Red Hat Software Collections 3

3.1 リリースノート

Red Hat Software Collections 3.1 のリリースノート

Last Updated: 2024-02-28

Red Hat Software Collections 3 3.1 リリースノート

Red Hat Software Collections 3.1 のリリースノート

Lenka Špačková
Red Hat Customer Content Services
lspackova@redhat.com

Jaromír Hradílek
Red Hat Customer Content Services
jhradilek@redhat.com

Eliška Slobodová
Red Hat Customer Content Services

法律上の通知

概要

Red Hat Software Collections 3.1 リリースノートには、Red Hat Software Collections 3.1 の主な機能が記載されており、既知の問題に関する重要な情報が含まれています。 Red Hat Developer Toolset コレクションは、Red Hat Developer Toolset リリースノート および Red Hat Developer Toolset ユーザーガイド に記載されています。

目次

第1章 RED HAT SOFTWARE COLLECTIONS 3.1

この章では、Red Hat Software Collections 3.1 のコンテンツセットの概要を説明します。コンポーネントのリストとその説明、このバージョンでの変更点のまとめ、関連する互換性情報の文書化、既知の問題のリストを提供します。

1.1. RED HAT SOFTWARE COLLECTIONS について

アプリケーションによっては、最新の新機能を使用するために、より新しいバージョンのソフトウェアコンポーネントが必要になることがあります。**Red Hat Software Collections**は、基本的な Red Hat Enterprise Linux システムに含まれる同等のバージョンよりも新しい、またはこのシステムで最初に利用可能になった動的プログラミング言語、データベースサーバー、およびさまざまな関連パッケージのセットを提供する Red Hat 製品です。

Red Hat Software Collections 3.1 は Red Hat Enterprise Linux 7 で利用可能であり、選択された新しいコンポーネントおよび以前にリリースされたコンポーネントは Red Hat Enterprise Linux 6 でも利用可能です。Red Hat Software Collections の一部として配布されているコンポーネントの完全なリストとその機能の簡単な概要については、「主な特長」 を参照してください。

Red Hat Software Collections では、Red Hat Enterprise Linux 6 または Red Hat Enterprise Linux 7 で提供されるデフォルトのシステムツールは置き換えられません。その代わりに、並列のツールセットが **/opt/** ディレクトリーにインストールされ、提供された **scl** ユーティリティーを使用してユーザーがアプリケーションごとにオプションで有効にできます。たとえば、Perl または PostgreSQL のデフォルトのバージョンは、ベース Red Hat Enterprise Linux システムが提供するバージョンのままになります。

すべての Red Hat Software Collections コンポーネントは、Red Hat Enterprise Linux サブスクリプション契約で完全にサポートされ、機能的に完全で、実稼働環境での使用を目的としています。重要なバグ修正とセキュリティーエラータは、Red Hat Enterprise Linux と同様に、各メジャーバージョンのリリースから少なくとも 2 年間は Red Hat Software Collections サブスクライバーに発行されます。各メジャーリリースストリームでは、選択したコンポーネントの各バージョンは後方互換性を維持します。個別コンポーネントのサポート期間の詳細は、Red Hat Software Collections Product Life Cycle を参照してください。

1.1.1. Red Hat Developer Toolset

Red Hat Developer Toolsetは、個別の Software Collection として同梱される Red Hat Software Collections の一部です。Red Hat Developer Toolset の詳細については、Red Hat Developer Toolset Release Notes と Red Hat Developer Toolset User Guide を参照してください。

1.2. 主な特長

表1.1「Red Hat Software Collections 3.1 コンポーネント」 には、Red Hat Software Collections 3.1 リリース時にサポートされているすべてのコンポーネントが記載されています。

表1.1 Red Hat Software Collections 3.1 コンポーネント

コンポーネント	Software Collection	説明

コンポーネント	Software Collection	説明
Red Hat Developer Toolset 7.1	devtoolset-7	Red Hat Developer Toolset は、Red Hat Enterprise Linux プラットフォームで作業する開発者向けに設計されています。**GNU Compiler Collection**、**GNU Debugger**、その他の開発用ツールやデバッグ用ツール、およびパフォーマンス監視ツールの現行バージョンを提供します。コンポーネントの完全なリストは、『Red Hat Developer Toolset User Guide』 の中の表、Red Hat Developer Toolset Components を参照してください。
Eclipse 4.6.3[a]	rh-eclipse46	Eclipse Foundation の Neon リリーストレインをベースにした**統合開発環境**Eclipse のリリースです。**Eclipse** は以前、Red Hat Developer Toolset のコンポーネントとして提供されていました。Software Collection は、rh-java-common のコンポーネントに依存しています。
Perl 5.24.0	rh-perl524	Perl のリリース。これは、システム管理ユーティリティーと Web プログラミングに一般的に使用される高レベルのプログラミング言語です。rh-perl524Software Collection は、追加のユーティリティー、スクリプト、そして**MySQL および PostgreSQL のデータベースコネクター**を提供します。これには、httpd24 Software Collection でのみサポートされる Perl モジュール **DateTime** と Apache httpd モジュール **mod_perl** が含まれます。さらに、CPAN モジュールを簡単にインストールするための **cpanm** ユーティリティーが提供されます。
Perl 5.26.1 [a]	rh-perl526	Perl のリリース。これは、システム管理ユーティリティーと Web プログラミングに一般的に使用される高レベルのプログラミング言語です。rh-perl526Software Collection は、追加のユーティリティー、スクリプト、そして**MySQL および PostgreSQL のデータベースコネクター**を提供します。これには、httpd24 Software Collection でのみサポートされる Perl モジュール **DateTime** と Apache httpd モジュール **mod_perl** が含まれます。さらに、CPAN モジュールを簡単にインストールするための **cpanm** ユーティリティーが提供されます。rh-perl526 パッケージはアップストリームに合わせて調整されます。インタープリターは perl526-perl パッケージにより提供されますが、perl-interpreter パッケージはコアモジュールもインストールします。
PHP 7.0.27	rh-php70	PEAR 1.10 を搭載した PHP 7.0 のリリースで、言語機能の強化と**パフォーマンスの向上**が図られています。
PHP 7.1.8 [a]	rh-php71	PEAR 1.10、**APCu 5.1.8**および強化された言語機能を備えた PHP 7.1 のリリース。

コンポーネント	Software Collection	説明
Python 2.7.13	python27	多くの追加ユーティリティーを備えた Python 2.7 のリリース。この Python バージョンでは、順序付けされたディクショナリータイプ、高速な I/O 操作、Python 3 との前方互換性など、さまざまな機能および機能拡張が提供されます。python27 Software Collections には **Python 2.7.13 インタープリター** が含まれています。これは、Web アプリケーションおよび **mod_wsgi** (httpd24 Software Collection でのみサポート)、MySQL データベースコネクターおよび PostgreSQL データベースコネクター、ならびに **numpy** および **scipy** のプログラミングに役立つ拡張ライブラリーセットです。
Python 3.5.1	rh-python35	rh-python35 Software Collection には、**Python 3.5.1 インタープリター**、Web アプリケーションのプログラミングに便利な拡張ライブラリー群、**mod_wsgi** (httpd24 Software Collection でのみサポート)、PostgreSQL データベースコネクター、**numpy** および **scipy** が含まれています。
Python 3.6.3	rh-python36	rh-python36 Software Collection には Python 3.6.3 が含まれ、**f-strings、変数アノテーションの構文、非同期ジェネレーターおよび内包表記** など多くの新機能が導入されています。さらに、Web アプリケーションのプログラミングに便利な拡張ライブラリーのセットが含まれています (httpd24 ソフトウェアコレクションと併せてサポート)、PostgreSQL データベースコネクター、ならびに **numpy** および **scipy** が含まれます。
Ruby 2.3.6	rh-ruby23	Ruby 2.3 のリリースです。このバージョンでは、Ruby 2.2、Ruby 2.0.0、Ruby 1.9.3 とのソースレベルでの後方互換性を維持しつつ**ソースファイル内のすべての文字列リテラルをフリーズするコマンドラインオプション、安全なナビゲーション演算子、および複数のパフォーマンス**の向上が導入されています。
Ruby 2.4.3	rh-ruby24	Ruby 2.4 のリリースです。このバージョンでは、**ハッシュテーブルの改良、新しいデバッグ機能、Unicode の大文字小文字のマッピングのサポート、OpenSSL 1.1.0 のサポート**など、複数のパフォーマンスの改善と強化が行われています。Ruby 2.4.0 は、Ruby 2.3、Ruby 2.2、Ruby 2.0.0、Ruby 1.9.3 とのソースレベルでの後方互換性を維持しています。
Ruby 2.5.0 [a]	rh-ruby25	Ruby 2.5 のリリース。このバージョンでは、複数のパフォーマンスの向上と新機能が提供されます。たとえば、 **rescue、else、および ensure キーワードによるブロックの使用の簡素化、新しい yield_self メソッド、ブランチおよびメソッド範囲測定のサポート、新しい Hash#slice および Hash#transform_keys メソッド**です。Ruby 2.5.0 は、Ruby 2.4 とソースレベルの後方互換性を維持します。

コンポーネント	Software Collection	説明
Ruby on Rails 4.2.6	rh-ror42	Ruby 言語で書かれた Web アプリケーションフレームワークである Ruby on Rails 4.2 のリリース。今回のリリースでは、**Active Job、非同期メール、Adequate Record、Web Console、外国語キーのサポート**などの機能が追加されています。Software Collection は、rh-ruby23 および rh-nodejs4 コレクションとともにサポートされます。
Ruby on Rails 5.0.1	rh-ror50	Ruby 言語で書かれた Web アプリケーションフレームワークの最新版である Ruby on Rails 5.0 をリリースしました。注目すべき新機能としては、**Action Cable、API モード、Rake ではなく rails CLI の独占使用、ActionRecord の属性**などがあります。Software Collection は、rh-ruby24 および rh-nodejs6 コレクションとともにサポートされます。
Scala 2.10.6 [a]	rh-scala210	オブジェクト指向言語と関数型言語の機能を統合した、Java プラットフォーム用の汎用プログラミング言語 Scala をリリースしました。
MariaDB 10.1.29	rh-mariadb101	Red Hat Enterprise Linux ユーザー向けの MySQL の代替となる MariaDB のリリース。あらゆる実用的な目的で、MySQL は MariaDB とバイナリー互換性があり、データ変換なしで MySQL と置き換えることができます。このバージョンでは、**Galera Cluster のサポート**が追加されています。
MariaDB 10.2.8	rh-mariadb102	Red Hat Enterprise Linux ユーザー向けの MySQL の代替となる MariaDB のリリース。あらゆる実用的な目的で、MySQL は MariaDB とバイナリー互換性があり、データ変換なしで MySQL と置き換えることができます。このバージョンでは、**MariaDB バックアップ、フラッシュバック、再帰的共通テーブル式のサポート、ウィンドウ関数、JSON 関数** が追加されています。
MongoDB 3.2.10	rh-mongodb32	**NoSQL データベースに分類されるクロスプラットフォームのドキュメント指向データベースシステム**である MongoDB のリリースです。Software Collection には、mongo-java-driver パッケージバージョン 3.2.1 が含まれています。
MongoDB 3.4.9	rh-mongodb34	MongoDB のリリース。これは、NoSQL データベースとして分類されるクロスプラットフォームのドキュメント指向のデータベースシステムです。本リリースでは、新しいアーキテクチャーのサポートが導入され、**メッセージ圧縮と decimal128 タイプのサポート、照合機能の向上**などが追加されます。

コンポーネント	Software Collection	説明
MongoDB 3.6.3 [a]	rh-mongodb36	MongoDB のリリース。これは、NoSQL データベースとして分類されるクロスプラットフォームのドキュメント指向のデータベースシステムです。本リリースでは、**変更ストリーム、再試行可能な書き込み、および JSON スキーマおよび** その他の機能が導入されました。
MySQL 5.7.21	rh-mysql57	パフォーマンスの向上を含む多くの新機能と機能強化を提供する、MySQL のリリース。
PostgreSQL 9.5.9	rh-postgresql95	PostgreSQL のリリースで、**行レベルのセキュリティー制御**を含む多くの機能強化が行われ、レプリケーションの進捗状況の追跡が導入され、カラム数の多い大規模テーブルの扱いが改善され、ソートマシンやマルチ CPU マシンのパフォーマンスが向上しました。
PostgreSQL 9.6.5	rh-postgresql96	PostgreSQL のリリース。順次スキャン、結合、および集計の並列実行が導入され、同期レプリケーション、全文検索、ディレーションドライバー、postgres_fdw の機能が強化され、パフォーマンスが改善されています。
PostgreSQL 10.3 [a]	rh-postgresql10	PostgreSQL のリリースには、パフォーマンスが大幅に向上し、 **publish および subscribe キーワードを使用した論理レプリケーション、SCRAM-SHA-256 メカニズムに基づく強力なパスワード認証** などの新機能が多数含まれています。
Node.js 4.6.2	rh-nodejs4	Chrome の V8 JavaScript エンジンで構築された JavaScript ランタイムを提供する Node.js と、JavaScript 用パッケージマネージャー**npm 2.15.1** のリリースです。このバージョンでは、API の強化、複数のセキュリティーおよびバグの修正、**SPDY プロトコルのバージョン 3.1 のサポート**がされています。
Node.js 6.11.3	rh-nodejs6	Node.js のリリース。複数の API の機能拡張、パフォーマンスとセキュリティーの向上、**ECMAScript 2015 のサポート**、および **npm 3.10.9** を提供します。
Node.js 8.9.4 [a]	rh-nodejs8	**V8 エンジンバージョン 6.0、npm 5.6.0、npx、セキュリティーの強化、実験的な N-API のサポート、パフォーマンスの向上**など、複数の API の強化と新機能を提供する Node.js のリリースです。
nginx 1.8.1	rh-nginx18	nginx は、高い同時性、パフォーマンス、低いメモリー使用量に重点を置いたウェブおよびプロキシーサーバーです。このバージョンでは、**バックエンドの SSL 証明書の検証、syslog へのロギング、I/O リクエストをオフロードするためのスレッドプールのサポート、またはハッシュロードバランシング方式**など、多くの新機能が導入されています。

コンポーネント	Software Collection	説明
nginx 1.10.2	rh-nginx110	nginx は、高い同時性、パフォーマンス、低いメモリー使用量に重点を置いたウェブおよびプロキシーサーバーです。このバージョンでは、**ダイナミックモジュールのサポート、HTTP/2 のサポート、Perl の統合、多数のパフォーマンスの向上**など、多くの新機能が導入されています。
nginx 1.12.1 [a]	rh-nginx112	nginx は、高い同時性、パフォーマンス、低いメモリー使用量に重点を置いたウェブおよびプロキシーサーバーです。このバージョンでは、**IP トランスペアレンシー、TCP/UDP ロードバランシングの改善、キャッシングパフォーマンスの向上**など、数多くの新機能が導入されています。
Apache httpd 2.4.27	httpd24	Apache HTTP Server (httpd) のリリース。これには、高パフォーマンスの **イベントベースの処理モデル、強化された SSL モジュール、および FastCGI サポート** が含まれます。**mod_auth_kerb** モジュールも含まれています。
Varnish Cache 4.0.3	rh-varnish4	**高性能な HTTP リバースプロキシー**である Varnish Cache のリリースです。**Varnish Cache** は、ファイルまたはファイルのフラグメントをメモリーに保存し、今後、同等のリクエストに対する応答時間とネットワーク帯域幅の消費を削減するために使用されます。
Varnish Cache 5.2.1 [a]	rh-varnish5	高パフォーマンスの HTTP リバースプロキシーである Varnish Cache のリリース。このバージョンには、個別の VCL ファイルと VCL ラベルを使用した **Varnish 設定の shard ディレクター、実験的な HTTP/2 サポート、および改善** が含まれています。
Maven 3.3.9	rh-maven33	主に Java プロジェクトで使用される**ソフトウェアのプロジェクトの管理および状況把握ツール**である Maven のリリースです。このバージョンでは、**コア拡張メカニズムの改善**など、さまざまな機能強化が行われています。
Maven 3.5.0 [a]	rh-maven35	Maven のリリース (ソフトウェアプロジェクト管理および内包表記ツール)。このリリースでは、新しいアーキテクチャーのサポートと、**カラー化されたログ**を含む多くの新機能が導入されています。
Git 2.9.3	rh-git29	分散アーキテクチャーを備えた**分散リビジョン管理システム** である Git のリリース。クライアントサーバーモデルを使用する集中型バージョン管理システムとは対照的に、Git は Git リポジトリーの各作業コピーが完全なリビジョン履歴で正確なコピーになるようにします。
Redis 3.2.4	rh-redis32	**永続的なキーバリューデータベース**である Redis 3.2 のリリースです。

コンポーネント	Software Collection	説明
HAProxy 1.8.4 [a]	rh-haproxy18	HAProxy 1.8 のリリース (TCP および HTTP ベースのアプリケーションの信頼できる高パフォーマンスな **ネットワークロードバランサー**) です。
Common Java Packages	rh-java-common	Software Collection は、他のコレクションで使用されている**共通の Java ライブラリーとツール**を提供します。rh-java-common Software Collection は、devtoolset-4、devtoolset-3、rh-maven33、maven30、rh-mongodb32、rh-mongodb26、thermostat1、rh-thermostat16、rh-eclipse46 コンポーネントで必要とされるものであり、ユーザーが直接インストールすることは想定されていません。

[a] この Software Collection は、Red Hat Enterprise Linux 7 でのみ利用できます。

これまでリリースされた Software Collections は同じディストリビューションチャンネルで引き続き利用できます。終了したコンポーネントを含む Software Collections はすべて、表1.2「利用可能なすべての Software Collections」 に記載されています。サポートされなくなった Software Collections にはアスタリスク (*) が付いています。

個々のコンポーネントのサポート期間の詳細は、Red Hat Software Collections Product Life Cycle を参照してください。以前にリリースされたコンポーネントの詳細は、Red Hat Software Collections の以前のバージョンの Release Notes を参照してください。

表1.2 利用可能なすべての Software Collections

コンポーネント	Software Collection	可用性	RHEL7 でサポートされるアーキテクチャー
Red Hat Software Collections 3.1 の新コンポーネント			
Perl 5.26.1	rh-perl526	RHEL7	x86_64、s390x、aarch64、ppc64le
Ruby 2.5.0	rh-ruby25	RHEL7	x86_64、s390x、aarch64、ppc64le
MongoDB 3.6.3	rh-mongodb36	RHEL7	x86_64、s390x、aarch64、ppc64le
Varnish Cache 5.2.1	rh-varnish5	RHEL7	x86_64、s390x、aarch64、ppc64le
PostgreSQL 10.3	rh-postgresql10	RHEL7	x86_64、s390x、aarch64、ppc64le
HAProxy 1.8.4	rh-haproxy18	RHEL7	x86_64

