークパケットの **conntrack** マークまたは Diffserv コードポイント (DSCP) の値をソケットバッファーの **mark** メタデータフィールドにコピーできます。その結果、**conntrack** マークまたは DSCP 値に基づく条件を使用して、トラフィックをフィルタリングできます。詳細については、**tc-ctinfo(8)** の man ページを参照してください。

(BZ#2027894)

Microsoft Azure で毎起動時に cloud-init がネットワーク設定を更新するようになりました

仮想マシンがオフラインのときに管理者がネットワークインターフェイスの設定を更新しても、Microsoft Azure はインスタンス ID を変更しません。今回の機能強化により、仮想マシンの起動時に **cloud-init** サービスが常にネットワーク設定を更新し、Microsoft Azure 上の RHEL が最新のネットワーク設定を使用するようになります。

結果として、追加の検索ドメインなどのインターフェイスの設定を手動で設定すると、仮想マシンの再起動時に **cloud-init** が それらをオーバーライドする可能性があります。詳細と回避策は、cloud-init-22.1-5 が起動ごとにネットワーク設定を更新する の解決策を参照してください。

(BZ#2144898)

PTP ドライバーが仮想クロックとタイムスタンプをサポートするようになりました

この機能拡張により、Precision Time Protocol (PTP) ドライバーは、**/sys/class/ptp/ptp*/n_vclocks** に書き込むことで、フリーランニング PHC の上に仮想 PTP ハードウェアクロック (PHC) を作成できます。その結果、ユーザーは 1 つのインターフェイスでハードウェアタイムスタンプを使用して複数のドメイン同期を実行できます。

(BZ#2066451)

firewalld がバージョン 1.1.1 にリベースされました

firewalld パッケージがバージョン 1.1.1 にアップグレードされました。このバージョンでは、以前のバージョンから複数のバグ修正と機能拡張が行われました。

新機能:

- リッチルールは、ユーザー空間のロギング用に NetFilter-log (NFLOG) ターゲットをサポートします。RHEL には NFLOG 対応のロギングデーモンがないことに注意してください。ただし、**tcpdump -i nflog** コマンドを使用して、必要なログを収集できます。
- **ingress-zones=HOST** および **egress-zones={ANY, source based zone}** を使用したポリシーでのポート転送がサポートされます。

その他の主な変更点は次の通りです。

- **afp**、**http3**、**jellyfin**、**netbios-ns**、**ws-discovery**、**ws-discovery-client** サービスがサポートされます。
- **policy** オプションの Z Shell でのタブ補完とサブオプション。

(BZ#2040689)

NetworkManager が advmss、rto_min、および quickack ルート属性をサポートするようになりました

この機能拡張により、管理者は次の属性を使用して **ipv4.routes** 設定を設定できます。

- **rto_min** (TIME) - ルート宛先と通信する際の最小 TCP 再送信タイムアウトをミリ秒単位の設定。
- **quickack** (BOOL) - TCP クイック ACK を有効または無効にするルートごとの設定。
- **advmss** (NUMBER) - TCP 接続の確立時にルート宛先に最大セグメントサイズ (MSS) をアドバタイズ。指定しない場合、Linux は最初のホップデバイスの最大転送単位 (MTU) から計算されたデフォルト値を使用します。

上記の属性を使用して **ipv4.routes** の新しい機能を実装する利点は、**dispatcher** スクリプトを実行する必要がないことです。

上記のルート属性で接続を有効にすると、そのような変更がカーネルに設定されることに注意してください。

(BZ#2068525)

nmstate で 802.ad vlan-protocol がサポートされます

nmstate API は、802.ad **vlan-protocol** オプションを使用した **linux-bridge** インターフェイスの作成をサポートするようになりました。この機能により、サービスタグ VLAN の設定が可能になります。次の例は、**yaml** 設定ファイルでのこの機能の使用法を示しています。

```
---
interfaces:
  - name: br0
    type: linux-bridge
    state: up
    bridge:
      options:
        vlan-protocol: 802.1ad
```

```
port:
  - name: eth1
    vlan:
      mode: trunk
      trunk-tags:
      - id: 500
```

(BZ#2084474)

firewalld サービスは、ローカルホストから発信された NAT パケットを別のホストおよびポートに転送できます。

firewalld サービスを実行する localhost から送信されたパケットを別の宛先ポートと IP アドレスに転送できます。この機能は、たとえば、**loopback** デバイスのポートをコンテナーまたは仮想マシンに転送する場合に役立ちます。この変更の前は、**firewalld** は別のホストから発信されたパケットを受信した場合にのみポートを転送できました。詳細と設定例については、DNAT を使用して HTTPS トラフィックを別のホストに転送する を参照してください。

(BZ#2039542)

NetworkManager が ifcfg-rh からキーファイルへの移行をサポートするようになりました

ユーザーは、既存の接続プロファイルファイルを **ifcfg-rh** 形式からキーファイル形式に移行できます。このようにして、すべての接続プロファイルが 1 つの場所に優先形式で配置されます。鍵ファイル形式には、次の利点があります。

- NetworkManager がネットワーク設定を表現する方法によく似ています
- 将来の RHEL リリースとの互換性を保証
- より読みやすく
- すべての接続プロファイルをサポート

接続を移行するには、次を実行します。

```
# nmcli connection migrate
```

ifcfg-rh ファイルは、RHEL 9 の存続期間中は正しく機能することに注意してください。ただし、設定をキーファイル形式に移行することで、RHEL 9 以降の互換性が保証されます。

詳細は、**nmcli(1)**、nm-settings-keyfile(5)、および nm-settings-ifcfg-rh(5) の man ページを参照してください。

(BZ#2059608)

DHCP および IPv6 自動設定属性が nmstate API に追加されました。

この機能拡張により、次の属性のサポートが nmstate API に追加されます。

- RFC 2132 および 4361 で説明されている DHCPv4 接続の **dhcp-client-id**。
- RFC 8415 で説明されている DHCPv6 接続の **dhcp-duid**。
- IPv6 自動設定の **addr-gen-mode**。この属性を次のように設定できます。
  - **eui64** (RFC 4862 で説明されているとおり)

- **stable-privacy** (RFC 7217 で説明されているとおり)

(BZ#2082043)

NetworkManager は、RHEL 9 で WEP がサポートされないことを明示するようになりました。

RHEL 9.0 以降の **wpa_supplicant** パッケージには、非推奨のセキュアではない Wired Equivalent Privacy (WEP) セキュリティーアルゴリズムが含まれなくなりました。この機能拡張により、NetworkManager が更新され、変更が反映されます。たとえば、**nmcli device wifi list** コマンドは、リストの最後にある WEP アクセスポイントを灰色で返し、WEP で保護されたネットワークに接続すると意味のあるエラーメッセージが返されるようになりました。

安全な暗号化のために、Wi-Fi Protected Access 2 (WPA2) および WPA3 認証を備えた Wi-Fi ネットワークのみを使用してください。

(BZ#2030997)

MPTCP コードが更新されました

カーネルの MultiPath TCP (MPTCP) コードが更新され、アップストリームの Linux 5.19.この更新では、以前のバージョンに対して多数のバグ修正と拡張が行われています。

- **FASTCLOSE** オプションが追加され、完全なスリーウェイハンドシェイクなしで MPTCP 接続を閉じることができます。
- 最初のハンドシェイク後でも TCP へのフォールバックを有効にする **MP_FAIL** オプションが追加されました。
- 管理情報ベース (MIB) カウンターを追加することで、監視機能が改善されました。
- MPTCP リスナーソケットの監視サポートが追加されました。**ss** ユーティリティーを使用してソケットを監視します。

(BZ#2079368)

4.9. カーネル

RHEL 9.1 のカーネルバージョン

Red Hat Enterprise Linux 9.1 は、カーネルバージョン 5.14.0-162 で配布されます。

(BZ#2125549)

list_lru のメモリー消費が最適化されました

内部カーネルデータ構造 **list_lru** は、カーネル i ノードとファイルのディレクトリーエントリーの「最近使用されていない」ステータスを追跡します。以前は、**list_lru** に割り当てられた構造体の数は、マウントポイント数および現在のメモリー **cgroups** 数に正比例していました。これらの数値は両方とも、実行中のコンテナーの数とともに増加し、メモリー消費は **O(n^2)** になります。この場合の **n** は実行中のコンテナーの数です。この更新により、システム内の **list_lru** のメモリー消費が **O(n)** に最適化されます。その結果、特に実行中のコンテナーが多数あるシステムで、ユーザーアプリケーションに十分なメモリーを使用できるようになりました。

(BZ#2013413)

BPF が Linux カーネルバージョン 5.16 にリベースされました。

Berkeley Packet Filter (BPF) 機能が Linux カーネルバージョン 5.16 にリベースされ、複数のバグ修正と機能拡張が行われました。以下は、主な変更点です。

- 内部 BPF プログラムセクションの処理と**libbpf** ユーザー空間ライブラリーの **bpf_program__set_attach_target()** API を合理化しました。**bpf_program__set_attach_target()** API は、BPF ベースのプログラムで BTF ベースの接続ターゲットを設定します。
- 宣言にタグ付けできる **BTF_KIND_TAG** kind のサポートが追加されました。
- **bpf_get_branch_snapshot()** ヘルパーのサポートが追加されました。これにより、トレースプログラムはハードウェアから最後の分岐レコード (LBR) をキャプチャできるようになります。
- **libbpf** ユーザー空間ライブラリーにレガシー **kprobe** イベントサポートが追加されました。これにより、レガシーインターフェイスを介して **kprobe** トレースポイントイベントを作成できます。
- **__sk_buff** ヘルパー関数を使用し、BPF 固有の構造を介してハードウェアタイムスタンプにアクセスする機能が追加されました。
- **i40e** および **ice** ドライバーのサポートにより、**AF_XDP** バッファープールにおける RX バッファー割り当て用バッチインターフェイスのサポートが追加されました。
- 最近マージされたレガシー **kprobe** を補完するために、**libbpf** ユーザー空間ライブラリーにレガシー **uprobe** のサポートが追加されました。
- **bpf_trace_vprintk()** が可変 **printk** ヘルパーとして追加されました。
- **libbpf** 1.0 の取り組みの一環として、より厳密な BPF プログラムセクション名の処理のために **libbpf** オプトインが追加されました。
- **perf RB** などの特殊なマップを検索し、作成中に BTF タイプ識別子を内部的に削除するための、**libbpf** サポートが追加されました。
- セット内に要素が存在するかどうかをテストする **Bloomfilter** BPF マップタイプが追加されました。
- BPF からのカーネルモジュール関数呼び出しに対するサポートが追加されました。
- ライトスケルトンでタイプレスの弱い **ksym** に対するサポートが追加されました。
- **BTF_KIND_DECL_TAG** kind に対するサポートが追加されました。

実行中のカーネルで使用可能な BPF 機能の完全なリストの詳細については、**bpftool feature** コマンドを使用してください。

(BZ#2069045)

BTF データがカーネルモジュールに配置されるようになりました

BPF Type Format (BTF) は、BPF プログラムおよびマップに関連するデバッグ情報をエンコードするメタデータ形式です。以前は、カーネルモジュールの BTF データは **kernel-debuginfo** パッケージに格納されていました。そのため、カーネルモジュールに BTF を使用するには、対応する **kernel-debuginfo** パッケージをインストールする必要がありました。今回の更新により、BTF データがカーネルモジュールに直接配置されるようになりました。その結果、BTF を機能させるために追加のパッケージをインストールする必要はありません。

(BZ#2097188)

kernel-rt ソースツリーが RHEL 9.1 ツリーに更新されました。

kernel-rt ソースが更新され、最新の Red Hat Enterprise Linux カーネルソースツリーを使用するようになりました。リアルタイムパッチセットも、最新のアップストリームバージョン **v5.15-rt** に更新されました。これらの更新は、バグ修正および機能強化を多数提供します。

(BZ#2061574)

ARM、AMD、および Intel 64 ビットアーキテクチャーで動的プリエンプティブスケジューリングが有効になりました

RHEL 9 は、ARM、AMD および Intel 64 ビットアーキテクチャーで動的スケジューリング機能を提供します。この機能拡張により、コンパイル時ではなく、起動時または実行時にカーネルのプリエンプションモードを変更できるようになりました。**/sys/kernel/debug/sched/preempt** ファイルには現在の設定が含まれており、**runtime** の変更が可能です。

DYNAMIC_PREEMPT オプションを使用すると、起動時に **preempt=** 変数を **none**、**voluntary** または **full** に設定できます。デフォルトは、**voluntary** プリエンプションです。動的プリエンプティブ処理を使用すると、デフォルトのプリエンプションモデルをオーバーライドして、スケジューリングレイテンシーを改善できます。

(BZ#2065226)

stalld がバージョン 1.17 にリベースされました

stall デーモンを提供する **stalld** プログラムは、Linux システムでオペレーティングシステムスレッドの枯渇状態を防ぐためのメカニズムです。このバージョンは、スレッドの枯渇状態を監視します。枯渇は、スレッドが枯渇しきい値より長く CPU 実行キューにある場合に発生します。

この **stalld** バージョンには、以前のバージョンからの改善とバグ修正が多数含まれています。注目すべき変更には、実行可能な停止寸前のタスクを検出する機能が含まれます。

stalld が枯渇スレッドを検出すると、プログラムはスレッドのスケジューリングクラスを **SCHED_DEADLINE** ポリシーに変更します。これにより、指定された CPU がスレッドを実行するためのわずかな時間がスレッドに与えられます。**timeslice** が使用されると、スレッドは元のスケジューリングポリシーに戻り、**stalld** は引き続きスレッドの状態を監視します。

(BZ#2107275)

tpm2-tools パッケージが tpm2-tools-5.2-1 バージョンにリベースされました

tpm2-tools パッケージがバージョン **tpm2-tools-5.2-1** にリベースされました。このアップグレードでは、多くの重要な昨日書く古布とバグ修正が行われました。以下は、主な変更点です。

- **tpm2_createprimary** および **tpm2_create** ツールを使用したプライマリーオブジェクト作成時の公開鍵出力に対するサポートが追加されました。
- 公開鍵の出力形式を出力する **tpm2_print** ツールのサポートが追加されました。**tpm2_print** は、トラステッドプラットフォームモジュール (TPM) データ構造をデコードし、囲まれた要素を出力します。
- 64 KB を超えるログを読み取る **tpm2_eventlog** ツールに対するサポートが追加されました。
- セッション属性の表示と設定をサポートする **tpm2_sessionconfig** ツールが追加されました。

重要な変更の詳細については、 **/usr/share/doc/tpm2-tools/Changelog.md** ファイルを参照してください。

(BZ#2090748)

Intel E800 デバイスが iWARP および RoCE プロトコルをサポートするように

この機能拡張により、**enable_iwarp** および **enable_roce** devlink パラメーターを使用して、iWARP または RoCE プロトコルのサポートをオンまたはオフにできるようになりました。この必須機能を使用すると、いずれかのプロトコルを使用してデバイスを設定できます。Intel E800 デバイスは、同じポートで両方のプロトコルを同時にサポートしません。

特定の E800 デバイスの iWARP プロトコルを有効または無効にするには、まずカードの PCI の場所を確認します。

```
$ lspci | awk '/E810/ {print $1}'
44:00.0
44:00.1
$
```

次に、プロトコルを有効または無効にします。devlink コマンドの引数として、カードの最初のポートに **pci/0000:44:00.0** を使用し、2 番目のポートに **pci/0000:44:00.1** を使用できます。

```
$ devlink dev param set pci/0000:44:00.0 name enable_iwarp value true cmode runtime
$ devlink dev param set pci/0000:44:00.0 name enable_iwarp value false cmode runtime
```

特定の E800 デバイスの RoCE プロトコルを有効または無効にするには、上記のようにカードの PCI の場所を確認します。次に、以下のコマンドの 1 つを使用します。

```
$ devlink dev param set pci/0000:44:00.0 name enable_roce value true cmode runtime
$ devlink dev param set pci/0000:44:00.0 name enable_roce value false cmode runtime
```

(BZ#2096127)

4.10. ブートローダー

GRUB は新しい鍵で署名されています

セキュリティー上の理由から、GRUB は新しい鍵で署名されるようになりました。結果として、RHEL ファームウェアをバージョン FW1010.30 (またはそれ以降) または FW1020 に更新して、セキュアブート機能を有効にした IBM Power Systems のリトルエンディアンバリアントを起動できるようにする必要があります。

(BZ#2074761)

IBM POWER で仮想マシンを起動すると、設定可能なディスクアクセスが再試行する

IBM POWER アーキテクチャーで論理パーティション (**lpar**) 仮想マシン (VM) を起動すると、GRUB ブートローダーがリモートディスクへのアクセスを再試行する回数を設定できるようになりました。再試行回数を下げると、特定の状況で起動が遅くなる可能性があります。

以前は、起動時にディスクアクセスが失敗したときに GRUB がディスクへのアクセスを 20 回試行していました。これにより、低速なストレージエリアネットワーク (SAN) ディスクに接続されている **lpar** システムで、Live Partition Mobility (LPM) 移行を実行すると、問題が発生しました。その結果、20 回の再試行が完了するまで、起動に非常に長い時間がかかっていた可能性があります。

今回の更新で、**ofdisk_retries** GRUB オプションを使用して、ディスクアクセスの再試行回数を減らすように設定できるようになりました。詳細は、IBM POWER で仮想マシンを起動する際のディスクアクセスの再試行設定 を参照してください。

その結果、**lpar** ブートは LPM の後、POWER で遅くならず、障害が発生したディスクなしで **lpar** システムが起動するようになりました。

(BZ#2070725)

4.11. ファイルシステムおよびストレージ

Stratis は、作成時にファイルシステムのサイズを設定できるようになりました

ファイルシステムの作成時に必要なサイズを設定できるようになりました。以前は、自動デフォルトサイズは 1 TiB でした。この機能強化により、ユーザーは任意のファイルシステムサイズを設定できます。下限は 512 MiB を下回ってはなりません。

(BZ#1990905)

Stratis プールのオーバープロビジョニング管理の改善

シンプロビジョニングの管理が改善されたことで、警告が改善され、プールメタデータ用のスペースが正確に割り当てられ、シンプール管理の予測可能性、全体的な安全性、および信頼性が向上しました。新しい個別モードは、オーバープロビジョニングを無効にします。この機能強化により、ユーザーはオーバープロビジョニングを無効にして、ファイルシステムが完全に満杯の場合でも、すべてのファイルシステムをサポートするのに十分なスペースがプールに含まれるようにすることができます。

(BZ#2040352)

Stratis は、改善された個々のプール管理を提供するようになりました

停止した個々の Stratis プールを停止および開始できるようになりました。以前は、**stratisd** は、検出したすべてのデバイスに対して使用可能なすべてのプールを開始しようとしました。この機能強化により、Stratis 内の個々のプールのより柔軟な管理、より優れたデバッグおよび回復機能が提供されます。単一プールのリカバリーおよびメンテナンス操作を実行するためにシステムを再起動する必要がなくなりました。

(BZ#2039960)

マルチパスデバイスパスのプロトコル固有の設定を有効化

以前は、プロトコルごとに最適な設定が異なるため、個々のプロトコルごとにオプションを設定しないと設定を正しく設定できませんでした。この機能強化により、ユーザーは、パストランスポートプロトコルに基づいてマルチパスデバイスパスを設定できるようになりました。**/etc/multipath.conf** ファイルの **overrides** セクションの **protocol** サブセクションを使用して、プロトコルに基づいてマルチパスデバイスパスを正しく設定します。

(BZ#2084365)

新しい libnvme 機能ライブラリー

以前は、NVMe ストレージコマンドラインインターフェイスユーティリティー (**nvme-cli**) にすべてのヘルパー関数と定義が含まれていました。この機能強化により、新しい **libnvme** ライブラリーが RHEL 9.1 にもたらされます。ライブラリーには以下が含まれます。

- NVMe 仕様構造の型定義
- 列挙とビットフィールド
- コマンドとペイロードを構築、ディスパッチ、およびデコードするヘルパー関数
- NVMe デバイスを接続、スキャン、および管理するためのユーティリティー

今回の更新により、ユーザーはコードや **nvme-stas** などの複数のプロジェクトやパッケージを複製する必要がなくなり、この共通ライブラリーを利用できるようになりました。

(BZ#2099619)

新しいライブラリー libnvme が利用可能になりました

今回の更新により、nvme-cli は 2 つの異なるプロジェクトに分割されました。* **nvme-cli** には、**nvme** ツールに固有のコードのみが含まれるようになりました * **libnvme** ライブラリーには、NVMe 仕様構造、列挙型、ビットフィールド、構築するヘルパー関数、ディスパッチ、コマンドとペイロードのデコード、および NVMe デバイスを接続、スキャン、および管理するためのユーティリティーのすべての型定義が含まれるようになりました。

(BZ#2090121)

4.12. 高可用性およびクラスター

Red Hat OpenStack Platform で高可用性がサポートされました

Red Hat OpenStack Platform で高可用性クラスターを設定できるようになりました。この機能をサポートするために、Red Hat は次の新しいクラスターエージェントを提供します。

- **fence_openstack**: OpenStack 上の HA クラスターのフェンスエージェント
- **openstack-info**: OpenStack 上の HA クラスターに必要な **openstack-info** クローンリソースを設定するためのリソースエージェント
- **openstack-virtual-ip**: 仮想 IP アドレスリソースを設定するリソースエージェント
- **openstack-floating-ip**: フローティング IP アドレスリソースを設定するリソースエージェント
- **openstack-cinder-volume**: ブロックストレージリソースを設定するリソースエージェント

(BZ#2121838)

pcs は、システムの再起動を必要としないマルチパス SCSI デバイス更新をサポートします

pcs stonith update-scsi-devices コマンドを使用して、マルチパス SCSI デバイスを更新できるようになりました。このコマンドは、同じノードで実行されている他のクラスターリソースを再起動することなく、SCSI デバイスを更新します。

(BZ#2024522)

クラスター UUID をサポートします

クラスターのセットアップ中に、**pcs** コマンドがすべてのクラスターの UUID を生成するようになりました。クラスター名は一意のクラスター識別子ではないため、複数のクラスターを管理する場合に、クラスター UUID を使用して同じ名前のクラスターを識別できます。

pcs cluster config [show] コマンドを使用して、現在のクラスター UUID を表示できます。**pcs cluster config uuid generate** コマンドを使用して、UUID を既存のクラスターに追加するか、すでに UUID が存在する場合は再生成できます。

(BZ#2054671)

設定済みリソースを再作成する pcs コマンドを表示する新しい pcs resource config コマンドオプションが追加されました

pcs resource config コマンドで、**--output-format=cmd** オプションが使用可能になりました。このオプションを指定すると、別のシステムで設定済みリソースを再作成するために使用できる **pcs** コマンドが表示されます。

(BZ#2058251)

設定済みフェンスデバイスを再作成する pcs コマンドを表示する新しい pcs stonith config コマンドオプションが追加されました

pcs stonith config コマンドで、**--output-format=cmd** オプションが使用可能になりました。このオプションを指定すると、別のシステムで設定済みのフェンスデバイスを再作成するために使用できる **pcs** コマンドが表示されます。

(BZ#2058252)

Pacemaker がバージョン 2.1.4 にリベースされました

Pacemaker パッケージは、Pacemaker 2.1.4 のアップストリームバージョンにアップグレードされました。主な変更点は、以下のとおりです。

- **multiple-active** リソースパラメーターは、**stop_unexpected** の値を受け入れるようになりました。**multiple-active** リソースパラメーターは、リソースが複数のノードでアクティブになっていてはならない時にアクティブになっている場合のリカバリー動作を決定します。デフォルトでは、リソースが本来あるべき場所で正常に実行されている場合でも、この状況ではリソースを完全に再起動する必要があります。このパラメーターの **stop_unexpected** の値は、マルチアクティブリソースの予期しないインスタンスのみが停止されるように指定します。ユーザーは、サービスとそのリソースエージェントが、完全に再起動しなくても追加のアクティブインスタンスで機能することを確認する必要があります。

- Pacemaker は、**allow-unhealthy-node** リソースメタ属性をサポートするようになりました。このメタ属性が **true** に設定されている場合、ノードの正常性が低下したことでリソースがノードから強制的に切り離されることはありません。正常性リソースにこの属性が設定されている場合、クラスターはノードの正常性が回復したかどうかを自動的に検出し、リソースをノードに戻すことができます。

- ユーザーは、**pcs acl group** コマンドを使用して、システムグループのアクセス制御リスト (ACLS) を指定できるようになりました。これまで Pacemaker では個々のユーザーに ACL を指定できましたが、ローカルポリシーを使用してシステムグループに ACL を指定し、そのグループ内のすべてのユーザーに適用する方が簡単で適している場合があります。以前のリリースにもこのコマンドはありましたが、効果はありませんでした。

(BZ#2072108)

クラスターパッケージで Samba が自動的にインストールされなくなりました

このリリース以降、RHEL High Availability Add-On のパッケージをインストールしても、Samba パッケージは自動的にインストールされなくなりました。そのため、Samba パッケージを削除しても HA パッケージが自動的に削除されません。クラスターで Samba リソースを使用している場合は、手動でインストールする必要があります。

(BZ#1826455)

4.13. 動的プログラミング言語、WEB サーバー、およびデータベースサーバー

nodejs:18 モジュールストリームに完全に対応しました。

以前はテクノロジープレビューとして利用できた **nodejs:18** モジュールストリームは、RHSA-2022:8832 アドバイザリーのリリースで完全に対応しています。**nodejs:18** モジュールストリームでは、LTS (Long Term Support) バージョンの **Node.js 18.12** が提供されるようになりました。

RHEL 9.1 に含まれる **Node.js 18** は、**Node.js 16** のバグ修正およびセキュリティー修正と共に多くの新機能を提供します。

主な変更点は、以下のとおりです。

- **V8** エンジンがバージョン 10.2 にアップグレードされました。
- **npm** パッケージマネージャーがバージョン 8.19.2 にアップグレードされました。
- **Node.js** が実験的な新しい **fetcj** API を提供するようになりました。
- **Node.js** は、新しい実験的な **node:test** モジュールを提供するようになりました。これにより、Test Anything Protocol (TAP) 形式で結果を報告するテストの作成が容易になります。
- **Node.js** は、IPv4 よりも IPv6 アドレスを優先するようになりました。

nodejs:18 モジュールストリームをインストールするには、以下を使用します。

```
# dnf module install nodejs:18
```

(BZ#2083072)

新しいモジュールストリーム: php:8.1

RHEL 9.1 は、**PHP 8.1** を新しい **php:8.1** モジュールストリームとして追加します。

PHP 8.1 では、以下が可能です。

- 列挙 (Enums) 機能を使用して、可能な値の離散数の 1 つに制限されるカスタム型を定義します。
- 初期化後のプロパティー変更を防ぐために、**readonly** 修飾子を使用してプロパティーを宣言します。
- ファイバー、フルスタック、中断可能な機能を使用します。

php:8.1 モジュールストリームをインストールするには、以下を実行します。

```
# dnf module install php:8.1
```

RHEL 9 での PHP の使用方法の詳細は、Using the PHP scripting language を参照してください。

(BZ#2070040)

新しいモジュールストリーム: ruby:3.1

RHEL 9.1 では、新しい **ruby:3.1.2** モジュールストリームに **Ruby 3.1** が導入されました。このバージョンでは、RHEL 9.0 で配布される **Ruby 3.0** に対するパフォーマンスの向上、バグおよびセキュリティー修正、および新機能が数多く追加されました。

主な機能拡張は、次のとおりです。

- **Interactive Ruby** (IRB) ユーティリティーは、オートコンプリート機能とドキュメントダイアログを提供するようになりました。
- **lib/debug.rb** を置き換える新しい **debug** gem は、パフォーマンスを改善し、リモートデバッグとマルチプロセス/マルチスレッドデバッグをサポートします。
- **error_highlight** gem は、バックトレースでエラーの詳細な場所を提供するようになりました。
- ハッシュリテラルデータ型とキーワード引数の値を省略できるようになりました。
- ピン Operator (**^**) がパターンマッチングで式を受け入れるようになりました。
- 1行のパターンマッチングで括弧を省略できるようになりました。
- 新しい実験的なインプロセス Just-in-Time (JIT) コンパイラーである YJIT が、AMD および Intel 64 ビットアーキテクチャーで利用可能になりました。
- **TypeProf For IDE** ユーティリティーが導入されました。これは、IDE 内の **Ruby** コード用の実験的な静的型分析ツールです。

Method Based Just-in-Time Compiler (MJIT) では、以下の通りパフォーマンスが改善されました。

- **Rails** のようなワークロードでは、デフォルトの最大 JIT キャッシュ値が 100 から 10000 に増加しました。
- クラスイベントの **TracePoint** が有効になっている場合、JIT を使用してコンパイルされたコードがキャンセルされなくなりました。

その他の主な変更点は次の通りです。

- **tracer.rb** ファイルが削除されました。
- バージョン 4.0 以降、**Psych** YAML パーサーはデフォルトで **safe_load** メソッドを使用します。

ruby:3.1 モジュールストリームをインストールするには、以下を使用します。

```
# dnf module install ruby:3.1
```

(BZ#2063773)

httpd がバージョン 2.4.53 にリベースされました

Apache HTTP Server がバージョン 2.4.53 に更新されました。これは、RHEL 9.0 で配布されたバージョン 2.4.51 に対するバグ修正、機能拡張、およびセキュリティー修正を提供します。

mod_proxy および **mod_proxy_connect** モジュールの主な変更点は次のとおりです。

- **mod_proxy**: コントローラー名の長さ制限が増加しました。
- **mod_proxy**: バックエンドとフロントエンドのタイムアウトを選択的に設定できるようになりました。
- **mod_proxy** : **SetEnv proxy-nohalfclose** パラメーターを設定して、TCP 接続のリダイレクトを無効にできるようになりました。

- **mod_proxy** および **mod_proxy_connect**: クライアントに送信した後にステータスコードを変更することは禁止されています。

さらに、LDAP インジェクションの脆弱性を防ぐのに役立つ新しい **ldap** 関数が式 API に追加されました。

(BZ#2079939)

httpd 設定の LimitRequestBody ディレクティブのデフォルトが新しくなりました

CVE-2022-29404 を修正するために、Apache HTTP Server の **LimitRequestBody** ディレクティブのデフォルト値が **0** (無制限) から 1 GiB に変更されました。

LimitRequestBody の値が **httpd** 設定ファイルで明示的に指定されていないシステムでは、**httpd** パッケージを更新すると、**LimitRequestBody** がデフォルト値の 1 GiB に設定されます。その結果、HTTP 要求本文の合計サイズがデフォルトの制限である 1 GiB を超える場合、**httpd** は **413 Request Entity Too Large** エラーコードを返します。

HTTP 要求メッセージ本文の新しいデフォルトの許容サイズがユースケースに不十分な場合は、それぞれのコンテキスト (サーバー、ディレクトリーごと、ファイルごと、または場所ごと) 内で **httpd** 設定ファイルを更新し、適切な制限をバイト単位で設定します。たとえば、新しい制限を 2 GiB に設定するには、次を使用します。

```
LimitRequestBody 2147483648
```

すでに **LimitRequestBody** ディレクティブで明示的な値を使用するように設定されているシステムは、この変更の影響を受けません。

(BZ#2128016)

新規パッケージ: httpd-core

RHEL 9.1 以降では、基本的な **httpd** 機能のみを必要とするシナリオ (コンテナーなど) で Apache HTTP Server の依存関係を制限するために、すべての必須ファイルを含む **httpd** バイナリーファイルが新しい **httpd-core** パッケージに移動されました。

httpd パッケージは、**mod_systemd**、**mod_brotli**、ドキュメントなど、**systemd** 関連のファイルを提供するようになりました。

この変更により、**httpd** パッケージは **httpd** Module Magic Number (MMN) 値を提供しなくなりました。代わりに、**httpd-core** パッケージが **httpd-mmn** 値を提供するようになりました。その結果、**httpd** パッケージから **httpd-mmn** を取得することはできなくなりました。

インストールされた **httpd** バイナリーの **httpd-mmn** 値を取得するには、**httpd-devel** パッケージの一部である **apxs** バイナリーを使用できます。**httpd-mmn** 値を取得するには、次のコマンドを使用します。

```
# apxs -q HTTPD_MMN
20120211
```

(BZ#2065677)

pcre2 がバージョン 10.40 にリベースされました

Perl 互換正規表現ライブラリー v2 を提供する **pcre2** パッケージがバージョン 10.40 に更新されました。

今回の更新では、**Perl 5.32** のそれぞれの変更に従って、lookaround アサーションでの **\K** エスケープシーケンスの使用が禁止されました。以前の動作に依存している場合は、**PCRE2_EXTRA_ALLOW_LOOKAROUND_BSK** オプションを使用できます。このオプションが設定されている場合、**\K** はポジティブアサーション内でのみ受け入れられ、ネガティブアサーションでは無視されることに注意してください。

(BZ#2086494)

4.14. コンパイラーおよび開発ツール

更新された GCC コンパイラーが RHEL 9.1 で利用可能になりました

システム GCC コンパイラーバージョン 11.2.1 が更新され、アップストリームの GCC で利用可能なバグ修正および機能拡張が数多く追加されました。

GNU コンパイラーコレクション (GCC) には、C、C++、および Fortran のプログラミング言語でアプリケーションを開発するためのツールが含まれます。

使用方法は、RHEL 9 での C および C++ アプリケーションの開発 を参照してください。

(BZ#2063255)

新しい GCC Toolset 12

GCC Toolset 12 は最新バージョンの開発ツールを提供するコンパイラーツールセットです。このツールセットは、**AppStream** リポジトリーにおいて、Software Collection の形式で、Application Streams として利用できます。

GCC コンパイラーがバージョン 12.1.1 に更新され、アップストリームの GCC で利用可能なバグ修正および機能拡張が数多く追加されました。

以下のツールおよびバージョンは、GCC Toolset 12 で利用できます。

ツール	バージョン
GCC	12.1.1
GDB	11.2
binutils	2.35
dwz	0.14
annobin	10.76

GCC Toolset 12 をインストールするには、root で以下のコマンドを実行します。

```
# dnf install gcc-toolset-12
```

GCC Toolset 12 のツールを実行するには、以下のコマンドを実行します。

```
$ scl enable gcc-toolset-12 tool
```

GCC Toolset バージョン 12 のツールバージョンが、このようなツールのシステムバージョンをオーバーライドするシェルセッションを実行するには、次のコマンドを実行します。

```
$ scl enable gcc-toolset-12 bash
```

詳細については、GCC ツールセット 12 を参照してください。

(BZ#2077465)

GCC Toolset 12: Annobin がバージョン 10.76 にリベースされました。

GCC Toolset 12 では、Annobin パッケージがバージョン 10.76 に更新されました。

主なバグ修正と機能拡張は、以下のとおりです。

- annocheck の新しいコマンドラインオプションは、別の方法でデバッグ情報を見つけることができない場合、**debuginfod** サービスの使用を避けるように指示します。**debuginfod** を使用すると annocheck で多くの情報が提供されますが、**debuginfod** サーバーが使用できない場合、annocheck のパフォーマンスが大幅に低下する可能性もあります。
- Annobin ソースは、必要に応じて設定および作成するのではなく、**meson** および **ninja** を使用してビルドできるようになりました。
- Annocheck は、Rust 1.18 コンパイラーによってビルドされたバイナリーをサポートするようになりました。

さらに、GCC Toolset 12 バージョンの Annobin で次の既知の問題が報告されています。

状況によっては、次のようなエラーメッセージでコンパイルが失敗する可能性があります。

```
cc1: fatal error: inaccessible plugin file
opt/rh/gcc-toolset-12/root/usr/lib/gcc/architecture-linux-gnu/12/plugin/gcc-annobin.so
expanded from short plugin name gcc-annobin: No such file or directory
```

この問題を回避するには、プラグインディレクトリーに **annobin.so** から **gcc-annobin.so** へのシンボリックリンクを作成します。

```
# cd /opt/rh/gcc-toolset-12/root/usr/lib/gcc/architecture-linux-gnu/12/plugin
# ln -s annobin.so gcc-annobin.so
```

architecture は、使用しているアーキテクチャーに置き換えます。

- **aarch64**
- **i686**
- **ppc64le**
- **s390x**
- **x86_64**

(BZ#2077438)

GCC Toolset 12: binutils がバージョン 2.38 にリベースされました

GCC Toolset 12 では、**elfutils** パッケージがバージョン 2.38 に更新されました。

主なバグ修正と機能拡張は、以下のとおりです。

- **binutils** パッケージのすべてのツールで、マルチバイト文字の存在を表示または警告するオプションがサポートされるようになりました。
- **readelf** および **objdump** ツールは、デフォルトで **debuginfo** ファイルへのリンクを自動的にたどるようになりました。この動作は、**readelf** の **--debug-dump=no-follow-links** オプションまたは **objdump** の **--dwarf=no-follow-links** オプションを使用して無効にすることができます。

(BZ#2077445)

GCC 12 以降は _FORTIFY_SOURCE レベル 3 をサポートします

この機能拡張により、ユーザーは、GCC バージョン 12 以降でビルドする場合に、コンパイラーコマンドラインで **-D_FORTIFY_SOURCE=3** を使用してアプリケーションをビルドできます。**_FORTIFY_SOURCE** レベル 3 では、ソースコード強化の範囲が改善されるため、コンパイラーコマンドラインで **-D_FORTIFY_SOURCE=3** を指定してビルドされたアプリケーションのセキュリティーが向上します。これは、GCC バージョン 12 以降、および **__builtin_dynamic_object_size** ビルトインを使用する RHEL 9 のすべての Clang でサポートされています。

(BZ#2033683)

DNS スタブリゾルバーオプションが no-aaaa オプションをサポートするようになりました

今回の機能拡張により、**glibc** は **/etc/resolv.conf** の **no-aaaa** スタブリゾルバーオプションと **RES_OPTIONS** 環境変数を認識するようになりました。このオプションが有効な場合、AAAA クエリーはネットワーク経由で送信されません。システム管理者は、診断目的で AAAA DNS ルックアップを無効にすることができます。たとえば、IPv4 のみのネットワークでの余分なルックアップが DNS の問題に影響しないように除外できます。

(BZ#2096191)

IBM Z シリーズ z16 のサポートを追加

IBM z16 プラットフォームで **s390** 命令セットがサポートされるようになりました。**IBM z16** は、**glibc** で **HWCAP_S390_VXRS_PDE2** と **HWCAP_S390_NNPA** という 2 つの追加のハードウェア機能を提供します。その結果、アプリケーションはこれらの機能を使用して、最適化されたライブラリーと機能を提供できるようになりました。

(BZ#2077838)

アプリケーションは、新しい glibc インターフェイスを介して再起動可能なシーケンス機能を使用できます

sched_getcpu 関数 (特に aarch64 で) を高速化するには、**glibc** でデフォルトで再起動可能なシーケンス (rseq) カーネル機能を使用する必要があります。アプリケーションが共有 rseq 領域を継続的に使用できるようにするために、**glibc** は、**glibc** 2.35 アップストリームバージョンで最初に追加された **__rseq_offset**、**__rseq_size**、および **__rseq_flags** シンボルを提供するようになりました。この機能強化により、**sched_getcpu** 関数のパフォーマンスが向上し、アプリケーションは新しい **glibc** インターフェイスを介して再起動可能なシーケンス機能を使用できるようになりました。

(BZ#2085529)

GCC Toolset 12: GDB がバージョン 11.2 にリベースされました。

GCC Toolset 12 では、GDB パッケージがバージョン 11.2 に更新されました。

主なバグ修正と機能拡張は、以下のとおりです。

- 64 ビット ARM アーキテクチャーメモリータグ付け拡張 (MTE) の新しいサポート。**memory-tag** プレフィックスが付いた新しいコマンドを参照してください。
- **-break-insert** および **-dprintf-insert** の **--qualified** オプション。このオプションは、全スコープで検索するのではなく、ユーザーのイベントの場所と完全に一致するものを探します。たとえば、**break --qualified foo** は、グローバルスコープで **foo** という名前のシンボルを探します。**--qualified** を指定しないと、GDB はすべてのスコープでその名前のシンボルを検索します。
- **--force-condition**: 現在無効になっていても、指定された条件が定義されます。
- **-break-condition --force**: MI コマンドと同様。
- **-file-list-exec-source-files** は、出力を制限するオプションの **REGEXP** を受け入れます。
- **.gdbinit** 検索パスには config ディレクトリーが含まれます。順序は次のとおりです。
  a. **$XDG_CONFIG_HOME/gdb/gdbinit**
  b. **$HOME/.config/gdb/gdbinit**
  c. **$HOME/.gdbinit**
- ~/**.config/gdb/gdbearlyinit** または ~/**.gdbearlyinit** のサポート。
- **-eix** および **-eiex** 早期初期化ファイルオプション。

ターミナルユーザーインターフェイス (TUI):

- ターミナルユーザーインターフェイス (TUI) ウィンドウ内でのマウス操作がサポートされます。
- フォーカスされたウィンドウで機能しないキーの組み合わせが GDB に渡されるようになりました。

新しいコマンド:

- **show print memory-tag-violations**
- **set print memory-tag-violations**
- **memory-tag show-logical-tag**
- **memory-tag with-logical-tag**
- **memory-tag show-allocation-tag**
- **memory-tag check**
- **show startup-quietly** および **set startup-quietly**: GDB スクリプトで **-q** または **-quiet** を指定する方法。早期初期化ファイルでのみ有効です。
- **show print type hex** および **set print type hex**: ストラクチャーメンバーのサイズまたはオフセットを 10 進法ではなく 16 進法で出力するように GDB に指示します。
- **show python ignore-environment** および **set python ignore-environment**: 有効にすると、GDB の Python インタープリターは、Python 実行可能ファイルに **-E** を渡すのと同じように、Python 環境変数を無視します。早期初期化ファイルでのみ有効です。

- **show python dont-write-bytecode** および **set python dont-write-bytecode**: **off** の場合、これらのコマンドは、Python 実行可能ファイルに **-B** を渡すのと同様に、GDB の Python インタープリターがインポートされたモジュールのバイトコードコンパイル済みオブジェクトを書き込むことを抑制します。早期初期化ファイルでのみ有効です。

変更したコマンド:

- **break LOCATION if CONDITION**: CONDITION が無効な場合、GDB はブレークポイントの設定を拒否します。**-force-condition** オプションはこれをオーバーライドします。
- **CONDITION -force N COND**: 前のコマンドと同じ。
- **Inferior ID**: ID が省略されている場合、このコマンドは現在の下位関連情報を出力します。それ以外の場合は、変更されません。
- **ptype[/FLAGS] TYPE | EXPRESSION**: **/x** フラグを使用して、struct メンバーのサイズとオフセットを出力するときに 16 進法での表記を使用します。**/d** フラグを使用して同じことを行いますが、10 進法を使用します。
- **info sources**: 出力が再構築されました。

Python API:

- 下位オブジェクトには、読み取り専用の **connection_num** 属性が含まれています。
- 新しい **gdb.Frame.level()** メソッド。
- 新しい **gdb.PendingFrame.level()** メソッド。
- **gdb.Stop** の代わりに **gdb.BreakpoiontEvent** が出力されます。

(BZ#2077494)

GDB は Power 10 PLT 命令をサポートします

GDB は、Power 10 PLT 命令をサポートするようになりました。今回の更新により、ユーザーは共有ライブラリー関数にステップインし、GDB バージョン 10.2-10 以降を使用してスタックバックトレースを検査できるようになりました。

(BZ#1870017)

dyninst パッケージ化がバージョン 12.1 にリベースされました

dyninst パッケージがバージョン 12.1 にリベースされました。主なバグ修正と機能拡張は、以下のとおりです。

- **glibc-2.35** の複数名前空間に対する初期サポート
- DWARF 並列解析の並列処理性を修正
- **CUDA** および **CDNA2** GPU バイナリーのサポートを改善
- IBM POWER Systems (リトルエンディアン) レジスターアクセスのサポートを改善
- PIE バイナリーのサポートを改善
- キャッチブロックの解析を修正

- 64 ビット Arm (**aarch64**) 浮動小数点レジスターへのアクセスを修正

(BZ#2057675)

新しいファイルセット /etc/profile.d/debuginfod.*

組織の debuginfod サービスをアクティベートするための新しい fileset を追加しました。システム全体の **debuginfod** クライアントアクティベーションを取得するには、URL を **/etc/debuginfod/FOO.urls** ファイルに追加する必要があります。

(BZ#2088774)

Rust Toolset がバージョン 1.62.1にリベースされました

Rust Toolset が、バージョン 1.62.1 に更新されました。主な変更点は、以下のとおりです。

- 分割代入では、代入の左側にある既存の変数にパターンを代入できます。たとえば、タプル代入は変数にスワップできます: **(a、b) = (b、a);**
- **core::arch::asm** マクロを使用して、インラインアセンブリが 64 ビット x86 および 64 ビット ARM でサポートされるようになりました。詳細については、リファレンスのインラインアセンブリ章 **/usr/share/doc/rust/html/reference/inline-assembly.html** (オンライン: https://doc.rust-lang.org/reference/inline-assembly.html) を参照してください。
- 列挙は、明示的にアノテーションが付けられた **#[default]** バリアントを使用して **Default** トレイトを派生できるようになりました。
- **Mutex**、**CondVar**、および **RwLock** は、pthreads ではなくカスタム **futex** ベースの実装を使用するようになり、Rust 言語保証によって新たな最適化が可能になりました。
- Rust は、新しく安定化された **Termination** トレイトを実装するユーザー定義型を含む、**main** からのカスタム終了コードをサポートするようになりました。
- Cargo は、依存関係機能のより詳細な制御をサポートしています。**dep:** プレフィックスは、それを機能として公開することなく、オプションの依存関係を参照できます。また、**?** は、**package-name?/feature-name** のように、その依存関係が他の場所で有効になっている場合にのみ依存関係の機能を有効にします。
- Cargo には、依存関係を **Cargo.toml** に追加するための新しい **cargo add** サブコマンドがあります。
- 詳細については、一連のアップストリームリリース発表を参照してください。
  - Announcing Rust 1.59.0
  - Announcing Rust 1.60.0
  - Announcing Rust 1.61.0
  - Announcing Rust 1.62.0
  - Announcing Rust 1.62.1

(BZ#2075337)

LLVM Toolset がバージョン 14.0.6 にリベース

LLVM Toolset はバージョン 14.0.6 にリベースされました。主な変更点は、以下のとおりです。

- 64 ビット x86 では、**AVX512-FP16** 命令のサポートが追加されました。
- Armv9-A、Armv9.1-A、および Armv9.2-A アーキテクチャーのサポートが追加されました。
- PowerPC では、IBM double-double 形式を表す **__ibm128** 型が追加されました。これは **__attribute__((mode (IF)))** としても使用できます。

clang の変更:

- **C++2b** の **if consteval** が実装されました。
- 64 ビット x86 では、**AVX512-FP16** 命令のサポートが追加されました。
- 実験段階の OpenCL C 3.0 および OpenCL 2021 の **C++** サポートが完了しました。
- **-E -P** プリプロセッサーの出力は、必ず空白行を省略します。これは、GCC の動作に一致します。以前は、最大 8 つの連続する空白行が出力に表示される可能性がありました。
- C89 だけでなく、**C99** 以降の標準で **-Wdeclaration-after-statement** をサポートします。これは、GCC の動作に一致します。注目すべきユースケースは、宣言とコードの混在を禁止するスタイルガイドのサポートですが、さらに新しい C 標準に移行したいと考えています。

詳細については、LLVM Toolset および Clang アップストリームのリリースノートを参照してください。

(BZ#2061041)

Go Toolset がバージョン 1.18.2 にリベース

Go Toolset はバージョン 1.18.2 にリベースされました。

主な変更点は、以下のとおりです。

- 以前のバージョンの Go との下位互換性を維持しながら generics を導入。
- 新しいファジングライブラリー。
- 新しい **debug/buildinfo** および **net/netip** パッケージ。
- **go get** ツールがパッケージをビルドまたはインストールしなくなりました。現在は、**go.mod** の依存関係のみを処理します。
- メインモジュールの **go.mod** ファイルで **go 1.17** 以降が指定されている場合、追加の引数なしで **go mod download** コマンドを使用すると、メインモジュールの **go.mod** ファイルで明示的に必要なモジュールのソースコードのみがダウンロードされます。推移的な依存関係のソースコードもダウンロードするには、**go mod download all** コマンドを使用します。
- **go mod vendor** サブコマンドが、出力ディレクトリーを設定する **-o** オプションをサポートするようになりました。
- **go mod tidy** コマンドが、インポートされた各パッケージを提供しているビルドリスト内のモジュールが 1 つだけであることを確認するためにソースコードが必要なモジュールの追加のチェックサムを **go.sum** ファイルに保持するようになりました。この変更は、メインモジュールの **go.mod** ファイルの Go バージョンに左右されません。

(BZ#2075169)

新しいモジュールストリーム: maven:3.8。

RHEL 9.1 では、新しいモジュールストリームとして **Maven 3.8** が導入されています。

maven:3.8 モジュールストリームをインストールするには、次を使用します。

```
# dnf module install maven:3.8
```

(BZ#2083112)

.NET バージョン 7.0 が利用可能

Red Hat Enterprise Linux 9.1 は、**.NET** バージョン 7.0 と共に配布されます。以下は、主な改善点です。

- IBM Power のサポート (**ppc64le**)

詳細は、.NET 7.0 RPM パッケージリリースノート および .NET 7.0 コンテナーリリースノート を参照してください。

(BZ#2112027)

4.15. IDENTITY MANAGEMENT

SSSD が SID リクエストのメモリーキャッシングをサポートするようになりました

この機能拡張により、SSSD は SID 要求のメモリーキャッシングをサポートするようになりました。これには、SID による GID および UID ルックアップと、その逆が含まれます。メモリーキャッシングにより、たとえば、Samba サーバーとの間で大量のファイルをコピーする場合などに、パフォーマンスが向上します。

(JIRA:RHELPLAN-123369)

ipaservicedelegationtarget および ipaservicedelegationrule Ansible モジュールが利用可能になりました

ipaservicedelegationtarget および **ipaservicedelegationrule ansible-freeipa** モジュールを使用して、たとえば、スマートカードで認証された Identity Management (IdM) ユーザーが以下を実行できるように Web コンソールクライアントを設定できます。

- 再認証を求められることなく、Web コンソールサービスが実行されている RHEL ホストで **sudo** を使用します。
- **SSH** を使用してリモートホストにアクセスし、再度認証を求められることなくホスト上のサービスにアクセスします。

ipaservicedelegationtarget および **ipaservicedelegationrule** モジュールは、制約付き委任とも呼ばれる Kerberos **S4U2proxy** 機能を利用します。IdM は伝統的にこの機能を使用して、Web サーバーフレームワークがユーザーに代わって LDAP サービスチケットを取得できるようにします。IdM-AD 信頼システムは、この機能を使用して cifs プリンシパルを取得します。

(JIRA:RHELPLAN-117109)

SSSD で、FAST の匿名 PKINIT がサポートされます

この機能拡張により、SSSD は Flexible Authentication via Secure Tunneling (FAST) の匿名 PKINITをサポートするようになりました。これは、Active Directory のKerberos アーマーとも呼ばれます。これまで、FAST を使用するには、必要なクレデンシャルを要求するために Kerberos キータブが必要でした。匿名 PKINIT を使用してこのクレデンシャルキャッシュを作成し、FAST セッションを確立できるようになりました。

匿名 PKINIT を有効にするには、次の手順を実行します。

1. **sssd.conf** ファイルの **[domain]** セクションで、**krb5_fast_use_anonymous_pkinit** を **true** に設定します。
2. SSSD を再起動します。
3. IdM 環境では、IdM ユーザーとしてログインすることで、FAST セッションの確率に匿名 PKINIT が使用されたか検証できます。FAST チケットを含むキャッシュファイルが作成され、その中の **Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS** が匿名 PKINIT の使用を示します。

```
klist /var/lib/sss/db/fast_ccache_IPA.VM
Ticket cache: FILE:/var/lib/sss/db/fast_ccache_IPA.VM
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS
Valid starting Expires Service principal
03/10/2022 10:33:45 03/10/2022 10:43:45 krbtgt/IPA.VM@IPA.VM
```

(JIRA:RHELPLAN-123368)

IdM が Random Serial Numbers をサポートするようになりました

今回の更新により、Identity Management (IdM) に **dogtagpki 11.2.0** が含まれるようになりました。これにより、Random Serial Numbers バージョン 3 (RSNv3) を使用できるようになります。**ipa-server-install** または **ipa-ca-install** の実行時に **--random-serial-numbers** オプションを使用して RSNv3 を有効にできます。RSNv3 を有効にすると、IdM は範囲管理なしで PKI の証明書とリクエストに対して完全にランダムなシリアル番号を生成します。RSNv3 を使用すると、大規模な IdM インストールでの範囲管理を回避し、IdM を再インストールする際の一般的な競合を防ぐことができます。

重要

RSNv3 は、新しい IdM インストールでのみサポートされます。有効にした場合、すべての PKI サービスで RSNv3 を使用する必要があります。

(BZ#747959)

IdM は、ユーザーパスワードの有効期限が切れた後に許可される LDAP バインド数の制限をサポートするようになりました

今回の機能拡張により、Identity Management (IdM) ユーザーのパスワードの有効期限が切れたときに許可される LDAP バインドの数を設定できます。

-1
: IdM は、ユーザーがパスワードをリセットする必要がある前に、ユーザーに無制限の LDAP バインドを許可します。これはデフォルト値で、以前の動作と一致します。

0
: この値は、パスワードの有効期限が切れると、すべての LDAP バインドを無効にします。適用された場合、ユーザーは自分のパスワードをすぐにリセットする必要があります。

1-MAXINT
: 有効期限が切れると、入力された値どおりの数のバインドが許可されます。

この値は、グローバルパスワードポリシーとグループポリシーで設定できます。

数はサーバーごとに保存されることに注意してください。

ユーザーが自分のパスワードをリセットするには、現在の期限切れパスワードにバインドする必要があります。ユーザーが有効期限後に許可されるバインドをすべて使い果たした場合、管理者がパスワードをリセットする必要があります。

(BZ#2091988)

新しい ipasmartcard_server および ipasmartcard_client ロールが追加されました

今回の更新により、**ansible-freeipa** パッケージは、スマートカード認証用に Identity Management (IdM) サーバーとクライアントを設定するための Ansible ロールを提供します。**ipasmartcard_server** および **ipasmartcard_client** ロールは、統合を自動化および簡素化するために **ipa-** advise スクリプトを置き換えます。他の **ansible-freeipa** ロールと同じインベントリーと命名スキームが使用されます。

(BZ#2076567)

IdM は、Windows Server 2022 での AD Trust の設定をサポートするようになりました

この機能拡張により、Identity Management (IdM) ドメインと、Windows Server 2022 を実行するドメインコントローラーを使用する Active Directory フォレストとの間でフォレスト間の信頼を確立できます。

(BZ#2122716)

ipa-dnskeysyncd および ipa-ods-exporter デバッグメッセージは、デフォルトで/var/log/messages に記録されなくなりました

以前は、LDAP から OpenDNSSEC への同期サービスである **ipa-dnskeysyncd** と、Identity Management (IdM) OpenDNSSEC エクスポーターサービスである **ipa-ods-exporter** は、デフォルトですべてのデバッグメッセージを **/var/log/messages** に記録していました。その結果、ログファイルが非常に大きくなっていました。今回の機能拡張により、**/etc/ipa/dns.conf** ファイルで **debug=True** を設定することにより、ログレベルを設定できます。詳細は、IdM 設定ファイルの man ページである **default.conf(5)** を参照してください。

(BZ#2083218)

samba がバージョン 4.16.1 にリベースされました。

samba パッケージがアップストリームバージョン 4.16.1 にアップグレードされ、以前のバージョンに対するバグ修正や機能強化が追加されました。

- デフォルトでは、**smbd** プロセスは新しい **samba-dcerpcd** プロセスをオンデマンドで自動的に開始し、分散コンピューティング環境/リモートプロシージャコール (DCERPC) を提供します。Samba 4.16 以降では、DCERPC を使用するために必ず **samba-dcerpcd** が必要であることに注意してください。**/etc/samba/smb.conf** ファイルの **[global]** セクションで **rpc start on demand helpers** 設定を無効にする場合、スタンドアロンモードで **samba-dcerpcd** を実行する **systemd** サービスユニットを作成する必要があります。
- Cluster Trivial Database (CTDB) の **recovery master** のロール名が、**leader** に変更されました。その結果、次のとおり **ctdb** サブコマンドの名前が変更されました。
  - **recmaster** から **leader** に変更
  - **setrecmasterrole** から **setleaderrole** に変更
- CTDB **recovery lock** 設定の名前が **cluster lock** に変更されました。
- CTDB は、リーダーブロードキャストと関連するタイムアウトを使用して、選択が必要かどうかを判断するようになりました。

Samba 4.11 以降はサーバーメッセージブロックバージョン 1 (SMB1) プロトコルが非推奨となり、今後のリリースで削除されることに注意してください。

Samba を起動する前にデータベースファイルがバックアップされます。**smbd**、**nmbd**、または**winbind** サービスが起動すると、Samba が **tdb** データベースファイルを自動的に更新します。Red Hat は、**tdb** データベースファイルのダウングレードをサポートしていないことに留意してください。

Samba を更新したら、**testparm** ユーティリティーを使用して **/etc/samba/smb.conf** ファイルを確認します。

更新する前に、upstream release notes で重要な変更の詳細を確認してください。

(BZ#2077487)

SSSD が Windows Server 2022 との直接統合をサポートするようになりました

この機能拡張により、SSSD を使用して、Windows Server 2022 を実行するドメインコントローラーを使用する Active Directory フォレストと RHEL システムを直接統合できます。

(BZ#2070793)

SSSD マルチスレッドパフォーマンスの向上

以前は、SSSD は、Red Hat Directory Server や Identity Management などのマルチスレッドアプリケーションからの並列リクエストをシリアル化していました。今回の更新により、**nss** や **pam** などの SSSD クライアントライブラリーがすべて修正され、リクエストはシリアル化されなくなりました。そのため、複数のスレッドからのリクエストを並行して実行できるようになり、パフォーマンスが向上しました。以前のシリアル化の動作を有効にするには、環境変数 **SSS_LOCKFREE** を **NO** に設定します。

(BZ#1978119)

Directory Server が Auto Membership プラグインタスクのキャンセルに対応

以前は、ディレクトリーサーバーの設定が複雑な場合 (大規模なグループ、複雑なルール、他のプラグインとの対話など)、Auto Membership プラグインタスクが原因でサーバーの CPU 使用率が高くなる可能性がありました。この機能強化により、Auto Membership プラグインタスクをキャンセルできます。その結果、パフォーマンスの問題は発生しなくなりました。

(BZ#2052527)

Directory Server は、ldapdelete 使用時に再帰的な削除操作をサポートするようになりました

この機能拡張により、Directory Server は **Tree Delete Control 1.2.840.113556.1.4.805** OpenLDAP コントロールをサポートするようになりました。その結果、**ldapdelete** ユーティリティーを使用して、親エントリーのサブエントリーを再帰的に削除できます。

(BZ#2057063)

Directory Server インストール時における基本的な複製オプションの設定に対応

この機能強化により、インスタンスのインストール時に **.inf** ファイルを使用して、認証情報や変更履歴のトリミングなどの基本的なレプリケーションオプションを設定することができるようになりました。

(BZ#2057066)

Directory Server は、root 以外のユーザーによるインスタンスの作成をサポートするようになりました

以前は、root 以外のユーザーは Directory Server インスタンスを作成できませんでした。この機能強化により、root 以外のユーザーは **dscreate ds-root** サブコマンドを使用して、通常どおり **dscreate**、**dsctl**、**dsconf** コマンドを使用して Directory Server インスタンスを作成および管理する環境を設定できます。

(BZ#1872451)

pki パッケージの名前を idm-pki に変更

以下の **pki** パッケージの名前が **idm-pki** に変更され、IDM パッケージと Red Hat Certificate System パッケージを区別しやすくなりました。

- **idm-pki-tools**
- **idm-pki-acme**
- **idm-pki-base**
- **idm-pki-java**
- **idm-pki-ca**
- **idm-pki-kra**
- **idm-pki-server**
- **python3-idm-pki**

(BZ#2139877)

4.16. グラフィックインフラストラクチャー

Matrox GPU で Wayland が有効になりました

デスクトップセッションは、Matrox GPU で Wayland バックエンドを有効にするようになりました。

以前のリリースでは、パフォーマンスやその他の制限により、Matrox GPU で Wayland が無効になっていました。これらの問題が修正されました。

デスクトップセッションは、Wayland から Xorg に切り替えることができます。詳細は Overview of GNOME environments を参照してください。

(BZ#2097308)

第 12 世代 Intel Core GPU に対応しました

今回のリリースで、12th Gen Intel Core CPU の複数の統合 GPU に対するサポートが追加されました。これには、以下の CPU モデルを含む Intel UHD グラフィックと Intel Xe 統合 GPU が含まれます。

- Intel Core i3 12100T から Intel Core i9 12900KS まで
- Intel Pentium Gold G7400 および G7400T
- Intel Celeron G6900 および G6900T
- Intel Core i5-12450HX から Intel Core i9-12950HXまで
- Intel Core i3-1220P から Intel Core i7-1280P まで

(JIRA:RHELPLAN-135601)

新しい AMD GPU のサポート

今回のリリースで、AMD Ryzen 6000 Series CPU の複数の AMD Radeon RX 6000 Series GPU と統合グラフィックスのサポートが追加されました。

以下の AMD Radeon RX 6000 Series GPU モデルがサポートされるようになりました。

- AMD Radeon RX 6400
- AMD Radeon RX 6500 XT
- AMD Radeon RX 6300M
- AMD Radeon RX 6500M

AMD Ryzen 6000 シリーズには、以下の CPU モデルで確認された統合 GPU が含まれています。

- AMD Ryzen 5 6600U
- AMD Ryzen 5 6600H
- AMD Ryzen 5 6600HS
- AMD Ryzen 7 6800U
- AMD Ryzen 7 6800H
- AMD Ryzen 7 6800HS
- AMD Ryzen 9 6900HS
- AMD Ryzen 9 6900HX
- AMD Ryzen 9 6980HS
- AMD Ryzen 9 6980HX

(JIRA:RHELPLAN-135602)

4.17. WEB コンソール

Web コンソールの更新進行状況ページで、自動再起動オプションがサポートされるようになりました

更新の進行状況ページに、**完了後に再起動する** スイッチが追加されました。これにより、更新のインストール後にシステムが自動的に再起動します。

(BZ#2056786)

4.18. RED HAT ENTERPRISE LINUX システムロール

network RHEL システムロールが、nmstate API を使用したネットワーク設定をサポートするようになりました

この更新により、**network** RHEL システムロールは **nmstate** API を介したネットワーク設定をサポート

するようになりました。ユーザーは、接続プロファイルを作成する代わりに、必要なネットワーク状態の設定をネットワークインターフェイスに直接適用できるようになりました。この機能により、ネットワークの部分的な設定も可能になります。これには次の利点があります。

- ネットワーク設定の複雑さの軽減
- 信頼できるネットワーク状態の変更適用方法
- ネットワーク設定全体の追跡が不要

(BZ#2072385)

network RHEL システムロールを使用して、IPoIB 機能で接続を作成できるようになりました

network RHEL システムロールの **infiniband** 接続タイプが、Internet Protocol over Infiniband (IPoIB) 機能をサポートするようになりました。この機能を有効にするには、**infiniband** の **p_key** オプションに値を定義します。**p_key** を指定する場合は、**network_connections** 変数の **interface_name** オプションを未設定にする必要があることに注意してください。**network** RHEL システムロールの以前の実装では、**infiniband** 接続タイプの **p_key** 値と **interface_name** オプションが適切に検証されませんでした。したがって、IPoIB 機能はこれまで機能しませんでした。詳細については、**/usr/share/doc/rhel-system-roles/network/** ディレクトリーにある README ファイルを参照してください。

(BZ#2086965)

HA Cluster RHEL システムロールが SBD フェンシングと Corosync 設定をサポートするようになりました

HA Cluster システムロールは、次の機能をサポートするようになりました。

SBD フェンシング

フェンシングは、HA クラスター設定の重要な部分です。SBD は、フェンシングが必要な場合にノードが確実に自己終了する手段を提供します。SBD フェンシングは、従来のフェンシングメカニズムが不可能な環境で特に役立ちます。HA Cluster システムロールを使用して SBD フェンシングを設定できるようになりました

Corosync 設定

HA Cluster システムロールは、トランスポート、圧縮、暗号化、リンク、トーテム、クォーラムなどの Corosync 設定をサポートするようになりました。これらの設定は、デフォルト設定が適切でない場合に、クラスター設定をお客様のニーズと環境に一致させるために必要です。

(BZ#2065337、BZ#2070452、BZ#2079626、BZ#2098212、BZ#2120709、BZ#2120712)

network RHEL ロールは、ルーティングルールのネットワーク設定を設定するようになりました

以前は、パケットの宛先アドレスフィールドに基づいてパケットをルーティングできましたが、ソースルーティングおよびその他のポリシールーティングルールを定義できませんでした。今回の機能拡張により、**network** RHEL ロールはルーティングルールをサポートし、ユーザーがパケット転送またはルート選択を制御できるようになります。

(BZ#2079622)

新しい previous:replaced 設定により、firewall システムロールがファイアウォール設定をデフォルトにリセットできるようになります

各マシンで既存のファイアウォール設定が異なる一連のマシンを管理するシステム管理者は、**firewall** ロールの **previous: replaced** 設定を使用して、すべてのマシンで同じファイアウォール設定を設定できるようになりました。**previous: replaced** 設定では、既存のすべてのファイアウォール設定を消去

し、それらを一貫した設定に置き換えることができます。

(BZ#2043010)

以前の設定をオーバーライドする postfix RHEL システムロールの新しいオプションが追加されました

グループ内で **postfix** 設定が異なるシステムグループを管理していると、設定を一貫させる必要があるバイアがあります。この機能拡張により、**postfix_conf** ディクショナリー内で **previous: replaced** オプションを指定して、既存の設定を削除し、クリーンな **postfix** インストール上に目的の設定を適用できます。その結果、既存の **postfix** 設定を消去して、管理対象のすべてのシステムで一貫性を確保できます。

(BZ#2065383)

拡張された microsoft.sql.server RHEL システムロール

次の新しい変数が、**microsoft.sql.server** RHEL システムロールで使用できるようになりました。

- 高可用性クラスターの設定を制御する **mssql_ha_** 接頭辞を持つ変数。
- 管理対象ノードで **mssql_tls_cert** および **mssql_tls_private_key** の値を検索するための **mssql_tls_remote_src** 変数。デフォルトの **false** 設定のままにすると、ロールは制御ノードでこれらのファイルを検索します。
- ファイアウォールポートを自動的に管理するための **mssql_manage_firewall** 変数。この変数が **false** に設定されている場合は、ファイアウォールポートを手動で有効にする必要があります。
- **mssql_pre_input_sql_file** 変数と **mssql_post_input_sql_file** 変数を使用して、SQL スクリプトをロールの実行前または実行後に実行するかどうかを制御します。これらの新しい変数は、SQL スクリプトの実行時間に影響を与えることができなかった以前の **mssql_input_sql_file** 変数に取って代わります。

(BZ#2066337)

logging RHEL システムロールが、ファイル入力で startmsg.regex および endmsg.regex オプションをサポートするようになりました

この機能拡張により、正規表現を使用して、ファイルからのログメッセージをフィルタリングできるようになりました。オプション **startmsg_regex** と **endmsg_regex** がファイルの入力に含まれるようになりました。**startmsg_regex** はメッセージの開始部分に一致する正規表現を表し、**endmsg_regex** はメッセージの最後の部分に一致する正規表現を表します。その結果、日時、優先度、重大度などのプロパティーに基づいてメッセージをフィルタリングできるようになりました。

(BZ#2112145)

sshd RHEL システムロールが、ドロップインディレクトリーの include ディレクティブを検証するようになりました

RHEL 9 の **sshd** RHEL システムロールは、ドロップインディレクトリー内のファイルのみを管理します。しかし、以前はメインの **sshd_config** ファイルからディレクトリーが組み込まれているかどうかは検証していませんでした。この更新により、ドロップインディレクトリーの include ディレクティブが **sshd_config** に含まれていることをロールが検証するようになりました。その結果、ロールは提供された設定をより確実に適用します。

(BZ#2052081)

sshd RHEL システムロールを /etc/ssh/sshd_config から管理できるようになりました

RHEL 9 管理対象ノードに適用される **sshd** RHEL システムロールが、SSHD 設定をドロップインディレクトリー (デフォルトでは **/etc/ssh/sshd_config.d/00-ansible_system_role.conf**) に配置するようになりました。以前は、**/etc/ssh/sshd_config** ファイルを変更すると、**00-ansible_system_role.conf** のデフォルト値が上書きされていました。今回の更新により、**00-ansible_system_role.conf** のシステムデフォルト値を保持しながら、**00-ansible_system_role.conf** の代わりに **/etc/ssh/sshd_config** を使用して SSHD を管理できるようになりました。

(BZ#2052086)

metrics ロールは、管理設定ファイル内で常に Ansible_managed コメントを使用します。

この更新により、**metrics** ロールは、Ansible 標準の **ansible_managed** 変数を使用して、設定ファイルに Ansiblemanaged コメントを挿入します。コメントは、**metrics** ロールによってファイルが上書きされるため、設定ファイルを直接編集してはならないことを示します。その結果、設定ファイルには、設定ファイルが Ansible によって管理されていることを示す宣言が含まれています。

(BZ#2065392)

storage RHEL システムロールが、プールメンバーの管理をサポートするようになりました

storage RHEL システムロールは、先にプールを削除せずに、既存の LVM プールからディスクを追加または削除できるようになりました。プールの容量を増やすために、**storage** RHEL システムロールは、プールに新しいディスクを追加し、別の用途のためにプール内で現在割り当てられているディスクを解放できます。

(BZ#2072742)

シンプロビジョニングされたボリュームが、storage RHEL システムロールでサポートされるようになりました

storage RHEL システムロールは、シンプロビジョニングされた LVM 論理ボリューム (LV) を作成し、管理できるようになりました。シンプロビジョニングされた LV は書き込み時に割り当てられます。これにより、後で必要に応じてシンプロビジョニングされた LV に提供される物理ストレージとしてボリュームを作成する際の柔軟性が向上します。LVM シンプロビジョニングでは、シン LV とそのスナップショットに共通するデータブロックが共有されるため、より効率的なスナップショットを作成できます。

(BZ#2072745)

storage RHEL システムロールで、キャッシュされたボリュームのサポートが強化されました

storage RHEL システムロールは、既存の LVM 論理ボリュームにキャッシュを割り当てることができるようになりました。LVM キャッシュを使用すると、LV のデータのサブセットを SSD などのより小さくて高速なデバイスに一時的に保存することで、より低速な論理ボリュームのパフォーマンスを向上させることができます。これにより、以前にキャッシュされていない既存のボリュームにキャッシュを追加 (接続) できるようになり、キャッシュされたボリュームを作成するために以前に追加されたサポートが強化されます。

(BZ#2072746)

logging RHEL システムロールが、template、severity、および facility オプションをサポートするようになりました

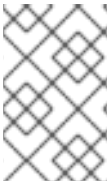
logging RHEL システムロールに、ファイル入力に対する新しい便利な **severity** および **facility** オプションと、ファイルおよび転送出力に対する新しい **template** オプションが追加されました。**template** オプションでは、**traditional** パラメーターを使用して従来の時刻形式を指定し、**syslog** パラメーター

を使用して syslog プロトコル 23 形式を指定し、**modern** パラメーター を使用してモダンスタイル形式を指定します。そうすることで、**logging** ロールを使用して、severity と facility でフィルタリングしたり、template で出力形式を指定したりできるようになりました。

(BZ#2075119)

RHEL システムロールが、ファクト収集が無効になっている Playbook でも利用できるようになりました

Ansible ファクト収集は、パフォーマンスまたはその他の理由により、環境内で無効になっている場合があります。以前は、このような設定で RHEL システムロールを使用することはできませんでした。今回の更新により、システムは設定内の **ANSIBLE_GATHERING=explicit** パラメーターと Playbook 内の **gather_facts: false** パラメーターを検出し、**setup:** モジュールを使用して、指定されたロールに必要なファクトのみを収集します (ファクトキャッシュから取得できない場合)。



注記

パフォーマンスのために Ansible ファクト収集を無効にしている場合は、代わりに Ansible ファクトキャッシングを有効にできます。これにより、ソースからそれらを取得する際にパフォーマンスに影響が及びません。

(BZ#2078989)

デフォルトで storage ロールの詳細レベルが低くなりました

storage ロール出力の詳細レベルがデフォルトで低くなりました。今回の更新により、ユーザーはストレージロール出力の詳細レベルを上げて、Ansible 詳細レベル 1 以上を使用している場合にのみデバッグ出力を生成できるようになりました。

(BZ#2079627)

masquerade または icmp_block_inversion を設定するときに、firewall RHEL システムロールに state パラメーターが不要になりました

カスタムファイアウォールゾーンを設定する場合、変数 **masquerade** と **icmp_block_inversion** はブール値で設定します。**true** の値は **state: present** を意味し、**false** の値は **state: Absent** を意味します。したがって、**masquerade** または **icmp_block_inversion** を設定する場合、**state** パラメーターは必要ありません。

(BZ#2093423)

firewall RHEL システムロールの absent および present 状態を使用して、サービスを追加、更新、削除できるようになりました

この機能拡張により、**present** の状態を使用してポート、モジュール、プロトコル、サービス、および宛先アドレスを追加したり、**absent** 状態を使用してそれらを削除したりできます。**firewall** RHEL システムロールで **absent** 状態と **present** 状態を使用するには、**permanent** オプションを **true** に設定してください。**permanent** オプションを **true** に設定すると、状態の設定は変更されるまで適用され、ロールをリロードしても影響を受けません。

(BZ#2100292)

firewall システムロールは、PCI デバイス ID を使用してゾーンにインターフェイスを追加または削除できます

firewall システムロールは、PCI デバイス ID を使用して、ネットワークインターフェイスをゾーンに割り当てたり、ゾーンから削除したりできるようになりました。以前は、インターフェイス名ではなく

PCI デバイス ID しか分からない場合、ユーザーはまず対応するインターフェイス名を識別し、**firewall** システムロールを使用する必要がありました。今回の更新により、**firewall** システムロールは PCI デバイス ID を使用して、ゾーン内のネットワークインターフェイスを管理できるようになりました。

(BZ#2100942)

firewall RHEL システムロールが、Ansible ファクトを提供できるようになりました

この機能拡張により、Playbook に引数なしの **firewall:** 変数を含めることで、すべてのシステムから **firewall** RHEL システムロールの Ansible ファクトを収集できるようになりました。Ansible ファクトのより詳細なバージョンを収集するには、**detailed: true** 引数を使用します。以下はその例です。

```
vars:
  firewall:
    detailed: true
```

(BZ#2115154)

selinux RHEL システムロールに seuser と selevel の設定を追加しました

SELinux コンテキストファイルシステムのマッピング設定時に、**seuser** および **selevel** パラメーターを設定する必要がある場合があります。今回の更新により、**selinux_fcontext** でオプション引数 **seuser** および **selevel** を使用し、SELinux コンテキストファイルシステムマッピングで SELinux ユーザーとレベルを指定できるようになりました。

(BZ#2115157)

カスタムリスニングポートを設定するための新しい cockpit システムロール変数

cockpit システムロールに、デフォルトの 9090 ポート以外のカスタムリスニングポートを設定できる **cockpit_port** 変数が導入されています。カスタムのリッスンポートを設定する場合は、Web コンソールがそのポートでリッスンできるように SELinux ポリシーを調整する必要があることに注意してください。

(BZ#2115152)

metrics ロールは、postfix パフォーマンスデータをエクスポートできるようになりました。

記録と詳細なパフォーマンス分析のために、**metrics** ロールで新しい **metrics_from_postfix** ブール変数を使用できるようになりました。今回の機能強化により、変数を設定すると、システムで **pmdapostfix** メトリックエージェントが有効になり、**postfix** に関する統計が利用可能になります。

(BZ#2051737)

Postfix ロールは、管理設定ファイル内で常に Ansible_managed コメントを使用します。

postfix ロールは **/etc/postfix/main.cf** 設定ファイルを生成します。今回の更新で、**postfix** ロールは、Ansible 標準の **ansible_managed** 変数を使用して、設定ファイルに Ansible managed コメントを挿入します。コメントは、**postfix** ロールがファイルを上書きする可能性があるため、設定ファイルを直接編集しないことを示しています。その結果、設定ファイルには、設定ファイルが Ansible によって管理されていることを示す宣言が含まれています。