Red Hat Software Collections 3.1 で更新されたコンポーネント			
Red Hat Developer Toolset 7.1	devtoolset-7	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64、ppc64le
PHP 7.0.27	rh-php70	RHEL6、RHEL7	x86_64
MySQL 5.7.21	rh-mysql57	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le
Apache httpd 2.4.27	httpd24	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le

Red Hat Software Collections 3.0 で最後に更新されたコンポーネント			
PHP 7.1.8	rh-php71	RHEL7	x86_64、s390x、aarch64、ppc64le
nginx 1.12.1	rh-nginx112	RHEL7	x86_64、s390x、aarch64、ppc64le
Python 3.6.3	rh-python36	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le
Maven 3.5.0	rh-maven35	RHEL7	x86_64、s390x、aarch64、ppc64le
MariaDB 10.2.8	rh-mariadb102	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le
PostgreSQL 9.6.5	rh-postgresql96	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le
MongoDB 3.4.9	rh-mongodb34	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64le
Node.js 8.9.4	rh-nodejs8	RHEL7	x86_64、s390x、aarch64、ppc64le

Red Hat Software Collections 2.4 で最後に更新されたコンポーネント			
Red Hat Developer Toolset 6.1	devtoolset-6	RHEL6、RHEL7	x86_64、s390x、aarch64、ppc64、ppc64le
Scala 2.10.6	rh-scala210	RHEL7	x86_64

Red Hat Software Collections 2.4 で最後に更新されたコンポーネント			
nginx 1.10.2	rh-nginx110	RHEL6、RHEL7	x86_64
Node.js 6.11.3	rh-nodejs6	RHEL6、RHEL7	x86_64
Ruby 2.4.3	rh-ruby24	RHEL6、RHEL7	x86_64
Ruby on Rails 5.0.1	rh-ror50	RHEL6、RHEL7	x86_64
Eclipse 4.6.3	rh-eclipse46	RHEL7	x86_64
Python 2.7.13	python27	RHEL6、RHEL7	x86_64
Thermostat 1.6.6	rh-thermostat16*	RHEL6、RHEL7	x86_64
Maven 3.3.9	rh-maven33	RHEL6、RHEL7	x86_64
Common Java Packages	rh-java-common	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.3 で最後に更新されたコンポーネント			
Git 2.9.3	rh-git29	RHEL6、RHEL7	x86_64
Redis 3.2.4	rh-redis32	RHEL6、RHEL7	x86_64
Perl 5.24.0	rh-perl524	RHEL6、RHEL7	x86_64
Python 3.5.1	rh-python35	RHEL6、RHEL7	x86_64
MongoDB 3.2.10	rh-mongodb32	RHEL6、RHEL7	x86_64
Ruby 2.3.6	rh-ruby23	RHEL6、RHEL7	x86_64
PHP 5.6.25	rh-php56*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.2 で最後に更新されたコンポーネント			
Red Hat Developer Toolset 4.1	devtoolset-4*	RHEL6、RHEL7	x86_64
MariaDB 10.1.29	rh-mariadb101	RHEL6、RHEL7	x86_64
MongoDB 3.0.11 アップグレードコレクション	rh-mongodb30upg*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.2 で最後に更新されたコンポーネント			
Node.js 4.6.2	rh-nodejs4	RHEL6、RHEL7	x86_64
PostgreSQL 9.5.9	rh-postgresql95	RHEL6、RHEL7	x86_64
Ruby on Rails 4.2.6	rh-ror42	RHEL6、RHEL7	x86_64
MongoDB 2.6.9	rh-mongodb26*	RHEL6、RHEL7	x86_64
Thermostat 1.4.4	thermostat1*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.1 で最後に更新されたコンポーネント			
Varnish Cache 4.0.3	rh-varnish4	RHEL6、RHEL7	x86_64
nginx 1.8.1	rh-nginx18	RHEL6、RHEL7	x86_64
Node.js 0.10	nodejs010*	RHEL6、RHEL7	x86_64
Maven 3.0.5	maven30*	RHEL6、RHEL7	x86_64
V8 3.14.5.10	v8314*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.0 で最後に更新されたコンポーネント			
Red Hat Developer Toolset 3.1	devtoolset-3*	RHEL6、RHEL7	x86_64
Perl 5.20.1	rh-perl520*	RHEL6、RHEL7	x86_64
Python 3.4.2	rh-python34*	RHEL6、RHEL7	x86_64
Ruby 2.2.9	rh-ruby22*	RHEL6、RHEL7	x86_64
Ruby on Rails 4.1.5	rh-ror41*	RHEL6、RHEL7	x86_64
MariaDB 10.0.33	rh-mariadb100*	RHEL6、RHEL7	x86_64
MySQL 5.6.40	rh-mysql56*	RHEL6、RHEL7	x86_64
PostgreSQL 9.4.14	rh-postgresql94*	RHEL6、RHEL7	x86_64
Passenger 4.0.50	rh-passenger40*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 2.0 で最後に更新されたコンポーネント			
PHP 5.4.40	php54*	RHEL6、RHEL7	x86_64
PHP 5.5.21	php55*	RHEL6、RHEL7	x86_64
nginx 1.6.2	nginx16*	RHEL6、RHEL7	x86_64
DevAssistant 0.9.3	devassist09*	RHEL6、RHEL7	x86_64

Red Hat Software Collections 1 で最後に更新されたコンポーネント			
Git 1.9.4	git19*	RHEL6、RHEL7	x86_64
Perl 5.16.3	perl516*	RHEL6、RHEL7	x86_64
Python 3.3.2	python33*	RHEL6、RHEL7	x86_64
Ruby 1.9.3	ruby193*	RHEL6、RHEL7	x86_64
Ruby 2.0.0	ruby200*	RHEL6、RHEL7	x86_64
Ruby on Rails 4.0.2	ror40*	RHEL6、RHEL7	x86_64
MariaDB 5.5.53	mariadb55*	RHEL6、RHEL7	x86_64
MongoDB 2.4.9	mongodb24*	RHEL6、RHEL7	x86_64
MySQL 5.5.52	mysql55*	RHEL6、RHEL7	x86_64
PostgreSQL 9.2.18	postgresql92*	RHEL6、RHEL7	x86_64

RHEL6: Red Hat Enterprise Linux 6

RHEL7: Red Hat Enterprise Linux 7

x86_64 : AMD64 および Intel 64 アーキテクチャー

s390x: IBM z Systems

aarch64: 64 ビット ARM アーキテクチャー

ppc64: IBM POWER、ビッグエンディアン

ppc64le: IBM POWER、リトルエンディアン

* リタイアしたコンポーネント: この Software Collection のサポートは終了しました。

上記の表には、非同期更新で利用できる最新バージョンがリスト表示されます。

Red Hat Software Collections 2.0 以降でリリースされた Software Collections には、その名前に **rh-** 接頭辞が含まれていることに注意してください。

1.3. RED HAT SOFTWARE COLLECTIONS 3.1 での変更点

1.3.1. 概要

アーキテクチャー
Red Hat Software Collections には、AMD64 および Intel 64 アーキテクチャー上で動作する Red Hat Enterprise Linux 7 用のパッケージが含まれていますが、一部の Software Collections は Red Hat Enterprise Linux 6 でもご利用いただけます。

さらに、Red Hat Software Collections 3.1 は、Red Hat Enterprise Linux 7 の以下のアーキテクチャーをサポートします。

- 64 ビット ARM アーキテクチャー
- IBM z Systems
- IBM POWER、リトルエンディアン

コンポーネントの完全なリストとそれらの可用性は、表1.2「利用可能なすべての Software Collections」 を参照してください。

新しい Software Collections
Red Hat Software Collections 3.1 では、以下の新しい Software Collection が追加されました。

- rh-perl526: 「Perl の変更点」 を参照してください。
- rh-ruby25: 「Ruby の変更点」 を参照してください。
- rh-mongodb36: 「MongoDB の変更点」 を参照してください。
- rh-postgresql10: 「PostgreSQL の変更点」 を参照してください。
- rh-varnish5: 「Varnish キャッシュの変更点」 を参照してください。
- rh-haproxy18: 「HAProxy の変更点」 を参照してください。

すべての新しい Software Collections は、Red Hat Enterprise Linux 7 でのみ利用できます。

Software Collections の更新
Red Hat Software Collections 3.1 では、以下のコンポーネントが更新されました。

- devtoolset-7: 「Red Hat Developer Toolset の変更点」 を参照してください。
- rh-php70: 「PHP の変更点」 を参照してください。
- rh-mysql57: 「MySQL の変更点」 を参照してください。
- httpd24: 「Apache httpd の変更点」 を参照してください。

Red Hat Software Collections コンテナーイメージ
以下のコンテナーイメージは Red Hat Software Collections 3.1 で新しく加わりました。

- rhscl/perl-526-rhel7
- rhscl/ruby-25-rhel7
- rhscl/mongodb-36-rhel7
- rhscl/varnish-5-rhel7
- rhscl/postgresql-10-rhel7

Red Hat Software Collections 3.1 で以下のコンテナーイメージが更新されました。

- rhscl/devtoolset-7-toolchain-rhel7
- rhscl/devtoolset-7-perftools-rhel7
- rhscl/php-70-rhel7
- rhscl/httpd-24-rhel7

Red Hat Software Collections のコンテナーイメージに関する詳細は、「Red Hat Software Collections コンテナーイメージ」 を参照してください。

1.3.2. Red Hat Developer Toolset の変更点

Red Hat Developer Toolset 7.1 では、以下のコンポーネントが Red Hat Developer Toolset の旧リリースと比較してアップグレードされています。

- **GCC**: バージョンを 7.3.1 へ

さらに、以下のコンポーネントに対するバグ修正の更新を利用できます。

- **GDB**
- **Valgrind**
- **elfutils**
- **binutils**
- **dwz**
- **memstomp**
- **make**
- **ltrace**
- **strace**
- **OProfile**
- **SystemTap**
- **Dyninst**

7.1 での変更点の詳細については、Red Hat Developer Toolset User Guide を参照してください。

1.3.3. Perl の変更点

新しいrh-perl526 Software Collection には、 **Perl 5.26.1**が含まれており、前バージョンと比較して多くのバグフィックスと機能強化が施されています。以下は、広範囲に及ぶ重要な変更の一部になります。

- セキュリティー上の理由により、**@INC** モジュールの検索パスから、現在のディレクトリー (**.**) が削除されました。
- 上記の動作上の変更によりファイルの読み込みに失敗した時に、**do** ステートメントが非推奨の警告を返すようになりました。
- 正規表現のパターンで、エスケープされていないリテラルの **{** 文字が使用できなくなりました。
- Unicode 9.0 に対応するようになった

Perl 5.26 の変更点の詳細は、バージョン 5.26.0 および 5.26.1 のアップストリーム変更ログを参照してください。

また、rh-perl526 パッケージもアップストリームに合わせて調整されます。rh-perl526-perl パッケージはコアモジュールもインストールしますが、**/usr/bin/perl** インタープリターは rh-perl526-perl-interpreter パッケージで提供されます。以前のリリースでは、perl パッケージには最小限のインタープリターのみが含まれ、perl-core パッケージにはインタープリターとコアモジュールの両方が含まれていました。

1.3.4. Ruby の変更点

新しい rh-ruby25 Software Collection は **Ruby 2.5.0** を提供します。このバージョンでは、以下の例のように、パフォーマンスの向上と多くの新機能および変更が導入されています。

- **begin** キーワードと **end** キーワードを追加することなく、**do** と **end** で区切られたブロック内で、**rescue**、**else**、および **ensure** キーワードを直接使用できるようになりました。
- ブロックの結果を返す新しい **yield_self** メソッドが追加されました。
- ブランチカバレッジおよびメソッドカバレッジ測定のサポートが追加されました。
- 新しい **Hash#slice** メソッドおよび **Hash#transform_keys** メソッドが実装されています。
- **Struct** サブクラスコンストラクターはキーワード引数を受け取ることができます。
- トップレベルの定数の検索が利用できなくなりました。

Ruby 2.5.0 は、**Ruby 2.4** とのソースレベルの下位互換性を維持します。

Ruby 2.5.0 の詳細な変更点については、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。

1.3.5. MongoDB の変更点

新しいrh-mongodb36Software Collection には **MongoDB 3.6.3** が含まれており、以前のバージョンに比べて多くのバグ修正と拡張機能が提供されています。最も重要な新機能は次のとおりです。

- ストリームを変更します。これにより、アプリケーションがリアルタイムデータの変更にアクセスして、すぐにそのデータに反応することが可能になります。
- 再試行可能な書き込み。これにより、MongoDB ドライバーが特定の書き込み操作を自動的に再試行できるようにします (ネットワークエラーが発生した場合など)。

- JSON スキーマ。JSON ドキュメントを定義する基準をユーザーに提供します。

また、以下のサブパッケージも更新されました。

- mongo-c-driver をバージョン 1.9.2 に変更しました。
- mongo-cxx-driver をバージョン 3.2.0 に変更しました。
- mongo-tools をバージョン 3.6.3 に変更しました。
- mongo-java-driver をバージョン 3.6.3 に変更しました。

MongoDB 3.6 の詳細な変更点については、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。

注記

rh-mongodb36-mongo-cxx-driver パッケージは、Red Hat Developer Toolset 6 の **GCC** を使用して **-std=gnu++14** オプションでビルドされています。C++11 (以降) の機能を使用する MongoDB C++ ドライバーの共有ライブラリーを使用するバイナリーは、Red Hat Developer Toolset 6 以降でもビルドする必要があります。Red Hat Developer Toolset 6 User Guideの C++ compatibility を参照してください。

rh-mongodb36 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-mongodb36-syspaths パッケージが含まれます。rh-mongodb36*-syspaths パッケージのインストール後に、rh-mongodb36* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。syspaths の詳細は、Red Hat Software Collections パッケージガイド を参照してください。

移行に関する説明は、「MongoDB 3.6 への移行」 を参照してください。

1.3.6. PostgreSQL の変更点

新しいrh-postgresql10Software Collection は **PostgreSQL 10.3** を提供します。このバージョンの主な機能強化は、以下のとおりです。

- **publish** キーワードおよび **subscribe** キーワードを使用した論理レプリケーション
- **SCRAM-SHA-256** メカニズムに基づくより強力なパスワード認証
- 宣言型テーブルのパーティション
- 改善されたクエリーの並列処理
- 重要な一般的なパフォーマンスの向上
- 改善された監視および制御

PostgreSQL 10.3 の変更点の詳細は、アップストリームのドキュメント を参照してください。

rh-postgresql10 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-postgresql10-syspaths パッケージが含まれます。rh-postgreqsl10*-syspaths パッケージのインストール後に、rh-postgreqsl10* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認す

る必要はありません。*-syspaths パッケージは、ベースの Red Hat Enterprise Linux システムと対応するパッケージと競合することに注意してください。syspaths の詳細は、Red Hat Software Collections パッケージガイドを参照してください。

移行の情報は、「PostgreSQL 10 への移行」 を参照してください。

1.3.7. Varnish キャッシュの変更点

新しい rh-varnish5 Software Collection には **Varnish Cache 5.2.1** が含まれており、以前のバージョンに比べて多くのバグ修正と拡張機能が提供されています。以下に例を示します。

- 個別の Varnish Configuration Language(VCL) ファイルと VCL ラベルを使用した設定の改善。これにより、設定での複雑さを軽減することができます。
- 実験的な HTTP/2 サポート
- **shard** ディレクター
- 安定性の改善

Varnish Cache 5.2.1 の変更点の詳細は、アップストリームの変更ログ を参照してください。upstream documentation および upgrading notes を参照してください。

1.3.8. HAProxy の変更点

新しい rh-haproxy18 Software Collection は **HAProxy 1.8.4** を提供します。このバージョンでの最も重要な変更点は以下の通りです。

- HTTP 要求および応答キャプチャー
- マルチプロセスピアとスティックテーブル
- 実行時に DNS を使用したサーバー IP 解決
- メールアラート
- シームレスサーバーの状態
- HTTP/2 サポート (実験的)
- マルチスレッド (実験的)
- シームレスなリロード
- サービス検出の DNS
- ランタイム API の改善

Red Hat Enterprise Linux 7 に含まれるバージョン 1.5 以降の **HAProxy** の変更点の詳細は、HAProxy Configuration Manual を参照してください。

1.3.9. PHP の変更点

rh-php70 Software Collection がバージョン 7.0.27 にアップグレードされ、Red Hat Software Collections 2.3 でリリースされたバージョンに対するバグ修正および機能拡張が数多く追加されました。

このリリースのバグ修正および機能拡張の詳細は、バージョン 7.0.27 以前のアップストリーム変更ログを参照してください。

1.3.10. MySQL の変更点

今回の更新で、rh-mysql57 Software Collection に、以下のアーキテクチャーのサポートが追加されました。

- 64 ビット ARM アーキテクチャー
- IBM z Systems
- IBM POWER、リトルエンディアン

1.3.11. Apache httpd の変更点

今回の更新で、httpd24-mod_auth_mellon パッケージが httpd24 Software Collection に追加されました。Apache HTTP サーバーの **mod_auth_mellon** モジュールは、SAML 2.0 フェデレーションプロトコルを実装する認証サービスです。モジュールは、アイデンティティープロバイダー (IdP) サーバーが生成したアサーションで受け取った属性に基づいてアクセスを付与します。

1.4. 互換性情報

Red Hat Software Collections 3.1 は、AMD64 および Intel 64 アーキテクチャー、64 ビット ARM アーキテクチャー、IBM z Systems、および IBM POWER、リトルエンディアン上の Red Hat Enterprise Linux 7 のサポートされるすべてのリリースで利用できます。

また、特定のコンポーネントが、AMD64 および Intel 64 アーキテクチャー上の Red Hat Enterprise Linux 6 の全サポートリリースに向けて提供されています。

利用可能なコンポーネントのリストは、表1.2「利用可能なすべての Software Collections」 を参照してください。

1.5. 既知の問題

httpd24 コンポーネント (BZ#1429006)

httpd 2.4.27 以降、**mod_http2** モジュールはデフォルトの **prefork** Multi-Processing Module (MPM) でサポートされなくなりました。HTTP/2 サポートを有効にするには、**/opt/rh/httpd24/root/etc/httpd/conf.modules.d/00-mpm.conf** で設定ファイルを編集し、**event** または **worker** MPM に切り替えます。