(BZ#2065393)

nbde-client RHEL システムロールが静的 IP アドレスをサポートするようになりました

以前のバージョンの RHEL では、静的 IP アドレスを使用してシステムを再起動し、**nbde_client** RHEL システムロールで設定すると、システムの IP アドレスが変更されました。今回の更新で

は、**nbde_client** ロールにより静的 IP アドレスを持つシステムがサポートされ、再起動後にその IP アドレスは変更されません。

デフォルトでは、**nbde_client** ロールは起動時に DHCP を使用し、システムの起動後に設定済みの静的 IP に切り替わることに注意してください。

(BZ#2070462)

4.19. 仮想化

RHEL Web コンソールは、OS のダウンロード の仮想マシンワークフローのオプションとして RHEL を備えています。

この機能拡張により、RHEL Web コンソールは、デフォルトの**OS のダウンロード** ワークフローを使用した RHEL 仮想マシン (VM) のインストールをサポートするようになりました。その結果、Web コンソール内で直接 RHEL OS を仮想マシンとしてダウンロードおよびインストールできます。

(JIRA:RHELPLAN-121982)

KVM アーキテクチャーコンプライアンスの向上

今回の更新により、KVM ハイパーバイザーのアーキテクチャーコンプライアンスが強化され、より厳格になりました。その結果、ハイパーバイザーは、Linux ベースおよびその他のオペレーティングシステムに対する将来の変更に対応できるようになりました。

(JIRA:RHELPLAN-117713)

ap-check が RHEL 9 で利用可能になりました

mdevctl ツールは、新しい **ap-check** サポートユーティリティーを提供するようになりました。**mdevctl** を使用して、**matrix** および **vfio-ap** デバイスだけでなく、仮想マシンへのパススルー使用が許可されている暗号化アダプターとドメインを永続的に設定できます。**mdevctl** を使用すると、IPL のたびにこれらのアダプター、ドメイン、デバイスを再設定する必要がなくなります。さらに、**mdevctl** は、ディストリビューターがそれらを再設定する他の方法を発明するのを防ぎます。

vfio-ap デバイスに対して **mdevctl** コマンドを呼び出すと、新しい **ap-check** サポートユーティリティーが **mdevctl** コマンドの一部として呼び出され、**vfio-ap** デバイス設定に対して追加の有効性チェックが実行されます。

さらに、**chzdev** ツールは、**vfio-ap** デバイスで使用できる AP リソースを決定する、システム全体の補助プロセッサー (AP) マスク設定を管理する機能を提供するようになりました。**chzdev** を使用すると、関連付けられた **udev** ルールを生成することで、これらの設定を永続化できます。**lszdev** を使用して、システム全体の AP マスク設定を照会できるようになりました。

(BZ#1870699)

open-vm-tools が 12.0.5 にリベースされました

open-vm-tools パッケージがバージョン 12.0.5 にアップグレードされ、多数の バグ修正および新機能が追加されました。最も注目すべきは、Salt Minion ツールをゲスト OS 変数で管理するためのサポートが追加されたことです。

(BZ#2061193)

IBM Z 上の一部の VM は、896 バイトを超えるカーネルコマンドラインで起動できるようになりました

以前は、RHEL 9 IBM Z ホストでの仮想マシン (VM) の起動は、VM のカーネルコマンドラインが 896

バイトより長い場合、常に失敗していました。今回の更新により、QEMU エミュレーターは 896 バイトを超えるカーネルコマンドラインを処理できるようになりました。その結果、VM カーネルがサポートしている場合、非常に長いカーネルコマンドラインで VM の QEMU ダイレクトカーネルブートを使用できるようになりました。具体的には、896 バイトを超えるコマンドラインを使用するには、VM で Linux カーネルバージョン 5.16-rc1 以降を使用する必要があります。

(BZ#2044218)

IBM Z の Secure Execution 機能がリモート認証をサポートするようになりました

IBM Z アーキテクチャーの Secure Execution 機能は、リモート認証をサポートするようになりました。**pvattest** ユーティリティーは、リモート認証要求を作成して、Secure Execution が有効になっているゲストの整合性を検証できます。

さらに、GISA を使用して Secure Execution でゲストに割り込みを挿入できるようになりました。

(BZ#2001936、BZ#2044300)

複数のスレッドを使用した VM メモリーの事前割り当て

次の例のように、ドメイン XML 設定で仮想マシン (VM) メモリー割り当て用に複数の CPU スレッドを定義できるようになりました。

```
<memoryBacking>
  <allocation threads='8'/>
</memoryBacking>
```

これにより、VM の起動時にメモリーページの割り当てに複数のスレッドが使用されるようになります。その結果、特に VM に大量の RAM が割り当てられ、hugepage によってバックアップされている場合、複数の割り当てスレッドが設定されている VM の起動が大幅に高速化されます。

(BZ#2064194)

RHEL 9 ゲストが SEV-SNP をサポートするようになりました

RHEL 9 をゲストオペレーティングシステムとして使用する仮想マシン (VM) で、Secure Nested Paging (SNP) 機能を備えた AMD Secure Encrypted Virtualization (SEV) を使用できるようになりました。他の利点の中でも、SNP はメモリー整合性保護を改善することで SEV を強化します。これにより、データの再生やメモリーの再マッピングなどのハイパーバイザーベースの攻撃を防ぐことができます。SEV-SNP が RHEL 9 仮想マシンで機能するには、仮想マシンを実行しているホストも SEV-SNP をサポートしている必要があることに注意してください。

(BZ#2169738)

4.20. クラウド環境の RHEL

cloud-init の新しい SSH モジュール

今回の更新で、**cloud-init** ユーティリティーに SSH モジュールが追加され、インスタンスの作成時にホストキーが自動的に生成されるようになりました。

この変更により、デフォルトの **cloud-init** 設定が更新されました。したがって、ローカルの変更があった場合は、/etc/cloud/cloud.cfg に "ssh_genkeytypes: ['rsa', 'ecdsa', 'ed25519']" 行が含まれていることを確認してください。

そうしないと、**sshd** サービスを起動できないイメージが **cloud-init** によって作成されます。この問題が発生した場合は、次の手順に従って問題を回避してください。

1. **/etc/cloud/cloud.cfg** ファイルに次の行が含まれていることを確認します。

```
ssh_genkeytypes:  ['rsa', 'ecdsa', 'ed25519']
```

2. **/etc/ssh/ssh_host_*** ファイルがインスタンスに存在するかどうかを確認します。
3. **/etc/ssh/ssh_host_*** ファイルが存在しない場合は、次のコマンドを使用してホストキーを生成します。

```
cloud-init single --name cc_ssh
```

4. sshd サービスを再起動します。

```
systemctl restart sshd
```

(BZ#2115791)

4.21. コンテナー

Container Tools パッケージが更新されました

Podman、Buildah、Skopeo、crun、および runc ツールを含む Container Tools パッケージが利用可能になりました。今回の更新で、以前のバージョンに対するバグ修正および機能拡張のリストが追加されました。

主な変更点は、以下のとおりです。

- **podman pod create** コマンドが、CPU とメモリーの制限の設定をサポートするようになりました。Pod 内のすべてのコンテナーに制限を設定できますが、Pod 内の個々のコンテナーには独自の制限を設定できます。
- **podman pod clone** コマンドは、既存の Pod のコピーを作成します。
- **podman play kube** コマンドは、**BlockDevice** および **CharDevice** ボリュームを使用したセキュリティーコンテキスト設定をサポートするようになりました。
- **podman play kube** によって作成された Pod は、**podman-kube@<service>.service** (たとえば、**systemctl --user start podman-play-kube@$(systemd-escape my.yaml).service** など) を使用して、systemd ユニットファイルによって管理できるようになりました。サービス)。
- **podman push** および **podman push manifest** コマンドが sigstore 署名をサポートするようになりました。
- **podman network --opt isolate** コマンドを使用して、Podman ネットワークを分離できるようになりました。

Podman がバージョン 4.2 にアップグレードされました。注目すべき変更点の詳細については、アップストリームリリースノート を参照してください。

(JIRA:RHELPLAN-118462)

GitLab Runner が Podman を使用して RHEL で利用できるようになりました

GitLab Runner 15.1 以降では、GitLab Runner Docker Executor でコンテナーランタイムとして Podman を使用できます。詳細については、GitLab のリリースノート を参照してください。

(JIRA:RHELPLAN-101140)

Podman が --health-on-failure オプションをサポートするようになりました

podman run および **podman create** コマンドで **--health-on-failure** オプションがサポートされるようになり、コンテナーのステータスが異常になったときに実行するアクションを決定できるようになりました。

--health-on-failure オプションは、次の 4 つのアクションをサポートします。

- **none**: アクションを実行しません。これがデフォルトのアクションです。
- **kill**: コンテナーを強制終了します。
- **restart**: コンテナーを再起動します。
- **stop**: コンテナーを停止します。

注記

restart アクションを **--restart** オプションと組み合わせないでください。systemd ユニット内で実行する場合は、systemd の再起動ポリシーを利用する代わりに **kill** または **stop** アクションを使用することを検討してください。

(BZ#2097708)

Netavark ネットワークスタックが利用可能になりました。

Netavark スタックは、コンテナーのネットワーク設定ツールです。RHEL 9 では、Netavark スタックは完全にサポートされ、デフォルトで有効になっています。

このネットワークスタックには、次の機能があります。

- JSON 設定ファイルを使用したコンテナーネットワークの設定
- ブリッジおよび MACVLAN インターフェイスを含むネットワークインターフェイスの作成、管理、および削除
- ネットワークアドレス変換 (NAT) やポートマッピングルールなどのファイアウォールの設定
- IPv4 および IPv6 (IPv4 and IPv6)
- 複数ネットワークのコンテナー機能の向上
- aardvark-dns project を使用したコンテナー DNS 解決

注記

同じバージョンの Netavark スタックと **aardvark-dns** 権限のある DNS サーバーを使用する必要があります。

(JIRA:RHELPLAN-132023)

新しいパッケージ: CRB リポジトリーの catatonit

新しい **catatonit** パッケージが CodeReady Linux Builder (CRB) リポジトリーで利用できるようになりました。**catatonit** パッケージは、コンテナーの最小限の init プログラムとして使用され、アプリケー

ションコンテナーイメージ内に含めることができます。CodeReady Linux Builder リポジトリーに含まれるパッケージは、サポート対象外であることに注意してください。

RHEL 9.0 以降、**podman-catonit** パッケージは AppStream リポジトリーで利用できることに注意してください。**podman-catatonit** パッケージは、Podman ツールでのみ使用されます。

(BZ#2074193)

第5章 外部カーネルパラメーターへの重要な変更

この章では、Red Hat Enterprise Linux 9.1 で配布されるカーネルの重要な変更点の概要をシステム管理者に提供します。変更には、たとえば、**proc** エントリー、**sysctl** および **sysfs** のデフォルト値、ブートパラメーター、カーネル設定オプション、または重要な動作の変更などが含まれます。

新しいカーネルパラメーター

allow_mismatched_32bit_el0 = [ARM64]

このパラメーターを使用すると、EL0 レベルでの 32 ビットサポートが一致しないシステムが 32 ビットアプリケーションを実行できるようになります。32 ビット EL0 をサポートする CPU のセットは、**/sys/devices/system/cpu/aarch32_el0** ファイルによって示されます。また、ホットアンプラグ操作を制限できます。
詳細については、**Documentation/arm64/asymmetric-32bit.rst** を参照してください。

arm64.nomte = [ARM64]

このパラメーターを使用すると、Memory Tagging Extension (MTE) サポートを無条件に無効にすることができます。

i8042.probe_defer = [HW]

このパラメーターを使用すると、**i8042** プローブエラーで遅延プローブを許可できます。

idxd.tc_override = [HW]

このパラメーターを **<bool>** 形式で使用すると、デバイスのデフォルトのトラフィッククラス設定をオーバーライドできます。
デフォルト値は **false** (**0**) に設定されます。

kvm.eager_page_split = [KVM,X86]

このパラメーターを使用すると、KVM がダーティロギング中にすべてのヒュージページをプロアクティブに分割するかどうかを制御できます。Eager page splitting は、巨大なページを遅延して分割するために必要な書き込み保護障害とメモリー管理ユニット (MMU) ロックの競合を排除することで、vCPU 実行の中断を減らします。
書き込みをほとんど実行しない、または VM メモリーの小さな領域にのみ書き込む VM ワークロードは、熱心なページ分割を無効にして、巨大なページを引き続き読み取りに使用できるようにすることでメリットが得られます。

積極的なページ分割の動作は、**KVM_DIRTY_LOG_INITIALLY_SET** オプションが有効か無効かによって異なります。

- 無効にすると、その **memslot** でダーティロギングが有効になっているときに、**memslot** 内のすべての huge ページが積極的に分割されます。

- 有効にすると、**KVM_CLEAR_DIRTY ioctl()** システムコール中に、ページがクリアされている場合にのみ、熱心なページ分割が実行されます。
  イーガーページ分割は現在、2 次元ページング (TDP) MMU によってマップされた Huge Page の分割のみをサポートしています。

  デフォルト値は **Y** (**on**) に設定されています。

kvm.nx_huge_pages_recovery_period_ms = [KVM]

このパラメーターを使用すると、KVM が 4 KiB ページを huge ページにザッピングする期間を制御できます。

- 値がゼロ以外の **N** の場合、KVM はページの一部を **N** ミリ秒ごとにザッピングします。

- 値が **0** の場合、KVM は比率に基づいて期間を選択し、ページが平均 1 時間後にザッピングされるようにします。
  デフォルト値は **0** に設定されています。

l1d_flush = [X86,INTEL]

このパラメーターを使用して、L1D ベースのスヌーピングの脆弱性の軽減を制御できます。
特定の CPU は、特定の条件下で、開示ガジェットに情報を転送できる CPU 内部バッファーに対するエクスプロイトに対して脆弱です。脆弱なプロセッサーでは、攻撃者が直接アクセス権を持たないデータにアクセスするために、キャッシュ側のチャネル攻撃で、予測的に転送されるデータを利用できます。

利用可能なオプションは **on** です。これは **enable the interface for the mitigation** を意味します。

mmio_stale_data = [X86,INTEL]

このパラメーターを使用して、プロセッサーメモリーにマップされた I/O (MMIO) の古いデータの脆弱性の緩和を制御できます。
Processor MMIO Stale Data は、MMIO 操作後にデータを公開できる脆弱性のクラスです。公開されたデータは、メタデータサーバー (MDS) および Transactional Asynchronous Abort (TAA) の影響を受ける同じ CPU バッファーで開始または終了する可能性があります。そのため、MDS や TAA と同様に、影響を受ける CPU バッファーをクリアすることで軽減できます。

利用可能なオプションは以下のとおりです。

- **full**: 脆弱な CPU で軽減策を有効にします
- **full,nosmt**: 緩和策を有効にし、脆弱な CPU で SMT を無効にします。
- **off**: 無条件に緩和を無効にします
  MDS または TAA の影響を受けるマシンでは、アクティブな MDS または TAA の軽減策によって **mmio_stale_data=off** を防ぐことができます。これらの脆弱性は同じメカニズムで軽減されるからです。したがって、この軽減策を無効にするには、**mds=off** と **tsx_async_abort=off** も指定する必要があります。

  このオプションを指定しないことは、**mmio_stale_data=full** と同等です。

  詳細については、**Documentation/admin-guide/hw-vuln/processor_mmio_stale_data.rst** を参照してください。

random.trust_bootloader={on,off} = [KNL]

このパラメーターを使用すると、カーネルの CRNG を完全にシードするためにブートローダー (利用可能な場合) によって渡されるシードの使用を信頼することを有効または無効にすることができます。デフォルトの動作は、**CONFIG_RANDOM_TRUST_BOOTLOADER** オプションによって制御されます。

rcupdate.rcu_task_collapse_lim = [KNL]

このパラメーターを使用すると、猶予期間の開始時に存在するコールバックの最大数を設定できます。これにより、RCU タスクフレーバーが単一のコールバックキューを使用するように折りたたむことができます。この切り替えは、**rcupdate.rcu_task_enqueue_lim** オプションがデフォルト値の **-1** に設定されている場合にのみ発生します。

rcupdate.rcu_task_contend_lim = [KNL]

このパラメーターを使用すると、RCU タスクフレーバーを CPU ごとのコールバックキューイングに切り替えるために必要な jiffy あたりのコールバックキューイング時のロック競合イベントの最小数を設定できます。この切り替えは、**rcupdate.rcu_task_enqueue_lim** オプションがデフォルト値の **-1** に設定されている場合にのみ発生します。

rcupdate.rcu_task_enqueue_lim = [KNL]

このパラメーターを使用すると、RCU フレーバーの RCU タスクファミリーに使用するコールバックキューの数を設定できます。デフォルト値の **-1** を使用して、コールバックキューの数を自動的かつ動的に調整できます。
このパラメーターは、テストでの使用を目的としています。

retbleed = [X86]

このパラメーターを使用して、リターン命令による任意の投機的コード実行 (RETBleed) 脆弱性の緩和を制御できます。利用可能なオプションは以下のとおりです。

- **off**: 緩和なし
- **auto**: 軽減策を自動的に選択します
- **auto,nosmt**: 緩和策を自動的に選択し、必要に応じて完全な緩和策として SMT を無効にします (STIBP を使用しない Zen1 以前のみ)。
- **ibpb**: 基本ブロック境界での短い投機ウィンドウも軽減します。安全で最高のパフォーマンスへの影響。
- **unret**: トレーニングされていないリターンサンクを強制的に有効にします。AMD f15h-f17h ベースのシステムでのみ有効です。
- **unret,nosmt**: **unret** オプションと同様に、STIBP が利用できない場合に SMT を無効にします。
  auto オプションを選択すると、実行時に CPU に応じて緩和方法が選択されます。

  このオプションを指定しないことは、**retbleed=auto** と同等です。

sev=option[,option...] = [X86-64]

詳細については、**Documentation/x86/x86_64/boot-options.rst** を参照してください。

更新されたカーネルパラメーター

acpi_sleep = [HW,ACPI]

フォーマット: { s3_bios、s3_mode、s3_beep、s4_hwsig、s4_nohwsig、old_ordering、nonvs、sci_force_enable、nobl }

- **s3_bios** および **s3_mode** の詳細については、**Documentation/power/video.rst** を参照してください。
- **s3_beep** はデバッグ用です。カーネルのリアルモードエントリーポイントが呼び出されるとすぐに、PC のスピーカーからビープ音が鳴ります。
- **s4_hwsig** により、カーネルは休止状態からの再開中に ACPI ハードウェア署名をチェックし、変更されている場合は再開を適切に拒否します。デフォルトの動作は、**s4_hwsig** オプションが有効になっていない限り、再開を許可し、署名が変更されたときに単に警告することです。
- **s4_nohwsig** は、再開中に ACPI ハードウェア署名が使用されたり、警告されたりするのを防ぎます。**old_ordering** は、デバイスを低電力状態にすることに関して、**_PTS** 制御メソッドの ACPI 1.0 順序付けを強制します。デフォルトでは、**_PTS** の ACPI 2.0 順序が使用されます。

- **nonvs** は、カーネルがサスペンド、ハイバネーション、およびレジューム中に ACPI NVS メモリーを保存および復元するのを防ぎます。

- **sci_force_enable** により、カーネルは S1/S3 からの再開時に直接 **SCI_EN** を設定します。この動作は ACPI 仕様に反していますが、一部の破損したシステムはそれなしでは機能しません。

- **nobl** は、システムのサスペンドとレジュームに関して何らかの点で正しく動作しないことが知られているシステムの内部拒否リストを無視します。このオプションを賢く使用してください。
  詳細については、**Documentation/power/video.rst** を参照してください。

crashkernel=size[KMG],high = [KNL, X86-64, ARM64]

このパラメーターを使用すると、次のように物理メモリー領域を上から割り当てることができます。

- システムに 4 GB を超える RAM がインストールされている場合、物理メモリー領域は 4 GB を超える可能性があります。

- システムに 4 GB 未満の RAM がインストールされている場合、物理メモリー領域は 4 GB 未満に割り当てられます (利用可能な場合)。
  crashkernel=X パラメーターが指定されている場合、このパラメーターは無視されます。

crashkernel=size[KMG],low = [KNL, X86-64]

crashkernel=X,high を渡すと、カーネルは 4 GB を超える物理メモリー領域を割り当てることができます。これにより、ある程度の低メモリー (たとえば、**swiotlb** は少なくとも 64M+32K 低メモリーが必要) と、32 ビットデバイスの DMA バッファーが使い果たされないようにするのに十分な余分な低メモリーを必要とするシステムで、2 番目のカーネルクラッシュが発生します。カーネルは、4 GB 未満に少なくとも 256 M を自動的に割り当てようとします。このパラメーターを使用すると、代わりに 2 番目のカーネルに 4 GB 未満の範囲を指定できます。

- **0:** 低割り当てを無効にします。**crashkernel=X,high** が使用されていない場合、または予約済みメモリーが 4 GB 未満の場合は無視されます。

crashkernel=size[KMG],low = [KNL, ARM64]

このパラメーターを使用すると、クラッシュダンプカーネルの DMA ゾーンに低い範囲を指定できます。**crashkernel=X,high** が使用されていない場合、または予約済みメモリーが DMA ゾーンにある場合は無視されます。

kvm.nx_huge_pages_recovery_ratio = [KVM]

このパラメーターを使用すると、4 KiB ページを定期的に巨大なページにザッピングする回数を制御できます。

- **0** はリカバリーを無効にします

- **N** KVM は、4 KiB ページの **1/N** を周期ごとに消去します。
  デフォルトは **60** に設定されます。

kvm-arm.mode = [KVM,ARM]

このパラメーターを使用して、KVM 操作モードの 1 つを選択できます。

- **none**: KVM を強制的に無効にします。

- **nvhe**: 標準の nVHE ベースのモードで、保護されたゲストはサポートされていません。

- **protected**: 状態がホストから非公開に保たれているゲストをサポートする **nVHE** ベースのモード。カーネルが EL2 レベルで実行されている場合は無効です。
  デフォルト値は、ハードウェアサポートに基づいて **VHE**/**nVHE** に設定されています。

mitigations = [X86,PPC,S390,ARM64]

このパラメーターを使用すると、CPU の脆弱性に対するオプションの軽減策を制御できます。これは、厳選されたアーキテクチャーに依存しないオプションのセットであり、それぞれが既存のアーキテクチャー固有のオプションの集合体です。

- **off**: オプションの CPU 軽減策をすべて無効にします。これによりシステムパフォーマンスが向上しますが、ユーザーを複数の CPU の脆弱性にさらす可能性もあります。
  - 同等: **nopti X86,PPC**、**kpti=0 ARM64**、**nospectre_v1 X86,PPC**、**nobp=0 S390**、**nospectre_v2 X86,PPC,S390,ARM64**、**spectre_v2_user=off X86**、**spec_store_bypass_disable =off X86,PPC**、**ssbd=force-off ARM64**、**l1tf=off X86**、**mds=off X86**、**tsx_async_abort=off X86**、**kvm.nx_huge_pages=off X86**、**no_entry_flush PPC** 、**no_uaccess_flush PPC**、**mmio_stale_data=off X86**
  - 例外: **kvm.nx_huge_pages=force** オプションが指定されている場合、これは **kvm.nx_huge_pages** には影響しません。
- **auto** (デフォルト): すべての CPU の脆弱性を軽減しますが、脆弱であっても SMT を有効のままにします。
  - 同等: (デフォルトの動作)
- **auto,nosmt**: すべての CPU の脆弱性を軽減し、必要に応じて SMT を無効にします。
  - 以下と同等: **l1tf=flush,nosmt X86**、**mds=full,nosmt X86**、**tsx_async_abort=full,nosmt X86**、**mmio_stale_data=full,nosmt X86**

rcu_nocbs[=cpu-list] = [KNL]

オプションの引数は CPU リストです。
CONFIG_RCU_NOCB_CPU=y でビルドされたカーネルでは、no-callback CPU モードを有効にできます。これにより、そのような CPU コールバックが softirq コンテキストで呼び出されるのを防ぎます。そのような CPU の RCU コールバックの呼び出しは、代わりに、その目的のために作成された **rcuox**/**N kthreads** にオフロードされます。ここで、**x** は RCU-preempt を表す **p**、RCU-sched を表す **s**、猶予期間を仲介する **kthread** を表す **g** です。**N** は CPU 番号です。これにより、オフロードされた CPU の OS ジッターが減少し、HPC およびリアルタイムのワークロードに役立ちます。また、非対称マルチプロセッサーのエネルギー効率も改善できます。

- **cpulist** が引数として渡された場合、指定された CPU のリストはブートからコールバックなしモードに設定されます。
- **=** 記号と **cpulist** 引数を省略した場合、起動時に CPU がコールバックなしモードに設定されることはありませんが、**cpusets** を使用して実行時にモードを切り替えることができます。

rcutree.kthread_prio = [KNL,BOOT]

このパラメーターを使用すると、CPU ごとの RCU **kthreads** (**rcuc**/**N**) の **SCHED_FIFO** 優先度を設定できます。この値は、RCU ブーストスレッド (**rcub**/**N**) と RCU 猶予期間 **kthreads** (**rcu_bh**、**rcu_preempt**、および **rcu_sched**) の優先度にも使用されます。

- **RCU_BOOST** が設定されている場合、有効な値は 1 から 99 であり、デフォルトは **1** で、最も優先度の低い優先度です。

- **RCU_BOOST** が設定されていない場合、有効な値は 0 ~ 99 で、デフォルトは **0**、非リアルタイム操作です。
  RCU_NOCB_CPU が設定されている場合、**NOCB** コールバック **kthreads** の優先度を調整する必要があります。

rcutorture.fwd_progress = [KNL]

このパラメーターを使用すると、この概念をサポートするタイプの RCU の RCU 猶予期間の前方進行テストに使用する **kthreads** の数を指定できます。
デフォルトは **1 kthread** に設定されています。ゼロ未満の値または CPU の数より大きい値を指定すると、その数の CPU が使用されます。

spectre_v2 = [X86]

このパラメーターを使用すると、Spectre バリアント 2 (間接分岐スペキュレーション) の脆弱性の緩和を制御できます。デフォルトの操作は、カーネルをユーザー空間攻撃から保護します。

- **on**: 無条件に有効にし、**spectre_v2_user=on** を意味します
- **off**: 無条件に無効にし、**spectre_v2_user=off** を意味します
- **auto**: カーネルが CPU モデルが脆弱かどうかを検出します
- **on** を選択すると、CPU、利用可能なマイクロコード、**CONFIG_RETPOLINE** 設定オプション、カーネルがビルドされたコンパイラーに応じて、実行時に緩和方法が選択されます (**auto** も可能です)。
- **on** を選択すると、ユーザー空間からユーザー空間へのタスク攻撃に対する軽減も有効になります。
- **off** を選択すると、カーネルとユーザー空間の保護の両方が無効になります。
- 特定の軽減策を手動で選択することもできます。
  - **retpoline**: 間接ブランチを置き換えます
  - **retpoline,generic**: Retpolines
  - **retpoline、lfence**: LFENCE;間接分岐
  - **retpoline,amd**: retpoline,lfence のエイリアス
  - **eibrs**: 拡張 IBRS
  - **eibrs,retpoline**: 強化された IBRS + Retpolines
  - **eibrs,lfence**: 強化された IBRS + LFENCE
  - **ibrs**: IBRS を使用してカーネルを保護する
    このオプションを指定しないことは、**spectre_v2=auto** と同等です。

新しい sysctl パラメーター

max_rcu_stall_to_panic

panic_on_rcu_stall を **1** に設定すると、**panic()** が呼び出される前に RCU がストールできる回数が決まります。**panic_on_rcu_stall** を **0** に設定すると、この値は無効になります。

perf_user_access = [ARM64]

このパラメーターを使用すると、**perf** イベントカウンターを読み取るためのユーザー空間アクセスを制御できます。

- **1** に設定すると、ユーザー空間はパフォーマンスモニターカウンターレジスターを直接読み取ることができます。
- デフォルトは **0** に設定されており、これは **access disabled** であることを意味します。詳細については、**Documentation/arm64/perf.rst** を参照してください。

gro_normal_batch

このパラメーターを使用すると、GRO の出力でバッチ処理するセグメントの最大数を設定できます。パケットが結合されたスーパーフレームとして、または GRO が結合しないことを決定した元のパケットとして GRO を出ると、NAPI ごとのリストに配置されます。このリストは、セグメント数が **gro_normal_batch** 制限に達すると、スタックに渡されます。

high_order_alloc_disable

このパラメーターを使用すると、オーダー 0 の割り当てを選択できます。デフォルトでは、ページフラグメントのアロケータは高次ページ (X86 システムでは order-3) を使用しようとします。デフォルトの動作では良好な結果が返されますが、特定の状況では、ページの割り当てと解放で競合が発生します。これは、上位ページが CPU ごとのリストに格納されていない古いカーネル (バージョン 5.14 以降) で特に当てはまりました。このパラメーターは現在、主に歴史的に重要なものとして存在しています。
デフォルト値は **0** です。

page_lock_unfairness

このパラメーターの値を指定することにより、ウェイターからページロックを盗むことができる回数を決定できます。このファイルで指定された回数だけロックが盗まれた後、**fair lock handoff** セマンティクスが適用され、ロックを取得できる場合にのみウェイターが起動されます。
デフォルト値は **5** です。

変更された sysctl パラメーター

urandom_min_reseed_secs

このパラメーターを使用して、**urandom** プールの再シード間の最小秒数を決定できます。このファイルは互換性のために書き込み可能ですが、このファイルに書き込んでも RNG の動作には影響しません。

write_wakeup_threshold

エントロピーカウントがこのしきい値をビット単位で下回ると、**/dev/random** ファイルへの書き込みを待機しているプロセスを起動できます。このファイルは互換性のために書き込み可能ですが、このファイルに書き込んでも RNG の動作には影響しません。

第6章 デバイスドライバー

6.1. 新しいドライバー

ネットワークドライバー

- プラットフォームファームウェアランタイム 更新 テレメトリードライバー (**pfr_telemetry**)
- プラットフォームファームウェアランタイム更新デバイスドライバー (**pfr_update**)
- MediaTek デバイスバージョン 0.1 の Bluetooth サポート (**btmtk**)
- MHI ホストインターフェイス (**mhi**)
- Modem Host Interface (MHI) PCI コントローラードライバー**spectre_v2_user=auto**
- IDXD driver dsa_bus_type driver (**idxd_bus**)
- AMD PassThru DMA ドライバー (**ptdma**)
- Mellanox FAN ドライバー (**mlxreg-fan**)
- Mellanox LED regmap ドライバー (l **eds-mlxreg**)
- Intel® LPSS ACP ドライバー (**intel-lpss-acpi**)
- Intel® LPSS PCI ドライバー (**intel-lpss-pci**)
- Intel® LPSS コアドライバー (**intel-lpss**)
- Maxlinear イーサネット GPY ドライバー (**mxl-gpy**)
- Realtek 802.11ax ワイヤレス 8852A ドライバー (**rtw89_8852a**)
- Realtek 802.11ax ワイヤレス 8852AE ドライバー (**rtw89_8852ae**)
- Intel® PMT Class ドライバー (**pmt_class**)
- Intel® PMT Crashlog ドライバー (**pmt_crashlog**)
- Intel® PMT Telemetry ドライバー (**pmt_telemetry**)
- Intel® スピード選択インターフェイス mailbox ドライバー (**isst_if_mbox_msr**)
- Intel® speed select interface pci mailbox driver (**isst_if_mbox_pci**)
- Intel® speed select interface mmio driver (**isst_if_mmio**)
- Intel® Software Defined Silicon ドライバー (**intel_sdsi**)
- Intel® Extended Capabilities aux バスドライバー (**intel_vsec**)
- ISH ISHTP eclite クライアント OPERATION ドライバー (**ishtp_eclite**)
- Acer Wireless Radio Control Driver (**acer-wireless**)
- AMD HSMP Platform Interface Driver (**amd_hsmp**)

- DESIGNWARE HS OTG Core (**dwc2**)
- Synopsys HAPS PCI Glue Layer (**dwc3-haps**)
- DesignWare USB3 PCI Glue Layer (**dwc3-pci**)
- DesignWare USB3 DRD Controller Driver (**dwc3**)
- xHCI Platform Host Controller Driver (**xhci-plat-hcd**)
- ON Semiconductor FSA4480 ドライバー (**fsa4480**)
- Richtek RT1719 Sink Only USBPD Controller Driver (**rt1719**)
- Willsemi WUSB3801 Type-C ポートコントローラードライバー (**wusb3801**)
- VFIO ベース PCI デバイスのコアドライバー (**vfio-pci-core**)
- AMD SEV ゲストドライバー (**sev-guest**)
- Mellanox ウォッチドッグドライバー (**mlx_wdt**)

グラフィックドライバーとその他のドライバー

- Cirrus Logic DSP サポート (**cs_dsp**)
- DRM DisplayPort ヘルパー (**drm_dp_helper**)
- DRM Buddy Allocator (**drm_buddy**)
- DRM SHMEM メモリ-マネジメントヘルパー (**drm_shmem_helper**)
- bochs dispi インターフェイスを使用する DRM ドライバー (**bochs**)
- Letsketch タブレットドライバー (**hid-letsketch**)
- Intel® スピードセレクトインターフェイスドライバー (**isst_if_common**)
- SiGma Micro HID ドライバー (**hid-sigmamicro**)
- Xiaomi Mi Silent Mouse (**hid-xiaomi**) のサイドボタンの修正
- DEC VSXXX-AA および -GA マウスと VSXXX-AB タブレット用のドライバー (**vsxxxaa**)
- Nvidia ラインカードプラットフォームドライバー (**mlxreg-lc**)
- Intel PCH Thermal ドライバー (**intel_pch_thermal**)
- Intel LPSS UART ドライバー (**8250_lpss**)

6.2. 更新されたドライバー

ネットワークドライバーの更新

- VMware vmxnet3 virtual NIC ドライバー (**vmxnet3**) がバージョン 1.7.0.0-k に更新されました。

ストレージドライバーの更新

- Emulex LightPulse Fibre Channel SCSI ドライバー (**lpfc**) がバージョン 14.2.0.5 に更新されました。
- MPI3 Storage Controller Device Driver (**mpi3mr**) がバージョン 8.0.0.69.0 に更新されました。
- LSI MPT Fusion SAS 3.0 デバイスドライバー (**mpt3sas**) がバージョン 40.100.00.00 に更新されました。
- Microchip Smart Family Controller (**smartpqi**) のドライバーがバージョン 2.1.18-045 に更新されました。

グラフィックおよびその他ドライバーの更新

- VMware SVGA デバイス (**vmwgfx**) のスタンドアロン drm ドライバーがバージョン 2.20.0.0 に更新されました。

第7章 利用可能な BPF 機能

この章では、Red Hat Enterprise Linux 9 のこのマイナーバージョンのカーネルで利用可能な **Berkeley Packet Filter** (**BPF**) 機能の完全なリストを提供します。表には次のリストが含まれます。

- システム設定とその他のオプション
- 利用可能なプログラムの種類とサポートされているヘルパー
- 利用可能なマップの種類

この章には、**bpftool feature** コマンドの自動生成された出力が含まれています。

表7.1 システム設定とその他のオプション

オプション	値
unprivileged_bpf_disabled	2 (特権ユーザーに限定された bpf() syscall、管理者は変更可能)
JIT コンパイラー	1 (有効)
JIT コンパイラーの強化	1 (権限のないユーザーに対して有効)
JIT コンパイラー kallsyms エクスポート	1 (ルートで有効)
非特権ユーザーの JIT のメモリー制限	264241152
CONFIG_BPF	y
CONFIG_BPF_SYSCALL	y
CONFIG_HAVE_EBPF_JIT	y
CONFIG_BPF_JIT	y
CONFIG_BPF_JIT_ALWAYS_ON	y
CONFIG_DEBUG_INFO_BTF	y
CONFIG_DEBUG_INFO_BTF_MODULES	y
CONFIG_CGROUPS	y
CONFIG_CGROUP_BPF	y
CONFIG_CGROUP_NET_CLASSID	y
CONFIG_SOCK_CGROUP_DATA	y

オプション	値
CONFIG_BPF_EVENTS	y
CONFIG_KPROBE_EVENTS	y
CONFIG_UPROBE_EVENTS	y
CONFIG_TRACING	y
CONFIG_FTRACE_SYSCALLS	y
CONFIG_FUNCTION_ERROR_INJECTION	y
CONFIG_BPF_KPROBE_OVERRIDE	n
CONFIG_NET	y
CONFIG_XDP_SOCKETS	y
CONFIG_LWTUNNEL_BPF	y
CONFIG_NET_ACT_BPF	m
CONFIG_NET_CLS_BPF	m
CONFIG_NET_CLS_ACT	y
CONFIG_NET_SCH_INGRESS	m
CONFIG_XFRM	y
CONFIG_IP_ROUTE_CLASSID	y
CONFIG_IPV6_SEG6_BPF	n
CONFIG_BPF_LIRC_MODE2	n
CONFIG_BPF_STREAM_PARSER	y
CONFIG_NETFILTER_XT_MATCH_BPF	m
CONFIG_BPFILTER	n
CONFIG_BPFILTER_UMH	n

オプション	値
CONFIG_TEST_BPF	m
CONFIG_HZ	1000
bpf() syscall	available
大きなプログラムサイズの制限	available

表7.2 利用可能なプログラムの種類とサポートされているヘルパー

プログラムの種類	利用可能なヘルパー
socket_filter	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_perf_event_output, bpf_skb_load_bytes, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_skb_load_bytes_relative, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
kprobe	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_perf_event_read, bpf_perf_event_output, bpf_get_stackid, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_get_numa_node_id, bpf_probe_read_str, bpf_perf_event_read_value, bpf_get_stack, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_send_signal, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_send_signal_thread, bpf_jiffies64, bpf_get_ns_current_pid_tgid, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_get_task_stack, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_get_attach_cookie, bpf_task_pt_regs, bpf_get_branch_snapshot

プログラムの種類	利用可能なヘルパー
sched_cls	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_skb_store_bytes, bpf_l3_csum_replace, bpf_l4_csum_replace, bpf_tail_call, bpf_clone_redirect, bpf_get_cgroup_classid, bpf_skb_vlan_push, bpf_skb_vlan_pop, bpf_skb_get_tunnel_key, bpf_skb_set_tunnel_key, bpf_redirect, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_get_tunnel_opt, bpf_skb_set_tunnel_opt, bpf_skb_change_proto, bpf_skb_change_type, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_change_tail, bpf_skb_pull_data, bpf_csum_update, bpf_set_hash_invalid, bpf_get_numa_node_id, bpf_skb_change_head, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_set_hash, bpf_skb_adjust_room, bpf_skb_get_xfrm_state, bpf_skb_load_bytes_relative, bpf_fib_lookup, bpf_skb_cgroup_id, bpf_skb_ancestor_cgroup_id, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_sk_fullsock, bpf_tcp_sock, bpf_skb_ecn_set_ce, bpf_get_listener_sock, bpf_skc_lookup_tcp, bpf_tcp_check_syncookie, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_tcp_gen_syncookie, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_sk_assign, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_csum_level, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_skb_cgroup_classid, bpf_redirect_neigh, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_redirect_peer, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_check_mtu, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock

プログラムの種類	利用可能なヘルパー
sched_act	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_skb_store_bytes, bpf_l3_csum_replace, bpf_l4_csum_replace, bpf_tail_call, bpf_clone_redirect, bpf_get_cgroup_classid, bpf_skb_vlan_push, bpf_skb_vlan_pop, bpf_skb_get_tunnel_key, bpf_skb_set_tunnel_key, bpf_redirect, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_get_tunnel_opt, bpf_skb_set_tunnel_opt, bpf_skb_change_proto, bpf_skb_change_type, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_change_tail, bpf_skb_pull_data, bpf_csum_update, bpf_set_hash_invalid, bpf_get_numa_node_id, bpf_skb_change_head, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_set_hash, bpf_skb_adjust_room, bpf_skb_get_xfrm_state, bpf_skb_load_bytes_relative, bpf_fib_lookup, bpf_skb_cgroup_id, bpf_skb_ancestor_cgroup_id, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_sk_fullsock, bpf_tcp_sock, bpf_skb_ecn_set_ce, bpf_get_listener_sock, bpf_skc_lookup_tcp, bpf_tcp_check_syncookie, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_tcp_gen_syncookie, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_sk_assign, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_csum_level, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_skb_cgroup_classid, bpf_redirect_neigh, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_redirect_peer, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_check_mtu, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
tracepoint	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_perf_event_read, bpf_perf_event_output, bpf_get_stackid, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_get_numa_node_id, bpf_probe_read_str, bpf_perf_event_read_value, bpf_get_stack, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_send_signal, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_send_signal_thread, bpf_jiffies64, bpf_get_ns_current_pid_tgid, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_get_task_stack, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_get_attach_cookie, bpf_task_pt_regs, bpf_get_branch_snapshot

プログラムの種類	利用可能なヘルパー
xdp	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_redirect, bpf_perf_event_output, bpf_csum_diff, bpf_get_current_task, bpf_get_numa_node_id, bpf_xdp_adjust_head, bpf_redirect_map, bpf_xdp_adjust_meta, bpf_xdp_adjust_tail, bpf_fib_lookup, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_skc_lookup_tcp, bpf_tcp_check_syncookie, bpf_tcp_gen_syncookie, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_check_mtu, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
perf_event	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_perf_event_read, bpf_perf_event_output, bpf_get_stackid, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_get_numa_node_id, bpf_probe_read_str, bpf_perf_event_read_value, bpf_perf_prog_read_value, bpf_get_stack, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_send_signal, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_send_signal_thread, bpf_jiffies64, bpf_read_branch_records, bpf_get_ns_current_pid_tgid, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_get_task_stack, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_get_attach_cookie, bpf_task_pt_regs, bpf_get_branch_snapshot

プログラムの種類	利用可能なヘルパー
cgroup_skb	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_perf_event_output, bpf_skb_load_bytes, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_skb_load_bytes_relative, bpf_skb_cgroup_id, bpf_get_local_storage, bpf_skb_ancestor_cgroup_id, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_sk_fullsock, bpf_tcp_sock, bpf_skb_ecn_set_ce, bpf_get_listener_sock, bpf_skc_lookup_tcp, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_sk_cgroup_id, bpf_sk_ancestor_cgroup_id, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
cgroup_sock	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_get_cgroup_classid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_sk_storage_get, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_get_netns_cookie, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs
lwt_in	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_cgroup_classid, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_pull_data, bpf_get_numa_node_id, bpf_lwt_push_encap, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock

プログラムの種類	利用可能なヘルパー
lwt_out	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_cgroup_classid, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_pull_data, bpf_get_numa_node_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
lwt_xmit	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_skb_store_bytes, bpf_l3_csum_replace, bpf_l4_csum_replace, bpf_tail_call, bpf_clone_redirect, bpf_get_cgroup_classid, bpf_skb_get_tunnel_key, bpf_skb_set_tunnel_key, bpf_redirect, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_get_tunnel_opt, bpf_skb_set_tunnel_opt, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_change_tail, bpf_skb_pull_data, bpf_csum_update, bpf_set_hash_invalid, bpf_get_numa_node_id, bpf_skb_change_head, bpf_lwt_push_encap, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_csum_level, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock

プログラムの種類	利用可能なヘルパー
sock_ops	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_setsockopt, bpf_sock_map_update, bpf_getsockopt, bpf_sock_ops_cb_flags_set, bpf_sock_hash_update, bpf_get_local_storage, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_tcp_sock, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_get_netns_cookie, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_load_hdr_opt, bpf_store_hdr_opt, bpf_reserve_hdr_opt, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
sk_skb	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_skb_store_bytes, bpf_tail_call, bpf_perf_event_output, bpf_skb_load_bytes, bpf_get_current_task, bpf_skb_change_tail, bpf_skb_pull_data, bpf_get_numa_node_id, bpf_skb_change_head, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_skb_adjust_room, bpf_sk_redirect_map, bpf_sk_redirect_hash, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_skc_lookup_tcp, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
cgroup_device	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_uid_gid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs

プログラムの種類	利用可能なヘルパー
sk_msg	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_cgroup_classid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_msg_redirect_map, bpf_msg_apply_bytes, bpf_msg_cork_bytes, bpf_msg_pull_data, bpf_msg_redirect_hash, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_msg_push_data, bpf_msg_pop_data, bpf_spin_lock, bpf_spin_unlock, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_get_netns_cookie, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
raw_tracepoint	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_perf_event_read, bpf_perf_event_output, bpf_get_stackid, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_get_numa_node_id, bpf_probe_read_str, bpf_perf_event_read_value, bpf_get_stack, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_send_signal, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_send_signal_thread, bpf_jiffies64, bpf_get_ns_current_pid_tgid, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_get_task_stack, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_task_pt_regs, bpf_get_branch_snapshot

プログラムの種類	利用可能なヘルパー
cgroup_sock_addr	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_get_cgroup_classid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_setsockopt, bpf_getsockopt, bpf_bind, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_skc_lookup_tcp, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_get_netns_cookie, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
lwt_seg6local	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_cgroup_classid, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_csum_diff, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_skb_pull_data, bpf_get_numa_node_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
lirc_mode2	サポート対象外
sk_reuseport	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_skb_load_bytes, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_socket_cookie, bpf_skb_load_bytes_relative, bpf_sk_select_reuseport, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs

プログラムの種類	利用可能なヘルパー
flow_dissector	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_skb_load_bytes, bpf_get_current_task, bpf_get_numa_node_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock
cgroup_sysctl	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_uid_gid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_sysctl_get_name, bpf_sysctl_get_current_value, bpf_sysctl_get_new_value, bpf_sysctl_set_new_value, bpf_strtol, bpf_strtoul, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs
raw_tracepoint_writable	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_perf_event_read, bpf_perf_event_output, bpf_get_stackid, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_get_numa_node_id, bpf_probe_read_str, bpf_perf_event_read_value, bpf_get_stack, bpf_get_current_cgroup_id, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_send_signal, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_send_signal_thread, bpf_jiffies64, bpf_get_ns_current_pid_tgid, bpf_get_current_ancestor_cgroup_id, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_get_task_stack, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_task_pt_regs, bpf_get_branch_snapshot

プログラムの種類	利用可能なヘルパー
cgroup_sockopt	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_get_current_uid_gid, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_tcp_sock, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_get_netns_cookie, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs
tracing	サポート対象外

プログラムの種類	利用可能なヘルパー
struct_ops	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_probe_read, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_skb_store_bytes, bpf_l3_csum_replace, bpf_l4_csum_replace, bpf_tail_call, bpf_clone_redirect, bpf_get_current_pid_tgid, bpf_get_current_uid_gid, bpf_get_current_comm, bpf_get_cgroup_classid, bpf_skb_vlan_push, bpf_skb_vlan_pop, bpf_skb_get_tunnel_key, bpf_skb_set_tunnel_key, bpf_perf_event_read, bpf_redirect, bpf_get_route_realm, bpf_perf_event_output, bpf_skb_load_bytes, bpf_get_stackid, bpf_csum_diff, bpf_skb_get_tunnel_opt, bpf_skb_set_tunnel_opt, bpf_skb_change_proto, bpf_skb_change_type, bpf_skb_under_cgroup, bpf_get_hash_recalc, bpf_get_current_task, bpf_current_task_under_cgroup, bpf_skb_change_tail, bpf_skb_pull_data, bpf_csum_update, bpf_set_hash_invalid, bpf_get_numa_node_id, bpf_skb_change_head, bpf_xdp_adjust_head, bpf_probe_read_str, bpf_get_socket_cookie, bpf_get_socket_uid, bpf_set_hash, bpf_setsockopt, bpf_skb_adjust_room, bpf_redirect_map, bpf_sk_redirect_map, bpf_sock_map_update, bpf_xdp_adjust_meta, bpf_perf_event_read_value, bpf_perf_prog_read_value, bpf_getsockopt, bpf_override_return, bpf_sock_ops_cb_flags_set, bpf_msg_redirect_map, bpf_msg_apply_bytes, bpf_msg_cork_bytes, bpf_msg_pull_data, bpf_bind, bpf_xdp_adjust_tail, bpf_skb_get_xfrm_state, bpf_get_stack, bpf_skb_load_bytes_relative, bpf_fib_lookup, bpf_sock_hash_update, bpf_msg_redirect_hash, bpf_sk_redirect_hash, bpf_lwt_push_encap, bpf_lwt_seg6_store_bytes, bpf_lwt_seg6_adjust_srh, bpf_lwt_seg6_action, bpf_rc_repeat, bpf_rc_keydown, bpf_skb_cgroup_id, bpf_get_current_cgroup_id, bpf_get_local_storage, bpf_sk_select_reuseport, bpf_skb_ancestor_cgroup_id, bpf_sk_lookup_tcp, bpf_sk_lookup_udp, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_msg_push_data, bpf_msg_pop_data, bpf_rc_pointer_rel, bpf_spin_lock, bpf_spin_unlock, bpf_sk_fullsock, bpf_tcp_sock, bpf_skb_ecn_set_ce, bpf_get_listener_sock, bpf_skc_lookup_tcp, bpf_tcp_check_syncookie, bpf_sysctl_get_name, bpf_sysctl_get_current_value, bpf_sysctl_get_new_value, bpf_sysctl_set_new_value, bpf_strtol, bpf_strtoul, bpf_sk_storage_get, bpf_sk_storage_delete, bpf_send_signal, bpf_tcp_gen_syncookie, bpf_skb_output, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_tcp_send_ack, bpf_send_signal_thread, bpf_jiffies64, bpf_read_branch_records, bpf_get_ns_current_pid_tgid, bpf_xdp_output, bpf_get_netns_cookie, bpf_get_current_ancestor_cgroup_id, bpf_sk_assign, bpf_ktime_get_boot_ns, bpf_seq_printf, bpf_seq_write, bpf_sk_cgroup_id, bpf_sk_ancestor_cgroup_id, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_csum_level, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_get_task_stack, bpf_load_hdr_opt, bpf_store_hdr_opt, bpf_reserve_hdr_opt, bpf_inode_storage_get, bpf_inode_storage_delete, bpf_d_path, bpf_copy_from_user, bpf_snprintf_btf, bpf_seq_printf_btf, bpf_skb_cgroup_classid, bpf_redirect_neigh, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_redirect_peer, bpf_task_storage_get, bpf_task_storage_delete, bpf_get_current_task_btf, bpf_bprm_opts_set, bpf_ktime_get_coarse_ns, bpf_ima_inode_hash, bpf_sock_from_file, bpf_check_mtu, bpf_for_each_map_elem, bpf_snprintf, bpf_sys_bpf, bpf_btf_find_by_name_kind, bpf_sys_close, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_get_func_ip, bpf_get_attach_cookie, bpf_task_pt_regs, bpf_get_branch_snapshot, bpf_skc_to_unix_sock, bpf_kallsyms_lookup_name

プログラムの種類	利用可能なヘルパー
ext	サポート対象外
lsm	サポート対象外
sk_lookup	bpf_map_lookup_elem, bpf_map_update_elem, bpf_map_delete_elem, bpf_ktime_get_ns, bpf_get_prandom_u32, bpf_get_smp_processor_id, bpf_tail_call, bpf_perf_event_output, bpf_get_current_task, bpf_get_numa_node_id, bpf_sk_release, bpf_map_push_elem, bpf_map_pop_elem, bpf_map_peek_elem, bpf_spin_lock, bpf_spin_unlock, bpf_probe_read_user, bpf_probe_read_kernel, bpf_probe_read_user_str, bpf_probe_read_kernel_str, bpf_jiffies64, bpf_sk_assign, bpf_ktime_get_boot_ns, bpf_ringbuf_output, bpf_ringbuf_reserve, bpf_ringbuf_submit, bpf_ringbuf_discard, bpf_ringbuf_query, bpf_skc_to_tcp6_sock, bpf_skc_to_tcp_sock, bpf_skc_to_tcp_timewait_sock, bpf_skc_to_tcp_request_sock, bpf_skc_to_udp6_sock, bpf_snprintf_btf, bpf_per_cpu_ptr, bpf_this_cpu_ptr, bpf_get_current_task_btf, bpf_ktime_get_coarse_ns, bpf_for_each_map_elem, bpf_snprintf, bpf_timer_init, bpf_timer_set_callback, bpf_timer_start, bpf_timer_cancel, bpf_task_pt_regs, bpf_skc_to_unix_sock

表7.3 利用可能なマップの種類

マップの種類	Available
ハッシュ	はい
array	はい
prog_array	はい
perf_event_array	はい
percpu_hash	はい
percpu_array	はい
stack_trace	はい
cgroup_array	はい
lru_hash	はい
lru_percpu_hash	はい
lpm_trie	はい
array_of_maps	はい

マップの種類	Available
hash_of_maps	はい
devmap	はい
sockmap	はい
cpumap	はい
xskmap	はい
sockhash	はい
cgroup_storage	はい
reuseport_sockarray	はい
percpu_cgroup_storage	はい
queue	はい
stack	はい
sk_storage	はい
devmap_hash	はい
struct_ops	いいえ
ringbuf	はい
inode_storage	はい
task_storage	はい

第8章 バグ修正

このパートでは、Red Hat Enterprise Linux 9.1 で修正された、ユーザーに重大な影響を与えるバグについて説明します。

8.1. インストーラーおよびイメージの作成

インストーラーは以前のバージョンのパッケージをインストールしなくなりました

以前は、インストーラーはインストールプロセス中に DNF 設定ファイルを正しくロードしませんでした。その結果、インストーラーは、RPM トランザクションで選択したパッケージの以前のバージョンをインストールすることがありました。

このバグは修正され、最新バージョンのパッケージのみがインストールリポジトリーからインストールされるようになりました。パッケージの最新バージョンをインストールできない場合、インストールは想定どおり失敗します。

(BZ#2053710)

stage2 でネットワーク設定を変更しても Anaconda のインストールは成功します

以前は、**rd.live.ram** ブート引数を使用すると、Anaconda は **initramfs** でメモリーにインストールイメージをフェッチするために使用される NFS マウントポイントをアンマウントしませんでした。その結果、stage2 でネットワーク設定が変更された場合、インストールプロセスが応答しなくなるか、タイムアウトエラーで失敗する可能性がありました。

この問題を修正するために、インストールイメージをメモリーにフェッチするために使用される NFS マウントポイントは、switchroot の前に **initramfs** でアンマウントされます。その結果、インストールプロセスは中断することなく完了します。

(BZ#2082132)

8.2. サブスクリプションの管理

FIPS モードで、 virt-who が ESX サーバーに正しく接続するようになりました

以前は、FIPS モードの RHEL 9 システムで **virt-who** ユーティリティーを使用すると、**virt-who** は ESX サーバーに接続できませんでした。その結果、**virt-who** は、設定されていても ESX サーバーを報告せず、以下のエラーメッセージをログに記録しました。

```
ValueError: [digital envelope routines] unsupported
```

今回の更新により、**virt-who** は FIPS モードを正しく処理するように修正され、前述の問題は発生しなくなりました。

(BZ#2054504)

8.3. ソフトウェア管理

DNF は、 Reason Change Action タイプのアイテムを含むトランザクションを正しくロールバックするようになりました

以前は、**Reason Change** Action タイプの項目を含むトランザクションで **dnf history rollback** コマンドを実行すると失敗しました。今回の更新で問題が修正され、**dnf 履歴のロールバック** が期待どおりに機能するようになりました。

(BZ#2053014)

8.4. シェルおよびコマンドラインツール

ReaR の vi コマンドで無限ループが発生しなくなりました

以前は、ReaR レスキューシステムには **vi** 実行可能ファイルが含まれておらず、**/bin/vi** スクリプトのみが含まれていました。その結果、**/bin/vi** スクリプトが呼び出されると、無限ループが発生しました。今回の更新により、ReaR レスキューシステムには実際の **vi** 実行可能ファイル **/usr/libexec/vi** が含まれるようになり、**vi** コマンドを実行しても無限ループが発生しなくなりました。

(BZ#2097437)

PXE 出力メソッドを使用する ReaR は、出力ファイルを rsync OUTPUT_URL 内に出力ファイルを保存する際に失敗しなくなりました

以前のリリースでは、**OUTPUT=PXE** および **BACKUP=RSYNC** オプションを使用した **OUTPUT_URL** 変数の処理が削除されていました。その結果、**OUTPUT_URL** に rsync の場所を使用すると、ReaR は **initrd** およびカーネルファイルをこの場所にコピーできず、それらは **BACKUP_URL** で指定された場所にアップロードされました。今回の更新により、RHEL 8.4 以前のリリースの動作が復元されました。ReaR は、rsync を使用して、指定された **OUTPUT_URL** の宛先に必要なファイルを作成します。

(BZ#2115958)

/etc/fstab で UUID を更新しない場合、ReaR はエラーメッセージの表示に失敗しなくなりました

これまで ReaR では、Universally Unique Identifier (UUID) が異なる場合に **/etc/fstab** の UUID を新しく作成されたパーティションの UUID と一致するように更新できなかった場合、リカバリー中にエラーメッセージを表示しませんでした。これは、レスキューイメージがバックアップと同期していない場合に発生する可能性がありました。今回の更新により、復元された基本システムファイルが再作成されたシステムと一致しない場合、リカバリー中にエラーメッセージが表示されます。

(BZ#2083272)

ReaR は、NetBackup バージョン 9 を使用したシステムの復元をサポートするようになりました

以前は、NetBackup バージョン 9 以降で NetBackup (NBU) メソッドを使用してシステムを復元すると、ライブラリーやその他のファイルが見つからないために失敗していました。今回の更新により、**NBU_LD_LIBRARY_PATH** 変数に必要なライブラリーパスが含まれ、レスキューシステムに必要なファイルが組み込まれるようになり、ReaR は NetBackup メソッドを使用できるようになりました。

(BZ#2120736)

ReaR では、シンボリックリンクターゲットの欠落に関する誤ったエラーメッセージが表示されなくなりました

以前のリリースでは、ReaR はレスキューイメージの作成時に、**/usr/lib/modules/** の **build** および **source** シンボリックリンクのターゲットがないことを示す誤ったエラーメッセージを表示していました。この状況は安全性に問題がないため、エラーメッセージを無視しても問題はありませんでした。今回の更新により、この状況で ReaR は、シンボリックリンクターゲットが欠落していることを示す誤ったエラーメッセージを報告しなくなりました。

(BZ#2119501)

パラメーターのない cmx 操作により CIM クライアントがクラッシュしなくなりました

cmx 操作はメソッドを呼び出し、XML を返します。パラメーターは呼び出されたメソッドの名前を指定します。以前は、追加のパラメーターなしで **cmx** 操作を実行すると、コマンドライン **sblim-wbemcli** Common Information Model (CIM) クライアントがクラッシュしていました。今回の更新により、**cmx** 操作には、呼び出されたメソッドの名前を定義するパラメーターが必要になりました。このパラメーターなしで **cmx** 操作を呼び出すとエラーメッセージが表示され、CIM クライアントはクラッシュしなくなりました。

(BZ#2083577)

free コマンドは、使用済みメモリーに新しい計算方法を使用します

以前は、**free** ユーティリティーで使用されたメモリーの計算では、合計メモリーから空き領域、キャッシュ領域、バッファー領域を減産していました。そのため、使用していたメモリーと他のツールの結果を比較すると、**free** ユーティリティーが共有メモリーを計算しないことで不一致が生じました。今回の更新で、**free** コマンドは新しい計算メソッドを使用して要求不可能なキャッシュも考慮に入れ、空きメモリーの状態を明確に示すようになりました。使用されるメモリーは利用不可メモリーとなり、これには仮想メモリーにある **tmpfs** オブジェクトも含まれます。

(BZ#2003033)

8.5. インフラストラクチャーサービス

Unbound は SHA-1 ベースの RSA 署名を検証しなくなりました

以前は、OpenSSL は DEFAULT システム全体の暗号化ポリシーで SHA-1 ベースの RSA 署名を検証しませんでした。その結果、Unbound がそのような署名を検証しようとすると、OpenSSL からのエラーにより解決が失敗しました。今回の更新で、Unbound はすべての RSA/SHA1 (アルゴリズム番号 5) および RSASHA1-NSEC3-SHA1 (アルゴリズム番号 7) 署名の検証サポートを無効にし、クエリーを解決します。これにより、すべてのシステム全体の暗号化ポリシーの下で結果が安全でなくなることに注意してください。

(BZ#2071543)

8.6. セキュリティー

OpenSSH キーの生成は FIPS 互換インターフェイスを使用します

OpenSSH で使用される OpenSSL 暗号化ライブラリーは、レガシーとモダンの 2 つのインターフェイスを提供します。以前は、OpenSSH はキー生成にレガシーインターフェイスを使用していましたが、これは連邦情報処理標準 (FIPS) 要件に準拠していませんでした。今回の更新で、**ssh-keygen** ユーティリティーは、低レベルの FIPS 非互換 API ではなく、FIPS 準拠 API を使用するようになりました。そのため、OpenSSH キーの生成は FIPS に準拠しています。

(BZ#2087121)

FIPS で承認されていない暗号化が FIPS モードの OpenSSL で機能しなくなりました

これまで、FIPS 承認されていない暗号は、システム設定に関係なく OpenSSL ツールキットで機能していました。したがって、システムが FIPS モードで実行している場合に無効にする必要のある暗号化アルゴリズムと暗号を、以下の例のように使用できていました。

- RSA 鍵交換を使用する TLS 暗号化スイートが機能する。
- 公開鍵の暗号化および復号化用の RSA ベースのアルゴリズムは、PKCS#1 および SSLv23 パディングを使用したり、2048 ビットより短い鍵を使用したりしても機能する。

この更新では、FIPS によって承認されていない暗号化が FIPS モードの OpenSSL で動作しないように修正が含まれています。

(BZ#2053289)

任意の曲線を指定できるオプションを OpenSSL から削除しました

以前は、明示的な曲線パラメーターの安全性のチェックが不完全でした。そのため、**p** 値が十分な大きさの任意の楕円曲線が RHEL で機能していました。今回の更新で、チェックで明示的な曲線パラメーターがよく知られている曲線のいずれかと一致するか検証できるようになりました。その結果、明示的な曲線パラメーターを使用して任意の曲線を指定するオプションが OpenSSL から削除されました。任意の明示的曲線を指定するパラメーターファイル、秘密鍵、公開鍵、および証明書は OpenSSL では機能しなくなります。明示的な曲線パラメーターを使用して、P-224、P-256、P-384、P-521、**secp256k1** などのよく知られている曲線のいずれかを指定した場合も、FIPS 以外のモードで引き続きサポートされます。

(BZ#2066412)

OpenSSL req は、秘密鍵の暗号化に AES-256-CBC を使用します

以前は、OpenSSL **req** ツールは、3DES アルゴリズムを使用して秘密鍵ファイルを暗号化していました。3DES アルゴリズムは安全ではなく、暗号化モジュールの現在の FIPS 140 規格では使用できないため、**req** は代わりに AES-256-CBC アルゴリズムを使用して暗号化された秘密鍵ファイルを生成します。全体的な PKCS#8 ファイル形式は変更されません。

(BZ#2063947)

FFDHE の使用時に OpenSSL が接続に失敗しなくなりました

以前は、有限フィールドベースの Diffie-Hellman ephemeral (FFDHE) 鍵交換メカニズムを使用する TLS 接続は、クライアントから FFDHE 鍵共有を処理するときに失敗することがありました。これは、OpenSSL のチェックが過度に制限されていることが原因でした。その結果、OpenSSL サーバーは **internal_error** アラートで接続を中止しました。今回の更新により、OpenSSL はより小さくても準拠したクライアントキー共有を受け入れるようになりました。その結果、OpenSSL と他の実装の間の接続は、FFDHE 鍵交換を使用しているときにランダムに中止されなくなりました。

(BZ#2004915)

OpenSSL ベースのアプリケーションが、Turkish ロケールで正しく動作するようになりました。

OpenSSL ライブラリーは大文字と小文字を区別しない文字列比較関数を使用するため、OpenSSL ベースのアプリケーションはトルコ語ロケールで正しく機能せず、チェックを省略すると、このロケールを使用するアプリケーションがクラッシュしました。この更新プログラムは、大文字と小文字を区別しない文字列比較のために Portable Operating System Interface (POSIX) ロケールを使用するためのパッチを提供します。その結果、curl などの OpenSSL ベースのアプリケーションはトルコ語のロケールで正しく機能します。

(BZ#2071631)

SELinux ポリシーに insights-client のパーミッションが追加されました

新しい **insights-client** サービスには、以前の **selinux-policy** バージョンにはないパーミッションが必要です。そのため、**insights-client** の一部のコンポーネントが正しく機能せず、アクセスベクターキャッシュ (AVC) エラーメッセージを報告していました。今回の更新で、SELinux ポリシーに新しいパーミッションが追加されました。その結果、AVC エラーを報告せずに **insights-client** が正常に実行されます。

(BZ#2081425、BZ#2077377、BZ#2087765、BZ#2107363)

SELinux staff_u ユーザーは誤って unconfined_rに誤って切り替えることができなくなりました

以前は、**secure_mode** ブール値を有効にすると、**staff_u** ユーザーが **unconfined_r** ロールに切り替えることができましたが、これは想定外の動作でした。これにより、**staff_u** ユーザーは、システムのセキュリティーに影響する特権操作を実行できました。今回の更新で、SELinux ポリシーが修正され、**staff_u** ユーザーは誤って **unconfined_r** に切り替えることができなくなりました。

(BZ#2076681)

OpenSCAP が使用可能なメモリーをチェックするときに誤ったエラーを生成しなくなりました

以前は、一部の XCCDF ルールを評価するときに、OpenSCAP がエラーメッセージ **Failed to check available memory** を誤って表示し、無効なスキャン結果を生成していました。たとえば、これはルール **accounts_user_dot_no_world_writable_programs**、**accounts_user_dot_group_ownership**、**accounts_users_home_files_permissions** で発生しました。今回の更新で、エラー処理のバグが修正され、エラーメッセージは実際の障害に対してのみ表示されるようになりました。

(BZ#2109485)

fagenrules --load が正常に動作するようになりました

以前は、**fapolicyd** サービスはシグナルのハングアップ (SIGHUP) を正しく処理しませんでした。そのため、**fapolicyd** は SIGHUP を受信した後に終了し、**fagenrules --load** コマンドが正しく機能しませんでした。この更新では、その問題が修正されました。その結果、**fagenrules --load** が正常に機能し、ルール更新時に **fapolicyd** を手動で再起動する必要がなくなりました。

(BZ#2070655)

8.7. ネットワーク

インスタンスは、Alibaba Cloud で nm-cloud-setup サービスを開始した後もプライマリー IP アドレスを保持するようになりました。

以前は、Alibaba Cloud でインスタンスを起動すると、**nm-cloud-setup** サービスは、複数の IPv4 アドレスがある場合、誤った IP アドレスをプライマリー IP アドレスとして設定していました。そのため、これは発信接続の IPv4 ソースアドレスの選択に影響を与えていました。今回の更新で、セカンダリー IP アドレスを手動で設定すると、**NetworkManager** パッケージは **primary-ip-address** メタデータからプライマリー IP アドレスを取得し、プライマリー IP アドレスとセカンダリー IP アドレスの両方を正しく設定するようになりました。

(BZ#2079849)

NetworkManager ユーティリティーは、手動で追加した IPv6 アドレスの正しい順序を強制します。

通常、IPv6 アドレスの順序は、ソースアドレス選択の優先度に影響します。たとえば、発信 TCP 接続を確立する場合などです。以前は、**manual**、**dhcpv6**、および **autoconf6** メソッドで追加された IPv6 アドレスの相対的な優先度は、正しくありませんでした。今回の更新で問題が修正され、優先順位の優先順位が **manual** > **dhcpv6** > **autoconf6** のロジックを反映するようになりました。また、**ipv6.addresses** 設定下のアドレスの順序が逆になり、最初に追加されたアドレスの優先度が最も高くなります。

(BZ#2097293)

8.8. カーネル

ネットワークソケットに対するタグ付けが再び機能するようになりました

これまで、**cgroup** v2 に相当するものがない特定のレガシー **cgroup** v1 コントローラー (**net_prio** や **net_cls** など) は、**cgroup** v2 ソケットタグ付けが混在する **cgroup** v1/v2 コントローラーと一緒にマウントされた場合に、**cgroup** v2 ソケットのタグ付けに干渉していました。そのため、**net_prio** または **net_cls** v1 コントローラーを使用する **cgroup** v1/v2 の混合環境では、**cgroup** v2 を使用した適切なネットワークソケットのタグ付けを無効になりました。今回の更新により、この制限がなくなり、cgroup v1/v2 の混合環境におけるネットワークソケットのタグ付けを使用できるようになりました。

(BZ#2060150)

kexec-tools パッケージがデフォルトの crashkernel メモリー予約値をサポートするようになりました

kexec-tools パッケージは、デフォルトの **クラッシュカーネル** メモリー予約値を維持するようになりました。**kdump** サービスはデフォルト値を使用して、各カーネルのクラッシュカーネルメモリーを予約します。この実装により、システムの使用可能なメモリーが 4 GB 未満の場合、**kdump** のメモリー割り当ても改善されます。

デフォルトの **crashkernel** 値によって予約されたメモリーがシステムで十分でない場合、**kdumpctl Estimator** コマンドを使用して、クラッシュをトリガーせずに推定値を取得できます。予測される **crashkernel=** 値は正確でない可能性があり、適切な **crashkernel=** 値を設定するための参照として利用できます。

(BZ#1959203)

システムは、動的 LPAR 操作を正常に実行できる

以前は、次のいずれかの条件が満たされた場合、ユーザーはハードウェア管理コンソール (HMC) から動的論理パーティション (DLPAR) 操作を実行できませんでした。

- Secure Boot 機能は、整合性モードでカーネル **lockdown** メカニズムを暗黙的に有効にしました。
- カーネル **ロックダウン** メカニズムは、整合性モードまたは機密性モードで手動で有効にされました。

RHEL 9 では、カーネルの **lockdown** により、**/dev/mem** 文字デバイスファイルからアクセスできるシステムメモリーへの RunTimeAbstraction Services (RTAS) アクセスが完全にブロックされました。正しく機能するために、いくつかの RTAS は **/dev/mem** への書き込みアクセスが必要です。そのため、RTAS 呼び出しが適切に実行されず、ユーザーには以下のエラーメッセージが表示されます。

> HSCL2957 Either there is currently no RMC connection between the management console and the partition <LPAR name> or the partition does not support dynamic partitioning operations. Verify the network setup on the management console and the partition and ensure that any firewall authentication between the management console and the partition has occurred. Run the management console diagrmc command to identify problems that might be causing no RMC connection.