なお、HTTP/2 のサーバープッシュ機能は、64 ビットの ARM アーキテクチャー、IBM z Systems、IBM POWER のリトルエンディアンでは動作しません。

httpd24 コンポーネント (BZ#1327548)

mod_ssl モジュールは、Red Hat Enterprise Linux 6 または Red Hat Enterprise Linux 7.3 以前で ALPN プロトコルに対応していません。そのため、ALPN を使用した TLS 接続の HTTP/2 へのアップグレードをサポートするクライアントは、HTTP/1.1 のサポートに制限されます。

httpd24 コンポーネント、BZ#1224763

FastCGI Process Manager (PHP-FPM) で **mod_proxy_fcgi** モジュールを使用する場合は、**httpd** は正しいポート **9000** ではなく、デフォルトで FastCGI プロトコルのポート **8000** を使用します。この問題を回避するには、正しいポートを明示的に指定します。

httpd24 コンポーネント、BZ#1382706

SELinux が有効になっている場合、**LD_LIBRARY_PATH**環境変数は、**httpd** によって呼び出される CGI スクリプトには渡されません。そのため、**httpd** が実行する CGI スクリプトの **/opt/rh/httpd24/service-environment** ファイルで有効にした Software Collections から実行ファイルを呼び出すことができない場合があります。この問題を回避するには、CGI スクリプト内から希望どおりに **LD_LIBRARY_PATH** 設定します。

httpd24 コンポーネント

httpd24 Software Collection からの Apache Portable Runtime (APR) および APR-util ライブラリーに対する外部アプリケーションのコンパイルはサポートされていません。LD_LIBRARY_PATH 環境変数は、この Software Collection のいずれのアプリケーションでも不要であるため、httpd24 に設定されていません。

rh-python34、rh-python35、rh-python36 コンポーネント、BZ#1499990

Babel がタイムゾーンのサポートに使用する **pytz** モジュールは、rh-python34、rh-python35、そしてrh-python36Software Collectionsには含まれていません。そのため、ユーザーが **Babel** から dates モジュールをインポートしようとすると、トレースバックが返されます。この問題を回避するには、**pip install pytz** コマンドを使用して、**pypi** パブリックリポジトリーから **pip** パッケージマネージャーを介して **pytz** をインストールします。

rh-python36 コンポーネント ( )

64 ビットの ARM アーキテクチャー、IBM z Systems、および IBM POWER のリトルエンディアンで、**numpy**が提供する特定の複雑な三角関数が正しくない値を返すことがありました。AMD64 および Intel 64 のアーキテクチャーは、この問題の影響を受けません。

python27 コンポーネント (BZ#1330489)

python27-python-pymongo パッケージがバージョン 3.2.1 に更新されました。このバージョンは、これまでに同梱されているバージョン 2.5.2 と完全に互換性がないことに注意してください。

python27 コンポーネント、

Red Hat Enterprise Linux 7 では、ユーザーが python27-python-debuginfo パッケージをインストールしようとすると、**/usr/src/debug/Python-2.7.5/Modules/socketmodule.c** ファイルがコアシステムにインストールされている python-debuginfo パッケージの対応するファイルと競合します。その結果、python27-python-debuginfo のインストールに失敗します。この問題を回避するには、python-debuginfo パッケージをアンインストールしてから、python27-python-debuginfo パッケージをインストールしてください。

scl-utils コンポーネント

scl-utils パッケージのアーキテクチャー固有のマクロのバグにより、64 ビット ARM アーキテクチャーおよび IBM POWER のリトルエンディアンでは、**<collection>/root/usr/lib64/** ディレクトリーに正しいパッケージの所有権がありません。したがって、Software Collection がアンインストールされると、このディレクトリーは削除されません。この問題を回避するには、Software Collection を削除する際に **<collection>/root/usr/lib64/** を手動で削除します。

rh-ruby24、rh-ruby23 コンポーネント

RubyGem のインストールパスは、複数の Software Collection が有効になる順番に応じて決定されます。Red Hat Software Collections 2.3 に同梱されている **Ruby 2.3.1**以降、依存する Collection をサポートするために、必要な順序が変更されました。そのため、Software Collection を間違った順序で指定すると、RPM ビルド中に **gem** のインストールに使用される **RubyGem** パスが無効になり

ます。たとえば、RPM spec ファイルに **scl Enable rh-ror50 rh-nodejs6** が含まれている場合、ビルドが失敗するようになりました。この問題を回避するには、rh-ror50 Software Collection を最後に有効にします。たとえば、**scl enable rh-nodejs6 rh-ror50** のようにします。

rh-maven35、rh-maven33 コンポーネント

ユーザーが Red Hat Enterprise Linux システムバージョンの maven-local パッケージと rh-maven35-maven-local package または rh-maven33-maven-local package の両方をインストールしている場合、rh-maven35 または rh-maven33 Software Collection から実行される Java RPM パッケージビルド用のツール **XMvn** が、ベースシステムから設定ファイルを読み取ろうとして失敗します。この問題を回避するには、ベースの Red Hat Enterprise Linux システムから maven-local パッケージをアンインストールします。

rh-nodejs4 コンポーネント、BZ#1316626

/opt/rh/rh-nodejs4/root/usr/share/licenses/ ディレクトリーはどのパッケージによっても所有されていません。そのため、rh-nodejs4 コレクションをアンインストールするとき、このディレクトリーは削除されません。この問題を回避するには、rh-nodejs4 をアンインストールした後、手動でディレクトリーを削除してください。

perl コンポーネント

複数の **mod_perl.so** ライブラリーをインストール することはできません。したがって、複数の **Perl** Software Collection から **mod_perl** モジュールを使用することはできません。

nodejs010 コンポーネント

nodejs010 Software Collection で提供されている共有ライブラリー (**libcares**、**libhttp_parser** および **libuv**) の接頭辞が正しくありません。そのため、対応するシステムライブラリーとの間でコンフリクトが発生する可能性があります。

nodejs-hawk コンポーネント

nodejs-hawk パッケージは、CryptoJS プロジェクトから採用された SHA-1 および SHA-256 アルゴリズムの実装を使用しています。今回のリリースでは、クライアント側の JavaScript が難読化されています。今後の修正では、CryptoJS ライブラリーから直接暗号機能を使用することになります。

postgresql コンポーネント)

Red Hat Enterprise Linux 6 の postgresql92、rh-postgresql94、rh-postgresql95 パッケージは **sepgsql** モジュールを提供しません。この機能は Red Hat Enterprise Linux 6 では利用できない libselinux バージョン 2.0.99 のインストールを必要とするからです。

httpd、mariadb、mongodb、mysql、nodejs、perl、php55、rh-php56、python、ruby、ror、thermostat、およびv8314コンポーネント、BZ#1072319

httpd24、mariadb55、rh-mariadb100、mongodb24、rh-mongodb26、mysql55、rh-mysql56、nodejs010、perl516、rh-perl520、php55、rh-php56、python27、python33、rh-python34、ruby193、ruby200、rh-ruby22、ror40、rh-ror41、thermostat1、または v8314 パッケージをアンインストールする際に、依存するパッケージの所有権により、アンインストールの順番が関係することがあります。そのため、一部のディレクトリーおよびファイルはシステム上に残される可能性があるため、削除されない可能性があります。

rh-mysql57、rh-mysql56、rh-mariadb100、rh-mariadb101 コンポーネント、BZ#1194611

rh-mysql57-mysql-server、rh-mysql56-mysql-server、rh-mariadb100-mariadb-server、および rh-mariadb101-mariadb-server パッケージは、デフォルトで**test**データベースを提供しなくなりました。このデータベースは初期化中には作成されませんが、付与テーブルは、**test** がデフォルトで作

成されたものと同じ値で事前に入力されます。その結果、**test** または **test_*** データベースが後で作成されると、これらのデータベースへのアクセス権限は、新規データベースのデフォルトよりも制限されません。

また、ベンチマークを実行している場合、**run-all-tests** スクリプトは、サンプルパラメーターを使用してもそのままでは機能しません。テストを実行する前にテストデータベースを作成し、**--database** パラメーターにデータベース名を指定する必要があります。パラメーターが指定されていない場合、デフォルトで **test** が取得されますが、**test** データベースが存在することを確認する必要があります。

mongodb24 コンポーネント

Red Hat Software Collections 1.2 のmongodb24 Software Collection は、Red Hat Software Collections 3.1 に同梱されているrh-java-common およびmaven30 Software Collection では再構築できません。さらに、mongodb24-build および mongodb24-scldevel パッケージは maven30-javapackages-tools および maven30-maven-local packages の要求が満たされていないため、Red Hat Software Collections 3.1 にはインストールできません。mongodb24-scldevel パッケージのインストール時に、壊れた依存関係が報告され、**yum --skip-broken** コマンドでスキップされるパッケージが多すぎます。rh-mongodb26 Software Collection に更新することを推奨します。

mariadb、mysql、postgresql、mongodb コンポーネント)

Red Hat Software Collections 3.1 には、**MySQL 5.6**、**MySQL 5.7**、**MariaDB 10.0**、**MariaDB 10.1**、**MariaDB 10.2**、**PostgreSQL 9.4**、PostgreSQL **9.5**、**PostgreSQL 9.6**、**PostgreSQL 10**、**MongoDB 2.6**、**MongoDB 3.2**、**MongoDB 3.4**、および **MongoDB 3.6** データベースが含まれています。Red Hat Enterprise Linux 6 のコアは、以前のバージョンの **MySQL** および **PostgreSQL** データベース (クライアントライブラリーおよびデーモン) を提供します。コア Red Hat Enterprise Linux 7 は、**MariaDB** および **PostgreSQL** データベース (クライアントライブラリーおよびデーモン) の以前のバージョンを提供します。クライアントライブラリーは、動的言語、ライブラリーなどのデータベースコネクターにも使用されます。

PostgreSQL コンポーネントの Red Hat Software Collections データベースパッケージにパッケージ化されたクライアントライブラリーは、サーバーユーティリティーおよびデーモンの目的にのみ含まれているため、使用する予定はありません。代わりに、ユーザーはコアシステムで提供されるシステムライブラリーとデータベースコネクターを使用することが想定されます。

クライアントライブラリーとデーモンの間で使用されるプロトコルはデータベースバージョン間で安定するため、たとえば **PostgreSQL 9.4** デーモンまたは **9.5** デーモンで **PostgreSQL 9.2** クライアントライブラリーを使用すると期待どおりに機能します。

コアの Red Hat Enterprise Linux 6 および Red Hat Enterprise Linux 7 には **MongoDB** のクライアントライブラリーが含まれていません。アプリケーションにこのクライアントライブラリーを使用するには、Red Hat Software Collections からのクライアントライブラリーを使用し、この **MongoDB** クライアントライブラリーにリンクされたアプリケーションを実行するたびに **scl enable ...** 呼び出しを常に使用する必要があります。

mariadb、mysql、mongodb コンポーネント ( )

MariaDB、MySQL、および MongoDB は、ログファイルの作成時に **/opt/*Provider*/*collection*/root** 接頭辞を使用しません。ログファイルは、**/opt/*provider*/*collection*/root/var/log/** ではなく、**/var/opt/*provider*/*collection*/log/** ディレクトリーに保存されることに注意してください。

rh-eclipse46 コンポーネント、

サードパーティーの更新サイトからプラグインがインストールされていると、**Eclipse** が起動に失敗し、ワークスペースのログファイルに **NullPointerException** が記録されることがあります。この問題を回避するには、**-clean** オプションを指定して **Eclipse** を再起動します。以下に例を示します。

```
~]$ scl enable rh-eclipse46 "eclipse -clean"
```

rh-eclipse46 コンポーネント、

Eclipse Docker Tooling に、構文の強調表示と基本的なコマンドの自動補完機能を備えた **Dockerfile エディター** が導入されています。**Build Image Wizard** を開いて **Edit Dockerfile** ボタンを押すと、切り離された **Dockerfile エディター** ウィンドウでファイルが開きます。ただし、このウィンドウには **Cancel** ボタンと **Save** ボタンがありません。この問題を回避するには、**Ctrl**+**S** を押して変更を保存するか、エディター内で右クリックしてコンテキストメニューを表示し、**Save** オプションを選択します。変更をキャンセルするには、ウィンドウを閉じます。

rh-eclipse46 コンポーネント、

Red Hat Enterprise Linux 7.2 では、**Eclipse** の **Perf Profile View** の設定に使用される perf ツールのバグにより、ビュー内の一部の項目が Eclipse Editor のそれぞれの位置に正しくリンクされません。プロファイリングは期待どおりに機能しますが、**Perl Profile View** の一部をクリックしてエディター内の関連する位置に移動することはできません。

rh-thermostat16 コンポーネント

デスクトップアプリケーションファイルのタイプミスが原因で、ユーザーはデスクトップアイコンを使用して **Thermostat** を起動できません。この問題を回避するには、**/usr/share/applications/rh-thermostat16-thermostat.desktop** ファイルを次のように変更します。

```
[Desktop Entry]
Version=1.0
Type=Application
Name=%{thermostat_desktop_app_name}
Comment=A monitoring and serviceability tool for OpenJDK
Exec=/opt/rh/rh-thermostat16/root/usr/share/thermostat/bin/thermostat local
Icon=thermostat
```

必要に応じて、以下を行ってください。

```
[Desktop Entry]
Version=1.0
Type=Application
Name=Thermostat-1.6
Comment=A monitoring and serviceability tool for OpenJDK
Exec=scl enable rh-thermostat16 "thermostat local"
Icon=rh-thermostat16-thermostat
```

または、コマンドラインから **Thermostat** を実行します。

```
$ scl enable rh-thermostat16 "thermostat local"
```

その他の注意事項

rh-ruby22、rh-ruby23、rh-python34、rh-python35、rh-php56、rh-php70 コンポーネント

読み取り専用 NFS で Software Collections を使用すると、いくつかの制限があります。

- rh-ruby22 またはrh-ruby23 Software Collection が読み取り専用の NFS 上にあるときは、Ruby の gems はインストールできません。したがって、たとえば、**gem install ab** コマンドを使用して ab gem をインストールしようとすると、以下のようなエラーメッセージが表

示されます。

```
ERROR:  While executing gem ... (Errno::EROFS)
    Read-only file system @ dir_s_mkdir - /opt/rh/rh-ruby22/root/usr/local/share/gems
```

または **bundle update** または **bundle install** コマンドを実行して、ユーザーが外部ソースから gem を更新またはインストールしようとすると、同じ問題が発生します。

- Python Package Index (PyPI) を使用して読み取り専用 NFS に Python パッケージをインストールすると、この **pip** コマンドが失敗し、以下のようなエラーメッセージが表示されます。

```
Read-only file system: '/opt/rh/rh-python34/root/usr/lib/python3.4/site-packages/ipython-3.1.0.dist-info'
```

- **pear** コマンドを使用した読み取り専用 NFS への PEAR (PHP Extension and Application Repository) からパッケージのインストールに失敗し、エラーメッセージが表示されます。

```
Cannot install, php_dir for channel "pear.php.net" is not writeable by the current user
```

これは想定される動作です。

httpd コンポーネント

Apache の言語モジュールは、**Apache httpd** の Red Hat Software Collections バージョンでのみサポートされ、Red Hat Enterprise Linux のシステムバージョン **httpd** では対応していません。たとえば、rh-python35 コレクションの **mod_wsgi** モジュールは httpd24 コレクションでのみ使用できます。

すべてのコンポーネント

Red Hat Software Collections 2.0 以降、設定ファイル、変数データ、および各 Collections のランタイムデータは、以前のバージョンの Red Hat Software Collections とは異なるディレクトリーに保存されます。

coreutils、util-linux、screen コンポーネント

su、**login**、**screen** などの一部のユーティリティーは、すべてのケースで環境設定をエクスポートせず、予期せぬ結果になる可能性があります。そのため、**su** の代わりに**sudo** を使用し、**/etc/sudoers** ファイルに **env_keep** 環境変数を設定することを推奨します。下の順序でコマンドを実行できます。例を以下に示します。

```
su -l postgres -c "scl enable rh-postgresql94 psql"
```

以下の代わりとなります。

```
scl enable rh-postgresql94 bash
su -l postgres -c psql
```

screen、**login** などのツールを使用する場合は、以下のコマンドを使用して環境設定を保存できます。

```
source /opt/rh/<collection_name>/enable
```

php54 コンポーネント、

Red Hat Software Collections の **Alternative PHP Cache (APC)** は、ユーザーデータキャッシュにのみ提供されていることに注意してください。opcode キャッシュには、**Zend OPcache** が提供されます。

python コンポーネント

ユーザーがpython27、python33、rh-python34、rh-python35 Software Collection から複数のscldevel パッケージをインストールしようとすると、トランザクションチェックのエラーメッセージが返されます。パッケージ (**%scl_python**, **%scl_*prefix*_python**) が提供するマクロファイルのセットをユーザーが 1 つだけインストールできるため、これは想定される動作です。

PHP コンポーネント)

ユーザーがphp54、php55、rh-php56、rh-php70 Software Collection から複数の scldevel パッケージをインストールしようとすると、トランザクションチェックのエラーメッセージが返されます。パッケージ (**%scl_php**, **%scl_*prefix*_php**) が提供するマクロファイルのセットをユーザーが 1 つしかインストールできないため、これは想定される動作です。

ruby コンポーネント

ユーザーがruby193、ruby200、rh-ruby22、rh-ruby23 Software Collection から複数の scldevel パッケージをインストールしようとすると、トランザクションチェックのエラーメッセージが返されます。パッケージ (**%scl_ruby**, **%scl_*prefix*_ruby**) が提供するマクロファイルのセットをユーザーが 1 つだけインストールできるため、これは想定される動作です。

perl コンポーネント

ユーザーがperl516、rh-perl520、rh-perl524 Software Collections から複数の scldevel パッケージをインストールしようとすると、トランザクションチェックのエラーメッセージが返されます。パッケージ (**%scl_perl**, **%scl_*prefix*_perl**) が提供するマクロファイルのセットをユーザーが 1 つだけインストールできるため、これは想定される動作です。

nginx コンポーネント

ユーザーがnginx16 およびrh-nginx18 Software Collection から複数の scldevel パッケージをインストールしようとすると、トランザクションチェックのエラーメッセージが返されます。パッケージ (**%scl_nginx**, **%scl_*prefix*_nginx**) が提供するマクロファイルのセットをユーザーが 1 つだけインストールできるため、これは想定される動作です。

nodejsコンポーネント

nodejs010 Software Collection をインストールする場合、 gcc パッケージがすでにインストールされていない限り、nodejs010 はベースの Red Hat Enterprise Linux システムに **GCC** を依存関係としてインストールします。

rh-eclipse46 コンポーネント

Red Hat Enterprise Linux 7 の Eclipse SWT グラフィカルライブラリーは GTK 3.x を使用しています。Eclipse**Dark Theme** は GTK 3.x 上でまだ完全に安定していないため、このテーマはテクノロジープレビューとみなされ、サポートされていません。Red Hat Technology Previews の詳細については、https://access.redhat.com/support/offerings/techpreview/ を参照してください。 は修正されていません。