この更新では、**lockdown** に対して非常に狭い PowerPC 固有の例外を提供することで問題が修正されました。この例外により、RTAS が必要な **/dev/mem** エリアにアクセスできます。その結果、上述のシナリオで問題が検出されなくなりました。

(BZ#2046472)

リングバッファーの値を rx から maxに設定した後、カーネル警告は表示されません

カーネルは、クリーンな入力を想定する内部関数が再利用された初期化済みの構造で呼び出された場合に、**Missing unregister, handled but fix driver** という警告メッセージを生成していました。今回の更新では、構造を再登録する前に再初期化することで、この問題が修正されています。

(BZ#2054379)

8.9. ブートローダー

grubby が引数を将来のカーネルに渡すようになりました

新しいバージョンのカーネルをインストールすると、**grubby** ツールは、以前のカーネルバージョンからのカーネルコマンドライン引数を渡しませんでした。その結果、GRUB ブートローダーがユーザー設定を無視していました。今回の修正により、新しいカーネルバージョンのインストール後も、ユーザー設定が永続化されるようになりました。

(BZ#1978226)

8.10. ファイルシステムおよびストレージ

ジャーナルエントリーがジャーナル書き込みを停止しなくなりました

以前は、VDO ドライバーで、デバイスマッパーの一時停止操作中およびデバイス操作の再開後、一部のジャーナルブロックは、メタデータの更新がすでに行われていても、再利用する前にそれらの更新を待機しているとマークされることがありました。ジャーナルが同じ物理ブロックにラップアラウンドするのに十分なジャーナルエントリーが作成されたとき、それは利用できませんでした。ブロックが使用可能になるのを待って、ジャーナルの書き込みが停止することがありましたが、これはこれまでは発生しませんでした。その結果、VDO デバイスの一部の操作にサスペンドまたはレジュームサイクルが含まれる場合は、一部のジャーナルの更新後にデバイスがフリーズ状態になりました。このデバイスの状態になる前のジャーナルの更新は、VDO 内の以前の割り当てパターン、および着信書き込みまたは破棄パターンに依存していたため、予測できませんでした。今回の更新により、データをストレージに保存する一時停止または再開サイクルの後、内部データ構造の状態がリセットされ、ロックアップが発生しなくなりました。

(BZ#2064802)

データデバイスを追加しても、アサーションエラーがトリガーされなくなりました

以前は、デバイスをキャッシュに追加する場合、Stratis は初期化直後にキャッシュを使用しませんでした。その結果、**stratisd** サービスは、ユーザーが追加のデータデバイスをプールに追加しようとするたびに、アサーション失敗メッセージを返しました。今回の修正により、初期化の直後にキャッシュが使用されるようになり、アサーションエラーは発生しなくなりました。

(BZ#2007018)

暗号化されたプールに新しいデータデバイスを追加する際のエラーを解決しました。

以前は、ユーザーが **--trust-url** オプションで指定された tang サーバーで Clevis bind コマンドを使用して、暗号化されたデータデバイスで暗号化されたプールを初期化するたびに、**stratisd** は内部データに Clevis tang 設定の拇印部分を含めませんでした。構造。その結果、新しいデータデバイスをプールに追加しようとしたときにエラーが発生しました。今回の更新により、**stratisd** の内部データ構造に Clevis tang 設定の拇印部分が含まれるようになりました。

(BZ#2005110)

AMD EPYC システムの Broadcom イニシエーターから NVMe 名前空間に接続する場合、デフォルト以外の IOMMU 設定が不要になりました。

デフォルトでは、RHEL カーネルは AMD ベースのプラットフォームで IOMMU を有効にします。以前は、**lpfc** ドライバーは scatter-gather リストアクセサーマクロを使用しませんでした。そのため、AMD プロセッサーを持つ特定のサーバーで、転送の長さの不一致が原因で I/O が失敗するなどの NVMe I/O の問題が発生していました。

今回の更新で、Broadcom イニシエーターから NVMe 名前空間に接続するために、カーネルコマンドラインオプションを使用して IOMMU をパススルーモードにする必要がなくなりました。

(BZ#2073541)

8.11. 高可用性およびクラスター

pcs が stonith-watchdog-timeoutの値を検証するようになりました

以前は、**stonith-watchdog-timeout** プロパティーを SBD 設定と互換性のない値に設定できました。これにより、フェンスループが発生したり、アクションが終了していなくても、クラスターがフェンシングアクションが成功したとみなす可能性がありました。この修正により、不正な設定を防ぐために、**pcs** 設定時に **stonith-watchdog-property** の値を検証するようになりました。

(BZ#2058246)

pcs が新しい Booth チケットの作成時に mode オプションを認識するようになりました

以前は、新しい Booth チケットを追加するときにユーザーが **mode** オプションを指定すると、**pcs** がエラー **invalid booth ticket option 'mode'** を報告していました。今回の修正により、Booth チケットの作成時に **mode** オプションを指定できるようになりました。

(BZ#2058243)

pcs がリソースと stonith リソースを区別するようになりました

これまで、一部の **pcs** コマンドは、リソースと stonith リソースを区別しませんでした。そのため、stonith リソースに **pcs resource** サブコマンドを使用し、stonith リソースではないリソースに対して **pcs stonith** サブコマンドを使用できました。これにより、ユーザーが混乱したり、リソースの誤設定が発生したりしていました。今回の更新で、リソースタイプが一致しない場合は **pcs** が警告を表示するようになりました。

(BZ#1301204)

8.12. コンパイラーおよび開発ツール

glibc が、NSS モジュールの読み込み後に errno を復元するようになりました

これまで、最後の NSS (Name Service Switch) モジュールがデータを提供しなかった場合、**glibc** の NSS 実装は **getpwent()** などの関数を使用して、データベースの列挙中に errno を誤って設定していました。その結果、これらの列挙関数を使用するアプリケーションはエラーを誤って確認し、失敗していました。**glibc** は、NSS モジュールの読み込み後に errno を復元するようになり、その結果、これらの関数を使用するアプリケーションが失敗しなくなりました。

(BZ#2063142)

監査インターフェイスは、x8 レジスタと AArch64 の NEON レジスタの全幅を保存および復元するようになりました。

以前は、ダイナミックローダーの監査インターフェイスの実装にバグがあったため、**AArch64** の保存されたレジスタの状態が、プロシージャコールの標準と比較して不完全でした。このバグは修正され、監査インターフェイスは x8 レジスタと **AArch64** の NEON レジスタの全幅を保存および復元するようになりました。動的ローダー監査インターフェイスを使用するアプリケーションは、**AArch64** の x8 レジスターを検査して影響を与えることができるようになりました。この新しい x8 レジスタを使用し、**AArch64** の NEON レジスタの全幅にアクセスするには、新しいバージョンのインターフェイスを使用するように監査モジュールを再コンパイルする必要があります (LAV_CURRENT は 2 です)。

(BZ#2003291)

POWER9 向けに最適化された strncpy 関数が誤った結果を返さなくなりました

以前は、POWER9 の strncpy 関数は、埋め込み用の NUL バイトのソースとして正しいレジスターを使用していませんでした。その結果、出力バッファーには、NUL パディングではなく、初期化されていないレジスタコンテンツが含まれていました。今回の更新で、strncpy 関数が修正され、出力バッファーの末尾が NUL バイトで正しくパディングされるようになりました。

(BZ#2091549)

glibc memmem 関数の Valgrind オーバーライドが IBMz15 アーキテクチャーにインストールされました

以前は、**glibc memmem** 関数の valgrind オーバーライドがないと、以下の警告が誤って検出されていました。

```
Conditional jump or move depends on uninitialised value(s)
```

今回の更新には **glibc memmem** 関数の valgrind オーバーライドが含まれており、IBMz15 アーキテクチャーの valgrind で実行しているプログラムで **memmem** 関数を使用しても警告のご検出は発生しなくなりました。

(BZ#1993976)

8.13. IDENTITY MANAGEMENT

ipa user-del --preserve user_login 出力に、ユーザーの削除は示されなくなりました

以前は、**ipa user-del --preserve user_login** コマンドを実行してユーザーアカウントを保存すると、出力に **Deleted user "user_login"** というメッセージが誤って返されていました。今回の更新で、出力は **Preserved user “user_login”** を返すようになりました。

(BZ#2100227)

RHEL 9 Kerberos クライアント (Heimdal KDC シナリオ) で PKINIT ユーザー認証が正しく動作するようになりました

以前は、Heimdal Kerberos Distribution Center (KDC) に対する RHEL 9 Kerberos クライアントでの IdM ユーザーの PKINIT 認証に失敗していました。この失敗は、Kerberos クライアントが RHEL 9 の SHA-1 アルゴリズムの非推奨コンテキストで必要な **supportedCMSTypes** フィールドに対応していなかったために発生しました。

今回の更新で、RHEL 9 Kerberos クライアントは PKINIT 中に、**sha512WithRSAEncryption** および **sha256WithRSAEncryption** を含む署名アルゴリズムのリストを **supportedCMSTypes** として Heimdal KDC に送信します。Heimdal KDC は **sha512WithRSAEncryption** を使用するため、PKINIT 認証が正しく機能します。

(BZ#2068935)

LDAP グループのメンバーリスト内の読み取り不能オブジェクトの処理

この更新の前は、SSSD が LDAP グループのメンバーリスト内の読み取り不能オブジェクトを一貫して処理していなかったため、読み取り不能オブジェクトが原因でエラーが発生したり、特定の状況で読み取り不能オブジェクトが無視されたりしていました。

今回の更新により、SSSD には、この動作を変更するための新しいオプション **ldap_ignore_unreadable_references** が追加されました。**ldap_ignore_unreadable_references** オプションが **false** に設定されている場合には、読み取り不能オブジェクトによってエラーが発生し、**true** に設定されている場合、読み取り不能オブジェクトは無視されます。デフォルトは **false** に設定されており、元の一貫性のない動作のために、更新後に一部のグループルックアップが失敗する場合があります。この場合、**/etc/sssd/sssd.conf** ファイルの対応する **[domain/name of the domain]** セクションで **ldap_ignore_unreadable_references = True** と指定します。

これにより、読み取り不能オブジェクトを一貫した方法で処理でき、新しい **ldap_ignore_unreadable_references** オプションを使用して動作を調整できます。

(BZ#2069376)

8.14. デスクトップ

アクティベーションキーを使用したサブスクリプションの登録が修正されました。

以前は、アクティベーションキーを使用して Red Hat サブスクリプションを **Settings** に登録できませんでした。この **Settings** では、**Register** を押すと、以下のエラーが表示されます。

> Failed to register system; Failed to RegisterWithActivationKeys: Unknown arguments: dict_keys(['enable_content'])

今回の更新で問題が修正され、**Settings** で、アクティベーションキーを使用してサブスクリプションを登録できるようになりました。

(BZ#2100467)

8.15. グラフィックインフラストラクチャー

X.org で X11 SECURITY エクステンションが有効になりました

以前は、X.org ディスプレイサーバーは X11 **SECURITY** エクステンションを提供していませんでした。そのため、このエクステンションを使用するアプリケーションは予期せず終了していました。

今回の更新で、X.org が X11 **SECURITY** エクステンションを有効にするようになりました。その結果、エクステンションに依存するアプリケーションが期待どおりに機能するようになりました。

(BZ#1894612)

VGA ディスプレイを備えた Matrox GPU が期待どおりに動作するようになりました

このリリースより前は、次のシステム設定を使用した場合、ディスプレイにグラフィカル出力が表示されませんでした。

- Matrox MGA G200 ファミリーの GPU
- VGA コントローラーで接続されたディスプレイ
- UEFI のレガシーモードへの切り替え

したがって、この設定で RHEL を使用またはインストールできませんでした。

今回の更新により、**mgag200** ドライバーが大幅に書き直され、その結果、グラフィック出力が期待どおりに機能するようになりました。

(BZ#2100898)

8.16. WEB コンソール

Web コンソールを使用した USB ホストデバイスの削除が期待どおりに機能するようになりました

以前は、USB デバイスを仮想マシン (VM) に接続すると、USB デバイスのデバイス番号とバス番号が仮想マシンに渡された後に変更されていました。その結果、Web コンソールを使用してこのようなデバイスを削除すると、デバイスとバス番号の相関が正しくないため失敗していました。今回の更新で問題が修正され、Web コンソールを使用して USB ホストデバイスを削除できるようになりました。

(JIRA:RHELPLAN-109067)

Web コンソールを使用した複数のホストデバイスの接続が想定通りに機能するようになりました

以前は、Web コンソールを使用して仮想マシンに接続する複数のデバイスを選択すると、1 つのデバイスのみが接続され、残りは無視されていました。今回の更新で問題が修正され、Web コンソールを使用して複数のホストデバイスを同時に接続できるようになりました。

(JIRA:RHELPLAN-115603)

8.17. RED HAT ENTERPRISE LINUX システムロール

network RHEL ロールは、設定ファイルで ansible_managed パラメーターを管理します。

以前は、Ansible ロールは、**network** ロールの管理対象設定ファイルに正しい **ansible_managed** ヘッダーを提供できませんでした。そのため、システム管理者は、どのファイルが Ansible によって管理されているかについて不明な状態となっていました。今回の修正により、ロール管理ファイルには正しい **ansible_managed** ヘッダーが設定され、システム管理者は、どのファイルを Ansible が管理しているか明確に分かるようになりました。

(BZ#2065382)

正しいボンディングモードの active-backup をサポートするようにタイプミスを修正する

以前は、**active-backup** ボンディングモードを指定する際に InfiniBand ポートをサポートする際に、タイプミス (**active_backup**) がありました。このタイプミスが原因で、接続は InfiniBand ボンディングポートの正しいボンディングモードをサポートできませんでした。この更新では、ボンディングモードを **active-backup** に変更することで、タイプミスを修正しています。これで、接続は InfiniBand ボンディングポートを正常にサポートします。

(BZ#2065394)

IPRouteUtils.get_route_tables_mapping() 関数が空白シーケンスを受け入れるようになりました。

以前は、**/etc/iproute2/rt_tables** などの **iproute2** ルーティングテーブルデータベースのパーサーは、**254 main** の形式でファイルにエントリーをアサートしていたため、数値 ID と名前は 1 つのスペースのみで区切られていました。その結果、パーサーはルートテーブル名とテーブル ID 間のすべてのマッピングをキャッシュできませんでした。これにより、ユーザーはルートテーブル名を定義して静的

ルートをルートテーブルに追加できませんでした。今回の更新で、パーサーはテーブル ID とテーブル名の間に空白シーケンスを受け入れるようになりました。その結果、パーサーはルートテーブル名とテーブル ID 間のすべてのマッピングをキャッシュし、ユーザーはルートテーブル名を定義して静的ルートをルートテーブルに追加できます。

(BZ#2115886)

forward_port パラメーターが、string オプションと dict オプションの両方を受け入れるようになりました。

以前は、**firewall** RHEL システムロールで、**forward_port** パラメーターは **string** オプションのみを受け入れていました。しかし、ロールのドキュメントでは、**string および dict** の両オプションがサポートされているとしていました。そのため、ドキュメントを読み込んだユーザーはエラーを経験していました。このバグは、**forward_port** で両方のオプションを受け入れるようにすることで修正されました。その結果、ユーザーはドキュメントに従いポート転送を設定できるようになりました。

(BZ#2100605)

metrics ロールによる設定がシンボリックリンクを正しくたどるようになりました

mssql pcp パッケージがインストールされると、**mssql.conf** ファイルは **/etc/pcp/mssql/** に配置され、シンボリックリンク **/var/lib/pcp/pmdas/mssql/mssql.conf** のターゲットになります。ただし、以前の **metrics** ロールはシンボリックリンクをたどって、**mssql.conf** を設定する代わりにシンボリックリンクを上書きしていました。その結果、**metrics** ロールを実行すると、シンボリックリンクが通常のファイルに変更され、**/var/lib/pcp/pmdas/mssql/mssql.conf** ファイルのみ設定の影響を受けました。これによりシンボリックリンクが失敗し、メインの設定ファイル **/etc/pcp/mssql/mssql.conf** は設定の影響を受けませんでした。この問題は修正され、シンボリックリンクをたどる **follow: yes** オプションが **metrics** ロールに追加されました。その結果、**metrics** ロールはシンボリックリンクを保持し、メイン設定ファイルを正しく設定します。

(BZ#2060523)

kernel_settings configobj はマネージドホストで利用できます。

以前は、**kernel_settings** ロールはマネージドホストに **python3-configobj** パッケージをインストールしませんでした。そのため、**configobj** Python モジュールが見つからないことを示すエラーが返されました。この修正により、ロールは **python3-configobj** パッケージがマネージドホストに存在し、**kernel_settings** ロールが期待どおりに機能することを保証します。

(BZ#2060525)

ボリュームの mount_options パラメーターがボリュームに対して有効になりました

以前は、ボリュームの有効なパラメーターの一覧からパラメーターが誤って削除されていました。そのため、ユーザーはボリュームに **mount_options** パラメーターを設定できませんでした。今回のバグ修正により、**mount_options** パラメーターが有効なパラメーターの一覧に戻され、エラーをキャッチするようにコードがリファクタリングされました。その結果、**storage** RHEL システムロールは、ボリュームの **mount_options** パラメーターを設定できます。

(BZ#2083376)

storage RHEL システムロールは、LVM ボリュームの striped および RAID0 レベルを正しくサポートするようになりました

以前は、**storage** RHEL システムロールは、LVM ボリュームに対応していないため、RAID レベルの **striped** および **raid0** を誤って報告していました。これは修正され、ロールは LVM でサポートされているすべての RAID レベルの LVM ボリュームを正しく作成できるようになりました。これには、**raid0**、**raid1**、**raid4**、**raid5**、**raid6**、**raid10**、**striped**、および **mirror** が含まれます。

(BZ#2083410)

metrics RHEL システムロールの README およびドキュメントで、特定のバージョンの RHEL においてロールがサポートする Redis および Grafana バージョンが明記されるようになりました

以前のバージョンでは、サポートされていないバージョンの Redis および Grafana で **metrics** ロールを使用しようとすると、ロールは失敗していました。この更新により、どのバージョンの RHEL でどのバージョンの Redis と Grafana がロールによってサポートされているかについて、ドキュメントが明確化されました。そのため、サポート対象外のバージョンの Redis および Grafana をサポート対象外のプラットフォームで使用することを回避できます。

(BZ#2100286)

ssh および sshd RHEL システムロールの RSA 鍵最小ビット長オプションが追加されました

誤って短い RSA 鍵を使用すると、システムが攻撃に対してより脆弱になる可能性があります。この更新により、**ssh** および **sshd** RHEL システムロールの **RequiredRSASize** オプションを使用して、OpenSSH クライアントおよびサーバーの RSA 鍵の最小ビット長を設定できるようになりました。

(BZ#2109998)

nbde_client RHEL システムロールが、追加の Dracut コマンドラインパラメーターの指定時に適切な間隔を使用するようになりました

Dracut フレームワークでは、カーネルコマンドラインパラメーターなどの追加のパラメーターを指定する場合には、適切な間隔が必要です。パラメーターが適切な間隔で指定されていない場合、Dracut は指定された追加パラメーターをカーネルコマンドラインに追加しない場合があります。この更新により、**nbde_client** RHEL システムロールは、アドオン Dracut 設定ファイルの作成時に適切な間隔を使用するようになりました。その結果、ロールは Dracut コマンドラインパラメーターを正しく設定します。

(BZ#2115156)

tlog RHEL システムロールが SSSD によって正しくオーバーレイされるようになりました

以前は、**tlog** RHEL システムロールは、System Security Services Daemon (SSSD) ファイルプロバイダーと有効な **authselect** オプション **with-files-domain** に依存して、**nsswitch.conf** ファイルに正しい **passwd** エントリーを設定していました。RHEL 9.0 では、SSSD はデフォルトではファイルプロバイダーを暗黙的に有効にせず、SSSD による **tlog-rec-session** シェルオーバーレイは機能しませんでした。今回の修正により、**tlog** ロールは **nsswitch.conf** を更新して、**tlog-rec-session** が SSSD によって正しくオーバーレイされるようになりました。

(BZ#2071804)

metrics RHEL システムロールが、設定の更新後に pmie および pmlogger サービスを自動的に再起動するようになりました

以前は、**pmie** および **pmlogger** サービスは、設定が変更され、ハンドラーの実行を待機した後、再起動しませんでした。これにより、他の **metrics** サービスでエラーが発生し、**pmie** と **pmlogger** の設定を実行時の動作と一致させる必要がありました。今回の更新により、ロールは設定の更新直後に **pmie** と **pmlogger** を再起動し、それらの設定は依存メトリックサービスの実行時の動作と一致し、正しく機能します。

(BZ#2100294)

8.18. 仮想化

負荷が大きい場合に、仮想マシンのネットワークトラフィックのパフォーマンスが低下しなくなりました

以前は、RHEL 仮想マシンは、高レベルのネットワークトラフィックを処理する際のパフォーマンスが低下していました。基礎となるコードが修正され、上記の状況でもネットワークトラフィックのパフォーマンスが期待どおりに機能するようになりました。

(BZ#1945040)

8.19. クラウド環境の RHEL

Hyper-V 仮想マシンに接続されたネットワークアダプターの SR-IOV 機能が確実に動作するようになりました

以前は、シングルルート I/O 仮想化 (SR-IOV) が有効になっているネットワークアダプターを MicrosoftHyper-V ハイパーバイザーで実行されている RHEL 9 仮想マシン (VM) に接続すると、SR-IOV 機能が正しく機能しない場合がありました。Hyper-V 固有の memory-mapped I/O (MMIO) 割り当てコードのバグが修正され、Hyper-V 仮想マシンで SR-IOV 機能が想定どおりに機能するようになりました。

(BZ#2030922)

SR-IOV は、Azure 上の ARM 64 RHEL 9 仮想マシンで準最適に動作しなくなりました

これまで、SR-IOV ネットワーキングデバイスは、Microsoft Azure プラットフォームで実行されている ARM 64 RHEL 9 仮想マシンで想定されるよりも、全体ではるかに低く、レイテンシーは高くなっていました。この問題は修正され、影響を受ける仮想マシンが期待どおりに動作するようになりました。

(BZ#2068432)

8.20. コンテナー

podman system connection add および podman image scp が失敗しなくなりました

Podman は、RSA 鍵交換に SHA-1 ハッシュを使用します。これまで、RHEL 9 では SHA-1 ハッシュが鍵交換に受け入れられないため、RSA キーを使用するマシン間の通常の SSH 接続が機能し、一方で、**podman system connection add** および **podman image scp** コマンドは同じ RSA 鍵を使用して機能しませんでした。今回の更新で、この問題が修正されました。

(JIRA:RHELPLAN-121180)

ベータ版 GPG キーで署名されたコンテナーイメージがプルできるように

以前は、RHEL ベータコンテナーイメージをプルすると、Podman は次のエラーメッセージで失敗していました。**Error: Source image rejected: None of the signatures were accepted** (エラー: ソースイメージが拒否されました: 署名が受け入れられませんでした)。現在のビルドでは、RHEL ベータ版の GPG キーをデフォルトで信頼しないように設定されているため、イメージのプルに失敗していました。今回の更新により、**/etc/containers/policy.json** ファイルは、信頼できるキーを含むファイルのリストを受け入れる新しい **keyPaths** フィールドをサポートします。このため、GA および Beta GPG キーで署名されたコンテナーイメージがデフォルト設定で受け入れられるようになりました。

(BZ#2094015)

Podman がコンテナー X509: 不明な機関によって署名された証明書のプルに失敗しなくなりました

以前は、独自の CA 証明書によって署名された独自の内部レジストリーがある場合、その証明書をホストマシンにインポートする必要がありました。そうでない場合は、エラーが発生します。

```
x509: certificate signed by unknown authority
```

今回の更新で、この問題が修正されています。

(BZ#2027576)

リポジトリー ID が一致しないために DNF と YUM が失敗することがなくなりました

以前は、DNF および YUM リポジトリー ID は、DNF または YUM が期待する形式と一致しませんでした。たとえば、次の例を実行すると、エラーが発生しました。

```
# podman run -ti ubi8-ubi
# dnf debuginfo-install dnsmasq
...
This system is not registered with an entitlement server. You can use subscription-manager to register.
```

今回の更新で、この問題が修正されています。接尾辞 **--debug-rpms** がすべてのデバッグリポジトリー名に追加され (例: **ubi-8-appstream-debug-rpms**)、接尾辞 **-rpms** がすべての UBI リポジトリー名に追加されました (例: **ubi-8-appstream-rpms)**)。

詳細については、ユニバーサルベースイメージ (UBI): イメージ、リポジトリー、パッケージ、およびソースコード を参照してください。

(BZ#2120378)

第9章 テクノロジープレビュー

ここでは、Red Hat Enterprise Linux 9 で利用可能なテクノロジープレビューのリストを提示します。

テクノロジープレビューに対する Red Hat のサポート範囲の詳細は、テクノロジープレビューのサポート範囲 を参照してください。

9.1. シェルおよびコマンドラインツール

ReaR は、64 ビット IBM Z アーキテクチャーでテクノロジープレビューとして利用できます。

Basic Relax and Recover (ReaR) 機能が、64 ビットの IBM Z アーキテクチャーでテクノロジープレビューとして利用できるようになりました。IBM Z では、z/VM 環境でのみ ReaR レスキューイメージを作成できます。論理パーティション (LPAR) のバックアップおよび復元はテストされていません。

現在利用できる出力方法は、Initial Program Load (IPL) のみです。IPL は、**zIPL** ブートローダーで使用できるカーネルと初期 ramdisk (initrd) を生成します。

警告

現在、レスキュープロセスは、システムに接続したすべての DASD (Direct Attached Storage Devices) を再フォーマットします。システムのストレージデバイスに貴重なデータが存在する場合は、システムの復旧を行わないでください。これには、レスキュー環境で起動するのに使用された **zIPL** ブートローダー、ReaR カーネル、および initrd で準備されたデバイスも含まれます。必ずコピーを保管してください。

詳細は、64 ビット IBM Z アーキテクチャーで ReaR レスキューイメージの使用 を参照してください。

(BZ#2046653)

RHEL 9 でテクノロジープレビューとして利用可能な GIMP

GNU Image Manipulation Program (GIMP) 2.99.8 が、テクノロジープレビューとして RHEL 9 で利用できるようになりました。**gimp** パッケージバージョン 2.99.8 は、改善された一連の改良を含むリリース前のバージョンですが、機能のセットが制限され、安定性の保証は保証されません。公式の GIMP 3 のリリース後すぐに、今回のリリース前のバージョンの更新として RHEL 9 に導入されます。

RHEL 9 では、RPM パッケージとして **gimp** を簡単にインストールできます。

(BZ#2047161)

9.2. セキュリティー

gnutls がテクノロジープレビューとして KTLS を使用するようになる

更新された **gnutls** パッケージは、テクノロジープレビューとして、暗号化チャネルでのデータ転送を加速するためにカーネル TLS (KTLS)を使用できます。KTLS を有効にするには、**modprobe** コマンドを使用して **tls.ko** カーネルモジュールを追加し、以下の内容でシステム全体の暗号化ポリシー用の新しい設定ファイル **/etc/crypto-policies/local.d/gnutls-ktls.txt** を作成します。

-

```
[global]
ktls = true
```

現在のバージョンは、TLS **KeyUpdate** メッセージによるトラフィックキーの更新をサポートしていません。これは、AES-GCM 暗号スイートのセキュリティーに影響を与えることに注意してください。詳細は、RFC 7841 - TLS 1.3 ドキュメントを参照してください。

(BZ#2042009)

9.3. ネットワーク

WireGuard VPN はテクノロジープレビューとして利用可能になる

Red Hat がサポートしていないテクノロジープレビューとして提供している WireGuard は、Linux カーネルで実行する高パフォーマンスの VPN ソリューションです。最新の暗号を使用し、その他の VPN ソリューションよりも簡単に設定できます。さらに、WireGuard のコードベースが小さくなり、攻撃の影響が減るため、セキュリティーが向上します。

詳細は Setting up a WireGuard VPN を参照してください。

(BZ#1613522)

NetworkManager を使用した Multipath TCP の設定がテクノロジープレビューとして利用可能になりました

今回の更新で、NetworkManager ユーティリティーが Multipath TCP (MPTCP) 機能を提供するようになりました。**nmcli** コマンドを使用して MPTCP を制御し、その設定を永続化できます。

詳細は、Understanding Multipath TCP: High availability for endpoints and the networking highway of the future および RFC 8684: TCP Extensions for Multipath Operation with Multiple Addresses を参照してください。

(BZ#2029636)

KTLS がテクノロジープレビューとして利用可能になる

RHEL は、テクノロジープレビューとして KTLS (Kernel Transport Layer Security) を提供します。KTLS は、AES-GCM 暗号化のカーネルで対称暗号化アルゴリズムまたは複号アルゴリズムを使用して TLS レコードを処理します。KTLS には、この機能を提供するネットワークインターフェイスコントローラー (NIC) に TLS レコード暗号化をオフロードするインターフェイスも含まれています。

(BZ#1570255)

systemd-resolved サービスがテクノロジープレビューとして利用可能です。

systemd-resolved サービスは、ローカルアプリケーションに名前解決を提供します。このサービスは、DNS スタブリゾルバー、LLMNR (Link-Local Multicast Name Resolution)、およびマルチキャスト DNS リゾルバーとレスポンダーのキャッシュと検証を実装します。

systemd-resolved は、サポートされていないテクノロジープレビューであることに注意してください。

(BZ#2020529)

9.4. カーネル

カーネルの Intel データストリーミングタブレットドライバーがテクノロジープレビューとして利用可能になりました。

カーネルの Intel データストリーミングアクセラレータードライバー (IDXD) は、現在テクノロジープレビューとして利用できます。これは IntelCPU 統合アクセラレータであり、プロセスアドレス空間 ID (pasid) 送信と共有仮想メモリー (SVM) を備えた共有ワークキューが含まれています。

(BZ#2030412)

SGX がテクノロジープレビューとして利用可能

Software Guard Extensions(SGX) は、ソフトウェアコードおよび公開および修正からのデータを保護する Intel® テクノロジーです。RHEL カーネルは、SGX v1 および v1.5 の機能を部分的に提供します。バージョン 1 では、**Flexible Launch Control** メカニズムを使用するプラットフォームが SGX テクノロジーを使用できるようにします。

(BZ#1874182)

Soft-iWARP ドライバーがテクノロジープレビューとして利用可能に

Soft-iWARP(siw) は、Linux 用のソフトウェア、インターネットワイドエリア RDMA プロトコル (iWARP)、カーネルドライバーです。soft-iWARP は、TCP/IP ネットワークスタックで iWARP プロトコルスイートを実装します。このプロトコルスイートはソフトウェアで完全に実装されており、特定のリモートダイレクトメモリーアクセス (RDMA) ハードウェアを必要としません。soft-iWARP を使用すると、標準のイーサネットアダプターを備えたシステムが iWARP アダプターまたは他のシステムに接続でき、すでに Soft-iWARP がインストールされている別のシステムに接続できます。

(BZ#2023416)

9.5. ファイルシステムおよびストレージ

DAX がテクノロジープレビューとして ext4 および XFS で利用可能になる

RHEL 9 では、DAX ファイルシステムがテクノロジープレビューとして提供されています。DAX は、アプリケーションが永続メモリーをそのアドレス空間に直接マップするための手段を提供します。DAX を使用するには、システムに何らかの形式の永続メモリー (通常は 1 つ以上の不揮発性デュアルインラインメモリーモジュール (NVDIMM) の形式) が必要であり、DAX 互換ファイルシステムを NVDIMM 上に作成する必要があります。)。また、ファイルシステムは **dax** マウントオプションでマウントする必要があります。これにより、dax をマウントしたファイルシステムのファイルの **mmap** が、アプリケーションのアドレス空間にストレージを直接マッピングされます。

(BZ#1995338)

Stratis はテクノロジープレビューとして利用可能です

Stratis はローカルストレージマネージャーです。ユーザーへの追加機能を備えたストレージプールに管理されたファイルシステムを提供します。

- スナップショットおよびシンプロビジョニングを管理する
- 必要に応じてファイルシステムのサイズを自動的に大きくする
- ファイルシステムを維持する

Stratis ストレージを管理するには、バックグランドサービス **stratisd** と通信する **stratis** ユーティリティーを使用します。

Stratis はテクノロジープレビューとして提供されます。

詳細は、Stratis ドキュメントの Setting up Stratis file systems を参照してください。

(BZ#2041558)

NVMe-oF Discovery Service 機能がテクノロジープレビューとして利用可能になりました。

NVMexpress.org Technical Proposals (TP) 8013 および 8014 で定義されている NVMe-oF Discovery Service の機能は、テクノロジープレビューとして利用できます。これらの機能をプレビューするには、**nvme-cli 2.0** パッケージを使用して、TP-8013 または TP-8014 を実装する NVMe-oF ターゲットデバイスにホストを割り当てます。TP-8013 および TP-8014 の詳細は、https://nvmexpress.org/developers/nvme-specification/ の Web サイトの NVM Express 2.0 Ratified TPs を参照してください。

(BZ#2021672)

NVMe-stas パッケージがテクノロジープレビューとして利用可能になりました

Linux の Central Discovery Controller (CDC) クライアントである **nvme-stas** パッケージがテクノロジープレビューとして利用できるようになりました。これは、非同期イベント通知 (AEN)、自動化された NVMe サブシステム接続制御、エラー処理とレポート、および Automatic (**zeroconf**) 手動設定を処理します。

このパッケージは、Storage Appliance Finder (**stafd**)と Storage Appliance Connector (**stacd**)の 2 つのデーモンで構成されています。

(BZ#1893841)

9.6. コンパイラーおよび開発ツール

jmc-core および owasp-java-encoder がテクノロジープレビューとして利用可能に

RHEL 9 には、**jmc-core** パッケージおよび **owasp-java-encoder** パッケージでテクノロジープレビューとして配布されます。

jmc-core は、Java Development Kit (JDK) Mission Control のコア API を提供するライブラリーです。これには、JDK Flight Recording ファイルの解析および書き込み用のライブラリーや、Java Discovery Protocol (JDP) による Java Virtual Machine (JVM) 検出のライブラリーが含まれます。

owasp-java-encoder パッケージは、Java の高パフォーマンスな低オーバーヘッドコンテキストエンコーダーのコレクションを提供します。

(BZ#1980981)

9.7. IDENTITY MANAGEMENT

DNSSEC が IdM でテクノロジープレビューとして利用可能

統合 DNS のある Identity Management (IdM) サーバーは、DNS プロトコルのセキュリティーを強化する DNS に対する拡張セットである DNS Security Extensions (DNSSEC) を実装するようになりました。IdM サーバーでホストされる DNS ゾーンは、DNSSEC を使用して自動的に署名できます。暗号鍵は、自動的に生成およびローテートされます。

DNSSEC で DNS ゾーンを保護する場合は、以下のドキュメントを参照することが推奨されます。

- DNSSEC Operational Practices, Version 2
- Secure Domain Name System (DNS) Deployment Guide

- DNSSEC Key Rollover Timing Considerations

統合 DNS のある IdM サーバーは、DNSSEC を使用して、他の DNS サーバーから取得した DNS 回答を検証することに注意してください。これが、推奨される命名方法に従って設定されていない DNS ゾーンの可用性に影響を与える可能性があります。

(BZ#2084180)

Identity Management JSON-RPC API がテクノロジープレビューとして利用可能

Identity Management (IdM) では API が利用できます。API を表示するために、IdM は、テクノロジープレビューとして API ブラウザーも提供します。

以前では、複数のバージョンの API コマンドを有効にするために、IdM API が拡張されました。これらの機能拡張により、互換性のない方法でコマンドの動作が変更することがありました。IdM API を変更しても、既存のツールおよびスクリプトを引き続き使用できるようになりました。これにより、以下が可能になります。

- 管理者は、管理しているクライアント以外のサーバーで、IdM の以前のバージョンもしくは最近のバージョンを使用できます。
- サーバーで IdM のバージョンを変更しても、開発者は特定バージョンの IdM コールを使用できます。

すべてのケースでサーバーとの通信が可能になります。たとえば、ある機能向けの新オプションが新しいバージョンに追加されていて、通信の一方の側でこれを使用していたとしても、特に問題はありません。

API の使用方法は Identity Management API を使用して IdM サーバーに接続する (テクノロジープレビュー) を参照してください。

(BZ#2084166)

RHEL IdM では、ユーザー認証をテクノロジープレビューとして外部 ID プロバイダーに委任できます

RHEL IdM では、OAuth 2 デバイス認証フローをサポートする外部 ID プロバイダー (IdP) にユーザーを関連付けることができるようになりました。これらのユーザーは、RHEL 9.1 で利用可能な SSSD バージョンで認証すると、外部 IdP で認証と承認を実行した後、Kerberos チケットを使用して RHEL IdMSingle Sign-On 機能を受け取ります。

主な変更には以下のものがあります。

- **ipa idp-*** コマンドによる外部 IdP への参照の追加、変更、および削除
- **ipa user-mod --user-auth-type=idp** コマンドを使用したユーザーの IdP 認証の有効化

追加情報については、外部 ID プロバイダーを使用した IdM への認証 を参照してください。

(BZ#2069202)

sssd-idp サブパッケージがテクノロジープレビューとして利用可能になりました

SSSD の **sssd-idp** サブパッケージには、Identity Management (IdM) サーバーに対して OAuth2 認証を実行するクライアント側のコンポーネントである **oidc_child** プラグインおよび krb5 **idp** プラグインが含まれます。この機能は、RHEL 8.7 以降、および RHEL 9.1 以降の IdM サーバーでのみ使用できます。

(BZ#2065693)

SSSD の内部 krb5 idp プラグインがテクノロジープレビューとして利用可能になりました

SSSD krb5 **idp** プラグインを使用すると、OAuth2 プロトコルを使用して外部アイデンティティープロバイダー (IdP) に対して認証できます。この機能は、RHEL 8.7 以降、および RHEL 9.1 以降の IdM サーバーでのみ使用できます。

(BZ#2056482)

ACME がテクノロジープレビューとして利用可能

Automated Certificate Management Environment (ACME) サービスが、テクノロジープレビューとして Identity Management (IdM) で利用可能になりました。ACME は、自動化識別子の検証および証明書の発行に使用するプロトコルです。この目的は、証明書の有効期間を短縮し、証明書のライフサイクル管理での手動プロセスを回避することにより、セキュリティーを向上させることです。

RHEL では、ACME サービスは Red Hat Certificate System (RHCS) PKI ACME レスポンダーを使用します。RHCS ACME サブシステムは、IdM デプロイメントのすべての認証局 (CA) サーバーに自動的にデプロイされますが、管理者が有効にするまでリクエストに対応しません。RHCS は、ACME 証明書を発行する際に **acmeIPAServerCert** プロファイルを使用します。発行された証明書の有効期間は 90 日です。ACME サービスの有効化または無効化は、IdM デプロイメント全体に影響します。

重要

ACME は、すべてのサーバーが RHEL 8.4 以降を実行している IdM デプロイメントでのみ有効にすることが推奨されます。以前の RHEL バージョンには ACME サービスが含まれていないため、バージョンが混在するデプロイメントで問題が発生する可能性があります。たとえば、ACME のない CA サーバーは、異なる DNS サブジェクト代替名 (SAN) を使用しているため、クライアント接続が失敗する可能性があります。

警告

現在、RHCS は期限切れの証明書を削除しません。ACME 証明書は 90 日後に期限切れになるため、期限切れの証明書が蓄積され、パフォーマンスに影響を及ぼす可能性があります。

- IdM デプロイメント全体で ACME を有効にするには、**ipa-acme-manage enable** コマンドを使用します。

```
# ipa-acme-manage enable
The ipa-acme-manage command was successful
```

- IdM デプロイメント全体で ACME を無効にするには、**ipa-acme-manage disable** コマンドを使用します。

```
# ipa-acme-manage disable
The ipa-acme-manage command was successful
```

- ACME サービスがインストールされ、有効または無効であるかを確認するには、**ipa-acme-manage status** コマンドを使用します。

```
# ipa-acme-manage status
ACME is enabled
The ipa-acme-manage command was successful
```

(BZ#2084181)

9.8. デスクトップ

64 ビット ARM アーキテクチャーの GNOME がテクノロジープレビューとして利用できるようになりました。

GNOME デスクトップ環境は、テクノロジープレビューとして 64 ビット ARM アーキテクチャーで利用できます。

VNC を使用して 64 ビット ARM サーバーのデスクトップセッションに接続できるようになりました。その結果、グラフィカルアプリケーションを使用してサーバーを管理できます。

64 ビット ARM では、限定されたグラフィカルアプリケーションのセットを使用できます。以下に例を示します。

- Firefox Web ブラウザー
- Red Hat Subscription マネージャー (**subscription-manager-cockpit**)
- ファイアウォール設定 (**firewall-config**)
- ディスク使用状況アナライザー (**baobab**)

Firefox を使用して、サーバー上の Cockpit サービスに接続できます。

LibreOffice などの特定のアプリケーションは、コマンドラインインターフェイスのみを提供し、グラフィカルインターフェイスは無効になっています。

(JIRA:RHELPLAN-27394)

テクノロジープレビューとして利用可能な IBM Z アーキテクチャー用の GNOME

GNOME デスクトップ環境は、テクノロジープレビューとして IBM Z アーキテクチャーで利用できます。

VNC を使用して IBM Z サーバーのデスクトップセッションに接続できるようになりました。その結果、グラフィカルアプリケーションを使用してサーバーを管理できます。

IBM Z では、限定されたグラフィカルアプリケーションのセットを使用できます。たとえば、次のようになります。

- Firefox Web ブラウザー
- Red Hat Subscription マネージャー (**subscription-manager-cockpit**)
- ファイアウォール設定 (**firewall-config**)
- ディスク使用状況アナライザー (**baobab**)

Firefox を使用して、サーバー上の Cockpit サービスに接続できます。

LibreOffice などの特定のアプリケーションは、コマンドラインインターフェイスのみを提供し、グラフィカルインターフェイスは無効になっています。

(JIRA:RHELPLAN-27737)

9.9. WEB コンソール

Stratis が RHEL Web コンソールでテクノロジープレビューとして利用可能

今回の更新で、Red Hat Enterprise Linux Web コンソールは、Stratis ストレージをテクノロジープレビューとして管理できるようになりました。

Stratis の詳細は、Stratis とは を参照してください。

(JIRA:RHELPLAN-122345)

9.10. 仮想化

RHEL VM は、ARM64 プロセッサーで実行されている VMware ESXi インスタンスにデプロイできるようになりました

テクノロジープレビューとして、RHEL 仮想マシンを 64 ビット ARM ベースのプロセッサーで実行されている VMware ESXi ハイパーバイザーインスタンスにデプロイできるようになりました。

(JIRA:RHELPLAN-95456)

KVM 仮想マシンの AMD SEV および SEV-ES

RHEL 9 は、テクノロジープレビューとして、KVM ハイパーバイザーを使用する AMD EPYC ホストマシンに、セキュア暗号化仮想化 (SEV) 機能を提供します。仮想マシンで有効になっている場合は、SEV が仮想マシンのメモリーを暗号化して、ホストから仮想マシンへのアクセスを防ぎます。これにより、仮想マシンのセキュリティーが向上します。

さらに、強化された SEV (Encrypted State) バージョンの SEV (SEV-ES) もテクノロジープレビューとして提供されます。SEV-ES は、仮想マシンの実行が停止すると、すべての CPU レジスターの内容を暗号化します。これにより、ホストが仮想マシンの CPU レジスターを変更したり、そこから情報を読み取ったりできなくなります。

SEV および SEV-ES は、第 2 世代の AMD EPYC CPU (コードネーム Rome) 以降のみで動作することに注意してください。また、RHEL 9 には SEV および SEV-ES の暗号化が含まれますが、SEV および SEV-ES のセキュリティー証明は含まれません。

(JIRA:RHELPLAN-65217)

ARM 64 で仮想化が利用可能に

テクノロジープレビューとして、ARM 64 CPU を使用してシステムに KVM 仮想マシンを作成できるようになりました。

(JIRA:RHELPLAN-103993)

AMD64、Intel 64、および ARM 64 で virtio-mem が利用できるようになりました

RHEL 9 では、テクノロジープレビューとして、AMD64、Intel 64、および ARM 64 システムに **virtio-mem** 機能が追加されました。**virtio-mem** を使用すると、仮想マシンでホストメモリーを動的に追加または削除できます。

virtio-mem を使用するには、仮想マシンの XML 設定で **virtio-mem** メモリーデバイスを定義し、**virsh update-memory-device** コマンドを使用して、仮想マシンの実行中にメモリーデバイスのサイズ変更を要求します。このようなメモリーデバイスが実行中の仮想マシンに公開される現在のメモリーサイズを表示するには、仮想マシンの XML 設定を表示します。

(BZ#2014487, BZ#2044162, BZ#2044172)

Intel vGPU がテクノロジープレビューとして利用可能になる

テクノロジープレビューとして、物理 Intel GPU デバイスを、**mediated devices** と呼ばれる複数の仮想デバイスに分割できるようになりました。この仲介デバイスは、仮想 GPU として複数の仮想マシンに割り当てることができます。これにより、この仮想マシンが、1 つの物理 Intel GPU のパフォーマンスを共有します。

この機能は非推奨であり、今後の RHEL リリースでは完全に削除される予定であることに注意してください。

(JIRA:RHELDOCS-17050)

入れ子仮想マシンの作成

> Nested KVM virtualization is provided as a Technology Preview for KVM virtual machines (VMs) running on Intel, AMD64, and IBM Z hosts with RHEL 9. With this feature, a RHEL 7, RHEL 8, or RHEL 9 VM that runs on a physical RHEL 9 host can act as a hypervisor, and host its own VMs.