1.6. 非推奨の機能

httpd24 コンポーネント (BZ#1434053)

以前では、名前ベースの SSL 仮想ホスト選択が必要な SSL/TLS 設定で、**Host:** ヘッダーで提供されるホスト名が Server Name Indication (SNI) ヘッダーで提供されるホスト名と一致していなければ、**mod_ssl** モジュールは **400 Bad Request** エラーのあるリクエストを拒否していました。選択されたバーチャルホスト間で設定された SSL/TLS セキュリティーパラメーターが同じであれば、アップストリーム **mod_ssl** の動作に合わせて、そのようなリクエストは拒否されなくなりました。

第2章 インストール

本章では、コンテンツセットへのアクセス方法、システムへの Red Hat Software Collections 3.1 のインストール方法、および Red Hat Software Collections の再構築方法について詳しく説明します。

2.1. RED HAT SOFTWARE COLLECTIONS へのアクセス

Red Hat Software Collections コンテンツセットは、https://access.redhat.com/solutions/472793 に記載されている Red Hat Enterprise Linux 6 および Red Hat Enterprise Linux 7 サブスクリプションをご利用いただけます。Red Hat Subscription Management (RHSM) でシステムを登録する方法は、Using and Configuring Red Hat Subscription Manager を参照してください。RHSM を使用して Red Hat Software Collections を有効にする方法は、「Red Hat Subscription Management の使用」 を参照してください。

Red Hat Software Collections 2.2 以降、Red Hat Software Collections および Red Hat Developer Toolset のコンテンツは、(特に Server および Workstation 向けに) https://access.redhat.com/downloads にて ISO 形式でも提供されています。なお、「Optional チャネルのパッケージ」 に掲載されている **Optional** チャネルを必要とするパッケージは、ISO イメージからはインストールできません。

注記

Optional チャネルを必要とするパッケージは、ISO イメージからはインストールできません。**Optional** チャネルの有効化を必要とするパッケージのリストは、「Optional チャネルのパッケージ」 に記載されています。

ベータコンテンツは ISO 形式では使用できません。

2.1.1. Red Hat Subscription Management の使用

システムが Red Hat Subscription Management に登録されている場合は、以下の手順を実施して、Red Hat Software Collections のリポジトリーへのアクセスを提供するサブスクリプションを割り当て、リポジトリーを有効にします。

1. システムで利用可能なサブスクリプションのリストを表示し、Red Hat Software Collections を提供するサブスクリプションのプール ID を判別します。これを行うには、**root** で次のコマンドを実行します。

   ```
   subscription-manager list --available
   ```

   このコマンドは、使用可能なサブスクリプションごとに、その名前、一意の識別子、有効期限、およびそれに関連するその他の詳細を表示します。プール ID は、**Pool Id** で始まる行にリスト表示されます。

2. **root** で以下のコマンドを実行して、適切なサブスクリプションをシステムに割り当てます。

   ```
   subscription-manager attach --pool=pool_id
   ```

   pool_id を、直前のステップで確認したプール ID に置き換えます。システムに割り当てているサブスクリプションのリストを随時確認するには、**root** で以下を入力します。

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

3. 利用可能な Yum list リポジトリーのリストを表示して、リポジトリーメタデータを取得し、Red Hat Software Collections リポジトリーの正確な名前を決定します。**root** で以下のコマンドを実行します。

```
subscription-manager repos --list
```

または、**yum repolist all** を簡単なリストに対して実行します。

リポジトリー名は、使用している Red Hat Enterprise Linux のバージョンによって異なり、以下のフォーマットに基づいています。

```
rhel-variant-rhscl-6-rpms
rhel-variant-rhscl-6-debug-rpms
rhel-variant-rhscl-6-source-rpms

rhel-server-rhscl-6-eus-rpms
rhel-server-rhscl-6-eus-source-rpms
rhel-server-rhscl-6-eus-debug-rpms

rhel-variant-rhscl-7-rpms
rhel-variant-rhscl-7-debug-rpms
rhel-variant-rhscl-7-source-rpms

rhel-server-rhscl-7-eus-rpms
rhel-server-rhscl-7-eus-source-rpms
rhel-server-rhscl-7-eus-debug-rpms
```

variant を、Red Hat Enterprise Linux システムのバリアント (つまり **server** または **workstation** に置き換えます。Red Hat Software Collections は、 **Client** または **ComputeNode** バリアントではサポートされないことに注意してください。

4. **root** で以下のコマンドを実行して、適切なリポジトリーを有効にします。

```
subscription-manager repos --enable repository
```

サブスクリプションがシステムに割り当てられたら、「Red Hat Software Collections のインストール」の説明に従って Red Hat Software Collections をインストールできます。Red Hat Subscription Management を使用してシステムを登録し、サブスクリプションに関連付ける方法は、Using and Configuring Red Hat Subscription Manager を参照してください。

注記

RHN によるサブスクリプションは利用できなくなりました。

2.1.2. Optional チャネルのパッケージ

Red Hat Software Collections 3.1 パッケージの中には、パッケージの完全なインストールを完了するために、**Optional** チャネルの有効化を必要とするものがあります。システムをこのチャンネルにサブスクライブする方法の詳細については、Red Hat Subscription Management の場合はhttps://access.redhat.com/solutions/392003、システムが RHN Classic に登録されている場合はhttps://access.redhat.com/solutions/70019 の関連ナレッジベース記事を参照してください。

Optional チャネルの有効化を必要とする Red Hat Enterprise Linux 6 の Software Collections パッケージを以下の表に示します。

表2.1 Red Hat Enterprise Linux 6 の Optional チャネルの有効化を必要とするパッケージ

Software Collection のパッケージ	Optional チャネルの必須パッケージ
devtoolset-6-dyninst-testsuite	glibc-static
devtoolset-7-dyninst-testsuite	glibc-static
rh-git29-git-all	cvsps, perl-Net-SMTP-SSL
rh-git29-git-cvs	cvsps
rh-git29-git-email	perl-Net-SMTP-SSL
rh-git29-perl-Git-SVN	perl-YAML、subversion-perl
rh-mariadb101-boost-devel	libicu-devel
rh-mariadb101-boost-examples	libicu-devel
rh-mariadb101-boost-static	libicu-devel
rh-mongodb30upg-boost-devel	libicu-devel
rh-mongodb30upg-boost-examples	libicu-devel
rh-mongodb30upg-boost-static	libicu-devel
rh-mongodb30upg-yaml-cpp-devel	libicu-devel
rh-mongodb32-boost-devel	libicu-devel
rh-mongodb32-boost-examples	libicu-devel
rh-mongodb32-boost-static	libicu-devel
rh-mongodb32-yaml-cpp-devel	libicu-devel
rh-mongodb34-boost-devel	libicu-devel
rh-mongodb34-boost-examples	libicu-devel
rh-mongodb34-boost-static	libicu-devel
rh-mongodb34-yaml-cpp-devel	libicu-devel

Software Collection のパッケージ	Optional チャネルの必須パッケージ
rh-php56-php-imap	libc-client
rh-php56-php-recode	recode
rh-php70-php-imap	libc-client
rh-php70-php-recode	recode

Red Hat Enterprise Linux 7 の **Optional** チャネルを必要とする Software Collections パッケージを以下の表に示します。

表2.2 Red Hat Enterprise Linux 7 の Optional チャネルの有効化を必要とするパッケージ

Software Collection のパッケージ	Optional チャネルの必須パッケージ
devtoolset-7-dyninst-testsuite	glibc-static
devtoolset-7-gcc-plugin-devel	libmpc-devel
httpd24-mod_ldap	apr-util-ldap
rh-eclipse46	ruby-doc
rh-eclipse46-eclipse-dltk-ruby	ruby-doc
rh-eclipse46-eclipse-dltk-sdk	ruby-doc
rh-eclipse46-eclipse-dltk-tests	ruby-doc
rh-git29-git-all	cvsps
rh-git29-git-cvs	cvsps
rh-git29-perl-Git-SVN	subversion-perl
rh-perl520-perl-Pod-Perldoc	groff

Optional チャネルの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。詳細は、ナレッジベースの記事 https://access.redhat.com/articles/1150793 を参照してください。

2.2. RED HAT SOFTWARE COLLECTIONS のインストール

Red Hat Software Collections は、Red Hat Enterprise Linux に含まれる標準のパッケージ管理ツールを使用して、インストール、更新、アンインストールが可能な RPM パッケージのコレクションとして配布されます。Red Hat Software Collections をシステムにインストールするには、有効なサブスクリプ

ションが必要です。システムを適切なサブスクリプションに関連付け、Red Hat Software Collections にアクセスする方法は、「Red Hat Software Collections へのアクセス」 を参照してください。

Red Hat Software Collections 3.1 を使用するには、ベータリリースを含む以前のプレリリースバージョンを削除する必要があります。以前のバージョンの Red Hat Software Collections 3.1 をインストールしていた場合は、システムからアンインストールし、「Red Hat Software Collections のアンインストール」 および「Software Collection の個別インストール」 セクションで説明されているように新しいバージョンをインストールしてください。

Red Hat Enterprise Linux 6 から Red Hat Enterprise Linux 7 へのインプレースアップグレードは、Red Hat Software Collections ではサポートされていません。したがって、アップグレード後にインストールされた Software Collections が正しく動作しない可能性があります。Red Hat Enterprise Linux 6 から Red Hat Enterprise Linux 7 にアップグレードする場合は、すべての Red Hat Software Collections パッケージを削除し、インプレースアップグレードを実行し、Red Hat Software Collections リポジトリーを更新し、再度 Software Collections パッケージをインストールすることが強く推奨されます。アップグレードする前に、すべてのデータのバックアップを作成することが推奨されます。

2.2.1. Software Collection の個別インストール

表1.1「Red Hat Software Collections 3.1 コンポーネント」 に記載されている Software Collection をインストールするには、シェルプロンプトで **root** として次のように入力して、対応するメタパッケージをインストールします。

```
yum install software_collection...
```

software_collection を、インストールする Software Collections のスペース区切りリストに置き換えます。たとえば、php54 および rh-mariadb100 をインストールし、**root** として以下を入力します。

```
~]# yum install rh-php56 rh-mariadb100
```

これにより、選択した Software Collection のメインメタパッケージと、必要なパッケージの依存関係がインストールされます。追加モジュールなどの追加パッケージをインストールする方法は、「オプションパッケージのインストール」 を参照してください。

2.2.2. オプションパッケージのインストール

Red Hat Software Collections の各コンポーネントは、デフォルトでインストールされていない複数のオプションパッケージとともに配布されます。特定の Software Collection に含まれていて、システムにインストールされていないパッケージのリストを表示するには、シェルプロンプトで次のコマンドを実行します。

```
yum list available software_collection-\*
```

これらのオプションのパッケージのいずれかをインストールするには、**root** で以下を入力します。

```
yum install package_name...
```

package_name を、インストールするパッケージのリストに置き換えます。例えば、rh-perl520-perl-CPAN と rh-perl520-perl-Archive-Tar をインストールするには、次のように入力します。

```
~]# yum install rh-perl524-perl-CPAN rh-perl524-perl-Archive-Tar
```

2.2.3. デバッグ情報のインストール

Red Hat Software Collections パッケージのデバッグ情報をインストールするには、yum-utils パッケージがインストールされていることを確認し、**root** で以下のコマンドを入力します。

```
debuginfo-install package_name
```

例えば、rh-ruby22-ruby パッケージのデバッグ情報をインストールするには、次のように入力します。

```
~]# debuginfo-install rh-ruby22-ruby
```

これらのパッケージを含むリポジトリーにアクセスできる必要があることに注意してください。システムが Red Hat Subscription Management に登録されている場合は、「Red Hat Subscription Management の使用」 で説明しているように、**rhel-*variant*-rhscl-6-debug-rpms** または **rhel-*variant*-rhscl-7-debug-rpms** リポジトリーを有効化します。debuginfo パッケージへのアクセス方法はhttps://access.redhat.com/solutions/9907を参照してください。

2.3. RED HAT SOFTWARE COLLECTIONS のアンインストール

Software Collections コンポーネントをアンインストールするには、**root** で次のコマンドを実行します。

```
yum remove software_collection\*
```

software_collection を、アンインストールする Software Collection コンポーネントに置き換えます。

Red Hat Software Collections が提供するパッケージをアンインストールしても、これらのツールの Red Hat Enterprise Linux システムバージョンには影響がないことに注意してください。

2.4. RED HAT SOFTWARE COLLECTIONS の再構築

<collection>-build パッケージはデフォルトでは提供されません。コレクションを再構築して、**rpmbuild --define 'scl foo'** コマンドを使用しない場合には、最初にメタパッケージを再構築する必要があります。これにより、<collection>-build パッケージが提供されます。

既存のコレクションは、異なる内容で再構築しないでください。既存のコレクションに新しいパッケージを追加するには、新しいパッケージを含む新しいコレクションを作成し、元のコレクションからのパッケージに依存する必要があります。元のコレクションは、変更せずに使用する必要があります。

Software Collections のビルドに関する詳細は、Red Hat Software Collections パッケージガイド を参照してください。

第3章 使用法

この章では、Red Hat Software Collections 3.1 を再構築して使用し、Red Hat Software Collections を使用するアプリケーションをデプロイするために必要な手順を説明します。

3.1. RED HAT SOFTWARE COLLECTIONS の使用

3.1.1. Software Collection からの実行可能ファイルの実行

特定の Software Collection から実行ファイルを実行するには、シェルプロンプトで以下のコマンドを入力します。

```
scl enable software_collection... 'command...'
```

または、以下のコマンドを使用します。

```
scl enable software_collection... -- command...
```

software_collection を、使用する Software Collections のスペース区切りのリストに置き換え、*command* を、実行するコマンドに置き換えます。たとえば、perl516 Software Collection から Perl インタープリターで **hello.pl** という名前が付けられたファイルに保存されている Perl プログラムを実行するには、以下を入力します。

```
~]$ scl enable rh-perl524 'perl hello.pl'
Hello, World!
```

この **scl** ユーティリティーを使用してコマンドを実行すると、同等の Red Hat Enterprise Linux システムの代わりに、選択した Software Collection から実行可能なものを使用して実行できます。Red Hat Software Collections で配布される Software Collections の完全リストは、表1.1「Red Hat Software Collections 3.1 コンポーネント」を参照してください。

3.1.2. デフォルトで Software Collection を使用したシェルセッションの実行

Red Hat Enterprise Linux の同等のものよりも選択した Software Collection の実行可能ファイルで新しいシェルセッションを開始するには、シェルプロンプトで次のように入力します。

```
scl enable software_collection... bash
```

software_collection を、使用する Software Collections のスペース区切りリストに置き換えます。例えば、python27 と rh-postgresql95 Software Collections をデフォルトにして新しいシェルセッションを開始するには、次のように入力します。

```
~]$ scl enable python27 rh-postgresql95 bash
```

現行セッションで有効になっている Software Collections のリストは、**$X_SCLS** 環境変数に保存されます。以下に例を示します。

```
~]$ echo $X_SCLS
python27 rh-postgresql95
```

Red Hat Software Collections で配布される Software Collections の完全リストは、表1.1「Red Hat Software Collections 3.1 コンポーネント」を参照してください。

3.1.3. Software Collection からのシステムサービスの実行

システムサービスを含む Software Collections は、対応する init スクリプトを **/etc/rc.d/init.d/** ディレクトリーにインストールします。現行のセッションでそのようなサービスを起動するには、シェルプロンプトで **root** として以下を入力します。

```
service software_collection-service_name start
```

software_collection を、Software Collection および *service_name* を、開始するサービスの名前に置き換えます。システムの起動時にこのサービスが自動的に開始するように設定するには、**root** で以下のコマンドを入力します。

```
chkconfig software_collection-service_name on
```

たとえば、rh-postgresql95 Software Collection から **postgresql** サービスを、ランレベル 2、3、4、5 で有効にして起動するには、**root** で以下を入力します。

```
~]# service rh-postgresql95-postgresql start
Starting rh-postgresql95-postgresql service:               [  OK  ]
~]# chkconfig rh-postgresql95-postgresql on
```

Red Hat Enterprise Linux 6 でシステムサービスを管理する方法の詳細については、Red Hat Enterprise Linux 6 Deployment Guideを参照してください。Red Hat Software Collections で配布される Software Collections の完全リストは、表1.1「Red Hat Software Collections 3.1 コンポーネント」 を参照してください。

3.2. SOFTWARE COLLECTION からの手動ページへのアクセス

すべての Software Collection には、このコンポーネントの内容を説明する一般的な man ページが含まれています。各 man ページにはコンポーネントと同じ名前が付いており、**/opt/rh** ディレクトリーにあります。

Software Collection の man ページを確認するには、以下のコマンドを入力します。

```
scl enable software_collection 'man software_collection'
```

software_collection を、特定の Red Hat Software Collections コンポーネントに置き換えます。例えば、rh-mariadb101 のマニュアルページを表示するには、次のように入力します。

```
~]$ scl enable rh-mariadb101 "man rh-mariadb101"
```

3.3. RED HAT SOFTWARE COLLECTIONS を使用するアプリケーションのデプロイ

通常、以下の 2 つの方法のいずれかを使用して、実稼働環境の Red Hat Software Collections のコンポーネントに依存するアプリケーションをデプロイすることができます。

- 必要な Software Collections およびパッケージをすべて手動でインストールしてから、アプリケーションをデプロイする、または
- アプリケーション用の新しい Software Collection を作成し、必要な Software Collections およびその他のパッケージをすべて依存関係として指定する

個々の Red Hat Software Collections コンポーネントを手動でインストールする方法は、「Red Hat Software Collections のインストール」 を参照してください。Red Hat Software Collections の使用方法に関する詳細は、「Red Hat Software Collections の使用」 を参照してください。カスタム Software Collection を作成する方法や、既存のソフトウェアを拡張する方法については、Red Hat Software Collections Packaging Guideを参照してください。

3.4. RED HAT SOFTWARE COLLECTIONS コンテナーイメージ

Red Hat Software Collections に基づくコンテナーイメージには、アプリケーション、デーモン、およびデータベースが含まれます。イメージは、Red Hat Enterprise Linux 7 Server および Red Hat Enterprise Linux Atomic Host で実行できます。使用方法は、Using Red Hat Software Collections 3 Container Imagesを参照してください。Red Hat Software Collections バージョン 2.4 以前の Red Hat Software Collections バージョン 2.4 をベースとしたコンテナーイメージの詳細は、Using Red Hat Software Collections 2 Container Images を参照してください。

以下のコンテナーイメージは Red Hat Software Collections 3.1 で利用可能です。

- rhscl/devtoolset-7-toolchain-rhel7
- rhscl/devtoolset-7-perftools-rhel7
- rhscl/httpd-24-rhel7
- rhscl/mongodb-36-rhel7
- rhscl/perl-526-rhel7
- rhscl/php-70-rhel7
- rhscl/postgresql-10-rhel7
- rhscl/ruby-25-rhel7
- rhscl/varnish-5-rhel7

以下のコンテナーイメージは Red Hat Software Collections 3.0 をベースとしています。

- rhscl/mariadb-102-rhel7
- rhscl/mongodb-34-rhel7
- rhscl/nginx-112-rhel7
- rhscl/nodejs-8-rhel7
- rhscl/php-71-rhel7
- rhscl/postgresql-96-rhel7
- rhscl/python-36-rhel7

以下のコンテナーイメージは、Red Hat Software Collections 2.4 に基づいています。

- rhscl/devtoolset-6-toolchain-rhel7
- rhscl/devtoolset-6-perftools-rhel7

- rhscl/nginx-110-rhel7
- rhscl/nodejs-6-rhel7
- rhscl/python-27-rhel7
- rhscl/ruby-24-rhel7
- rhscl/ror-50-rhel7
- rhscl/thermostat-16-agent-rhel7 (EOL)
- rhscl/thermostat-16-storage-rhel7 (EOL)

以下のコンテナーイメージは、Red Hat Software Collections 2.3 に基づいています。

- rhscl/mysql-57-rhel7
- rhscl/perl-524-rhel7
- rhscl/redis-32-rhel7
- rhscl/mongodb-32-rhel7
- rhscl/php-56-rhel7
- rhscl/python-35-rhel7
- rhscl/ruby-23-rhel7

以下のコンテナーイメージは、Red Hat Software Collections 2.2 に基づいています。

- rhscl/devtoolset-4-toolchain-rhel7
- rhscl/devtoolset-4-perftools-rhel7
- rhscl/mariadb-101-rhel7
- rhscl/nginx-18-rhel7
- rhscl/nodejs-4-rhel7
- rhscl/postgresql-95-rhel7
- rhscl/ror-42-rhel7
- rhscl/thermostat-1-agent-rhel7 (EOL)
- rhscl/varnish-4-rhel7

以下のコンテナーイメージは、Red Hat Software Collections 2.0 に基づいています。

- rhscl/mariadb-100-rhel7
- rhscl/mongodb-26-rhel7
- rhscl/mysql-56-rhel7

- rhscl/nginx-16-rhel7 (EOL)
- rhscl/passenger-40-rhel7
- rhscl/perl-520-rhel7
- rhscl/postgresql-94-rhel7
- rhscl/python-34-rhel7
- rhscl/ror-41-rhel7
- rhscl/ruby-22-rhel7
- rhscl/s2i-base-rhel7

EOL(End of Life) と表示されているイメージはサポート対象外となります。

第4章 個別の SOFTWARE COLLECTIONS の詳細

本章では、特定の Software Collections の詳細に重点を置き、これらのコンポーネントに関する追加情報を提供します。

4.1. RED HAT DEVELOPER TOOLSET

Red Hat Developer Toolset は、Red Hat Enterprise Linux プラットフォームで作業する開発者向けに設計されています。Red Hat Developer Toolset は、現在のバージョンの**GNU Compiler Collection**、**GNU Debugger**、およびその他の開発、デバッグ、パフォーマンス監視ツールを提供します。他の Software Collections と同様に、追加のツールセットが **/opt/** ディレクトリーにインストールされます。これらのツールは、提供された **scl** ユーティリティーを使用してオンデマンドでユーザーが有効にします。他の Software Collections と同様に、これらのツールの Red Hat Enterprise Linux システムバージョンを置き換えることはありません。また、**scl** ユーティリティーを使用して明示的に呼び出されない限り、これらのシステムバージョンを優先して使用することもできます。

機能の概要については、『Red Hat Developer Toolset Release Notes 』のMain Features セクションを参照してください。

コンポーネントの完全なリストについては、『Red Hat Developer Toolset User Guide 』 のRed Hat Developer Toolset Components表を参照してください。

なお、Red Hat Developer Toolset 3.1 以降、Red Hat Developer Toolset にはrh-java-common Software Collection が必要です。

4.2. ECLIPSE 4.6.3

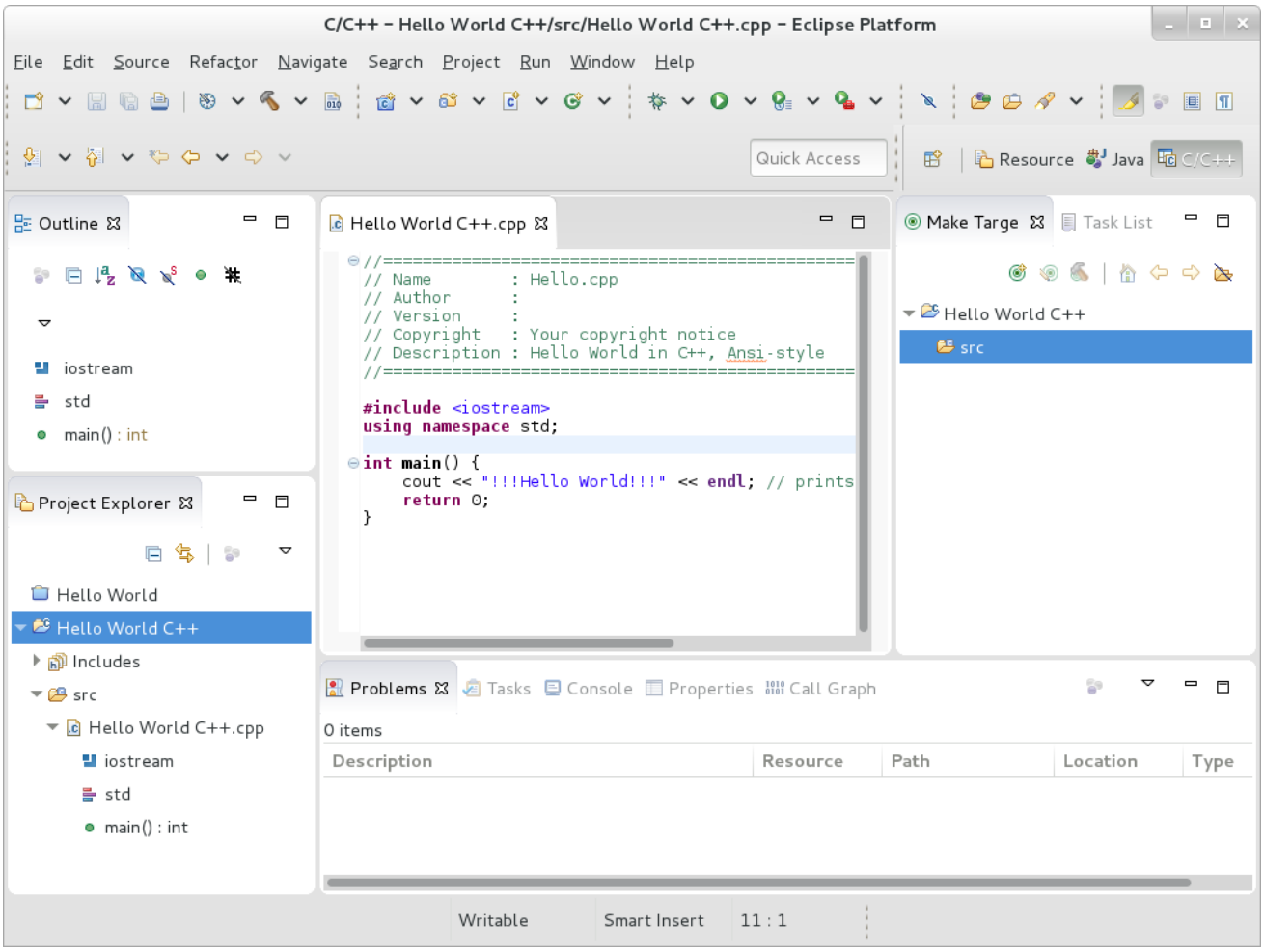
Red Hat Enterprise Linux 7 で利用可能なrh-eclipse46 Software Collection には、Eclipse Foundation の Neon release train をベースにした**Eclipse 4.6.3** が含まれています。この統合開発環境は、以前は Red Hat Developer Toolset の一部として提供されていました。なお、rh-eclipse46 Software Collection にはrh-java-common Collection が必要です。

注記

Eclipse の新しいバージョンが、Red Hat Developer Tools オファリングの rh-eclipse47 コンポーネントとして提供されています。詳しくは、Using Eclipse をご覧ください。

Eclipse は、開発プロセスの各フェーズにツールを提供する強力な開発環境です。豊富な開発エクスペリエンスを実現するためにさまざまな異種ツールを統一された環境に統合し、完全に設定可能なユーザーインターフェイスを提供して、さまざまな方法で拡張できるプラグイン可能なアーキテクチャーを特長としています。たとえば、**Valgrind** プラグインを使用すると、プログラマーは、**Eclipse** ユーザーインターフェイスを介してコマンドラインで実行されるメモリープロファイリングを実行できます。

図4.1 Eclipse セッションの例



[D]

Eclipse は、コマンドラインツールを使用した従来の対話に代わるグラフィカル開発環境を提供するため、コマンドラインインターフェイスを使用しない開発者にとっては歓迎される代わりの手段となります。従来のほぼコマンドラインベースの Linux ツール (**gcc** や **gdb** など) および **Eclipse** は、プログラミングに対して 2 つの異なるアプローチを提供します。

なお、Red Hat JBoss Middleware 用のアプリケーションを開発する場合や、OpenShift Tools のサポートが必要な場合は、Red Hat JBoss Developer Studio を使用することを推奨します。

表4.1 rh-eclipse46 Software Collection に含まれる Eclipse コンポーネント

パッケージ	説明
rh-eclipse46-eclipse-cdt	C や C++での開発のための機能やプラグインを提供する C/C++ Development Tooling(CDT)。
rh-eclipse46-eclipse-changelog	チェンジログファイルを作成、管理するための ChangeLog プラグイン。
rh-eclipse46-eclipse-egit	EGit は、Git リポジトリーと対話するための機能とプラグインを提供する **Eclipse** のチームプロバイダーです。

パッケージ	説明
rh-eclipse46-eclipse-emf	**Eclipse** Modeling Framework (EMF)。構造化データモデルに基づいてアプリケーションを構築できます。
rh-eclipse46-eclipse-epp-logging	Eclipse エラー報告ツール。
rh-eclipse46-eclipse-gcov	GCov プラグイン 。**GCov** テストカバレッジプログラムを **Eclipse** と統合します。
rh-eclipse46-eclipse-gef	既存のアプリケーションモデルからリッチなグラフィカルエディターを作成することができる Graphical Editing Framework(GEF) です。
rh-eclipse46-eclipse-gprof	Gprof プラグイン 。**Gprof** パフォーマンス分析ユーティリティーを **Eclipse** と統合します。
rh-eclipse46-eclipse-jdt	**Eclipse** Java 開発ツール (JDT) プラグイン。
rh-eclipse46-eclipse-jgit	JGit、**Git** リビジョン管理システムの Java 実装。
rh-eclipse46-eclipse-manpage	Man Page プラグイン。**Eclipse** で man ページを表示できるようにします。
rh-eclipse46-eclipse-mpc	**Eclipse** Marketplace Client。
rh-eclipse46-eclipse-mylyn	**Eclipse** のタスク管理システムである Mylyn。
rh-eclipse46-eclipse-oprofile	OProfile プラグイン。**OProfile** と **Eclipse** を統合します。
rh-eclipse46-eclipse-pde	**Eclipse** プラグインを開発するためのプラグイン開発環境です。
rh-eclipse46-eclipse-perf	Perf プラグイン。**perf** ツールを **Eclipse** と統合します。
rh-eclipse46-eclipse-ptp	同期プロジェクトをサポートするPTP projectのサブセット。
rh-eclipse46-eclipse-pydev	**Eclipse** 用のフル機能の Python IDE。
rh-eclipse46-eclipse-remote	拡張可能なリモートサービスのフレームワークを提供する Remote Services プラグインです。
rh-eclipse46-eclipse-rpm-editor	**Eclipse** Spec File Editor。RPM 仕様ファイルを管理できます。
rh-eclipse46-eclipse-rse	Remote System Explorer (RSE) フレームワーク。**Eclipse** からリモートシステムを操作できるようになります。
rh-eclipse46-eclipse-systemtap	SystemTap プラグイン。**SystemTap** と **Eclipse** を統合します。
rh-eclipse46-eclipse-valgrind	Valgrind プラグイン。**Valgrind** と **Eclipse** を統合します。

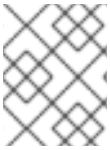
パッケージ	説明
rh-eclipse46-eclipse-webtools	**Eclipse** Webtools プラグイン。

4.2.1. Eclipse のインストール

Eclipse 開発環境は、RPM パッケージのコレクションとして提供されます。rh-eclipse46 Software Collection をインストールするには、**root** として次のコマンドを入力します。

```
yum install rh-eclipse46
```

利用可能なコンポーネントのリストについては、表4.1「rh-eclipse46 Software Collection に含まれる Eclipse コンポーネント」 を参照してください。



注記

rh-eclipse46 Software Collection は、C、C++、Java の開発を完全にサポートしていますが、Fortran プログラミング言語の**サポートはありません**。

4.2.2. Eclipse の使用

rh-eclipse46 Software Collection を起動するには、パネルから **Applications → Programming → Red Hat Eclipse** を選択するか、Shell プロンプトで以下を入力します。

```
scl enable rh-eclipse46 eclipse
```

Eclipse の起動中に、 ワークスペース、つまりプロジェクトを保存するディレクトリーを選択するように求められます。デフォルトのオプションである ~/**workspace**/ を使用するか、**Browse** ボタンをクリックしてファイルシステムを参照し、カスタムディレクトリーを選択できます。さらに、**Use this as the default and do not ask again** チェックボックスを選択して、次回この開発環境を実行するときに **Eclipse** がこのダイアログボックスを表示しないようにすることもできます。完了したら、**OK** ボタンをクリックして選択を確認し、起動を続行します。

4.2.2.1. Red Hat Developer Toolset Toolchain の使用

Red Hat Developer Toolset の **GNU Compiler Collection** と **binutils** をサポートする rh-eclipse46 Software Collection を使用するには、devtoolset-7-toolchain パッケージがインストールされていることを確認し、「Eclipse の使用」 で説明されているようにアプリケーションを実行します。rh-eclipse46 Collection は、デフォルトで Red Hat Developer Toolset ツールチェーンを使用します。

お使いのシステムに devtoolset-7-toolchain パッケージをインストールする方法の詳細は、Red Hat Developer Toolset User Guide をご覧ください。

重要

以前に Red Hat Enterprise Linux 版の**GNU Compiler Collection**でビルドしたプロジェクトで作業する場合は、以前のビルド結果をすべて破棄してください。これを行うには、**Eclipse** でプロジェクトを開き、メニューから **Project → Clean** を選択します。

4.2.2.2. Red Hat Enterprise Linux Toolchain の使用

Red Hat Enterprise Linux で配布されるツールチェーンへのサポートのある rh-eclipse46 Software Collection を使用するには、プロジェクトの設定を変更して、Red Hat Enterprise Linux システムバージョンの **gcc**、**g++**, および **as** への絶対パスを使用します。

現在のプロジェクトでツールの Red Hat Enterprise Linux システムバージョンを明示的に使用するように **Eclipse** を設定するには、以下の手順を実行します。

1. C/C++ パースペクティブで、メインメニューバーから **Project → Properties** を選択して、プロジェクトプロパティーを開きます。
2. ダイアログボックスの左側にあるメニューで、**C/C++ Build → Settings** をクリックします。
3. **Tool Settings** タブを選択します。
4. C プロジェクトで作業している場合は、以下を行います。
    1. **GCC C Compiler** または **Cross GCC Compiler** を選択し、**Command** フィールドの値を次のように変更します。

        ```
        /usr/bin/gcc
        ```

    2. **GCC C Linker** または **Cross GCC Linker** を選択し、**Command** フィールドの値を次のように変更します。

        ```
        /usr/bin/gcc
        ```

    3. **GCC Assembler** または **Cross GCC Assembler** を選択し、**Command** フィールドの値を次のとおりに変更します。

        ```
        /usr/bin/as
        ```

    C++ プロジェクトで作業している場合:

    1. **GCC C++ Compiler** または **Cross G++ Compiler** を選択し、**Command** フィールドの値を次のように変更します。

        ```
        /usr/bin/g++
        ```

    2. **GCC C Compiler** または **Cross GCC Compiler** を選択し、**Command** フィールドの値を次のように変更します。

        ```
        /usr/bin/gcc
        ```

    3. **GCC C++ Linker** または **Cross G++ Linker** を選択し、**Command** フィールドの値を次のように変更します。

        ```
        /usr/bin/g++
        ```

    4. **GCC Assembler** または **Cross GCC Assembler** を選択し、**Command** フィールドの値を次のとおりに変更します。

        ```
        /usr/bin/as
        ```

5. **OK** ボタンをクリックすると、設定変更が保存されます。

4.2.3. 関連情報

Eclipse とそのすべての機能の詳細な説明は、本書の範囲を超えています。詳細は、以下に記載のドキュメントを参照してください。

インストールされているドキュメント

- **Eclipse** には、統合された各機能およびツールに関する幅広いドキュメントを提供する **ヘルプ** システムが組むこまれています。これにより、新規開発者が使いこなせるようになるまでの初期投資を大幅に軽減することができます。このヘルプセクションの使用方法は、以下にリンクされている『Red Hat Enterprise Linux Developer Guide 』 で詳しく説明されています。