(JIRA:RHELDOCS-17040)

9.11. クラウド環境の RHEL

RHEL Confidential VMs がテクノロジープレビューとして Azure で利用可能になりました

更新された RHEL カーネルを使用すると、Microsoft Azure で機密仮想マシン (VM) をテクノロジープレビューとして作成して実行できるようになりました。ただし、Azure での起動中に RHEL 機密 VM イメージを暗号化することはまだできません。

(JIRA:RHELPLAN-122321)

9.12. コンテナー

イメージに署名するための複数の信頼できる GPG キーの機能は、テクノロジープレビューとして利用できます。

/etc/containers/policy.json ファイルは、信頼できるキーを含むファイルのリストを受け入れる新しい **keyPaths** フィールドをサポートします。このため、GA および Beta GPG キーで署名されたコンテナーイメージがデフォルト設定で受け入れられるようになりました。

以下に例を示します。

```
"registry.redhat.io": [
        {
            "type": "signedBy",
            "keyType": "GPGKeys",
            "keyPaths": ["/etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release", "/etc/pki/rpm-gpg/RPM-GPG-
```

```
KEY-redhat-beta"]
        }
]
```

(JIRA:RHELPLAN-129327)

sigstore 署名がテクノロジープレビューとして利用可能になりました

Podman 4.2 以降では、コンテナーイメージ署名の sigstore 形式を使用できます。sigstore 署名はコンテナーイメージと共にコンテナーレジストリーに格納されるため、イメージ署名を格納するために別の署名サーバーを用意する必要はありません。

(JIRA:RHELPLAN-74672)

podman-machine コマンドはサポート対象外です。

仮想マシンを管理するための **podman-machine** コマンドは、テクノロジープレビューとしてのみ利用可能です。代わりに、コマンドラインから直接 Podman を実行してください。

(JIRA:RHELDOCS-16861)

第10章 非推奨になった機能

ここでは、Red Hat Enterprise Linux 9 で **非推奨** となった機能の概要を説明します。

非推奨の機能は、本製品の今後のメジャーリリースではサポートされない可能性が高く、新たに実装することは推奨されません。特定のメジャーリリースにおける非推奨機能の最新情報は、そのメジャーリリースの最新版のリリースノートを参照してください。

非推奨の機能のサポートステータスは、Red Hat Enterprise Linux 9 では変更されていません。サポート期間の詳細は、Red Hat Enterprise Linux Life Cycle および Red Hat Enterprise Linux Application Streams Life Cycle を参照してください。

現行および今後のメジャーリリースでは、非推奨のハードウェアコンポーネントの新規実装は推奨されません。ハードウェアドライバーの更新は、セキュリティーと重大な修正のみに行われます。Red Hat では、このようなハードウェアの早期交換を推奨します。

パッケージが非推奨となり、使用の継続が推奨されない場合があります。製品からパッケージが削除されることもあります。その場合には、製品のドキュメントで、非推奨となったパッケージと同様、同一、またはより高度な機能を提供する最近のパッケージが指定され、詳しい推奨事項が記載されます。

RHEL 8 で使用され、RHEL 9 で **削除された** 機能の詳細はRHEL 9 の導入における検討事項 を参照してください。

10.1. インストーラーおよびイメージの作成

非推奨のキックスタートコマンド

以下のキックスタートコマンドが非推奨になりました。

- **timezone --ntpservers**
- **timezone --nontp**
- **logging --level**
- **%packages --excludeWeakdeps**
- **%packages --instLangs**
- **%Anaconda**
- **pwpolicy**

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

(BZ#1899167)

10.2. シェルおよびコマンドラインツール

ReaR 設定ファイルでの TMPDIR 変数の設定が非推奨になる

export TMPDIR=... などのステートメントを使用して、**/etc/rear/local.conf** または **/etc/rear/site.conf** ReaR 設定ファイルで **TMPDIR** 環境変数を設定することは、機能せず非推奨です。

ReaR 一時ファイルのカスタムディレクトリーを指定するには、ReaR を実行する前にシェル環境で変数をエクスポートします。たとえば、**export TMPDIR=...** ステートメントを実行してから、同じシェルセッションまたはスクリプトで **rear** コマンドを実行します。

Jira:RHELDOCS-18049

10.3. セキュリティー

SHA-1 は暗号化の目的で非推奨になる

暗号化を目的とした SHA-1 メッセージダイジェストの使用は、RHEL 9 では非推奨になりました。SHA-1 によって生成されたダイジェストは、ハッシュ衝突の検出に基づく多くの攻撃の成功例が記録化されているため、セキュアであるとは見なされません。RHEL コア暗号コンポーネントは、デフォルトで SHA-1 を使用して署名を作成しなくなりました。RHEL 9 のアプリケーションが更新され、セキュリティー関連のユースケースで SHA-1 が使用されないようになりました。

例外の中でも、HMAC-SHA1 メッセージ認証コードと Universal Unique Identifier (UUID) 値は、SHA-1 を使用して作成できます。これは、これらのユースケースが現在セキュリティーリスクをもたらさないためです。SHA-1 は、Kerberos や WPA-2 など、相互運用性および互換性に関する重要な懸念事項に関連する限られたケースでも使用できます。詳細は、RHEL 9 セキュリティーの強化ドキュメント の FIPS 140-3 に準拠していない暗号化を使用する RHEL アプリケーションのリスト を参照してください。

既存またはサードパーティーの暗号署名を検証するために SHA-1 を使用する必要がある場合は、次のコマンドを入力して有効にできます。

```
# update-crypto-policies --set DEFAULT:SHA1
```

または、システム全体の暗号化ポリシーを **LEGACY** ポリシーに切り替えることもできます。**LEGACY** は、セキュアではない他の多くのアルゴリズムも有効にすることに注意してください。

(JIRA:RHELPLAN-110763)

RHEL 9 で SCP が非推奨に

SCP (Secure Copy Protocol) には既知のセキュリティー脆弱性があるため、非推奨となりました。SCP API は RHEL 9 ライフサイクルで引き続き利用できますが、システムセキュリティーが低下します。

- **scp** ユーティリティーでは、SCP はデフォルトで SSH ファイル転送プロトコル (SFTP) に置き換えられます。
- OpenSSH スイートは、RHEL 9 では SCP を使用しません。
- **libssh** ライブラリーで SCP が非推奨に

(JIRA:RHELPLAN-99136)

Digest-MD5 SASL では非推奨となりました。

SASL (Simple Authentication Security Layer) フレームワークの Digest-MD5 認証メカニズムは非推奨になり、将来バージョンのメジャーリリースでは **cyrus-sasl** パッケージから削除される可能性あり

(BZ#1995600)

OpenSSL が MD2、MD4、MDC2、Whirlpool、RIPEMD160、Blowfish、CAST、DES、IDEA、RC2、RC4、RC5、SEED、および PBKDF1 を非推奨化

OpenSSL プロジェクトは、安全でない、一般的でない、またはその両方であるという理由で、一連の暗号アルゴリズムを非推奨にしました。Red Hat もそれらのアルゴリズムの使用を推奨せず、RHEL 9 では、新しいアルゴリズムを使用するために暗号化されたデータを移行するためにそれらを提供しています。ユーザーは、自分のシステムのセキュリティーのためにこれらのアルゴリズムに依存してはいけません。

MD2、MD4、MDC2、Whirlpool、RIPEMD160、Blowfish、CAST, DES、IDEA、RC2、RC4、RC5、SEED、および PBKDF1 のアルゴリズムの実装が、OpenSSL のレガシープロバイダーに移行されました。

レガシープロバイダーをロードし、非推奨のアルゴリズムのサポートを有効にする方法については、**/etc/pki/tls/openssl.cnf** 設定ファイルを参照してください。

(BZ#1975836)

/etc/system-fips が非推奨に

/etc/system-fips ファイルで FIPS モードが削除されることを示すサポートにより、ファイルは今後の RHEL バージョンに含まれなくなります。FIPS モードで RHEL をインストールするには、システムのインストール時に **fips=1** パラメーターをカーネルコマンドラインに追加します。**fips-mode-setup --check** コマンドを使用して、RHEL が FIPS モードで動作しているかどうかを確認できます。

(JIRA:RHELPLAN-103232)

libcrypt.so.1 が非推奨に

libcrypt.so.1 ライブラリーは現在非推奨であり、RHEL の将来のバージョンで削除される可能性があります。

(BZ#2034569)

fapolicyd.rules が非推奨になる

実行ルールの許可と拒否を含むファイルの **/etc/fapolicyd/rules.d/** ディレクトリーは、**/etc/fapolicyd/fapolicyd.rules** ファイルを置き換えます。**fagenrules** スクリプトは、このディレクトリー内のすべてのコンポーネントルールファイルを **/etc/fapolicyd/compiled.rules** ファイルにマージするようになりました。**/etc/fapolicyd/fapolicyd.trust** のルールは引き続き **fapolicyd** フレームワークによって処理されますが、下位互換性を確保するためのみに使用されます。

(BZ#2054740)

10.4. ネットワーク

RHEL 9 でネットワークチームが非推奨になりました

teamd サービスおよび **libteam** ライブラリーは、Red Hat Enterprise Linux 9 では非推奨になり、次回のメジャーリリースでは削除される予定です。代替として、ネットワークチームの代わりにボンディングを設定します。

Red Hat は、機能が類似するボンディングとチームの機能を 2 つ管理しなくてもいいように、カーネルベースのボンディングに注力しています。ボンディングコードは、顧客の採用率が高く、堅牢で、活発なコミュニティー開発が行われています。その結果、ボンディングコードは拡張、更新されます。

ボンディングにチームを移行する方法は、Migrating a network team configuration to network bond を参照してください。

(BZ#1935544)

ifcfg 形式の NetworkManager 接続プロファイルが非推奨に

RHEL 9.0 以降では、**ifcfg** 形式の接続プロファイルは非推奨になりました。次の RHEL メジャーリリースでは、この形式のサポートが削除されます。ただし、RHEL 9 では、既存のプロファイルを変更すると、NetworkManager は引き続きこの形式で既存のプロファイル処理および更新します。

デフォルトでは、NetworkManager は接続プロファイルをキーファイル形式で **/etc/NetworkManager/system-connections/** ディレクトリーに保存するようになりました。**ifcfg** 形式とは異なり、キーファイル形式は、NetworkManager が提供するすべての接続設定をサポートします。キーファイル形式とプロファイルの移行方法の詳細は、NetworkManager connection profiles in keyfile format を参照してください。(BZ#1894877)

firewalld の iptables バックエンドが非推奨に

RHEL 9 では、**iptables** フレームワークは非推奨になりました。結果として、**iptables** バックエンドと、**firewalld** の **直接インターフェイス** も非推奨になりました。**直接インターフェイス** の代わりに、**firewalld** のネイティブ機能を使用して、必要なルールを設定できます。

(BZ#2089200)

10.5. カーネル

RHEL 9 で ATM カプセル化が非推奨になりました

非同期転送モード (ATM) カプセル化により、ATM アダプテーションレイヤー 5(AAL-5) のレイヤー 2(ポイントツーポイントプロトコル、イーサネット) またはレイヤー 3(IP) 接続が可能になります。Red Hat は、RHEL7 以降 ATMNIC ドライバーのサポートを提供していません。ATM 実装のサポートは RHEL 9 で廃止されています。これらのプロトコルは現在、ADSL テクノロジーをサポートし、メーカーによって段階的に廃止されているチップセットのみで使用されています。したがって、ATM カプセル化は Red Hat Enterprise Linux 9 では非推奨です。

詳細については、PPP Over AAL5、Multiprotocol Encapsulation over ATM Adaptation Layer 5 、および Classical IP and ARPoverATM を参照してください。

(BZ#2058153)

10.6. ファイルシステムおよびストレージ

lvm2-activation-generator およびその生成されたサービスが RHEL 9.0 で削除される

lvm2-activation-generator プログラムとその生成されたサービス **lvm2-activation**、**lvm2-activation-early**、および **lvm2-activation-net** は、RHEL 9.0 で削除されています。サービスをアクティベートするために使用される **lvm.conf event_activation** 設定は機能しなくなりました。ボリュームグループを自動アクティブ化する唯一の方法は、イベントベースのアクティブ化です。

(BZ#2038183)

10.7. 動的プログラミング言語、WEB サーバー、およびデータベースサーバー

libdb が非推奨になりました

RHEL 8 および RHEL 9 は、現在、LGPLv2 ライセンスで配布される Berkeley DB (**libdb**) バージョン 5.3.28 を提供しています。アップストリームの Berkeley DB バージョン 6 は、より厳しい AGPLv3 ライセンスで利用できます。

libdb パッケージは、RHEL 9 で非推奨になり、将来バージョンの RHEL では利用できない可能性があります。

また、RHEL 9 では、**libdb** から暗号アルゴリズムが削除され、RHEL 9 では複数の **libdb** 依存関係が削除されています。

libdb のユーザーは、別の鍵値データベースに移行することが推奨されます。詳細は、ナレッジベースの記事 Available replacements for the deprecated Berkeley DB (libdb) in RHEL を参照してください。

(BZ#1927780, BZ#1974657, JIRA:RHELPLAN-80695)

10.8. コンパイラーおよび開発ツール

2048 より小さいサイズのキーは、openssl 3.0 で廃止されました。

2048 ビットより小さい鍵サイズは **openssl** 3.0 で廃止され、Go の FIPS モードでは機能しなくなりました。

(BZ#2111072)

一部の PKCS1 v1.5 モードが非推奨になりました

一部の **PKCS1** v1.5 モードは、**FIPS-140-3** で暗号化が承認されておらず、無効になっています。Go の FIPS モードでは機能しなくなります。

(BZ#2092016)

10.9. IDENTITY MANAGEMENT

OpenDNSSec の SHA-1 が非推奨になりました

OpenDNSSec は、**SHA-1** アルゴリズムを使用したデジタル署名および認証レコードのエクスポートに対応しています。**SHA-1** アルゴリズムの使用に対応しなくなりました。RHEL 9 リリースでは、OpenDNSSec の **SHA-1** が非推奨になり、今後のマイナーリリースで削除される可能性があります。また、OpenDNSSec のサポートは、Red Hat Identity Management との統合に限定されます。OpenDNSSec はスタンドアロンでは対応していません。

(BZ#1979521)

SSSD 暗黙的なファイルプロバイダードメインは、デフォルトで無効になっています。

/etc/shadow などのローカルファイルからユーザー情報を取得する SSSD 暗黙的な **ファイル** プロバイダードメイン、および **/etc/group** からグループ情報を取得する SSSD 暗黙的な <g id="1">ファイル</g> プロバイダードメインは、デフォルトで無効になりました。

SSSD を使用してローカルファイルからユーザーおよびグループ情報を取得するには、次のコマンドを実行します。

1. SSSD を設定します。以下のいずれかのオプションを選択します。
    a. **sssd.conf** 設定ファイルで **id_provider=files** を使用して、ローカルドメインを明示的に設定します。

```
[domain/local]
id_provider=files
...
```

b. **sssd.conf** 設定ファイルで **enable_files_domain=true** を設定して、**ファイル** プロバイダーを有効にします。

```
[sssd]
enable_files_domain = true
```

2. ネームサービススイッチを設定します。

```
# authselect enable-feature with-files-provider
```

(JIRA:RHELPLAN-100639)

-h および -p オプションは、OpenLDAP クライアントユーティリティーで廃止されました。

アップストリームの OpenLDAP プロジェクトは、そのユーティリティーで **-h** および **-p** オプションを廃止し、代わりに **-H** オプションを使用して LDAP URI を指定することを推奨しています。その結果、RHEL 9 では、すべての OpenLDAP クライアントユーティリティーでこれら 2 つのオプションが廃止されました。**-h** および **-p** オプションは、将来のリリースで RHEL 製品から削除される予定です。

(JIRA:RHELPLAN-137660)

SMB1 プロトコルは Samba では非推奨に

Samba 4.11 以降、安全でない Server Message Block バージョン 1 (SMB1) プロトコルは非推奨となり、今後のリリースでは削除される予定です。

セキュリティーを向上させるために、デフォルトでは、Samba サーバーおよびクライアントユーティリティーで SMB1 が無効になっています。

Jira:RHELDOCS-16612

10.10. デスクトップ

GTK 2 が非推奨になりました

レガシー GTK 2 ツールキットと、以下の関連パッケージが非推奨になりました。

- **adwaita-gtk2-theme**
- **gnome-common**
- **gtk2**
- **gtk2-immodules**
- **hexchat**

現在、他にも複数のパッケージが GTK 2 に依存しています。今後の RHEL メジャーリリースで非推奨パッケージへの依存が発生しないよう、これらは変更されます。

GTK 2 を使用するアプリケーションを維持する場合、Red Hat は、アプリケーションを GTK 4 に移植することを推奨します。

(JIRA:RHELPLAN-131882)

10.11. グラフィックインフラストラクチャー

X.org Server が非推奨に

X.org ディスプレイサーバーは非推奨になり、今後の RHEL のメジャーリリースで削除される予定です。ほとんどの場合、デフォルトのデスクトップセッションは **Wayland** セッションになりました。

X11 プロトコルは、**XWayland** バックエンドを使用して完全にサポートされたままです。その結果、**X11** を必要とするアプリケーションは **Wayland** セッションで実行できます。

Red Hat は、**Wayland** セッションの残りの問題、改善点の解決に取り組んでいます。**Wayland** の未解決の問題については、既知の問題 セクションを参照してください。

ユーザーセッションは **X.org** バックエンドに戻すことができます。詳細は、GNOME 環境と表示プロトコルの選択 を参照してください。

(JIRA:RHELPLAN-121048)

Motif は非推奨になりました

アップストリームの Motif コミュニティーでの開発は非アクティブであるため、Motif ウィジェットツールキットは RHEL で非推奨になりました。

開発バリアントおよびデバッグバリアントを含む、以下の Motif パッケージが非推奨になりました。

- **motif**
- **openmotif**
- **openmotif21**
- **openmotif22**

さらに、**motif-static** パッケージが削除されました。

Red Hat は、GTK ツールキットを代替として使用することを推奨します。GTK は Motif と比較してメンテナンス性が高く、新機能を提供します。

(JIRA:RHELPLAN-98983)

10.12. RED HAT ENTERPRISE LINUX システムロール

RHEL 9 ノードでチームを設定すると、networking システムロールが非推奨の警告を表示します

ネットワークチーミング機能は、RHEL 9 では非推奨になりました。その結果、RHEL 8 コントローラーの **networking** RHEL システムロールを使用して RHEL 9 ノードにネットワークチームを設定すると、非推奨に関する警告が表示されます。

(BZ#1999770)

10.13. 仮想化

SHA1 ベースの署名を使用した SecureBoot イメージ検証が非推奨になりました

UEFI (PE/COFF) 実行ファイルでの SHA1 ベースの署名を使用した SecureBoot イメージ検証の実行は非推奨になりました。代わりに、Red Hat は、SHA2 アルゴリズムまたはそれ以降に基づく署名を使用することを推奨します。

(BZ#1935497)

仮想マシンスナップショットのサポートが限定されました

仮想マシンのスナップショットの作成は、現在、UEFI ファームウェアを使用していない仮想マシンのみでサポートされています。さらに、スナップショット操作中に QEMU モニターがブロックされる可能性があり、これは特定のワークロードのハイパーバイザーのパフォーマンスに悪影響を及ぼします。

また、現在の仮想マシンスナップショットの作成メカニズムは非推奨となり、Red Hat は実稼働環境での仮想マシンスナップショットの使用を推奨していないことにも注意してください。ただし、新しい VM スナップショットメカニズムは開発中であり、RHEL 9 の将来のマイナーリリースで完全に実装される予定です。

(JIRA:RHELPLAN-15509, BZ#1621944)

virt-manager が非推奨になりました。

Virtual Machine Manager アプリケーション (**virt-manager**) は非推奨になっています。RHEL Web コンソール (**Cockpit**) は、後続のリリースで置き換えられる予定です。したがって、GUI で仮想化を管理する場合は、Web コンソールを使用することが推奨されます。ただし、**virt-manager** で利用可能な機能によっては、RHEL Web コンソールで利用できない場合があります。

(JIRA:RHELPLAN-10304)

libvirtd が非推奨に

モノリシック **libvirt** デーモン **libvirtd** は、RHEL 9 で非推奨になり、RHEL の将来のメジャーリリースで削除される予定です。ハイパーバイザーで仮想化を管理するために **libvirtd** を引き続き使用できることに注意してください。ただし、Red Hat では、新しく導入されたモジュラー **libvirt** デーモンに切り替えることを推奨します。手順と詳細は、RHEL 9 の仮想化の設定と管理 に関するドキュメントを参照してください。

(JIRA:RHELPLAN-113995)

仮想フロッピードライバーが非推奨に

仮想フロッピーディスクデバイスを制御する **isa-fdc** ドライバーが非推奨になり、今後の RHEL ではサポートされなくなります。そのため、移行した仮想マシンとの前方互換性を確保するため、Red Hat では、RHEL 9 でホストされている仮想マシンでのフロッピーディスクデバイスの使用を推奨しません。

(BZ#1965079)

qcow2-v2 イメージ形式が非推奨になりました。

RHEL 9 では、仮想ディスクイメージの qcow2-v2 形式が非推奨になり、将来バージョンの RHEL ではサポートされなくなります。また、RHEL 9 Image Builder は、qcow2-v2 形式のディスクイメージを作成できません。

Red Hat では、qcow2-v2 の代わりに、qcow2-v3 の使用を推奨しています。qcow2-v2 イメージを、それ以降の形式に変換する場合は、**qemu-img amend** コマンドを使用します。

(BZ#1951814)

レガシー CPU モデルは非推奨になりました

かなりの数の CPU モデルが非推奨になり、RHEL の将来のメジャーリリースで仮想マシン (VM) での使用がサポートされなくなります。非推奨のモデルは次のとおりです。

- Intel の場合: Intel Xeon 55xx および 75xx プロセッサーファミリー (Nehalem とも呼ばれます) より前のモデル
- AMD の場合: AMD Opteron G4 より前のモデル
- IBM Z の場合: IBM z14 より前のモデル

VM が非推奨の CPU モデルを使用しているかどうかを確認するには、**virsh dominfo** ユーティリティーを使用し、**Message** セクションで次のような行を探します。

```
tainted: use of deprecated configuration settings
deprecated configuration: CPU model 'i486'
```

(BZ#2060839)

10.14. コンテナー

RHEL 7 ホストでの RHEL 9 コンテナーの実行がサポート対象外

RHEL 7 ホストでは、RHEL 9 コンテナーの実行に対応していません。正常に動作するかもしれませんが、保証されません。

詳細は、Red Hat Enterprise Linux Container Compatibility Matrix を参照してください。

(JIRA:RHELPLAN-100087)

Podman 内の SHA1 ハッシュアルゴリズムが非推奨に

ルートレスネットワーク namespace のファイル名を生成するために使用される SHA1 アルゴリズムは Podman ではサポートされなくなりました。したがって、Podman 4.1.1 以降に更新する前に起動されたルートレスコンテナーは、ネットワークに参加している場合は (**slirp4netns** を使用するだけでなく) 再起動して、アップグレード後に起動したコンテナーに接続できるようにする必要があります。

(BZ#2069279)

rhel9/pause が非推奨に

rhel9/pause コンテナーイメージが非推奨になりました。

(BZ#2106816)

10.15. 非推奨のパッケージ

このセクションでは、非推奨となり、将来バージョンの Red Hat Enterprise Linux には含まれない可能性があるパッケージのリストを示します。

RHEL 8 と RHEL 9 との間でパッケージを変更する場合は、**RHEL 9 の導入における考慮事項** ドキュメントの パッケージの変更 を参照してください。

重要

非推奨パッケージのサポート状況は、RHEL 9 内でも変更されません。サポート期間の詳細は、Red Hat Enterprise Linux のライフサイクル および Red Hat Enterprise Linux アプリケーションストリームのライフサイクル を参照してください。

次のパッケージは RHEL 9 で非推奨になりました。

- iptables-devel
- iptables-libs
- iptables-nft
- iptables-nft-services
- iptables-utils
- libdb
- mcpp
- mod_auth_mellon
- python3-pytz
- xorg-x11-server-Xorg

第11章 既知の問題

このパートでは、Red Hat Enterprise Linux 9.1 の既知の問題について説明します。

11.1. インストーラーおよびイメージの作成

reboot --kexec コマンドおよび inst.kexec コマンドが、予測可能なシステム状態を提供しない

キックスタートコマンド **reboot --kexec** またはカーネル起動パラメーター **inst.kexec** で RHEL インストールを実行しても、システムの状態が完全な再起動と同じになるわけではありません。これにより、システムを再起動せずにインストール済みのシステムに切り替えると、予期しない結果が発生することがあります。

kexec 機能は非推奨になり、Red Hat Enterprise Linux の今後のリリースで削除されることに注意してください。

(BZ#1697896)

サードパーティーのツールを使用して作成した USB からインストールを起動する際に、Local Media のインストールソースが検出されない

サードパーティーツールを使用して作成した USB から RHEL インストールを起動すると、インストーラーは **Local Media** インストールソースを検出できません (**Red Hat CDN** のみが検出されます)。

この問題は、デフォルトの起動オプション **int.stage2=** が **iso9660** イメージ形式の検索を試みるためです。ただし、サードパーティーツールは、別の形式の ISO イメージを作成する可能性があります。

回避策として、以下のソリューションのいずれかを使用します。

- インストールの起動時に **Tab** キーをクリックしてカーネルコマンドラインを編集し、起動オプション **inst.stage2=** を **inst.repo=** に変更します。
- Windows で起動可能な USB デバイスを作成するには、Fedora Media Writer を使用します。
- Rufus などのサードパーティーツールを使用して起動可能な USB デバイスを作成し、最初に Linux システムで RHEL ISO イメージを再生成すると、サードパーティーのツールを使用して起動可能な USB デバイスを作成します。

指定の回避策を実行する手順の詳細は、RHEL 8.3 のインストール時にインストールメディアは自動検出されない を参照してください。

(BZ#1877697)

キックスタートコマンドの auth および authconfig で AppStream リポジトリーが必要になる

インストール中に、キックスタートコマンドの **auth** および **authconfig** で **authselect-compat** パッケージが必要になります。**auth** または **authconfig** を使用したときに、このパッケージがないとインストールに失敗します。ただし、設計上、 **authselect-compat** パッケージは AppStream リポジトリーでしか利用できません。

この問題を回避するには、BaseOS リポジトリーおよび AppStream リポジトリーがインストーラーで利用できることを確認するか、インストール中にキックスタートコマンドの **authselect** コマンドを使用します。

(BZ#1640697)

ドライバーディスクメニューが、コンソールでユーザー入力を表示できない

ドライバーディスクを使用したカーネルコマンドラインで **inst.dd** オプションを使用して RHEL インストールを開始すると、コンソールはユーザー入力を表示できません。その結果、アプリケーションがユーザー入力に応答せずにフリーズしているように見えますが、出力が表示されるため、ユーザーにとってわかりにくいます。ただし、この動作は機能に影響を与えず、**Enter** を押すとユーザー入力が登録されます。

回避策として、予想される結果を確認するには、コンソールでユーザー入力が存在しないことを無視し、入力の追加が終了したら **Enter** を押します。

(BZ#2109231)

Anaconda がアプリケーションとして実行されているシステムでの予期しない SELinux ポリシー

Anaconda がすでにインストールされているシステムでアプリケーションとして実行されている場合 (たとえば、**–image** anaconda オプションを使用してイメージファイルに別のインストールを実行する場合)、システムはインストール中に SELinux のタイプと属性を変更することを禁止されていません。そのため、SELinux ポリシーの特定の要素は、Anaconda が実行されているシステムで変更される可能性があります。この問題を回避するには、実稼働システムで Anaconda を実行せず、一時的な仮想マシンで実行します。そうすることで、実稼働システムの SELinux ポリシーは変更されません。**boot.iso** や **dvd.iso** からのインストールなど、システムインストールプロセスの一部として anaconda を実行しても、この問題の影響は受けません。

(BZ#2050140)

USB CD-ROM ドライブが Anaconda のインストールソースとして利用できない

USB CD-ROM ドライブがソースで、キックスタート **ignoredisk --only-use=** コマンドを指定すると、インストールに失敗します。この場合、Anaconda はこのソースディスクを見つけ、使用できません。

この問題を回避するには、**harddrive --partition=sdX --dir=/** コマンドを使用して USB CD-ROM ドライブからインストールします。その結果、インストールは失敗しなくなりました。

(BZ#1914955)

iso9660 ファイルシステムで、ハードドライブがパーティション分割されたインストールが失敗する

ハードドライブが **iso9660** ファイルシステムでパーティションが設定されているシステムには、RHEL をインストールできません。これは、**iso9660** ファイルシステムパーティションを含むハードディスクを無視するように設定されている、更新されたインストールコードが原因です。これは、RHEL が DVD を使用せずにインストールされている場合でも発生します。

この問題を回避するには、インストールの開始前に、キックスタートファイルに次のスクリプトを追加して、ディスクをフォーマットします。

メモ: 回避策を実行する前に、ディスクで利用可能なデータのバックアップを作成します。**wipefs** は、ディスク内の全データをフォーマットします。

```
%pre
wipefs -a /dev/sda
%end
```

その結果、インストールでエラーが発生することなく、想定どおりに機能します。

(BZ#1929105)

Anaconda が管理者ユーザーアカウントの存在の確認に失敗する

グラフィカルユーザーインターフェイスを使用して RHEL をインストールしている場合に、管理者アカウントが作成されていると、Anaconda が確認に失敗します。その結果、管理者ユーザーアカウントがなくても、システムをインストールできてしまう可能性があります。

この問題を回避するには、管理者ユーザーアカウントを設定するか、root パスワードを設定して、root アカウントのロックを解除します。その結果、インストール済みシステムで管理タスクを実行できます。

(BZ#2047713)

新しい XFS 機能により、バージョン 5.10 よりも古いファームウェアを持つ PowerNV IBM POWER システムが起動しなくなる

PowerNV IBM POWER システムは、ファームウェアに Linux カーネルを使用し、GRUB の代わりに Petitboot を使用します。これにより、ファームウェアカーネルのマウント **/boot** が発生し、Petitboot が GRUB 設定を読み取り、RHEL を起動します。

RHEL 9 カーネルでは、XFS ファイルシステムに **bigtime=1** 機能および **inobtcount=1** 機能が導入されています。これは、バージョン 5.10 よりも古いファームウェアのカーネルが理解できません。

この問題を回避するには、**/boot** に別のファイルシステム (ext4 など) を使用できます。

(BZ#1997832)

PReP のサイズが 4 または 8 MiB でない場合、RHEL をインストールできません

PowerPC Reference Platform (PReP) パーティションのサイズが 4kiB セクターを使用するディスク上の 4MiB または 8MiB と異なる場合、RHEL インストーラーはブートローダーをインストールできません。その結果、RHEL をディスクにインストールすることはできません。

この問題を回避するには、PReP パーティションのサイズが正確に 4 MiB または 8 MiB であり、サイズが別の値に丸められていないことを確認してください。これにより、インストーラーはディスクに RHEL をインストールできるようになりました。

(BZ#2026579)

マルチパスデバイスを使用したカスタムパーティション設定中に、誤ったディスク領域がインストーラーに表示されます

インストーラーは、カスタムパーティション設定中にマルチパスデバイスの個々のパスを除外しません。これにより、インストーラーがマルチパスデバイスへの個々のパスを表示し、ユーザーは作成したパーティションで使用するマルチパスデバイスへの個々のパスを選択できます。その結果、ディスク領域の誤った合計が表示されます。これは、各パスのサイズを全ディスク容量に加算して算出されます。

回避策として、カスタムパーティション設定中に個々のパスではなくマルチパスデバイスのみを使用し、誤って計算された合計ディスク容量を無視します。

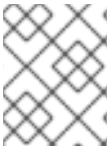
(BZ#2052938)

NVMe over Fibre Channel デバイスでインストールに失敗します

RHEL をインストールすると、インストーラーは、Fiber Channel デバイス上で NVMe (Non-volatile Memory Express) を選択できるようになります。インストールプロセス時のこのようなデバイスの使用はサポートされていません。その結果、インストールプロセスが失敗するか、インストールしたシステムが正常に起動できない場合があります。

この問題を回避するには、対話型インストール (テキストモードまたはグラフィカルモード) では

NVMe over Fibre Channel デバイスを使用しないでください。キックスタートインストールの実行時に、**ignoredisk --drives=<IGNORE_DISKS>** キックスタートコマンドを使用し、**<IGNORE_DISKS>** を NVMe over Fibre Channel デバイスに置き換えて NVMe over Fibre Channel デバイスを無視するようにシステムを設定します。または、**ignoredisk --only-use=<ONLY_USE_DISKS>** の **<ONLY_USE_DISKS>** をサポート対象のデバイスに置き換えて、インストール中にキックスタートが使用するディスクを定義することもできます。



注記

インストールは、NVMe over Fibre Channel デバイスでのみ失敗します。ローカルに接続された NVMe デバイスは正しく機能します。

ignoredisk キックスタートコマンドの詳細は、高度な RHEL 9 インストールの実行ガイドで Kickstart commands for handling storage を参照してください。

(BZ#2107346)

rpm-ostree ペイロードをインストールすると、RHEL for Edge インストーラーイメージがマウントポイントの作成に失敗します

RHEL for Edge インストーラーイメージなどで使用される **rpm-ostree** ペイロードをデプロイする場合、インストーラーはカスタムパーティションの一部のマウントポイントを適切に作成しません。その結果、インストールは以下のエラーで中止されます。

```
The command 'mount --bind /mnt/sysimage/data /mnt/sysroot/data' exited with the code 32.
```

この問題を回避するには、以下を実行します。

- 自動パーティション設定スキームを使用し、手動でマウントポイントを追加しないでください。
- マウントポイントは、**/var** ディレクトリー内のみに手動で割り当てます。たとえば、**/var/my-mount-point** や、/、**/boot**、**/var** などの標準ディレクトリーです。

その結果、インストールプロセスは正常に終了します。

(BZ#2125542)

ネットワークに接続されているが、DHCP または静的 IP アドレスが設定されていない場合、NetworkManager はインストール後に起動に失敗します

RHEL 9.0 以降、特定の **ip=** または kickstart ネットワーク設定が設定されていない場合、Anaconda はネットワークデバイスを自動的にアクティブ化します。Anaconda は、イーサネットデバイスごとにデフォルトの永続的な設定ファイルを作成します。接続プロファイルには、**ONBOOT** と **autoconnect** の値が **true** に設定されています。その結果、インストールされたシステムの起動中に、RHEL がネットワークデバイスをアクティブ化し、**networkManager-wait-online** サービスが失敗します。

回避策として、以下のいずれかを実行します。

- 使用する 1 つの接続を除いて、**nmcli** ユーティリティーを使用してすべての接続を削除します。以下に例を示します。

  a. すべての接続プロファイルを一覧表示します。

  ```
  # nmcli connection show
  ```

b. 不要な接続プロファイルを削除します。

```
# nmcli connection delete <connection_name>
```

<connection_name> を、削除する接続の名前に置き換えます。

- 特定の **ip=** またはキックスタートネットワーク設定が設定されていない場合は、Anaconda の自動接続ネットワーク機能を無効にします。

a. Anaconda GUI で、**Network & Host Name** に移動します。

b. 無効にするネットワークデバイスを選択します。

c. **Configure** をクリックします。

d. **General** タブで、**Connect automatically with priority** の選択を解除します。

e. **Save** をクリックします。

(BZ#2115783)

11.2. サブスクリプションの管理

コマンドの完了後、subscription-manager ユーティリティーがターミナルに不要なテキストを保持する

RHEL 9.1 以降、**subscription-manager** ユーティリティーは操作の処理中に進行状況情報を表示します。一部の言語 (通常は非ラテン語) では、操作の終了後に進行状況メッセージがクリアされない場合があります。その結果、ターミナルに古い進行状況メッセージの一部が表示される場合があります。

これは **subscription-manager** の機能障害ではないことに注意してください。

この問題を回避するには、次のいずれかの手順を実行します。

- ターミナルで subscription-manager コマンドを実行するときに **--no-progress-messages** オプションを含めます
- 次のコマンドを入力して、進行状況メッセージを表示せずに動作するように **subscription-manager** を設定します。

```
# subscription-manager config --rhsm.progress_messages=0
```

(BZ#2136694)

11.3. ソフトウェア管理

インストールプロセスが応答しなくなることがある

RHEL をインストールすると、インストールプロセスが応答しなくなることがあります。**/tmp/packaging.log** ファイルは、最後に以下のメッセージを表示します。

```
10:20:56,416 DDEBUG dnf: RPM transaction over.
```

この問題を回避するには、インストールプロセスを再起動します。

(BZ#2073510)