関連項目

- Using Eclipse: Red Hat Developer Tools の rh-eclipse47 コンポーネントの使用について説明しています。
- 『Red Hat Developer Toolset User Guide 』のRed Hat Developer Toolset の章では、Red Hat Developer Toolset の概要と、システムへのインストール方法の詳細を説明しています。
- 『Red Hat Developer Toolset User Guide 』のGNU Compiler Collection(GCC) の章では、C、C++、Fortran で書かれたプログラムをコマンドラインでコンパイルする方法が紹介されています。

4.3. RUBY ON RAILS 5.0

Red Hat Software Collections 3.1 はrh-ruby24 Software Collection を rh-ror50 Collection と共に提供します。

Ruby on Rails 5.0 をインストールするには、**root** で以下のコマンドを入力します。

```
yum install rh-ror50
```

rh-ror50 Software Collection のパッケージをインストールすると、依存関係にある rh-ruby24 と rh-nodejs6 が自動的に取り込まれます。

rh-nodejs6 Collection は、アセットパイプライン内の特定の gem が、 **sass** や **coffee-script** のソースファイルなどの Web リソースを後処理するために使用します。また、**Action Cable** フレームワークは、rh-nodejs6 Rails で **WebSocket** を処理するためのものです。

rh-nodejs6 を必要とせずに **rails s** コマンドを実行するには、**Gemfile** 内の **coffee-rails** gem と **uglifier** gem を無効にします。

Node.js なしで **Ruby on Rails** を実行するには、次のコマンドを実行します。実行すると、自動的に rh-ruby24 が有効になります。

```
scl enable rh-ror50 bash
```

すべての機能を備えた **Ruby on Rails** を実行するには、rh-nodejs6 Software Collection も有効にします。

```
scl enable rh-ror50 rh-nodejs6 bash
```

rh-ror50 Software Collection は、 rh-ruby24 とrh-nodejs6 のコンポーネントと共にサポートされています。

4.4. MONGODB 3.6

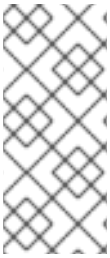
rh-mongodb36 Software Collection は、Red Hat Enterprise Linux 7 でのみ利用できます。Red Hat Enterprise Linux 6 で **MongoDB 3.4** を使用する方法は、 「MongoDB 3.4」 を参照してください。

rh-mongodb36 Collection をインストールするには、**root** で次のコマンドを入力します。

```
yum install rh-mongodb36
```

MongoDB シェルユーティリティーを実行するには、以下のコマンドを入力します。

```
scl enable rh-mongodb36 'mongo'
```



注記

rh-mongodb36-mongo-cxx-driver パッケージは、Red Hat Developer Toolset 6 の **GCC** を使用して **-std=gnu++14** オプションでビルドされています。C++11 (以降) の機能を使用する MongoDB C++ ドライバーの共有ライブラリーを使用するバイナリーは、Red Hat Developer Toolset 6 以降でもビルドする必要があります。Red Hat Developer Toolset 6 User Guide の C++ compatibility を参照してください。

MongoDB デーモンを起動するには、**root** で以下のコマンドを入力します。

```
systemctl start rh-mongodb36-mongod.service
```

システムの起動時に MongoDB デーモンを起動するには、**root** でこのコマンドを入力します。

```
systemctl enable rh-mongodb36-mongod.service
```

MongoDB シャードサーバーを起動するには、**root** で以下のコマンドを入力します。

```
systemctl start rh-mongodb36-mongos.service
```

システムの起動時に MongoDB シャードサーバーを起動するには、**root** でこのコマンドを入力します。

```
systemctl enable rh-mongodb36-mongos.service
```

少なくとも 1 つの設定サーバーを起動し、**mongos.conf** ファイルで指定しない限り、MongoDB シャーディングサーバーは機能しないことに注意してください。

4.5. MONGODB 3.4

rh-mongodb34 Collection をインストールするには、**root** で次のコマンドを入力します。

```
yum install rh-mongodb34
```

MongoDB シェルユーティリティーを実行するには、以下のコマンドを入力します。

```
scl enable rh-mongodb34 'mongo'
```

注記

rh-mongodb34-mongo-cxx-driver パッケージは、Red Hat Developer Toolset 6 の **GCC** を使用して **-std=gnu++14** オプションでビルドされています。C++11 (以降) の機能を使用する MongoDB C++ ドライバーの共有ライブラリーを使用するバイナリーは、Red Hat Developer Toolset 6 でもビルドする必要があります。Red Hat Developer Toolset 6 User Guide の C++ compatibility を参照してください。

Red Hat Enterprise Linux 6 上の MongoDB 3.4

Red Hat Enterprise Linux 6 を使用している場合は、以下の手順がシステムに適用されます。

MongoDB デーモンを起動するには、**root** で以下のコマンドを入力します。

```
service rh-mongodb34-mongod start
```

システムの起動時に MongoDB デーモンを起動するには、**root** でこのコマンドを入力します。

```
chkconfig rh-mongodb34-mongod on
```

MongoDB シャードサーバーを起動するには、**root** でこのコマンドを入力します。

```
service rh-mongodb34-mongos start
```

システムの起動時に MongoDB シャードサーバーを起動するには、**root** で以下のコマンドを入力します。

```
chkconfig rh-mongodb34-mongos on
```

少なくとも 1 つの設定サーバーを起動し、**mongos.conf** ファイルで指定しない限り、MongoDB シャーディングサーバーは機能しないことに注意してください。

Red Hat Enterprise Linux 7 の MongoDB 3.4

Red Hat Enterprise Linux 7 を使用する場合は、以下のコマンドを使用できます。

MongoDB デーモンを起動するには、**root** で以下のコマンドを入力します。

```
systemctl start rh-mongodb34-mongod.service
```

システムの起動時に MongoDB デーモンを起動するには、**root** でこのコマンドを入力します。

```
systemctl enable rh-mongodb34-mongod.service
```

MongoDB シャードサーバーを起動するには、**root** で以下のコマンドを入力します。

```
systemctl start rh-mongodb34-mongos.service
```

システムの起動時に MongoDB シャードサーバーを起動するには、**root** でこのコマンドを入力します。

```
systemctl enable rh-mongodb34-mongos.service
```

少なくとも 1 つの設定サーバーを起動し、**mongos.conf** ファイルで指定しない限り、MongoDB シャーディングサーバーは機能しないことに注意してください。

4.6. GIT

Git は、分散アーキテクチャーを備えた分散リビジョン管理システムです。クライアントサーバーモデルの集中型バージョン管理システムとは対照的に、Git は Git リポジトリーの各作業コピーが、完全なリビジョン履歴を持つ正確なコピーであることを保証します。これにより、変更内容を公式リポジトリーにプッシュする許可を得なくても、プロジェクトに取り組み、貢献することができるだけでなく、ネットワーク接続がなくても作業ができるようになります。詳細は、『Red Hat Enterprise Linux 7 開発者ガイド』 の Git の章 を参照してください。

4.7. MAVEN

Red Hat Enterprise Linux 7 でのみ利用可能なrh-maven35 Software Collection は、ソフトウェアのプロジェクト管理と理解を深めるツールを提供します。**Maven** はプロジェクトオブジェクトモデル (POM) の概念に基づいて、プロジェクトのビルド、レポート、およびドキュメントを一元的な情報から管理できます。

rh-maven35 Collection をインストールするには、**root** で以下のコマンドを入力します。

```
yum install rh-maven35
```

このコレクションを有効にするには、シェルプロンプトで以下のコマンドを入力します。

```
scl enable rh-maven35 bash
```

リモートリポジトリーやミラーなど、Maven のグローバル設定は、**/opt/rh/rh-maven35/root/etc/maven/settings.xml**ファイルを編集することでカスタマイズできます。

Maven の使用に関する詳細は、Maven ドキュメント を参照してください。プラグインの使用法については、こちらのセクション で説明します。個々のプラグインに関するドキュメントをお探しの場合は、プラグインのインデックス を参照してください。

4.8. パッセンジャー

rh-passenger40 Software Collection は、高速、堅牢、軽量に設計された Web およびアプリケーションサーバーである **Phusion Residential** を提供します。

rh-passenger40 Collection は、特に ruby193、ruby200、rh-ruby22 Software Collection など、複数のバージョンの **Ruby** を、ror40 または rh-ror41 コレクションを使用した **Ruby on Rails** とともにサポートしています。いずれかの **Ruby** Software Collection で **Passenger** を使用する前に、対応するパッケージを rh-passenger40 Collection (rh-passenger-ruby193、rh-passenger-ruby200、または rh-passenger-ruby22 パッケージ) からインストールしてください。

rh-passenger40 Software Collection は、 httpd24 Software Collection の **Apache httpd** とともに使用することもできます。そのためには、rh-passenger40-mod_passenger パッケージをインストールしてください。**Apache httpd** 設定の例については、デフォルト設定ファイル **/opt/rh/httpd24/root/etc/httpd/conf.d/passenger.conf** を参照してください。これには、単一の **Apache httpd** インスタンスで複数の **Ruby** バージョンを使用する方法が示されています。

さらに、rh-passenger40 Software Collection は、 nginx16 Software Collection の **nginx 1.6** Web サーバーとともに使用できます。**nginx 1.6** を rh-passenger40 とともに使用するには、Web アプリケーションのディレクトリーで次のコマンドを使用して、**Passenger** をスタンドアロンモードで実行します。

```
scl enable nginx16 rh-passenger40 'passenger start'
```

または、アップストリームのPassenger documentationに記載されているように、nginx16 設定ファイルを編集してください。

4.9. データベースコネクター

データベースコネクターパッケージは、データベースサーバーへのローカルまたはリモート接続に必要なデータベースクライアント機能を提供します。表4.2「言語とデータベース間の相互運用性」 特定のデータベースサーバーのコネクターを含む言語ランタイムを含む Software Collections をリスト表示します。

表4.2 言語とデータベース間の相互運用性

	Database				
言語 (Software Collection)	MariaDB	MongoDB	MySQL	PostgreSQL	Redis
rh-nodejs4	✗	✗	✗	✗	✗
rh-nodejs6	✗	✗	✗	✗	✗
rh-nodejs8	✗	✗	✗	✗	✗
rh-perl520	✓	✗	✓	✓	✗
rh-perl524	✓	✗	✓	✓	✗
rh-perl526	✓	✗	✓	✓	✗
rh-php56	✓	✓	✓	✓	✗
rh-php70	✓	✗	✓	✓	✗
rh-php71	✓	✗	✓	✓	✗

	Database				
言語 (Software Collection)	MariaDB	MongoDB	MySQL	PostgreSQL	Redis
python27	✓	✓	✓	✓	✗
rh-python34	✗	✓	✗	✓	✗
rh-python35	✓	✓	✓	✓	✗
rh-python36	✓	✓	✓	✓	✗
rh-ror41	✓	✓	✓	✓	✗
rh-ror42	✓	✓	✓	✓	✗
rh-ror50	✓	✓	✓	✓	✗
rh-ruby25	✓	✓	✓	✓	✗
✓	サポート対象		✗	サポート対象外	

第5章 マイグレーション

この章では、Red Hat Software Collections 3.1 に含まれるコンポーネントのバージョンへの移行に関する情報を提供します。

5.1. MARIADB 10.2 への移行

Red Hat Enterprise Linux 6 には、デフォルトの **MySQL** 実装として **MySQL 5.1** が含まれています。Red Hat Enterprise Linux 7 では、デフォルトの **MySQL** 実装として **MariaDB 5.5** が含まれています。**MariaDB** は、**MySQL** に代わるコミュニティー開発のドロップインに置き換えられます。**MariaDB 10.1**は、Red Hat Software Collections 2.2 以降、ソフトウェアコレクションとして提供されています。Red Hat Software Collections 3.1 は、**MariaDB 10.2** とともに配布されています。

Red Hat Enterprise Linux 6 と Red Hat Enterprise Linux 7 の両方で利用可能なrh-mariadb102 Software Collection は、コアシステムのmysql またはmariadb パッケージと競合しないため、rh-mariadb102 Software Collection をmysql またはmariadb パッケージと一緒にインストールすることが可能です。特定のリソースが競合しないようにするため、両方のバージョンを同時に実行することもできますが、ポート番号と **my.cnf** ファイルのソケットを変更する必要があります。さらに、rh-mariadb101 Collection がインストールされ、実行中でも、rh-mariadb102 Software Collection をインストールすることもできます。

MariaDB 5.5 または **MariaDB 10.0** を使用している場合は、まず rh-mariadb101 Software Collection にアップグレードする必要があります。これについては、Red Hat Software Collections 2.4 リリースノート で説明されています。

MariaDB 10.2 の詳細は、バージョン 10.2 での変更点およびアップグレードに関するアップストリームドキュメントを参照してください。

注記

rh-mariadb102 Software Collection では、NFS によるマウントや **scl register** コマンドによる動的登録はサポートしていません。

5.1.1. rh-mariadb101 および rh-mariadb102 Software Collections 間の主な相違点

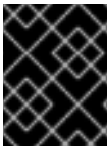
MariaDB 10.2 における主な変更は、Red Hat Software Collections 3.0 リリースノート を参照してください。

MariaDB 10.2 以降、**SQL_MODE** 変数の動作が変更になりました。詳細はアップストリームのドキュメント を参照してください。

複数のオプションがデフォルト値を変更しているか、非推奨または削除されました。詳細は、ナレッジベースの記事Migrating from MariaDB 10.1 to the MariaDB 10.2 Software Collection を参照してください。

rh-mariadb102 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-mariadb102-syspaths パッケージが含まれます。rh-mariadb102*-syspaths パッケージのインストール後に、rh-mariadb102* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。*-syspaths パッケージは、ベースの Red Hat Enterprise Linux システムと対応するパッケージと競合することに注意してください。

5.1.2. rh-mariadb101 から rh-mariadb102 Software Collection へのアップグレード



重要

アップグレードする前に、MariaDB データベースを含むすべてのデータのバックアップを作成します。

1. rh-mariadb101 データベースサーバーが実行している場合は停止します。

   サーバーを停止する前に、**innodb_fast_shutdown** オプションを **0** に設定し、**InnoDB** が完全なパージや挿入バッファーマージを含む低速なシャットダウンを実行します。アップストリームのドキュメント で、このオプションの詳細を参照してください。この操作は、通常のシャットダウンの場合よりも長い時間がかかる可能性があります。

   ```
   mysql -uroot -p -e "SET GLOBAL innodb_fast_shutdown = 0"
   ```

   rh-mariadb101 サーバーを停止します。

   ```
   service rh-mariadb101-mariadb stop
   ```

2. rh-mariadb102 Software Collection をインストールします。

   ```
   yum install rh-mariadb102-mariadb-server
   ```

   これらの Collections が競合しないため、rh-mariadb102 Software Collection のインストール中は rh-mariadb101 Software Collection をインストールすることができることに注意してください。

3. **/etc/opt/rh/rh-mariadb102/my.cnf** ファイルおよび **/etc/opt/rh/rh-mariadb102/my.cnf.d/** ディレクトリーに保存される rh-mariadb102 の設定を確認します。これを **/etc/opt/rh/rh-mariadb101/my.cnf** および **/etc/opt/rh/rh-mariadb101/my.cnf.d/** に保存されている rh-mariadb101 の設定と比較して、必要に応じて調整します。

4. rh-mariadb101 Software Collection のすべてのデータは、特に設定されていない限り、**/var/opt/rh/rh-mariadb101/lib/mysql/**ディレクトリーに保存されます。このディレクトリーのすべての内容を **/var/opt/rh/rh-mariadb102/lib/mysql/** にコピーします。コンテンツを移動することはできますが、アップグレードを続行する前にデータをバックアップすることを忘れないようにしてください。データが **mysql** ユーザーによって所有され、SELinux コンテキストが正しいことを確認します。

5. rh-mariadb102 データベースサーバーを起動します。

   ```
   service rh-mariadb102-mariadb start
   ```

6. データ移行を実行します。

   ```
   scl enable rh-mariadb102 mysql_upgrade
   ```

   root ユーザーに空ではないパスワードが定義されている場合 (パスワードを定義しておく必要あり) は、**-p** オプションを指定して **mysql_upgrade** ユーティリティーを呼び出してパスワードを指定する必要があります。

   ```
   scl enable rh-mariadb102 -- mysql_upgrade -p
   ```

5.2. MONGODB 3.6 への移行

Red Hat Software Collections 3.1 は、rh-mongodb36 Software Collection で提供される **MongoDB 3.6** とともにリリースされ、Red Hat Enterprise Linux 7 でのみ利用できます。

rh-mongodb36 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-mongodb36-syspaths パッケージが含まれます。rh-mongodb36*-syspaths パッケージのインストール後に、rh-mongodb36* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。syspaths の詳細は、Red Hat Software Collections パッケージガイド を参照してください。

5.2.1. MongoDB 3.4 と MongoDB 3.6 の主な相違点

一般的な変更点

rh-mongodb36 Software Collection では、以下のような重要な変更点が加えられています。

- NUMA (Non-Uniform Access Memory) ハードウェアでは、**numactl** コマンドを使用して起動するように **systemd** サービスを設定することができます。アップストリームの推奨事項 を参照してください。**numactl** コマンドで **MongoDB** を使用するには、numactl RPM パッケージをインストールし、**/etc/opt/rh/rh-mongodb36/sysconfig/mongod** および **/etc/opt/rh/rh-mongodb36/sysconfig/mongos** 設定ファイルを変更する必要があります。

互換性の変更点

MongoDB 3.6 には、**MongoDB** の以前のバージョンとの互換性に影響を与える可能性があるさまざまなマイナーな変更が含まれています。

- **MongoDB** バイナリーはデフォルトで **localhost** にバインドされるため、異なる IP アドレスでのリッスンを示的に有効にする必要があります。これは、**MongoDB** Software Collections で配布される **systemd** サービスのデフォルト動作であることに注意してください。
- MONGODB-CR 認証メカニズムが非推奨になりました。3.0 よりも前のバージョンの **MongoDB** で作成したユーザーの場合は、認証スキーマを SCRAM にアップグレードします。
- HTTP インターフェイスと REST API が削除されました。
- レプリカセットの Arbiter の優先度は **0** です。
- master-slave レプリケーションが非推奨になりました。

MongoDB 3.6 での詳細な互換性の変更点は、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。

後方互換性のない機能

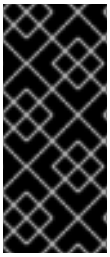
次の **MongoDB 3.6** 機能には後方互換性がないため、**featureCompatibilityVersion** コマンド を使用してバージョンを 3.6 に設定する必要があります。

- コレクションの UUID
- **$jsonSchema** ドキュメント検証
- ストリームの変更
- チャンク対応のセカンダリー

- バージョン 3.6 のクエリー機能を使用する定義、ドキュメントバリデーター、部分インデックスフィルターを表示する
- セッションおよび再試行可能な書き込み
- **authenticationRestrictions** のあるユーザーおよびロール

MongoDB 3.6 での後方互換性のない変更の詳細は、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。

5.2.2. rh-mongodb34 から rh-mongodb36 Software Collection へのアップグレード



重要

rh-mongodb34 から rh-mongodb36 Software Collection に移行する前に、デフォルトで **/var/opt/rh/rh-mongodb34/lib/mongodb/** ディレクトリーに保存される MongoDB データベースを含むすべてのデータをバックアップします。また、互換性の変更点 を参照して、アプリケーションおよびデプロイメントが MongoDB 3.6 と互換性があることを確認してください。

rh-mongodb36 Software Collection にアップグレードするには、以下の手順を実行します。

1. アップグレードできるようにするには、rh-mongodb34 インスタンスの **featureCompatibilityVersion** が **3.4** に設定されている必要があります。**featureCompatibilityVersion** を確認します。

```
~]$ scl enable rh-mongodb34 'mongo --host localhost --port 27017 admin' --eval
'db.adminCommand({getParameter: 1, featureCompatibilityVersion: 1})'
```

   mongod サーバーでアクセス制御が有効に設定されている場合は、**mongo** コマンドに **--username** および **--password** オプションを追加します。

2. rh-mongodb36 Software Collections から MongoDB サーバーおよびシェルをインストールします。

```
~]# yum install rh-mongodb36
```

3. MongoDB 3.4 サーバーを停止します。

```
~]# systemctl stop rh-mongodb34-mongod.service
```

4. データを新しい場所にコピーします。

```
~]# cp -a /var/opt/rh/rh-mongodb34/lib/mongodb/* /var/opt/rh/rh-
mongodb36/lib/mongodb/
```

5. **/etc/opt/rh/rh-mongodb36/mongod.conf** ファイルで **rh-mongodb36-mongod** デーモンを設定します。
6. MongoDB 3.6 サーバーを起動します。

```
~]# systemctl start rh-mongodb36-mongod.service
```

7. 後方互換性のない機能を有効にします。

```
~]$ scl enable rh-mongodb36 'mongo --host localhost --port 27017 admin' --eval
'db.adminCommand( { setFeatureCompatibilityVersion: "3.6" } )'
```

mongod サーバーでアクセス制御が有効に設定されている場合は、**mongo** コマンドに **--username** および **--password** オプションを追加します。

注記

アップグレード後、ダウングレードの可能性を最小限に抑えるために、焼き付き期間の間、下位互換性のない機能を有効にせずに、最初にデプロイメントを実行することが推奨されます。

アップグレードの詳細は、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。

レプリカセットのアップグレードに関する詳細は、アップストリームの MongoDB Manual を参照してください。

Sharded Cluster のアップグレードに関する詳細は、アップストリームの MongoDB マニュアル を参照してください。

5.3. MONGODB 3.4 への移行

Red Hat Enterprise Linux 6 および Red Hat Enterprise Linux 7 で利用可能な rh-mongodb34 Software Collection は **MongoDB 3.4** を提供します。

5.3.1. MongoDB 3.2 と MongoDB 3.4 との間の主な相違点

一般的な変更点
rh-mongodb34 Software Collection では、一般的な変更点が加えられています。ナレッジベースの記事Migrating from MongoDB 3.2 to MongoDB 3.4 に主な変更点が記載されています。詳細な変更については、upstream release notes を参照してください。

また、この Software Collection には rh-mongodb34-syspaths パッケージが含まれており、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージがインストールされます。rh-mongodb34*-syspaths パッケージのインストール後に、rh-mongodb34* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。syspaths の詳細は、Red Hat Software Collections パッケージガイド を参照してください。

互換性の変更点
MongoDB 3.4 には、**MongoDB** の以前のバージョンとの互換性に影響を与える可能性があるさまざまなマイナーな変更が含まれています。詳細は、ナレッジベースの記事Migrating from MongoDB 3.2 to MongoDB 3.4 およびupstream documentation を参照してください。

特に、次の **MongoDB 3.4** 機能には後方互換性がないため、**featureCompatibilityVersion** コマンドを使用してバージョンを **3.4** に設定する必要があります。

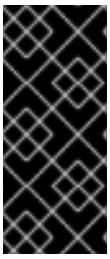
- 既存のコレクションまたはその他のビューから読み取り専用ビューを作成するためのサポート
- 照合、10 進データ、および大文字と小文字を区別しないインデックスのサポートを追加するインデックスバージョン **v: 2**

- 新しい **decimal** データ型による **decimal128** フォーマットのサポート

MongoDB 3.4 での後方互換性のない変更の詳細は、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。

5.3.2. rh-mongodb32 から rh-mongodb34 Software Collection へのアップグレード

MongoDB 3.4 にアップグレードして新機能を使用したら、バージョン 3.2.7 以前のバージョンにダウングレードできないことに注意してください。バージョン 3.2.8 以降にダウングレードできます。



重要

rh-mongodb32 から rh-mongodb34 Software Collection に移行する前に、デフォルトで **/var/opt/rh/rh-mongodb32/lib/mongodb/** ディレクトリーに保存される MongoDB データベースを含むすべてのデータをバックアップします。また、互換性の変更を参照して、アプリケーションおよびデプロイメントが MongoDB 3.4 と互換性があることを確認します。

rh-mongodb34 Software Collection にアップグレードするには、以下の手順を実行します。

1. rh-mongodb34 Software Collections から **MongoDB** サーバーおよびシェルをインストールします。

```
~]# yum install rh-mongodb34
```

2. MongoDB 3.2 サーバーを停止します。

```
~]# systemctl stop rh-mongodb32-mongod.service
```

Red Hat Enterprise Linux 6 システムで **service rh-mongodb32-mongodb stop** コマンドを使用します。

3. データを新しい場所にコピーします。

```
~]# cp -a /var/opt/rh/rh-mongodb32/lib/mongodb/* /var/opt/rh/rh-
mongodb34/lib/mongodb/
```

4. **/etc/opt/rh/rh-mongodb34/mongod.conf** ファイルで **rh-mongodb34-mongod** デーモンを設定します。

5. MongoDB 3.4 サーバーを起動します。

```
~]# systemctl start rh-mongodb34-mongod.service
```

Red Hat Enterprise Linux 6 では、代わりに **service rh-mongodb34-mongodb start** コマンドを使用します。

6. 後方互換性機能を有効にします。

```
~]$ scl enable rh-mongodb34 'mongo --host localhost --port 27017 admin' --eval
'db.adminCommand( { setFeatureCompatibilityVersion: "3.4" } )'
```

mongod サーバーでアクセス制御が有効に設定されている場合は、**mongo** コマンドに **--username** および **--password** オプションを追加します。

アップグレード後に、この機能を最初に有効にせずにデプロイメントを実行することが推奨されます。

アップグレードの詳細は、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。

レプリカセットのアップグレードに関する詳細は、アップストリームの MongoDB Manual を参照してください。

Sharded Cluster のアップグレードに関する詳細は、アップストリームの MongoDB マニュアル を参照してください。

5.4. MYSQL 5.7 への移行

Red Hat Enterprise Linux 6 には、デフォルトの **MySQL** 実装として **MySQL 5.1** が含まれています。Red Hat Enterprise Linux 7 では、デフォルトの **MySQL** 実装として **MariaDB 5.5** が含まれています。これらの基本バージョンに加えて、**MySQL 5.6** は、Red Hat Software Collections 2.0 以降、Red Hat Enterprise Linux 6 と Red Hat Enterprise Linux 7 の両方のソフトウェアコレクションとして利用可能になりました。

Red Hat Enterprise Linux 6 と Red Hat Enterprise Linux 7 の両方で利用可能なrh-mysql57 Software Collection は、コアシステムのmysql またはmariadb パッケージやrh-mysql56 Software Collection とは競合しないため、rh-mysql57 Software Collection を mysql、mariadb、rh-mysql56 パッケージと一緒にインストールすることができます。また、複数のバージョンを同時に実行することも可能ですが、特定リソースが競合しないようにするため、ポート番号と **my.cnf** ファイルのソケットを変更する必要があります。

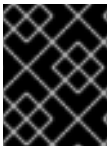
MySQL 5.7 には、**MySQL 5.6** からのみアップグレードできることに注意してください。以前のバージョンからアップグレードする必要がある場合は、まず **MySQL 5.6** にアップグレードしてください。**MySQL 5.6** にアップグレードする手順については、Red Hat Software Collections 2.2 リリースノート を参照してください。

5.4.1. MySQL 5.6 と MySQL 5.7 の注目すべき相違点

- mysql-bench サブパッケージは、rh-mysql57 Software Collection には含まれていません。
- **MySQL 5.7.7** 以降、デフォルトの SQL モードに **NO_AUTO_CREATE_USER** が含まれています。**GRANT** ステートメントではデフォルトでユーザーが作成されなくなったため、**CREATE USER** ステートメントを使用して MySQL アカウントを作成する必要があります。詳細は、upstream documentation を参照してください。

以前のバージョンと比較した **MySQL 5.7** の詳細な変更点については、アップストリームのドキュメント What Is New in MySQL 5.7 および Changes Affecting Upgrades to MySQL 5.7 を参照してください。

5.4.2. rh-mysql57 Software Collection へのアップグレード



重要

アップグレードする前に、MySQL データベースを含むすべてのデータのバックアップを作成します。

1. rh-mysql57 Software Collection をインストールします。

-

```
yum install rh-mysql57-mysql-server
```

2. **/etc/opt/rh/rh-mysql57/my.cnf** ファイルおよび **/etc/opt/rh/rh-mysql57/my.cnf.d/** ディレクトリーに保存される rh-mysql57 の設定を確認します。これを **/etc/opt/rh/rh-mysql56/my.cnf** と **/etc/opt/rh/rh-mysql56/my.cnf.d/** に保存されている rh-mysql56 の設定と比較して、必要に応じて調整します。

3. rh-mysql56 データベースサーバーがまだ実行中の場合は停止します。

```
service rh-mysql56-mysqld stop
```

4. rh-mysql56 Software Collection のすべてのデータは **/var/opt/rh/rh-mysql56/lib/mysql/** ディレクトリーに保存されます。このディレクトリーのすべての内容を **/var/opt/rh/rh-mysql57/lib/mysql/** にコピーします。コンテンツを移動することもできますが、アップグレードを続行する前にデータのバックアップを作成することを忘れないようにしてください。

5. rh-mysql57 データベースサーバーを起動します。

```
service rh-mysql57-mysqld start
```

6. データ移行を実行します。

```
scl enable rh-mysql57 mysql_upgrade
```

root ユーザーに空ではないパスワードが定義されている場合 (パスワードを定義しておく必要あり) は、**-p** オプションを指定して **mysql_upgrade** ユーティリティーを呼び出してパスワードを指定する必要があります。

```
scl enable rh-mysql57 -- mysql_upgrade -p
```

5.5. POSTGRESQL 10 への移行

Red Hat Software Collections 3.1 は **PostgreSQL 10** とともに配布され、Red Hat Enterprise Linux 7 でのみ利用できます。rh-postgresql10 Software Collection は、**PostgreSQL** または **PostgreSQL** Software Collection のベースの Red Hat Enterprise Linux システムバージョンと並行して、同じマシンに安全にインストールできます。複数のバージョンの **PostgreSQL** を同時にマシン上で実行することもできますが、別のポートまたは IP アドレスを使用し、SELinux ポリシーを調整する必要があります。以前のバージョンに移行する方法、または Red Hat Enterprise Linux 6 を使用する場合は、「PostgreSQL 9.6 への移行」を参照してください。

rh-postgresql10 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-postgresql10-syspaths パッケージが含まれます。rh-postgreqsl10*-syspaths パッケージのインストール後に、rh-postgreqsl10* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。*-syspaths パッケージは、ベースの Red Hat Enterprise Linux システムと対応するパッケージと競合することに注意してください。syspaths の詳細は、Red Hat Software Collections パッケージガイドを参照してください。

重要

PostgreSQL 10 に移行する前に、アップストリームの互換性情報 を参照してください。

以下の表は、postgresql パッケージが提供する **PostgreSQL** の Red Hat Enterprise Linux 7 システムバージョン、ならびに rh-postgresql96 および rh-postgresql10 Software Collection の異なるパスの概要を示しています。

表5.1 PostgreSQL パスの相違点

コンテンツ	postgresql	rh-postgresql96	rh-postgresql10
実行ファイル	/usr/bin/	/opt/rh/rh-postgresql96/root/usr/bin/	/opt/rh/rh-postgresql10/root/usr/bin/
ライブラリー	/usr/lib64/	/opt/rh/rh-postgresql96/root/usr/lib64/	/opt/rh/rh-postgresql10/root/usr/lib64/
ドキュメント	/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql/html/
PDF ドキュメント	/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql-docs/
Contrib ドキュメント	/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql-contrib/
ソース	未インストール	未インストール	未インストール
データ	/var/lib/pgsql/data/	/var/opt/rh/rh-postgresql96/lib/pgsql/data/	/var/opt/rh/rh-postgresql10/lib/pgsql/data/
バックアップエリア	/var/lib/pgsql/backups/	/var/opt/rh/rh-postgresql96/lib/pgsql/backups/	/var/opt/rh/rh-postgresql10/lib/pgsql/backups/
テンプレート	/usr/share/pgsql/	/opt/rh/rh-postgresql96/root/usr/share/pgsql/	/opt/rh/rh-postgresql10/root/usr/share/pgsql/
手順言語	/usr/lib64/pgsql/	/opt/rh/rh-postgresql96/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql10/root/usr/lib64/pgsql/
開発ヘッダー	/usr/include/pgsql/	/opt/rh/rh-postgresql96/root/usr/include/pgsql/	/opt/rh/rh-postgresql10/root/usr/include/pgsql/

コンテンツ	postgresql	rh-postgresql96	rh-postgresql10
他の共有データ	/usr/share/pgsql/	/opt/rh/rh-postgresql96/root/usr/share/pgsql/	/opt/rh/rh-postgresql10/root/usr/share/pgsql/
リグレッションテスト	/usr/lib64/pgsql/test/regress/ (-test パッケージ内)	/opt/rh/rh-postgresql96/root/usr/lib64/pgsql/test/regress/ (-test パッケージ内)	/opt/rh/rh-postgresql10/root/usr/lib64/pgsql/test/regress/ (-test パッケージ内)

5.5.1. Red Hat Enterprise Linux システムバージョンの PostgreSQL から PostgreSQL 10 Software Collection への移行

Red Hat Enterprise Linux 7 には **PostgreSQL 9.2** が同梱されています。**PostgreSQL** の Red Hat Enterprise Linux システムバージョンから rh-postgresql10 Software Collection にデータを移行するには、**pg_upgrade** ツールを使用して高速アップグレードを行うか (推奨)、データベースのデータを SQL コマンドでテキストファイルにダンプして新しいデータベースにインポートします。2 番目の方法は、通常かなり時間がかかり、手動による修正が必要になる場合があることに注意してください。このアップグレード方法の詳細は、PostgreSQL のドキュメント を参照してください。

重要

Red Hat Enterprise Linux システムバージョンの PostgreSQL から PostgreSQL 10 にデータを移行する前に、**デフォルトで /var/lib/pgsql/data/** ディレクトリーに格納される PostgreSQL データベースファイルを含む、すべてのデータをバックアップしてください。

手順5.1 pg_upgrade ツールを使用した高速アップグレード

PostgreSQL サーバーの高速アップグレードを実行するには、以下の手順を実行します。

1. 古い PostgreSQL サーバーを停止し、データが一貫性のない状態にあることを確認します。これを行うには、**root** で次のコマンドを実行します。

   ```
   systemctl stop postgresql.service
   ```

   サーバーが起動していないことを確認するには、以下を入力します。

   ```
   systemctl status postgresql.service
   ```

2. 古いディレクトリー **/var/lib/pgsql/data/** が存在することを確認します。

   ```
   file /var/lib/pgsql/data/
   ```

   データのバックアップを作成します。

3. 新しいデータディレクトリー **/var/opt/rh/rh-postgresql10/lib/pgsql/data/** が存在しないことを確認します。

   ```
   file /var/opt/rh/rh-postgresql10/lib/pgsql/data/
   ```

PostgreSQL 10 の新規インストールを実行している場合は、このディレクトリーはシステムに存在しないはずです。その場合は、**root** で以下のコマンドを実行してバックアップを作成します。

```
mv /var/opt/rh/rh-postgresql10/lib/pgsql/data{,-scl-backup}
```

4. **root** で以下のコマンドを実行して、新しいサーバーのデータベースデータをアップグレードします。

```
scl enable rh-postgresql10 -- postgresql-setup --upgrade
```

または、 **/opt/rh/rh-postgresql10/root/usr/bin/postgresql-setup --upgrade** コマンドを使用できます。

別のバージョンの PostgreSQL からのアップグレードには **--upgrade-from** オプションを使用できます。可能なアップグレードシナリオのリストは、**--upgrade-ids** オプションを使用して利用できます。

作成された **/var/lib/pgsql/upgrade_rh-postgresql10-postgresql.log** ログファイルを読み、アップグレード中に問題が発生したかどうかを確認することを推奨します。

5. **root** で新しいサーバーを起動します。

```
systemctl start rh-postgresql10-postgresql.service
```

また、以下のように **analyze_new_cluster.sh** スクリプトを実行することが推奨されます。

```
su - postgres -c 'scl enable rh-postgresql10 ~/analyze_new_cluster.sh'
```

6. 必要に応じて、システムの起動時に PostgreSQL 10 サーバーが自動的に起動するように設定できます。古いシステム PostgreSQL サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig postgresql off
```

PostgreSQL 10 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql10-postgresql on
```

7. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

手順5.2 ダンプおよびリストアアップグレードの実行

PostgreSQL サーバーのダンプおよび復元アップグレードを実行するには、以下の手順を実行します。

1. シェルプロンプトで、**root** で以下を入力し、古い PostgreSQL サーバーが実行中であることを確認します。

```
systemctl start postgresql.service
```