アップグレードによってアーキテクチャーを変更するパッケージのセキュリティー DNF アップグレードが失敗する

RHBA-2022:8295 アドバイザリーでリリースされた BZ#2108969 のパッチでは、次のリグレッションが導入されました。セキュリティフィルターを使用した DNF アップグレードは、アップグレードによってアーキテクチャが **noarch** に (逆もまた然り) 変更されたパッケージでは失敗します。その結果、システムが脆弱な状態になる可能性があります。

この問題を回避するには、セキュリティーフィルターを使用せずに通常のアップグレードを実行します。

(BZ#2108969)

11.4. シェルおよびコマンドラインツール

設定ファイルで TMPDIR 変数が設定されている場合、ReaR がリカバリー中に失敗する

/etc/rear/local.conf または **/etc/rear/site.conf** ReaR 設定ファイルで **TMPDIR** を設定してエクスポートすることは、機能せず非推奨です。

ReaR のデフォルト設定ファイル **/usr/share/rear/conf/default.conf** には、次の手順が記載されています。

```
# To have a specific working area directory prefix for Relax-and-Recover
# specify in /etc/rear/local.conf something like
#
# export TMPDIR="/prefix/for/rear/working/directory"
#
# where /prefix/for/rear/working/directory must already exist.
# This is useful for example when there is not sufficient free space
# in /tmp or $TMPDIR for the ISO image or even the backup archive.
```

上記の手順は正しく機能しません。これは、**TMPDIR** 変数がレスキュー環境で同じ値を持つためです。**TMPDIR** 変数で指定されたディレクトリーがレスキューイメージに存在しない場合、この値は不適切です。

そのため、**/etc/rear/local.conf** ファイルで **TMPDIR** を設定してエクスポートすると、レスキューイメージの起動時に次のエラーが発生します。

```
mktemp: failed to create file via template '/prefix/for/rear/working/directory/tmp.XXXXXXXXXX': No
such file or directory
cp: missing destination file operand after '/etc/rear/mappings/mac'
Try 'cp --help' for more information.
No network interface mapping is specified in /etc/rear/mappings/mac
```

または、**rear recover** の実行中に次のエラーが発生し、その後中断していました。

```
ERROR: Could not create build area
```

この問題を回避するには、カスタム一時ディレクトリーが必要な場合、ReaR を実行する前にシェル環境で変数をエクスポートして、ReaR 一時ファイル用のカスタムディレクトリーを指定します。たとえば、**export TMPDIR=...** ステートメントを実行してから、同じシェルセッションまたはスクリプトで **rear** コマンドを実行します。その結果、説明した設定でリカバリーが成功します。

Jira:RHEL-24847

ifcfg ファイルを使用したネットワークインターフェイスの名前変更に失敗する

RHEL 9 では、**initscripts** はデフォルトでインストールされません。その結果、**ifcfg** ファイルを使用したネットワークインターフェイスの名前変更に失敗します。この問題を解決するには、**udev** ルールを使用するか、ファイルをリンクしてインターフェイスの名前を変更することが推奨されます。詳細は、一貫したネットワークインターフェイスデバイスの命名 および **systemd.link(5)** の man ページを参照してください。

推奨される方法のいずれも使用できない場合は、**initscripts** パッケージをインストールします。

(BZ#2018112)

RHEL 9 では、 chkconfig パッケージがデフォルトでインストールされない

システムサービス用のランレベル情報を更新およびクエリーする **chkconfig** パッケージは、RHEL 9 ではデフォルトでインストールされません。

サービスを管理するには、**systemctl** コマンドを使用するか、**chkconfig** パッケージを手動でインストールします。

systemd の詳細は、systemd の管理 を参照してください。**systemctl** ユーティリティーの使用方法については、systemctl を使用したシステムサービスの管理 を参照してください。

(BZ#2053598)

11.5. インフラストラクチャーサービス

bind および unbound の両方が SHA-1- ベースの署名の検証を無効化する

bind および **unbound** コンポーネントは、すべての RSA/SHA1 (アルゴリズム番号 5) および RSASHA1-NSEC3-SHA1 (アルゴリズム番号 7) 署名の検証サポートを無効にし、署名の SHA-1 使用は DEFAULT システム全体の暗号化ポリシーで制限されます。

その結果、SHA-1、RSA/SHA1、および RSASHA1-NSEC3-SHA1 ダイジェストアルゴリズムで署名された特定の DNSSEC レコードは、Red Hat Enterprise Linux 9 で検証できず、影響を受けるドメイン名が脆弱になります。

この問題を回避するには、RSA/SHA-256 や楕円曲線キーなどの別の署名アルゴリズムにアップグレードします。

影響を受け脆弱なトップレベルドメインの詳細とリストについては、RSASHA1 で署名された DNSSEC レコードがソリューションを検証できない を参照してください。

(BZ#2070495)

同じ書き込み可能ゾーンファイルが複数のゾーンで使用されていると、named が起動しない

BIND では、複数のゾーンに同じ書き込み可能ゾーンファイルを使用することができません。そのため、**named** で変更可能なファイルへのパスを共有するゾーンが複数存在すると、**named** が起動できなくなります。この問題を回避するには、**in-view** 節を使用して、複数のビュー間で 1 つのゾーンを共有し、異なるゾーンに異なるパスを使用するようにします。たとえば、パスにビュー名を含めます。

書き込み可能なゾーンファイルは通常、動的更新が許可されたゾーン、スレーブゾーン、または DNSSEC が管理するゾーンで使用されることに注意してください。

(BZ#1984982)

コンソール keymap を設定するには、最小限のインストールで libxkbcommon ライブラリーが必要です。

RHEL 9 では、特定の **systemd** ライブラリーの依存関係が動的リンクから動的ロードに変換され、システムが実行時にライブラリーを開いて使用できるようになりました。今回の変更により、必要なライブラリーをインストールしない限り、このようなライブラリーに依存する機能は使用できなくなります。これは、最小限のインストール設定を使用するシステムにおけるキーボードレイアウトの設定にも影響します。その結果、**localectl --no-convert set-x11-keymap gb** コマンドに失敗します。

この問題を回避するには、**libxkbcommon** ライブラリーをインストールします。

```
# dnf install libxkbcommon
```

(BZ#2214130)

11.6. セキュリティー

OpenSSL は、PKCS #11 トークンが、生の RSA 署名または RSA-PSS 署名の作成に対応しているかどうかを検出しません。

TLS 1.3 プロトコルには、RSA-PSS 署名のサポートが必要です。PKCS#11 トークンが生の RSA または RSA-PSS 署名をサポートしていない場合、キーが **PKCS#11** トークンによって保持されていると、**OpenSSL** ライブラリーを使用するサーバーアプリケーションは **RSA** キーを処理できません。これにより、上記のシナリオで TLS 通信に失敗します。

この問題を回避するには、利用可能な最高の TLS プロトコルバージョンとして TLS バージョン 1.2 を使用するようにサーバーとクライアントを設定します。

(BZ#1681178)

OpenSSL が、生の RSA または RSA-PSS の署名に対応していない PKCS #11 トークンを誤って処理する

OpenSSL ライブラリーは、PKCS #11 トークンの鍵関連の機能を検出しません。したがって、生の RSA または RSA-PSS の署名に対応しないトークンで署名が作成されると、TLS 接続の確立に失敗します。

この問題を回避するには、**/etc/pki/tls/openssl.cnf** ファイルの **crypto_policy** セクションの末尾にある **.include** 行の後に、以下の行を追加します。

```
SignatureAlgorithms =
RSA+SHA256:RSA+SHA512:RSA+SHA384:ECDSA+SHA256:ECDSA+SHA512:ECDSA+SHA384
MaxProtocol = TLSv1.2
```

これにより、このシナリオで TLS 接続を確立できます。

(BZ#1685470)

特定の構文の使用時に scp はコピーされたファイルを空にします

scp ユーティリティーが Secure copy protocol (SCP) からよりセキュアな SSH ファイル転送プロトコル (SFTP) に変更されました。したがって、ある場所からファイルを同じ場所にコピーすると、ファイルの内容が消去されます。この問題は以下の構文に影響します。

scp localhost:/myfile localhost:/myfile

この問題を回避するには、この構文を使用して、ソースの場所と同じ宛先にファイルをコピーしないでください。

この問題は、以下の構文に対して修正されました。

- **scp /myfile localhost:/myfile**
- **scp localhost:~/myfile ~/myfile**

(BZ#2056884)

PSK 暗号スイートは FUTURE 暗号ポリシーでは機能しません

事前共有キー (PSK) 暗号スイートは、完全転送秘密 (PFS) キー交換方式を実行しているとは認識されません。結果として、**ECDHE-PSK** および **DHE-PSK** 暗号スイートは、**SECLEVEL=3** に設定された OpenSSL、たとえば **FUTURE** 暗号化ポリシーでは機能しません。回避策として、PSK 暗号スイートを使用するアプリケーションに対して、制限の少ない暗号化ポリシーを設定するか、セキュリティーレベル (**SECLEVEL**) を低く設定することができます。

(BZ#2060044)

GnuPG は、crypto-policies によって許可されていない場合でも、SHA-1 署名の使用を誤って許可します

GNU Privacy Guard (GnuPG) 暗号化ソフトウェアは、システム全体の暗号化ポリシーで定義されている設定に関係なく、SHA-1 アルゴリズムを使用する署名を作成および検証できます。したがって、**DEFAULT** の暗号化ポリシーで暗号化の目的で SHA-1 を使用できます。これは、署名に対するこのセキュアではないアルゴリズムのシステム全体での非推奨とは一致しません。

この問題を回避するには、SHA-1 を含む GnuPG オプションを使用しないでください。これにより、セキュアでない SHA-1 署名を使用して GnuPG がデフォルトのシステムセキュリティーを下げるのを防ぎます。

(BZ#2070722)

GPG-agent が FIPS モードで SSH エージェントとして動作しない

gpg-agent ツールは、FIPS モードが MD5 ダイジェストが無効であっても **ssh-agent** プログラムにキーを追加する際に MD5 フィンガープリントを作成します。その結果、**ssh-add** ユーティリティーは認証エージェントへのキーの追加に失敗します。

この問題を回避するには、**~/.gnupg/sshcontrol** ファイルを **gpg-agent --daemon --enable-ssh-support** コマンドを使用せずに作成します。たとえば、**gpg --list-keys** コマンドの出力を **<FINGERPRINT> 0** 形式で **~/.gnupg/sshcontrol** に貼り付けることができます。これにより、**gpg-agent** は SSH 認証エージェントとして機能します。

(BZ#2073567)

デフォルトの SELinux ポリシーにより、制限のない実行ファイルがスタックを実行可能にする

SELinux ポリシーの **selinuxuser_execstack** ブール値のデフォルトの状態は on です。これは、制限のない実行ファイルがスタックを実行可能にすることを意味します。実行可能ファイルはこのオプションを使用しないでください。また、ハードコーディングされていない実行ファイルや攻撃の可能性を示している可能性があります。ただし、他のツール、パッケージ、およびサードパーティー製品との互換性のため、Red Hat はデフォルトポリシーのブール値を変更できません。シナリオがそのような互換性の側面に依存しない場合は、コマンド **setsebool -P selinuxuser_execstack off** を入力して、ローカルポリシーでブール値をオフにすることができます。

(BZ#2064274)

キックスタートインストール時のサービス関連のルールの修正が失敗する場合があります。

キックスタートのインストール時に、OpenSCAP ユーティリティーで、サービス **enable** または **disable** 状態の修正が必要でないことが誤って表示されることがあります。これにより、OpenSCAP が、インストール済みシステムのサービスを非準拠状態に設定する可能性があります。回避策として、キックスタートインストール後にシステムをスキャンして修復できます。これにより、サービス関連の問題が修正されます。

(BZ#1834716)

SCAP 監査ルールの修正が失敗する

監査設定に関連する一部の SCAP ルールの bash 修復では、修正時に監査キーが追加されません。これは、以下のルールに適用されます。

- **audit_rules_login_events**
- **audit_rules_login_events_faillock**
- **audit_rules_login_events_lastlog**
- **audit_rules_login_events_tallylog**
- **audit_rules_usergroup_modification**
- **audit_rules_usergroup_modification_group**
- **audit_rules_usergroup_modification_gshadow**
- **audit_rules_usergroup_modification_opasswd**
- **audit_rules_usergroup_modification_passwd**
- **audit_rules_usergroup_modification_shadow**
- **audit_rules_time_watch_localtime**
- **audit_rules_mac_modification**
- **audit_rules_networkconfig_modification**
- **audit_rules_sysadmin_actions**
- **audit_rules_session_events**
- **audit_rules_sudoers**
- **audit_rules_sudoers_d**

そのため、関連する監査ルールがすでに存在しているものの、OVAL チェックに完全に準拠していない場合、修復により監査ルールの機能部分が修正され、パスとアクセスビットが監査キーを追加しません。したがって、生成される監査ルールは正常に機能しますが、SCAP ルールは FAIL を誤って報告します。この問題を回避するには、正しいキーを監査ルールに手動で追加します。

(BZ#2120978)

STIG プロファイルの SSH タイムアウトルールが誤ったオプションを設定する

OpenSSH の更新は、次の米国国防情報システム局のセキュリティー技術実装ガイド (DISA STIG) プロファイルのルールに影響を与えました。

- RHEL 9 用 DISA STIG (**xccdf_org.ssgproject.content_profile_stig**)
- RHEL 9 用、GUI の DISA STIG (**xccdf_org.ssgproject.content_profile_stig_gui**)

これらの各プロファイルでは、次の 2 つのルールが影響を受けます。

```
Title: Set SSH Client Alive Count Max to zero
CCE Identifier: CCE-90271-8
Rule ID: xccdf_org.ssgproject.content_rule_sshd_set_keepalive_0

Title: Set SSH Idle Timeout Interval
CCE Identifier: CCE-90811-1
Rule ID: xccdf_org.ssgproject.content_rule_sshd_set_idle_timeout
```

SSH サーバーに適用すると、これらの各ルールは、以前のように動作しなくなったオプション (**ClientAliveCountMax** および **ClientAliveInterval**) を設定します。その結果、OpenSSH は、これらのルールで設定されたタイムアウトに達したときに、アイドル状態の SSH ユーザーを切断しなくなりました。回避策として、これらのルールは、ソリューションが開発されるまで、DISA STIG for RHEL 9 および DISA STIG with GUI for RHEL 9 プロファイルから一時的に削除されました。

(BZ#2038978)

Keylime は、複数の IMA 測定ファイルにアクセスするシステムの認証に失敗する可能性があります

Keylime エージェントを実行するシステムが Integrity Measurement Architecture (IMA) によって測定された複数のファイルにすばやく連続してアクセスする場合、Keylime ベリファイアは IMA ログの追加を誤って処理する可能性があります。その結果、実行中のハッシュが正しいプラットフォーム設定レジスタ (PCR) の状態と一致せず、システムは設定証明に失敗します。現在、回避策はありません。

(BZ#2138167)

Keylim 測定ブートポリシー生成スクリプトにより、セグメンテーションエラーとコアダンプが発生することがある

Keylime で起動認証を測定するためのポリシーを生成する **create_mb_refstate** スクリプトは、提供された入力に応じて **tpm2_eventlog** ツールの出力を処理するときに、**LengthOfDevicePath** フィールドの値を使用する代わりに、**DevicePath** フィールドのデータ長を誤って計算する場合があります。その結果、スクリプトは誤って計算された長さを使用して無効なメモリーにアクセスしようとし、その結果、セグメンテーションエラーとコアダンプが発生します。Keylime の主な機能はこの問題の影響を受けませんが、測定されたブートポリシーを生成できない可能性があります。

この問題を回避するには、メジャーブートポリシーを使用しないか、**tpm2-tools** パッケージの **tpm2_eventlog** ツールを使用して取得したデータから手動でポリシーファイルを作成します。

(BZ#2140670)

一部の TPM 証明書により Keylime レジストラーがクラッシュする

本番デプロイメントで有効にする必要がある **tenant.conf** の **require_ek_cert** 設定オプションは、Keylime テナントが Trusted Platform Module (TPM) からの承認キー (EK) 証明書を必要とするかどうかを決定します。**require_ek_cert** を有効にして最初のアイデンティティクォートを実行すると、Kelime は、EK 証明書を Keylime TPM 証明書ストアに存在する信頼できる証明書と比較して、エージェント上

の TPM デバイスが本物であるかどうかを検証しようとします。ただし、ストア内の一部の証明書は不正な形式の x509 証明書であり、Keylime レジストラーがクラッシュする原因となります。**require_ek_cert** を **false** に設定し、EK 検証を実行する **ek_check_script** オプションでカスタムスクリプトを定義することを除いて、現在、この問題に対する簡単な回避策はありません。

(BZ#2142009)

11.7. ネットワーク

nm-cloud-setup サービスは、手動で設定されたセカンダリー IP アドレスをインターフェイスから削除する

クラウド環境から受け取った情報に基づいて、**nm-cloud-setup** サービスがネットワークインターフェイスを設定します。インターフェイスを手動で設定するには、**nm-cloud-setup** を無効にします。ただし、場合によっては、ホスト上の他のサービスもインターフェイスを設定できます。たとえば、これらのサービスはセカンダリー IP アドレスを追加できます。**nm-cloud-setup** がセカンダリー IP アドレスを削除しないようにするには、

1. **nm-cloud-setup** サービスおよびタイマーを停止して無効にします。

```
# systemctl disable --now nm-cloud-setup.service nm-cloud-setup.timer
```

2. 使用可能な接続プロファイルを表示します。

```
# nmcli connection show
```

3. 影響を受ける接続プロファイルを再アクティブ化します。

```
# nmcli connection up "<profile_name>"
```

その結果、このサービスは、手動で設定されたセカンダリー IP アドレスをインターフェイスから削除しなくなりました。

(BZ#2151040)

セッションキーの更新に失敗すると、接続が切断される

カーネルトランスポートレイヤーセキュリティー (kTLS) プロトコルは、対称暗号で使用されるセッションキーの更新をサポートしていません。その結果、ユーザーはキーを更新することができず、接続が切断されてしまいます。この問題を回避するには、kTLS を無効にしてください。その結果、この回避策により、セッションキーを正常に更新できます。

(BZ#2013650)

initscripts パッケージがデフォルトでインストールされない

デフォルトでは、**initscripts** パッケージはインストールされません。これにより、**ifup** ユーティリティーおよび **ifdown** ユーティリティーが利用できません。別の方法として、**nmcli connection up** コマンドおよび **nmcli connection down** コマンドを使用して、接続を有効および無効にします。提案された代替案がうまくいかない場合は、問題を報告し、**NetworkManager-initscripts-updown** パッケージをインストールしてください。これは、**ifup** および **ifdown** ユーティリティー用の NetworkManager ソリューションを提供します。

(BZ#2082303)

11.8. カーネル

Mellanox ConnectX-5 アダプターの使用中に mlx5 ドライバーが失敗します。

イーサネットスイッチデバイスドライバーモデル (**switchdev**) モードでは、デバイス管理フローステアリング (DMFS) パラメーターと **ConnectX-5** アダプターがサポートするハードウェアを使用して設定されていると、**mlx5** ドライバーが失敗します。その結果、次のエラーメッセージが表示されることがあります。

```
BUG: Bad page cache in process umount pfn:142b4b
```

この問題を回避するには、DMFS の代わりにソフトウェア管理フローステアリング (SMFS) パラメーターを使用する必要があります。

(BZ#2180665)

Secure Boot で fadump を有効にすると、GRUB Out of Memory (OOM) が発生する可能性があります。

Secure Boot 環境では、GRUB と PowerVM は、ブートメモリー用に、RMA (Real Mode Area)と呼ばれる 512 MB のメモリー領域を割り当てます。リージョンはブートコンポーネントに分割され、いずれかのコンポーネントが割り当てを超えると、メモリー不足の問題が発生します。

通常、デフォルトでインストールされている **initramfs** ファイルシステムと **vmlinux** シンボルテーブルは、このような障害を回避するために割り当て制限内に抑えられています。ただし、システムで Firmware Assisted Dump (FADump) が有効になっている場合は、デフォルトの **initramfs** サイズが増加して 95 MB を超える可能性があります。これにより、システムを再起動するたびに GRUB OOM 状態になります。

この問題を回避するには、Secure Boot と FADump を一緒に使用しないでください。この問題の回避方法は、https://www.ibm.com/support/pages/node/6846531 を参照してください。

(BZ#2149172)

kmod の weak-modules がモジュールの相互依存関係で機能しない

kmod パッケージによって提供される **weak-modules** スクリプトは、どのモジュールがインストールされたカーネルと kABI 互換であるかを判別します。ただし、モジュールのカーネル互換性をチェックしている間、**weak-modules は** モジュールシンボルの依存関係を、それらがビルドされたカーネルの上位リリースから下位リリースへと処理します。結果として、異なるカーネルリリースに対して構築された相互依存関係を持つモジュールは互換性がないと解釈される可能性があるため、**weak-modules** はこのシナリオでは機能しません。

この問題を回避するには、新しいカーネルをインストールする前に、最新のストックカーネルに対して追加のモジュールをビルドまたは配置します。

(BZ#2103605)

kdump サービスが IBM Z システムで initrd ファイルの構築に失敗する

64 ビットの IBM Z システムでは、**s390-** subchannels などの **znet** 関連の設定情報が非アクティブな **NetworkManager** 接続プロファイルに存在する場合、**kdump** サービスは初期 RAM ディスク (**initrd**) のロードに失敗します。その結果、**kdump** メカニズムは次のエラーで失敗します。

```
dracut: Failed to set up znet
kdump: mkdumprd: failed to make kdump initrd
```

回避策として、次のいずれかの解決策を使用してください。

- **znet** 設定情報を持つ接続プロファイルを再利用して、ネットワークボンディングまたはブリッジを設定します。

```
$ nmcli connection modify enc600 master bond0 slave-type bond
```

- 非アクティブな接続プロファイルからアクティブな接続プロファイルに **znet** 設定情報をコピーします。

  a. **nmcli** コマンドを実行して、**NetworkManager** 接続プロファイルを照会します。

```
# nmcli connection show

NAME                    UUID              TYPE    Device

bridge-br0           ed391a43-bdea-4170-b8a2 bridge   br0
bridge-slave-enc600  caf7f770-1e55-4126-a2f4 ethernet enc600
enc600               bc293b8d-ef1e-45f6-bad1 ethernet --
```

  b. 非アクティブな接続からの設定情報でアクティブなプロファイルを更新します。

```
#!/bin/bash
 inactive_connection=enc600
 active_connection=bridge-slave-enc600
 for name in nettype subchannels options; do
 field=802-3-ethernet.s390-$name
 val=$(nmcli --get-values "$field"connection show "$inactive_connection")
 nmcli connection modify "$active_connection" "$field" $val"
 done
```

  c. 変更を有効にするために **kdump** サービスを再起動します。

```
# kdumpctl restart
```

(BZ#2064708)

kdump メカニズムは、LUKS 暗号化ターゲットで vmcore ファイルをキャプチャーできない

Linux Unified Key Setup (LUKS) で暗号化されたパーティションを使用するシステムで **kdump** を実行する場合、システムには一定量の使用可能なメモリーが必要です。使用可能なメモリーが必要なメモリー量より少ない場合、**systemd-cryptsetup** サービスはパーティションのマウントに失敗します。その結果、2 番目のカーネルは LUKS 暗号化ターゲット上のクラッシュダンプファイル (**vmcore**) のキャプチャに失敗します。

kdumpctl Estimate コマンドを使用すると、**kdump** に必要な推奨メモリーサイズである **推奨クラッシュカーネル値** を照会できます。

この問題を回避するには、次の手順を使用して、LUKS 暗号化ターゲットで **kdump** に必要なメモリーを設定します。

1. 推定 **crashkernel** 値を出力します。

```
# kdumpctl estimate
```

2. **crashkernel** の値を増やして、必要なメモリー量を設定します。

```
# grubby --args=crashkernel=652M --update-kernel=ALL
```

3. システムを再起動して、変更を反映させます。

```
# reboot
```

これにより、LUKS で暗号化したパーティションがあるシステムで **kdump** が正常に機能します。

(BZ#2017401)

起動時にクラッシュカーネルメモリーの割り当てに失敗する

特定の Ampere Altra システムでは、利用可能なメモリーが 1 GB 未満の場合に、起動中に **kdump** の使用に対してクラッシュカーネルメモリーの割り当てに失敗します。その結果、**kdumpctl** コマンドは **kdump** サービスの起動に失敗します。

この問題を回避するには、以下のいずれかを実行します。

- **crashkernel** パラメーターの値を 240 MB 以上減らしてサイズ要件に合わせます (例: **crashkernel=240M**)。
- **crashkernel=x,high** オプションを使用して、**kdump** 用に 4 GB を超えるクラッシュカーネルメモリーを予約します。

その結果、Ampere Altra システムで **kdump** のクラッシュカーネルメモリー割り当てが失敗しなくなりました。

(BZ#2065013)

デフォルトでは、Delay Accounting 機能は SWAPIN および IO% 統計列を表示しません。

初期のバージョンとは異なり、**Delayed Accounting** 機能はデフォルトで無効になっています。その結果、**iotop** アプリケーションは **SWAPIN** および **IO%** 統計列を表示せず、次の警告を表示します。

```
CONFIG_TASK_DELAY_ACCT not enabled in kernel, cannot determine SWAPIN and IO%
```

taskstats インターフェイスを使用する **Delay Accounting** 機能は、スレッドグループに属するすべてのタスクまたはスレッドの遅延統計を提供します。タスク実行の遅延は、カーネルリソースが利用可能になるのを待つときに発生します。たとえば、空き CPU が実行されるのを待っているタスクです。統計は、タスクの CPU 優先度、I/O 優先度、および **rss** 制限値を適切に設定するのに役立ちます。

回避策として、実行時または起動時に **delayacct** 起動オプションを有効にできます。

- 実行時に **delayacct** を有効にするには、次のように入力します。

```
echo 1 > /proc/sys/kernel/task_delayacct
```

  このコマンドはシステム全体で機能を有効にしますが、このコマンドの実行後に開始したタスクに対してのみ有効であることに注意してください。

- 起動時に **delayacct** を永続的に有効にするには、次のいずれかの手順を使用します。
  - **/etc/sysctl.conf** ファイルを編集して、デフォルトのパラメーターをオーバーライドします。

a. 次のエントリーを **/etc/sysctl.conf** ファイルに追加します。

```
kernel.task_delayacct = 1
```

詳細は、Red Hat Enterprise Linux で sysctl 変数を設定する方法 を参照してください。

b. システムを再起動して、変更を反映させます。

- GRUB 2 設定ファイルを編集して、デフォルトのパラメーターをオーバーライドします。

a. **/etc/default/grub** ファイルの **GRUB _CMDLINE_LINUX** エントリーに **delayacct** オプションを追加します。

b. **grub2-mkconfig** ユーティリティーを実行して、ブート設定を再生成します。

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

詳細については、カーネルコマンドラインを永続的に変更するにはどうすればよいですか? を参照してください。.

c. システムを再起動して、変更を反映させます。

その結果、**iotop** アプリケーションは **SWAPIN** および **IO%** 統計列を表示します。

(BZ#2132480)

kTLS は、TLS 1.3 の NIC へのオフロードをサポートしない

Kernel Transport Layer Security(kTLS) は、TLS 1.3 の NIC へのオフロードをサポートしていません。そのため、NIC が TLS オフロードをサポートしていても、TLS 1.3 によるソフトウェア暗号化が使用されます。この問題を回避するには、オフロードが必要な場合は TLS 1.3 を無効にしてください。その結果、TLS 1.2 のみをオフロードすることができます。TLS 1.3 が使用されている場合、TLS 1.3 をオフロードすることができないため、パフォーマンスが低下します。

(BZ#2000616)

iwl7260-firmware により、Intel Wi-Fi 6 AX200、AX210、および Lenovo ThinkPad P1 Gen 4 で Wi-Fi が切断される

iwl7260-firmware または **iwl7260-wifi** ドライバーを RHEL 8.7 および/または RHEL 9.1 (およびそれ以降) によって提供されるバージョンに更新した後、ハードウェアが正しくない内部状態になります。状態を誤って報告します。その結果、Intel Wifi 6 カードが機能せず、次のエラーメッセージが表示される場合があります。

```
kernel: iwlwifi 0000:09:00.0: Failed to start RT ucode: -110
kernel: iwlwifi 0000:09:00.0: WRT: Collecting data: ini trigger 13 fired (delay=0ms)
kernel: iwlwifi 0000:09:00.0: Failed to run INIT ucode: -110
```

未確認の回避策は、システムの電源をオフにしてから再度オンにすることです。再起動しないでください。

(BZ#2129288)

64 ビット ARM CPU で正しくコンパイルされたドライバーでのプログラム失敗に関して dkms が誤った警告を出す

Dynamic Kernel Module Support (**dkms**) ユーティリティーは、64 ビット ARM CPU のカーネルヘッ

ダーが、ページサイズが 4 キロバイトのカーネルと 64 キロバイトのカーネルの両方で動作することを認識しません。その結果、**dkms** は、カーネルの更新時に **kernel-64k-devel** パッケージがインストールされていない場合、正しくコンパイルされたドライバーでプログラムが失敗した理由に関して誤った警告を出します。この問題を回避するには、**kernel-headers** パッケージをインストールします。このパッケージは、両タイプの ARM CPU アーキテクチャー用のヘッダーファイルを含むもので、**dkms** とその要件に特化したものではありません。

(JIRA:RHEL-25967)

11.9. ブートローダー

grubby の動作はドキュメントから逸脱している

grubby ツールを使用して新しいカーネルを追加し、引数を指定しない場合、**grubby** はデフォルトの引数を新しいエントリーに渡します。**--copy-default** 引数を渡さなくても、この動作が発生します。**--args** および **--copy-default** オプションを使用すると、これらの引数が、**汚い** ドキュメントに記載されているデフォルトの引数に追加されます。

ただし、**$tuned_params** などの追加の引数を追加すると、**--copy-default** オプションが呼び出されない限り、**grubby** ツールはこれらの引数を渡しません。

この状況では、次の 2 つの回避策があります。

- **root=** 引数を設定し、**--args** を空のままにします:

```
# grubby --add-kernel /boot/my_kernel --initrd /boot/my_initrd --args "root=/dev/mapper/rhel-root" --title "entry_with_root_set"
```

- または、**root=** 引数と指定された引数を設定しますが、デフォルトのものは設定しません:

```
# grubby --add-kernel /boot/my_kernel --initrd /boot/my_initrd --args "root=/dev/mapper/rhel-root some_args and_some_more" --title "entry_with_root_set_and_other_args_too"
```

(BZ#2127453)

11.10. ファイルシステムおよびストレージ

cloud-init によってプロビジョニングされ、NFSv3 マウントエントリーで設定された場合、Azure で RHEL インスタンスが起動しません

現在、仮想マシンが **cloud-init** ツールによってプロビジョニングされ、仮想マシンのゲストオペレーティングシステムで **/etc/fstab** ファイルに NFSv3 マウントエントリーがある場合、Microsoft Azure クラウドプラットフォームで RHEL 仮想マシンの起動に失敗します。

(BZ#2081114)

CHAP 認証の試行に失敗した後、no authentication メソッドを使用して iSCSI サーバーにログインできない

CHAP 認証を使用して iSCSI ディスクを追加し、間違った認証情報によりログイン試行に失敗した場合は、**no authentication** 方式でのディスクへの再ログインに失敗します。この問題を回避するには、現行セッションを閉じて、**no authentication** メソッドを使用してログインします。

(BZ#1983602)

デバイスマッパーマルチパスは NVMe/TCP ではサポートされない

nvme-tcp ドライバーで Device Mapper Multipath を使用すると、コールトレースの警告とシステムの不安定性が発生する可能性があります。この問題を回避するには、NVMe/TCP ユーザーはネイティブ NVMe マルチパスを有効にする必要があり、NVMe で **device-mapper-multipath** ツールを使用しないでください。

デフォルトでは、ネイティブ NVMe マルチパスは RHEL 9 で有効になっています。詳細は、Enabling multipathing on NVMe devices を参照してください。

(BZ#2033080)

blk-availability systemd サービスは、複雑なデバイススタックを非アクティブ化する

systemd では、デフォルトのブロック非アクティブ化コードは、仮想ブロックデバイスの複雑なスタックを常に正しく処理するとは限りません。一部の設定では、シャットダウン中に仮想デバイスが削除されない場合があり、エラーメッセージがログに記録されます。この問題を回避するには、次のコマンドを実行して、複雑なブロックデバイススタックを非アクティブ化します。

```
# systemctl enable --now blk-availability.service
```

その結果、複雑な仮想デバイススタックはシャットダウン中に正しく非アクティブ化され、エラーメッセージは生成されません。

(BZ#2011699)

supported_speeds sysfs 属性が誤った速度値を報告します

以前は、**qla2xxx** ドライバーの誤った定義により、HBA の **supported_speeds sysfs** 属性は、予想される 64 Gb/s の速度ではなく 20 Gb/s 速度を報告していました。そのため、HBA が 64 Gb/s リンク速度に対応していると、**sysfs supported_speeds** の値が正しくないため、報告された速度値に影響がありました。

しかし、HBA の **supported_speeds** sysfs 属性は、意図された 64 Gb/s ではなく100 Gb/s、意図された 128 Gb/s ではなく 50 Gb/s の速度を返すようになりました。これは報告された速度の値にのみ影響し、ファイバーチ接続で使用される実際のリンクレートが正しいことを確認します。

(BZ#2069758)

11.11. 動的プログラミング言語、WEB サーバー、およびデータベースサーバー

MySQL および MariaDB の --ssl-fips-mode オプションでは FIPS モードが変更されない

MySQL の **--ssl-fips-mode** オプションと RHEL の **MariaDB** は、アップストリームとは異なる動作をします。

RHEL 9 では、**--ssl-fips-mode** を **mysqld** デーモンまたは **mariadbd** デーモンの引数として使用する場合や、**MySQL** または **MariaDB** サーバー設定ファイルに **ssl-fips-mode** を使用すると、**--ssl-fips-mode** はこれらのデータベースサーバーの FIPS モードを変更しません。

代わりに、以下のようになります。

- **--ssl-fips-mode** を **ON** に設定すると、**mysqld** サーバーデーモンまたは **mariadbd** サーバーデーモンは起動しません。

- FIPS が有効なシステムで **--ssl-fips-mode** を **OFF** に設定すると、**mysqld** サーバーデーモンまたは **mariadbd** サーバーデーモンは FIPS モードで稼働します。

これは、特定のコンポーネントではなく、RHEL システム全体で FIPS モードを有効または無効にする必要があるためです。

したがって、RHEL の **MySQL** または **MariaDB** では **--ssl-fips-mode** オプションを使用しないでください。代わりに、FIPS モードが RHEL システム全体で有効になっていることを確認します。

- FIPS モードが有効な RHEL をインストールすることが推奨されます。インストール時に FIPS モードを有効にすると、システムは FIPS で承認されるアルゴリズムと継続的な監視テストですべての鍵を生成するようになります。FIPS モードで RHEL をインストールする方法は、FIPS モードでのシステムのインストール を参照してください。

- または、FIPS モードへのシステムの切り替え の手順に従って、RHEL システム全体の FIPS モードを切り替えることができます。

(BZ#1991500)

11.12. コンパイラーおよび開発ツール

64 ビット ARM アーキテクチャーの SystemTap で一部のシンボルベースのプローブが動作しない

カーネル設定は、**SystemTap** に必要な特定の機能を無効にします。したがって、一部のシンボルベースのプローブは、64 ビット ARM アーキテクチャーでは機能しません。その結果、影響を受ける **SystemTap** スクリプトが実行されないか、目的のプローブポイントでヒットが収集されない可能性があります。

このバグは、RHBA-2022:5259 アドバイザリーのリリースにより、残りのアーキテクチャーで修正されていることに注意してください。

(BZ#2083727)

11.13. IDENTITY MANAGEMENT

MIT Kerberos は PKINIT の ECC 証明書をサポートしません

MIT Kerberos は、初期認証 (PKINIT) の公開鍵暗号化における楕円曲線暗号化 (ECC) サポートの設計を説明するコメントドキュメントに、RFC5349 要求を実装していません。したがって、RHEL で使用される MIT **krb5-pkinit** パッケージは ECC 証明書に対応していません。詳細は、Elliptic Curve Cryptography (ECC) Support for Public Key Cryptography for Initial Authentication in Kerberos (PKINIT) を参照してください。

(BZ#2106043)

PKINIT が AD KDC に対して機能するように、DEFAULT:SHA1 サブポリシーを RHEL 9 クライアントに設定する必要があります

SHA-1 ダイジェストアルゴリズムは RHEL 9 で非推奨になり、初期認証 (PKINIT) の公開鍵暗号化の CMS メッセージは、より強力な SHA-256 アルゴリズムで署名されるようになりました。

しかし、Active Directory (AD) Kerberos Distribution Center (KDC) は引き続き SHA-1 ダイジェストアルゴリズムを使用して CMS メッセージに署名します。その結果、RHEL 9 Kerberos クライアントは、AD KDC に対して PKINIT を使用してユーザーを認証できません。

この問題を回避するには、次のコマンドを使用して、RHEL 9 システムで SHA-1 アルゴリズムのサポートを有効にします。

```
# update-crypto-policies --set DEFAULT:SHA1
```

(BZ#2060798)

RHEL 9 Kerberos エージェントが RHEL-9 以外および AD 以外の Kerberos エージェントと通信すると、ユーザーの PKINIT 認証に失敗します

クライアントまたは Kerberos Distribution Center (KDC) のいずれかの RHEL 9 Kerberos エージェントが、Active Directory (AD) エージェントではない RHEL-9 Kerberos エージェントとやりとりすると、ユーザーの PKINIT 認証に失敗します。この問題を回避するには、以下のいずれかのアクションを実行します。

- RHEL 9 エージェントの crypto-policy を **DEFAULT:SHA1** に設定して、SHA-1 署名の検証を許可します。

  ```
  # update-crypto-polices --set DEFAULT:SHA1
  ```

- RHEL 9 以外および AD 以外のエージェントを更新して、SHA-1 アルゴリズムを使用して CMS データを署名しないようにします。そのためには、Kerberos パッケージを SHA-1 の代わりに SHA-256 を使用するバージョンに更新します。
  - CentOS 9 Stream: krb5-1.19.1-15
  - RHEL 8.7: krb5-1.18.2-17
  - RHEL 7.9: krb5-1.15.1-53
  - Fedora Rawhide/36: krb5-1.19.2-7
  - Fedora 35/34: krb5-1.19.2-3

その結果、ユーザーの PKINIT 認証が正しく機能します。

他のオペレーティングシステムでは、エージェントが SHA-1 ではなく SHA-256 で CMS データを署名するように krb5-1.20 リリースであることに注意してください。

PKINIT が AD KDC に対して機能するように、DEFAULT:SHA1 サブポリシーを RHEL 9 クライアントに設定する必要がありますも併せて参照してください。

(BZ#2077450)

AD 信頼の FIPS サポートには、AD-SUPPORT 暗号サブポリシーが必要

Active Directory (AD) は、AES SHA-1 HMAC 暗号化タイプを使用します。これは、デフォルトで RHEL 9 の FIPS モードでは許可されていません。AD トラストで RHEL 9 ホストを使用する場合は、IdM ソフトウェアをインストールする前に、AESSHA-1HMAC 暗号化タイプのサポートを有効にしてください。

FIPS 準拠は技術的合意と組織的合意の両方を伴うプロセスであるため、**AD-SUPPORT** サブポリシーを有効にして技術的手段が AES SHA-1 HMAC 暗号化タイプをサポートできるようにする前に、FIPS 監査人に相談してから、RHEL IdM をインストールしてください。

```
# update-crypto-policies --set FIPS:AD-SUPPORT
```

(BZ#2057471)

Heimdal クライアントは、RHEL 9 KDC に対して PKINIT を使用してユーザーを認証できない

デフォルトでは、Heimdal Kerberos クライアントは、Internet Key Exchange (IKE) に Modular Exponential (MODP) Diffie-Hellman Group 2 を使用して、IdM ユーザーの PKINIT 認証を開始します。ただし、RHEL 9 の MIT Kerberos Distribution Center (KDC) は、MODP Group 14 および 16 のみに対応しています。

したがって、Heimdal クライアントで **krb5_get_init_creds: PREAUTH_FAILED** エラーが発生し、RHEL MIT KDC では **Key parameters not accepted** が発生します。

この問題を回避するには、Heimdal クライアントが MODP Group 14 を使用していることを確認してください。クライアント設定ファイルの **libdefaults** セクションで **pkinit_dh_min_bits** パラメーターを 1759 に設定します。

```
[libdefaults]
pkinit_dh_min_bits = 1759
```

その結果、Heimdal クライアントは、RHEL MIT KDC に対する PKINIT 事前認証を完了します。

(BZ#2106296)

FIPS モードの IdM は、NTLMSSP プロトコルを使用してフォレスト間で双方向の信頼を確立しません

FIPS モードが有効な Active Directory (AD)と Identity Management (IdM) との間で双方向のフォレスト間の信頼を確立すると、New Technology LAN Manager Security Support Provider (NTLMSSP) 認証が FIPS に準拠していないため、失敗します。FIPS モードの IdM は、認証の試行時に AD ドメインコントローラーが使用する RC4 NTLM ハッシュを受け入れません。

(BZ#2124243)

IdM から AD へのレルム間の TGS 要求が失敗します

IdM Kerberos チケットの 特権属性証明書 (PAC) 情報は、Active Directory (AD) でサポートされていない AES SHA-2 HMAC 暗号化で署名されるようになりました。

その結果、IdM から AD へのレルム間 TGS 要求 (双方向の信頼の設定) は、以下のエラーを出して失敗します。

```
"Generic error (see e-text) while getting credentials for <service principal>"
```

(BZ#2060421)

FIPS モードで IdM Vault 暗号化および復号化に失敗します

FIPS モードが有効な場合は、OpenSSL RSA-PKCS1v15 パディング暗号化がブロックされます。その結果、現在は IdM が PKCS1v15 パディングを使用してセッションキーをトランスポート証明書でラップするため、Identity Management (IdM) Vault が正しく機能しません。

(BZ#2089907)

ドメイン SID の不一致により、移行した IdM ユーザーがログインできない可能性があります

ipa migrate-ds スクリプトを使用して IdM デプロイメントから別のデプロイメントにユーザーを移行する場合、そのユーザーの以前のセキュリティー識別子 (SID) には現在の IdM 環境のドメイン SID がないため、ユーザーが IdM サービスを使用する際に問題が発生する可能性があります。たとえば、これらの

ユーザーは **kinit** ユーティリティーを使用して Kerberos チケットを取得できますが、ログインできません。この問題を回避するには、ナレッジベースの記事 Migrated IdM users unable to log in due to mismatching domain SIDs を参照してください。

(JIRA:RHELPLAN-109613)

referral mode で起動すると、Directory Server が予期せず終了する

バグにより、Directory Server ではグローバル参照モードが動作しません。**dirsrv** ユーザーとして **refer** オプションを指定して **ns-slapd** プロセスを開始すると、Directory Server はポート設定を無視し、予期せず終了します。**root** ユーザーが SELinux ラベルを変更し、サービスが将来通常モードで開始されないようにプロセスを実行しようとしています。回避策はありません。

(BZ#2053204)

Directory Server で接尾辞の referral の設定に失敗する。

Directory Server でバックエンド参照を設定すると、**dsconf <instance_name> backend suffix set --state referral** コマンドを使用したバックエンドの状態設定に失敗し、次のエラーが表示されます。

```
Error: 103 - 9 - 53 - Server is unwilling to perform - [] - need to set nsslapd-referral before moving to referral state
```

これにより、接尾辞の参照の設定に失敗します。この問題を回避するには、以下のコマンドを実行します。

1. **nsslapd-referral** パラメーターを手動で設定します。

```
# ldapmodify -D "cn=Directory Manager" -W -H ldap://server.example.com

dn: cn=dc\3Dexample\2Cdc\3Dcom,cn=mapping tree,cn=config
changetype: modify
add: nsslapd-referral
nsslapd-referral: ldap://remote_server:389/dc=example,dc=com
```

2. バックエンド状態を設定します。

```
# dsconf <instance_name> backend suffix set --state referral
```

その結果、回避策により、接尾辞の参照を設定できます。

(BZ#2063140)

dsconf ユーティリティーには、entryUUID プラグインの修正タスクを作成するオプションがありません。

dsconf ユーティリティーは、**entryUUID** プラグインの修正タスクを作成するオプションを提供しません。その結果、管理者は **dsconf** を使用して、既存のエントリーに **entryUUID** 属性を自動的に追加するタスクを作成することはできません。回避策として、タスクを手動で作成します。

```
# ldapadd -D "cn=Directory Manager" -W -H ldap://server.example.com -x

dn: cn=entryuuid_fixup_<time_stamp>,cn=entryuuid task,cn=tasks,cn=config
objectClass: top
objectClass: extensibleObject
```

```
basedn: <fixup base tree>
cn: entryuuid_fixup_<time_stamp>
filter: <filtered_entry>
```

タスクが作成された後、Directory Server は **entryUUID** 属性が欠落しているか無効であるエントリーを修正します。

(BZ#2047175)

ldap_id_use_start_tls オプションのデフォルト値を使用する場合の潜在的なリスク

ID ルックアップに TLS を使用せずに **ldap://** を使用すると、攻撃ベクトルのリスクが生じる可能性があります。特に、中間者 (MITM) 攻撃は、攻撃者が、たとえば、LDAP 検索で返されたオブジェクトの UID または GID を変更することによってユーザーになりすますことを可能にする可能性があります。

現在、TLS を強制する SSSD 設定オプション **ldap_id_use_start_tls** は、デフォルトで **false** に設定されています。セットアップが信頼できる環境で動作していることを確認し、**id_provider = ldap** に暗号化されていない通信を使用しても安全かどうかを判断してください。注記: **id_provider = ad** および **id_provider = ipa** は、SASL および GSSAPI によって保護された暗号化接続を使用するため、影響を受けません。

暗号化されていない通信を使用することが安全ではない場合は、**/etc/sssd/sssd.conf** ファイルで **ldap_id_use_start_tls** オプションを **true** に設定して TLS を強制します。デフォルトの動作は、RHEL の将来のリリースで変更される予定です。

(JIRA:RHELPLAN-155168)

11.14. デスクトップ

RHEL 9 にアップグレードすると、Firefox アドオンが無効になります

RHEL8 から RHEL 9 にアップグレードすると、Firefox で以前に有効にしたすべてのアドオンが無効になります。

この問題を回避するには、アドオンを手動で再インストールまたは更新します。その結果、アドオンは予想通りに有効になります。

(BZ#2013247)

User Creation 画面が応答しない

グラフィカルユーザーインターフェイスを使用して RHEL をインストールすると、User Creation の画面が応答しなくなります。そのため、インストール中にユーザーを作成するのが困難です。

この問題を回避するには、以下のソリューションのいずれかを使用してユーザーを作成します。

- VNC モードでインストールを実行し、VNC ウィンドウのサイズを変更します。
- インストールプロセスの完了後にユーザーを作成します。

(BZ#2122636)

RHEL 9 へのアップグレード後に VNC が実行されていない

RHEL8 から RHEL 9 にアップグレードした後、以前に有効にされていたとしても、VNC サーバーは起動に失敗します。

この問題を回避するには、システムのアップグレード後に **vncserver** サービスを手動で有効にします。

```
# systemctl enable --now vncserver@:port-number
```

その結果、VNC が有効になり、システムが起動するたびに期待どおりに起動します。

(BZ#2060308)

11.15. グラフィックインフラストラクチャー

Matrox G200e が VGA ディスプレイに出力を表示しない

以下のシステム設定を使用すると、ディスプレイにグラフィカル出力が表示されない場合があります。

- Matrox G200e GPU
- VGA コントローラーで接続されたディスプレイ

したがって、この設定で RHEL を使用またはインストールすることはできません。

この問題を回避するには、以下の手順に従います。

1. ブートローダーメニューにシステムを起動します。
2. **module_blacklist=mgag200** オプションをカーネルコマンドラインに追加します。

これにより、RHEL が起動し、予想どおりにグラフィカル出力が表示されますが、最大解像度は 16 ビットの色深度で 1024x768 に制限されます。

(BZ#1960467)

Wayland では X.org 設定ユーティリティーが動作しない

画面を操作するための X.org ユーティリティーは、Wayland セッションでは機能しません。特に、**xrandr** ユーティリティーは、処理、解像度、回転、およびレイアウトへのアプローチが異なるため、Wayland では機能しません。

(JIRA:RHELPLAN-121049)

NVIDIA ドライバーが X.org に戻る可能性がある

特定の条件下では、プロプライエタリー NVIDIA ドライバーは Wayland ディスプレイプロトコルを無効にし、X.org ディスプレイサーバーに戻ります。

- NVIDIA ドライバーのバージョンが 470 未満の場合。
- システムがハイブリッドグラフィックスを使用するラップトップの場合。
- 必要な NVIDIA ドライバーオプションを有効にしていない場合。

また、Wayland は有効になっていますが、NVIDIA ドライバーのバージョンが 510 未満の場合には、デスクトップセッションはデフォルトで X.org を使用します。

(JIRA:RHELPLAN-119001)

NVIDIA 使用時に Wayland で Night Light を利用できない

システムで独自の NVIDIA ドライバーが有効になっている場合、GNOME の **Night Light** 機能は Wayland セッションで使用できません。NVIDIA ドライバーは、現在 **Night Light** をサポートしていません。

(JIRA:RHELPLAN-119852)

11.16. WEB コンソール

VNC コンソールが特定の解像度で正しく動作しない

特定のディスプレイ解像度で Virtual Network Computing (VNC) コンソールを使用すると、マウスオフセットの問題が発生したり、インターフェイスの一部しか表示されない場合があります。そのため、VNC コンソールを使用できない場合があります。この問題を回避するには、VNC コンソールのサイズを拡大するか、代わりにコンソールタブのデスクトップビューアーを使用してリモートビューアーを起動します。

(BZ#2030836)

11.17. 仮想化

https または ssh 経由での仮想マシンのインストールに失敗する場合がある

現在、**virt-install** ユーティリティーは、https または ssh 接続を介して ISO ソースからゲストオペレーティングシステム (OS)をインストールしようとすると失敗します。たとえば、**virt-install--cdromhttps://example/path/to/image.iso** を使用します。仮想マシンを作成する代わりに、上述の操作は**internal error: process exited while connecting to monitor**(監視への接続中にプロセスが終了しました) というメッセージで予想外に終了します。

同様に、RHEL 9 Web コンソールを使用してゲスト OS をインストールすると失敗し、https または ssh URL を使用すると **Unknown driver 'https'** エラーが発生するか、**Download OS** 機能が表示されます。

この問題を回避するには、ホストに **qemu-kvm-block-curl** および **qemu-kvm-block-ssh** をインストールして、https および ssh プロトコルのサポートをそれぞれ有効にします。別の接続プロトコルまたは別のインストールソースを使用することもできます。

(BZ#2014229)

仮想マシンで NVIDIA ドライバーを使用すると Wayland が無効になる

現在、NVIDIA ドライバーは Wayland グラフィカルセッションと互換性がありません。これにより、NVIDIA ドライバーを使用する RHEL ゲストオペレーティングシステムは、Wayland を自動的に無効にし、代わりに Xorg セッションを読み込みます。これは主に以下のシナリオで生じます。

- NVIDIA GPU デバイスを RHEL 仮想マシンに渡す場合
- NVIDIA vGPU 仲介デバイスを RHEL 仮想マシンに割り当てる場合

(JIRA:RHELPLAN-117234)

Milan 仮想マシンの CPU タイプは、AMD Milan システムで利用できないことがあります。

一部の AMD Milan システムでは、Enhanced REP MOVSB (**erms**) および Fast Short REP MOVSB (**fsrm**) 機能フラグがデフォルトで BIOS で無効になっています。したがって、**Milan** CPU タイプは、これらのシステムで利用できない可能性があります。さらに、機能フラグ設定が異なる Milan ホスト間の仮想マシンのライブマイグレーションが失敗する可能性があります。これらの問題を回避するには、ホストの BIOS で **erms** および **fsrm** を手動で有効にします。

(BZ#2077767)

AVX を無効にすると、仮想マシンが起動できなくなる

Advanced Vector Extensions (AVX) をサポートする CPU を使用するホストマシンで、現在、AVX を明示的に無効にして VM を起動しようとすると失敗し、代わりに VM でカーネルパニックが発生します。

(BZ#2005173)

移行後に VNC が UEFI VM に接続できない

仮想マシン (VM) の移行中にメッセージキューを有効または無効にすると、移行の完了後に仮想ネットワークコンピューティング (VNC) クライアントが VM に接続できなくなります。

この問題は、Open Virtual Machine Firmware (OVMF) を使用する UEFI ベースの VM にのみ影響します。

(JIRA:RHELPLAN-135600)

フェイルオーバー virtio NIC には、Windows 仮想マシンで IP アドレスが割り当てられていない

現在、フェイルオーバー virtio NIC のみで Windows 仮想マシンを起動すると、仮想マシンは NIC に IP アドレスを割り当てることができません。したがって、NIC はネットワーク接続を設定できません。現在、回避策はありません。

(BZ#1969724)

ネットワークインターフェイスのリセット後に Windows VM が IP アドレスの取得に失敗する

ネットワークインターフェイスの自動リセット後に、Windows 仮想マシンが IP アドレスの取得に失敗することがあります。その結果、VM はネットワークに接続できません。この問題を回避するには、Windows デバイスマネージャーでネットワークアダプタードライバーを無効にしてから再度有効にします。

(BZ#2084003)

Broadcom ネットワークアダプターが、ライブマイグレーション後に Windows VM で正しく動作しない

現在、Broadcom、Qlogic、Marvell などの Broadcom デバイスファミリーのネットワークアダプターは、Windows 仮想マシン (VM) のライブマイグレーション中にホットアンプラグできません。その結果、移行が完了した後、アダプターが正しく動作しません。

この問題は、Single-root I/O virtualization (SR-IOV) を使用して Windows VM に接続されているアダプターにのみ影響します。

(BZ#2090712、BZ#2091528、BZ#2111319)

フェイルオーバー設定のある hostdev インターフェイスは、ホットアンプラグされた後にホットプラグすることはできない

フェイルオーバー設定の **hostdev** ネットワークインターフェイスを実行中の仮想マシン (VM) から削除した後、現在、インターフェイスを同じ実行中の VM に再接続することはできません。

(BZ#2052424)

フェイルオーバー VF を使用した VM のコピー後のライブマイグレーションが失敗する

現在、VM が仮想機能 (VF) フェイルオーバー機能が有効になっているデバイスを使用している場合、実行中の仮想マシン (VM) のコピー後移行の試行は失敗します。この問題を回避するには、コピー後の移行ではなく、標準の移行タイプを使用します。

(BZ#1817965)

ライブマイグレーション中にホストネットワークが VF と VM に ping できません

設定済みの仮想機能 (VF) で仮想マシン (仮想 SR-IOV ソフトウェアを使用する仮想マシンなど) のライブマイグレーションを行う場合、仮想マシンのネットワークは他のデバイスに表示されず、**ping** などのコマンドで仮想マシンに到達できません。ただし、移行が終了すると、問題は発生しなくなります。

(BZ#1789206)

多数のキューを使用すると、Windows 仮想マシンで障害が発生することがある

仮想 Trusted Platform Module (vTPM) デバイスが有効で、**マルチキュー virtio-net** 機能が 250 を超えるキューを使用するように設定されている場合、Windows 仮想マシン (VM) が失敗することがあります。

この問題は、vTPM デバイスの制限が原因で発生します。vTPM デバイスには、開いているファイル記述子の最大数に関するハードコーディングされた制限があります。新しいキューごとに複数のファイル記述子が開かれるため、内部の vTPM 制限を超えて VM が失敗する可能性があります。

この問題を回避するには、次の 2 つのオプションのいずれかを選択します。

- vTPM デバイスを有効のままにしますが、使用するキューは 250 未満にします。
- 250 を超えるキューを使用するには、vTPM デバイスを無効にします。

(BZ#2020146)

PCIe ATS デバイスが Windows 仮想マシンで動作しない

Windows ゲストオペレーティングシステムを使用して仮想マシン (VM) の XML 設定で PCIe アドレス変換サービス (ATS) デバイスを設定しても、ゲストが仮想マシンの起動後に ATS デバイスを有効にしません。これは、Windows が現在 **virtio** デバイス上の ATS をサポートしていないためです。

詳細は、Red Hat ナレッジベース を参照してください。

(BZ#2073872)

AMD SEV-SNP を搭載した仮想マシンで Kdump が失敗する

現在、Secure Nested Paging (SNP) 機能を備えた AMD Secure Encrypted Virtualization (SEV) を使用する RHEL 9 仮想マシン (VM) では kdump が失敗します。

(JIRA:RHEL-10019)

11.18. クラウド環境の RHEL

Nutanix AHV で LVM を使用する RHEL 9 仮想マシンのクローンを作成または復元すると、ルート以外のパーティションが表示されなくなる

Nutanix AHV ハイパーバイザーをホストとする仮想マシン (VM) で RHEL 9 ゲストオペレーティングシステムを実行する場合、スナップショットから VM を復元するか VM をクローンすると、ゲストが論理ボリューム管理 (LVM) を使用している場合は VM 内の非ルートパーティションを消失させることがあります。これにより、以下の問題が発生します。

- スナップショットから仮想マシンを復元すると、仮想マシンは起動できず、緊急モードに入ります。

- クローンを作成して作成した仮想マシンは起動できず、緊急モードに入ります。

これらの問題を回避するには、仮想マシンの緊急モードで以下を行います。

1. 以下の LVM システムデバイスファイルを削除します: **rm/etc/lvm/devices/system.devices**
2. LVM デバイス設定を再作成します。**vgimportdevices -a**
3. 仮想マシンを再起動します。

これにより、クローン化または復元された VM を正しく起動できます。

または、問題が発生しないようにするには、VM のクローンを作成する前、または VM のスナップショットを作成する前に、次の手順を実行します。

1. **/etc/lvm/lvm.conf** ファイルの **use_devicesfile = 0** 行のコメントを外します
2. 仮想マシンを再起動します。

(BZ#2059545)

ESXi で RHEL 9 ゲストをカスタマイズすると、ネットワークの問題が発生することがある

現在、VMware ESXi ハイパーバイザーでの RHEL 9 ゲストオペレーティングシステムのカスタマイズは、NetworkManager キーファイルでは正しく機能しません。その結果、ゲストがそのようなキーファイルを使用している場合、IP アドレスやゲートウェイなどのネットワーク設定が正しくなくなります。

詳細と回避策は、VMware ナレッジベース を参照してください。

(BZ#2037657)

VMware ホストの RHEL 仮想マシンで静的 IP を設定できません

現在、VMware ホストで RHEL を仮想マシンのゲストオペレーティングシステムとして使用すると、DatasourceOVF 機能は正しく機能しません。これにより、**cloud-init** ユーティリティーを使用して、仮想マシンのネットワークを静的 IP に設定し、仮想マシンを再起動すると、仮想マシンのネットワークが DHCP に変更されます。

(BZ#1750862)

11.19. サポート性

IBM Power Systems (Little Endian) で sos report を実行するとタイムアウトする

数百または数千の CPU を搭載した IBM Power Systems (Little Endian) で **sos report** コマンドを実行すると、**/sys/devices/system/cpu** ディレクトリーの膨大なコンテンツを収集する際のプロセッサープラグインはデフォルトのタイムアウトである 300 秒に達します。回避策として、それに応じてプラグインのタイムアウトを増やします。

- 1 回限りの設定の場合は、次を実行します。

```
# sos report -k processor.timeout=1800
```

- 永続的な変更を行うには、**/etc/sos/sos.conf** ファイルの **[plugin_options]** セクションを編集します。

```
[plugin_options]
```

```
# Specify any plugin options and their values here. These options take the form
# plugin_name.option_name = value
#rpm.rpmva = off
processor.timeout = 1800
```

値の例は 1800 に設定されています。特定のタイムアウト値は、特定のシステムに大きく依存します。プラグインのタイムアウトを適切に設定するには、次のコマンドを実行して、タイムアウトなしで 1 つのプラグインを収集するために必要な時間を最初に見積もることができます。

```
# time sos report -o processor -k processor.timeout=0 --batch --build
```

(BZ#1869561)

11.20. コンテナー

古いコンテナーイメージ内で systemd を実行すると動作しない

古いコンテナーイメージ (例:**centos:7**) で systemd を実行しても動作しません。

```
$ podman run --rm -ti centos:7 /usr/lib/systemd/systemd
 Storing signatures
 Failed to mount cgroup at /sys/fs/cgroup/systemd: Operation not permitted
 [!!!!!!] Failed to mount API filesystems, freezing.
```

この問題を回避するには、以下のコマンドを使用します。

```
# mkdir /sys/fs/cgroup/systemd
# mount none -t cgroup -o none,name=systemd /sys/fs/cgroup/systemd
# podman run --runtime /usr/bin/crun --annotation=run.oci.systemd.force_cgroup_v1=/sys/fs/cgroup -
-rm -ti centos:7 /usr/lib/systemd/systemd
```

(JIRA:RHELPLAN-96940)

付録A コンポーネント別のチケットリスト

本書には Bugzilla と JIRA ID が記載されています。一般にアクセス可能な Bugzilla バグには、チケットへのリンクが含まれます。

コンポーネント	チケット
389-ds-base	BZ#2052527、BZ#2057063、BZ#2057066、BZ#1872451、BZ#2053204、BZ#2063140、BZ#2047175
NetworkManager	BZ#2068525、BZ#2059608、BZ#2030997、BZ#2079849、BZ#2097293、BZ#2029636、BZ#1894877、BZ#2151040
anaconda	BZ#2059414、BZ#2053710、BZ#2082132、BZ#2050140、BZ#1877697、BZ#1914955、BZ#1929105、BZ#1997832、BZ#2052938、BZ#2107346、BZ#2125542、BZ#2115783
ansible-collection-microsoft-sql	BZ#2066337
ansible-collection-redhat-rhel_mgmt	BZ#2112434
ansible-freeipa	BZ#2076567
bind	BZ#1984982
catatonit	BZ#2074193
chrony	BZ#2047415, BZ#2051441
clevis	BZ#2107078
cloud-init	BZ#1750862
cockpit-appstream	BZ#2030836
cockpit	BZ#2056786
cronie	BZ#2090691
crypto-policies	BZ#2102774, BZ#2070604
cyrus-sasl	BZ#1995600
device-mapper-multipath	BZ#2084365、BZ#2033080、BZ#2011699

コンポーネント	チケット
distribution	BZ#2063773
dnf-plugins-core	BZ#2066646
dnf	BZ#2053014、BZ#2073510
dotnet7.0	BZ#2112027
dyninst	BZ#2057675
edk2	BZ#1935497
elfutils	BZ#2088774
fapolicyd	BZ#2100041, BZ#2054740, BZ#2070655
firefox	BZ#2013247
firewalld	BZ#2040689、BZ#2039542
frr	BZ#2069563
gcc-toolset-12-annobin	BZ#2077438
gcc-toolset-12-binutils	BZ#2077445
gcc-toolset-12-gcc	BZ#2077465
gcc-toolset-12-gdb	BZ#2077494
gcc	BZ#2063255
gdb	BZ#1870017
gdm	BZ#2097308
gimp	BZ#2047161
glibc	BZ#2033683、BZ#2096191、BZ#2063142、 BZ#2077838、BZ#2085529、BZ#2003291、BZ#2091549
gnome-settings-daemon	BZ#2100467
gnupg2	BZ#2070722, BZ#2073567

コンポーネント	チケット
gnutls	BZ#2042009
golang	BZ#2075169、BZ#2111072、BZ#2092016
grub2	BZ#2074761、BZ#2026579
grubby	BZ#1978226、BZ#1969362、BZ#2127453
httpd	BZ#2079939, BZ#2065677
ipa	BZ#747959、BZ#2091988、BZ#2083218、BZ#2100227、BZ#2084180、BZ#2084166、BZ#2069202、BZ#2057471、BZ#2124243、BZ#2089907
jmc-core	BZ#1980981
kdump-anaconda-addon	BZ#1959203、BZ#2017401
kernel-rt	BZ#2061574
kernel	JIRA:RHELPLAN-117713, BZ#2027894, BZ#2066451, BZ#2079368, BZ#2065226, BZ#2013413, BZ#2069045, BZ#2001936, BZ#2097188, BZ#2096127, BZ#2054379, BZ#2073541, BZ#2030922, BZ#1945040, BZ#2100898, BZ#2068432, BZ#2046472, BZ#1613522, BZ#1874182, BZ#1995338, BZ#1570255, BZ#2023416, BZ#2021672, BZ#2000616, BZ#2013650, BZ#2132480, BZ#2060150, BZ#2059545, BZ#2069758, BZ#1960467, BZ#2005173, BZ#2129288
kexec-tools	BZ#2064708、BZ#2065013
キーライム	BZ#2138167、BZ#2140670、BZ#2142009
kmod-kvdo	BZ#2064802
kmod	BZ#2103605
krb5	BZ#2068935, BZ#2106043, BZ#2060798, BZ#2077450, BZ#2106296, BZ#2060421
libdnf	BZ#2108969
libnvme	BZ#2099619
libsepol	BZ#2069718, BZ#2079276
libvirt	BZ#2064194、BZ#2014487

コンポーネント	チケット
libvpd	BZ#2051288
libxcrypt	BZ#2034569
llvm-toolset	BZ#2061041
lsvpd	BZ#2051289
lvm2	BZ#2038183
maven	BZ#2083112
mysql	BZ#1991500
nfs-utils	BZ#2081114
nmstate	BZ#2084474, BZ#2082043
nodejs	BZ#2083072
nss	BZ#2091905
nvme-cli	BZ#2090121
nvme-stas	BZ#1893841
open-vm-tools	BZ#2061193, BZ#2037657
opencryptoki	BZ#2044179
openscap	BZ#2109485
openssh	BZ#2066882, BZ#2087121, BZ#2056884
openssl	BZ#2060510、BZ#2053289、BZ#2066412、BZ#2063947、BZ#2004915、BZ#2058663、BZ#1975836、BZ#1681178、BZ#1685470、BZ#2060044、BZ#2071631
pacemaker	BZ#2121838, BZ#2072108
pause-container	BZ#2106816
pcre2	BZ#2086494

コンポーネント	チケット
pcs	BZ#2024522, BZ#2054671, BZ#2058251, BZ#2058252, BZ#2058246, BZ#2058243, BZ#1301204
php	BZ#2070040
pki-core	BZ#2084181
podman	BZ#2097708、 BZ#2027576、 BZ#2069279
policycoreutils	BZ#2115242
powerpc-utils	BZ#1920964
ppc64-diag	BZ#2051286
procps-ng	BZ#2052536, BZ#2003033
pykickstart	BZ#2083269
qemu-kvm	BZ#2044218、 BZ#1965079、 BZ#1951814、 BZ#2060839、 BZ#2014229、 BZ#2052424、 BZ#1817965、 BZ#1789206、 BZ#2090712、 BZ#2020146
rear	BZ#2111059, BZ#2097437, BZ#2115958, BZ#2083272, BZ#2120736, BZ#2119501
resource-agents	BZ#1826455
rhel-system-roles	BZ#2072385、BZ#2086965、BZ#2065337、BZ#2079622、BZ#2043010、BZ#2065383、BZ#2112145、BZ#2052081、BZ#2052086、BZ#2065392、BZ#2072742、BZ#2072745、BZ#2072746、BZ#2075119、BZ#2078989、BZ#2079627、BZ#2093423、BZ#2100292、BZ#2100942、BZ#2115154、BZ#2115157、BZ#2115152、BZ#2051737、BZ#2065382、BZ#2065394、BZ#2115886、BZ#2100605、BZ#2060523、BZ#2060525、BZ#2065393、BZ#2070462、BZ#2083376、BZ#2083410、BZ#2100286、BZ#2109998、BZ#2115156、BZ#2071804、BZ#2100294、BZ#1999770
rsyslog	BZ#2064318
rust	BZ#2075337
s390utils	BZ#1870699, BZ#1932480
samba	BZ#2077487、 Jira:RHELDOCS-16612
sblim-wbemcli	BZ#2083577

コンポーネント	チケット
scap-security-guide	BZ#2070563、BZ#2120978、BZ#2038978
selinux-policy	BZ#1965013, BZ#2081425, BZ#2076681, BZ#2064274
sos	BZ#1869561
sssd	BZ#1978119, BZ#2065693, BZ#2056482
stalld	BZ#2107275
stratisd	BZ#1990905、BZ#2040352、BZ#2039960、BZ#2007018、BZ#2005110、BZ#2041558
subscription-manager	BZ#2092014、BZ#2136694
systemd	BZ#2018112
systemtap	BZ#2083727
tigervnc	BZ#2060308
tpm2-tools	BZ#2090748
tuned	BZ#2093847
ubi8-container	BZ#2120378
udisks2	BZ#1983602
unbound	BZ#2087120、BZ#2071543、BZ#2070495
valgrind	BZ#1993976
virt-who	BZ#2054504
virtio-win	BZ#1969724、BZ#2084003
whois	BZ#2054043
xmlstarlet	BZ#2069689
xorg-x11-server	BZ#1894612

コンポーネント	チケット
その他	JIRA:RHELPLAN-92522, BZ#2125549, BZ#2128016, BZ#1937031, JIRA:RHELPLAN-121982, JIRA:RHELPLAN-95456, JIRA:RHELPLAN-122321, JIRA:RHELPLAN-118462, JIRA:RHELPLAN-101140, JIRA:RHELPLAN-132023, JIRA:RHELPLAN-123369, JIRA:RHELPLAN-117109, JIRA:RHELPLAN-130379, BZ#2049492, JIRA:RHELPLAN-130376, JIRA:RHELPLAN-122735, BZ#2070793, BZ#2122716, JIRA:RHELPLAN-123368, JIRA:RHELPLAN-135601, JIRA:RHELPLAN-135602, BZ#2139877, JIRA:RHELPLAN-122776, JIRA:RHELPLAN-121180, BZ#2094015, JIRA:RHELPLAN-109067, JIRA:RHELPLAN-115603, JIRA:RHELPLAN-65217, BZ#2020529, BZ#2030412, BZ#2046653, JIRA:RHELPLAN-103993, JIRA:RHELPLAN-122345, JIRA:RHELPLAN-129327, JIRA:RHELPLAN-74672, BZ#1927780, JIRA:RHELPLAN-110763, BZ#1935544, BZ#2089200, JIRA:RHELPLAN-15509, JIRA:RHELPLAN-99136, JIRA:RHELPLAN-103232, BZ#1899167, BZ#1979521, JIRA:RHELPLAN-100087, JIRA:RHELPLAN-100639, JIRA:RHELPLAN-10304, BZ#2058153, JIRA:RHELPLAN-113995, JIRA:RHELPLAN-121048, JIRA:RHELPLAN-98983, JIRA:RHELPLAN-131882, JIRA:RHELPLAN-137660, BZ#1640697, BZ#1697896, BZ#2047713, JIRA:RHELPLAN-96940, JIRA:RHELPLAN-117234, JIRA:RHELPLAN-119001, JIRA:RHELPLAN-119852, BZ#2077767, BZ#2053598, BZ#2082303, JIRA:RHELPLAN-121049, JIRA:RHELPLAN-109613, JIRA:RHELPLAN-135600, BZ#2149172

付録B 改訂履歴

0.2-6

2024 年 6 月 11 日火曜日、Brian Angelica (bangelic@redhat.com)

- 非推奨の機能 RHELDOCS-18049 (シェルとコマンドラインツール) を追加しました。

0.2-5

2024 年 6 月 11 日火曜日、Brian Angelica (bangelic@redhat.com)

- 既知の問題 JIRA:RHEL-24847 (シェルとコマンドラインツール) を追加しました。

0.2-4

2024 年 5 月 16 日 (木)、Gabriela Fialová (gfialova@redhat.com)

- 既知の問題 JIRA:RHEL-10019 (仮想化) を追加しました。

0.2-3

2024 年 3 月 14 日 (木)、Gabriela Fialová (gfialova@redhat.com)

- 既知の問題 JIRA:RHEL-25967 (カーネル) を追加しました。

0.2-2

2024 年 2 月 1 日 (木)、Gabriela Fialová (gfialova@redhat.com)

- KI BZ#1834716 (セキュリティー) を追加しました。

0.2-1

2023 年 11 月 13 日月曜日、Gabriela Fialová (gfialova@redhat.com)

- テクノロジープレビュー JIRA:RHELDOCS-17040 (仮想化) を追加しました。

0.2-0

2023 年 11 月 10 日金曜日、Gabriela Fialová (gfialova@redhat.com)

- RHEL ドキュメントへのフィードバックの提供に関するモジュールを更新しました。

0.1-9

2023 年 11 月 10 日金曜日、Gabriela Fialová (gfialova@redhat.com)

- テクノロジープレビュー JIRA:RHELDOCS-17050 (仮想化) を追加しました。

0.1-8

2023 年 10 月 13 日 (金) Gabriela Fialová (gfialova@redhat.com)

- テクノロジープレビュー JIRA:RHELDOCS-16861 (コンテナー) を追加しました。

0.1-7

2023 年 9 月 25 日、Gabriela Fialová (gfialova@redhat.com)

- KI BZ#2122636 (デスクトップ) を追加しました。

0.1-6

2023 年 9 月 8 日、Marc Muehlfeld (mmuehlfeld@redhat.com)

- 非推奨機能のリリースノート JIRA:RHELDOCS-16612 (Samba) を追加しました。
- JIRA の RHEL を反映して「Red Hat ドキュメントへのフィードバック」を更新しました。

0.1-5

2023 年 8 月 17 日、Gabriela Fialová (gfialova@redhat.com)

- Enh BZ#2136937 (Plumbers) を追加しました。

0.1-4

2023 年 8 月 7 日、Gabriela Fialová (gfialova@redhat.com)

- KI BZ#2214130 (CS)を追加しました。

0.1-3

2023 年 8 月 2 日、Marc Muehlfeld (mmuehlfeld@redhat.com)

- 非推奨機能のリリースノート BZ#1894877 (NetworkManager) を更新しました。

0.1-2

2023 年 7 月 25 日、Gabriela Fialová (gfialova@redhat.com)

- 既知の問題 BZ#2109231 (インストーラー) を追加しました。

0.1-1

2023 年 6 月 15 日 (木) Lucie Vařáková (lvarakova@redhat.com)

- 新機能 BZ#2070725 (ブートローダー) を追加
- その他の若干の更新

0.1-0

2023 年 5 月 17 日 (水) Gabriela Fialová (gfialova@redhat.com)

- 非推奨のパッケージ セクションのライフサイクル情報を更新

0.0-9

2023 年 4 月 27 日 (木) Gabriela Fialová (gfialova@redhat.com)

- 既知の問題 JIRA:RHELPLAN-155168 (アイデンティティー管理) を追加

0.0-8

2023 年 4 月 25 日、Lucie Vařáková (lvarakova@redhat.com)

- 既知の問題 BZ#2180665 (カーネル) を追加

0.0-7

2023 年 2 月 20 日 (月) Gabriela Fialová (gfialova@redhat.com)

- SAP 環境に関する情報を In-place upgrade from RHEL 8 to RHEL 9 に追加

0.0-6

2023 年 2 月 16 日 (木) Gabriela Fialová (gfialova@redhat.com)

- 既知の問題 BZ#2132480 (カーネル) を追加

0.0-5

2023 年 2 月 14 日 (火) Gabriela Fialová (gfialova@redhat.com)

- 外部カーネルパラメーターへの重要な変更 で書式設定を少々変更

0.0-4

2023 年 2 月 14 日 (火)、Marc Muehlfeld (mmuehlfeld@redhat.com)

- BZ#2144898 (ネットワーク) の拡張機能を追加

0.0-3

2022 年 12 月 7 日水曜日、Gabriela Fialová (gfialova@redhat.com)

- **nodejs:18** モジュールストリームの BZ#2083072 をテクノロジープレビューから完全に対応した機能 (動的プログラミング言語、Web サーバー、およびデータベースサーバー) へ移行

0.0-2

2022 年 11 月 16 日水曜日、Gabriela Fialová (gfialova@redhat.com)

- Red Hat Enterprise Linux 9.1 リリースノートのリリース

0.0-1

2022 年 9 月 28 日 (水) Gabriela Fialová (gfialova@redhat.com)

- Red Hat Enterprise Linux 9.1 Beta リリースノートのリリース