2. PostgreSQL データベースの全データをスクリプトファイルにダンプします。**root** で以下のコマンドを実行します。

```
su - postgres -c 'pg_dumpall > ~/pgdump_file.sql'
```

3. **root** で以下のコマンドを実行して、古いサーバーを停止します。

```
systemctl stop postgresql.service
```

4. 新規サーバーのデータディレクトリーを **root** として初期化します。

```
scl enable rh-postgresql10-postgresql -- postgresql-setup --initdb
```

5. **root** で新しいサーバーを起動します。

```
systemctl start rh-postgresql10-postgresql.service
```

6. 以前に作成した SQL ファイルからデータをインポートします。

```
su - postgres -c 'scl enable rh-postgresql10 "psql -f ~/pgdump_file.sql postgres"'
```

7. 必要に応じて、システムの起動時に PostgreSQL 10 サーバーが自動的に起動するように設定できます。古いシステム PostgreSQL サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig postgresql off
```

PostgreSQL 10 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql10-postgresql on
```

8. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

5.5.2. PostgreSQL 9.6 Software Collection から PostgreSQL 10 Software Collection への移行

rh-postgresql96 Software Collection から rh-postgresql10 Collection にデータを移行するには、**pg_upgrade** ツールを使用して高速アップグレードを行うか (推奨)、データベースのデータを SQL コマンドでテキストファイルにダンプして新しいデータベースにインポートします。2 番目の方法は、通常かなり時間がかかり、手動による修正が必要になる場合があることに注意してください。このアップグレード方法の詳細は、PostgreSQL のドキュメント を参照してください。

重要

PostgreSQL 9.6 から **PostgreSQL 10** にデータを移行する前に、デフォルトで **/var/opt/rh/rh-postgresql96/lib/pgsql/data/** ディレクトリーに格納される PostgreSQL データベースファイルを含む、すべてのデータをバックアップしてください。

手順5.3 pg_upgrade ツールを使用した高速アップグレード

PostgreSQL サーバーの高速アップグレードを実行するには、以下の手順を実行します。

1. 古い PostgreSQL サーバーを停止し、データが一貫性のない状態にあることを確認します。これを行うには、**root** で次のコマンドを実行します。

   ```
   systemctl stop rh-postgresql96-postgresql.service
   ```

   サーバーが起動していないことを確認するには、以下を入力します。

   ```
   systemctl status rh-postgresql96-postgresql.service
   ```

2. 古いディレクトリー **/var/opt/rh/postgresql96/lib/pgsql/data/** が存在することを確認します。

   ```
   file /var/opt/rh/rh-postgresql96/lib/pgsql/data/
   ```

   データのバックアップを作成します。

3. 新しいデータディレクトリー **/var/opt/rh/rh-postgresql10/lib/pgsql/data/** が存在しないことを確認します。

   ```
   file /var/opt/rh/rh-postgresql10/lib/pgsql/data/
   ```

   PostgreSQL 10 の新規インストールを実行している場合は、このディレクトリーはシステムに存在しないはずです。その場合は、**root** で以下のコマンドを実行してバックアップを作成します。

   ```
   mv /var/opt/rh/rh-postgresql10/lib/pgsql/data{,-scl-backup}
   ```

4. **root** で以下のコマンドを実行して、新しいサーバーのデータベースデータをアップグレードします。

   ```
   scl enable rh-postgresql10 -- postgresql-setup --upgrade --upgrade-from=rh-postgresql96-postgresql
   ```

   または、 **/opt/rh/rh-postgresql10/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql96-postgresql** コマンドを使用できます。

   別のバージョンの PostgreSQL からアップグレードする場合には、**--upgrade-from** オプションを使用できます。可能なアップグレードシナリオのリストは、**--upgrade-ids** オプションを使用して利用できます。

   作成された **/var/lib/pgsql/upgrade_rh-postgresql10-postgresql.log** ログファイルを読み、アップグレード中に問題が発生したかどうかを確認することを推奨します。

5. **root** で新しいサーバーを起動します。

   ```
   systemctl start rh-postgresql10-postgresql.service
   ```

   また、以下のように **analyze_new_cluster.sh** スクリプトを実行することが推奨されます。

   ```
   su - postgres -c 'scl enable rh-postgresql10 ~/analyze_new_cluster.sh'
   ```

6. 必要に応じて、システムの起動時に PostgreSQL 10 サーバーが自動的に起動するように設定できます。古い PostgreSQL 9.6サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig rh-postgresql96-postgreqsql off
```

PostgreSQL 10 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql10-postgresql on
```

7. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

手順5.4 ダンプおよびリストアアップグレードの実行

PostgreSQL サーバーのダンプおよび復元アップグレードを実行するには、以下の手順を実行します。

1. シェルプロンプトで、**root** で以下を入力し、古い PostgreSQL サーバーが実行中であることを確認します。

```
systemctl start rh-postgresql96-postgresql.service
```

2. PostgreSQL データベースの全データをスクリプトファイルにダンプします。**root** で以下のコマンドを実行します。

```
su - postgres -c 'scl enable rh-postgresql96 "pg_dumpall > ~/pgdump_file.sql"'
```

3. **root** で以下のコマンドを実行して、古いサーバーを停止します。

```
systemctl stop rh-postgresql96-postgresql.service
```

4. 新規サーバーのデータディレクトリーを **root** として初期化します。

```
scl enable rh-postgresql10-postgresql -- postgresql-setup --initdb
```

5. **root** で新しいサーバーを起動します。

```
systemctl start rh-postgresql10-postgresql.service
```

6. 以前に作成した SQL ファイルからデータをインポートします。

```
su - postgres -c 'scl enable rh-postgresql10 "psql -f ~/pgdump_file.sql postgres"'
```

7. 必要に応じて、システムの起動時に PostgreSQL 10 サーバーが自動的に起動するように設定できます。古い PostgreSQL 9.6サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig rh-postgresql96-postgresql off
```

PostgreSQL 10 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql10-postgresql on
```

8. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

5.6. POSTGRESQL 9.6 への移行

PostgreSQL 9.6 は、Red Hat Enterprise Linux 6 と Red Hat Enterprise Linux 7 の両方で利用できます。また、Red Hat Enterprise Linux 6 の **PostgreSQL 8.4**、Red Hat Enterprise Linux 7 の **PostgreSQL 9.2**、または以前のバージョンの Red Hat Software Collections でリリースされた任意のバージョンの **PostgreSQL** 9.2 と同時に、同じマシンに安全にインストールできます。複数のバージョンの **PostgreSQL** を同時にマシン上で実行することもできますが、別のポートまたは IP アドレスを使用し、SELinux ポリシーを調整する必要があります。

5.6.1. PostgreSQL 9.5 および PostgreSQL 9.6 間の主な違い

PostgreSQL 9.5 と **PostgreSQL 9.6** の間の最も注目すべき変更点は、アップストリームのリリースノート に記載されています。

rh-postgresql96 Software Collection には、バイナリー、スクリプト、man ページなどのシステム全体のラッパーを提供するパッケージをインストールする rh-postgresql96-syspaths パッケージが含まれます。rh-postgreqsl96*-syspaths パッケージのインストール後に、rh-postgreqsl96* パッケージによって提供されるバイナリーおよびスクリプトが正しく動作するかを **scl enable** コマンドを使用して確認する必要はありません。*-syspaths パッケージは、ベースの Red Hat Enterprise Linux システムと対応するパッケージと競合することに注意してください。syspaths の詳細は、Red Hat Software Collections パッケージガイドを参照してください。

以下の表は、**PostgreSQL** (postgresql) の Red Hat Enterprise Linux システムバージョン、ならびに postgresql92、rh-postgresql95、および rh-postgresql96 Software Collections のさまざまなパスの概要を示しています。Red Hat Enterprise Linux 6 で配布される **PostgreSQL 8.4** のパスと、Red Hat Enterprise Linux 7 に同梱された **PostgreSQL 9.2** のシステムバージョンは同じです。rh-postgresql94 Software Collection のパスは rh-postgresql95 と似ています。

表5.2 PostgreSQL パスの相違点

コンテンツ	postgresql	postgresql92	rh-postgresql95	rh-postgresql96
実行ファイル	/usr/bin/	/opt/rh/postgresql92/root/usr/bin/	/opt/rh/rh-postgresql95/root/usr/bin/	/opt/rh/rh-postgresql96/root/usr/bin/
ライブラリー	/usr/lib64/	/opt/rh/postgresql92/root/usr/lib64/	/opt/rh/rh-postgresql95/root/usr/lib64/	/opt/rh/rh-postgresql96/root/usr/lib64/
ドキュメント	/usr/share/doc/postgresql/html/	/opt/rh/postgresql92/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql95/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql/html/

コンテンツ	postgresql	postgresql92	rh-postgresql95	rh-postgresql96
PDF ドキュメント	/usr/share/doc/postgresql-docs/	/opt/rh/postgresql92/root/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql95/root/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql-docs/
Contrib ドキュメント	/usr/share/doc/postgresql-contrib/	/opt/rh/postgresql92/root/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql95/root/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql96/root/usr/share/doc/postgresql-contrib/
ソース	未インストール	未インストール	未インストール	未インストール
データ	/var/lib/pgsql/data/	/opt/rh/postgresql92/root/var/lib/pgsql/data/	/var/opt/rh/rh-postgresql95/lib/pgsql/data/	/var/opt/rh/rh-postgresql96/lib/pgsql/data/
バックアップエリア	/var/lib/pgsql/backups/	/opt/rh/postgresql92/root/var/lib/pgsql/backups/	/var/opt/rh/rh-postgresql95/lib/pgsql/backups/	/var/opt/rh/rh-postgresql96/lib/pgsql/backups/
テンプレート	/usr/share/pgsql/	/opt/rh/postgresql92/root/usr/share/pgsql/	/opt/rh/rh-postgresql95/root/usr/share/pgsql/	/opt/rh/rh-postgresql96/root/usr/share/pgsql/
手順言語	/usr/lib64/pgsql/	/opt/rh/postgresql92/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql95/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql96/root/usr/lib64/pgsql/
開発ヘッダー	/usr/include/pgsql/	/opt/rh/postgresql92/root/usr/include/pgsql/	/opt/rh/rh-postgresql95/root/usr/include/pgsql/	/opt/rh/rh-postgresql96/root/usr/include/pgsql/
他の共有データ	/usr/share/pgsql/	/opt/rh/postgresql92/root/usr/share/pgsql/	/opt/rh/rh-postgresql95/root/usr/share/pgsql/	/opt/rh/rh-postgresql96/root/usr/share/pgsql/
リグレッションテスト	/usr/lib64/pgsql/test/regress/ (-test パッケージ内)	/opt/rh/postgresql92/root/usr/lib64/pgsql/test/regress/ (-test パッケージ内)	/opt/rh/rh-postgresql95/root/usr/lib64/pgsql/test/regress/ (-test パッケージ内)	/opt/rh/rh-postgresql96/root/usr/lib64/pgsql/test/regress/ (-test パッケージ内)

PostgreSQL 8.4 から **PostgreSQL 9.2** への変更点は、 Red Hat Software Collections 1.2 リリースノートを参照してください。**PostgreSQL 9.2** と **PostgreSQL 9.4** との間の主な変更点は、Red Hat Software Collections 2.0 リリースノートを参照してください。**PostgreSQL 9.4** と **PostgreSQL 9.5** の相違点は、Red Hat Software Collections 2.2 リリースノート を参照してください。

5.6.2. Red Hat Enterprise Linux システムバージョンの PostgreSQL から PostgreSQL 9.6 Software Collection への移行

Red Hat Enterprise Linux 6 には **PostgreSQL 8.4** が含まれ、Red Hat Enterprise Linux 7 には **PostgreSQL 9.2** が同梱されています。**PostgreSQL** の Red Hat Enterprise Linux システムバージョンから rh-postgresql96 Software Collection にデータを移行するには、**pg_upgrade** ツールを使用して高速アップグレードを行うか (推奨)、データベースのデータを SQL コマンドでテキストファイルにダンプして新しいデータベースにインポートします。2 番目の方法は、通常かなり時間がかかり、手動による修正が必要になる場合があることに注意してください。このアップグレード方法の詳細は、PostgreSQL のドキュメント を参照してください。以下の手順は、Red Hat Enterprise Linux 6 および Red Hat Enterprise Linux 7 の **PostgreSQL** の両方に適用できます。

重要

Red Hat Enterprise Linux システムバージョンの PostgreSQL から PostgreSQL 9.6 にデータを移行する前に、**デフォルトで /var/lib/pgsql/data/** ディレクトリーに格納される PostgreSQL データベースファイルを含む、すべてのデータをバックアップしてください。

手順5.5 pg_upgrade ツールを使用した高速アップグレード

PostgreSQL サーバーの高速アップグレードを実行するには、以下の手順を実行します。

1. 古い PostgreSQL サーバーを停止し、データが一貫性のない状態にあることを確認します。これを行うには、**root** で次のコマンドを実行します。

   ```
   service postgresql stop
   ```

   サーバーが起動していないことを確認するには、以下を入力します。

   ```
   service postgresql status
   ```

2. 古いディレクトリー **/var/lib/pgsql/data/** が存在することを確認します。

   ```
   file /var/lib/pgsql/data/
   ```

   データのバックアップを作成します。

3. 新しいデータディレクトリー **/var/opt/rh/rh-postgresql96/lib/pgsql/data/** が存在しないことを確認します。

   ```
   file /var/opt/rh/rh-postgresql96/lib/pgsql/data/
   ```

   PostgreSQL 9.6 の新規インストールを実行している場合は、このディレクトリーがシステムに存在しないはずです。その場合は、**root** で以下のコマンドを実行してバックアップを作成します。

   ```
   mv /var/opt/rh/rh-postgresql96/lib/pgsql/data{,-scl-backup}
   ```

4. **root** で以下のコマンドを実行して、新しいサーバーのデータベースデータをアップグレードします。

   ```
   scl enable rh-postgresql96 -- postgresql-setup --upgrade
   ```

または、 **/opt/rh/rh-postgresql96/root/usr/bin/postgresql-setup --upgrade** コマンドを使用できます。

別のバージョンの **PostgreSQL** からのアップグレードには **--upgrade-from** オプションを使用できます。可能なアップグレードシナリオのリストは、**--upgrade-ids** オプションを使用して利用できます。

作成された **/var/lib/pgsql/upgrade_rh-postgresql96-postgresql.log** ログファイルを読み、アップグレード中に問題が発生したかどうかを確認することを推奨します。

5. **root** で新しいサーバーを起動します。

```
service rh-postgresql96-postgresql start
```

また、以下のように **analyze_new_cluster.sh** スクリプトを実行することが推奨されます。

```
su - postgres -c 'scl enable rh-postgresql96 ~/analyze_new_cluster.sh'
```

6. 必要に応じて、システムの起動時に PostgreSQL 9.6 サーバーが自動的に起動するように設定できます。古いシステム PostgreSQL サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig postgresql off
```

PostgreSQL 9.6 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql96-postgresql on
```

7. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

手順5.6 ダンプおよびリストアアップグレードの実行

PostgreSQL サーバーのダンプおよび復元アップグレードを実行するには、以下の手順を実行します。

1. シェルプロンプトで、**root** で以下を入力し、古い PostgreSQL サーバーが実行中であることを確認します。

```
service postgresql start
```

2. PostgreSQL データベースの全データをスクリプトファイルにダンプします。**root** で以下のコマンドを実行します。

```
su - postgres -c 'pg_dumpall > ~/pgdump_file.sql'
```

3. **root** で以下のコマンドを実行して、古いサーバーを停止します。

```
service postgresql stop
```

4. 新規サーバーのデータディレクトリーを **root** として初期化します。

```
scl enable rh-postgresql96-postgresql -- postgresql-setup --initdb
```

5. **root** で新しいサーバーを起動します。

```
service rh-postgresql96-postgresql start
```

6. 以前に作成した SQL ファイルからデータをインポートします。

```
su - postgres -c 'scl enable rh-postgresql96 "psql -f ~/pgdump_file.sql postgres"'
```

7. 必要に応じて、システムの起動時に PostgreSQL 9.6 サーバーが自動的に起動するように設定できます。古いシステム PostgreSQL サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig postgresql off
```

PostgreSQL 9.6 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql96-postgresql on
```

8. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

5.6.3. PostgreSQL 9.5 Software Collection から PostgreSQL 9.6 Software Collection への移行

rh-postgresql95 Software Collection から rh-postgresql96 コレクションにデータを移行するには、**pg_upgrade** ツールを使用して高速アップグレードを行うか (推奨)、データベースのデータを SQL コマンドでテキストファイルにダンプして新しいデータベースにインポートします。2 番目の方法は、通常かなり時間がかかり、手動による修正が必要になる場合があることに注意してください。このアップグレード方法の詳細は、PostgreSQL のドキュメント を参照してください。

重要

PostgreSQL 9.5 から **PostgreSQL 9.6** にデータを移行する前に、デフォルトで **/var/opt/rh/rh-postgresql95/lib/pgsql/data/** ディレクトリーに格納される PostgreSQL データベースファイルを含む、すべてのデータをバックアップしてください。

手順5.7 pg_upgrade ツールを使用した高速アップグレード

PostgreSQL サーバーの高速アップグレードを実行するには、以下の手順を実行します。

1. 古い PostgreSQL サーバーを停止し、データが一貫性のない状態にあることを確認します。これを行うには、**root** で次のコマンドを実行します。

```
service rh-postgresql95-postgresql stop
```

サーバーが起動していないことを確認するには、以下を入力します。

```
service rh-postgresql95-postgresql status
```

2. 古いディレクトリー **/var/opt/rh/postgresql95/lib/pgsql/data/** が存在することを確認します。

```
file /var/opt/rh/rh-postgresql95/lib/pgsql/data/
```

データのバックアップを作成します。

3. 新しいデータディレクトリー **/var/opt/rh/rh-postgresql96/lib/pgsql/data/** が存在しないことを確認します。

```
file /var/opt/rh/rh-postgresql96/lib/pgsql/data/
```

PostgreSQL 9.6 の新規インストールを実行している場合は、このディレクトリーがシステムに存在しないはずです。その場合は、**root** で以下のコマンドを実行してバックアップを作成します。

```
mv /var/opt/rh/rh-postgresql96/lib/pgsql/data{,-scl-backup}
```

4. **root** で以下のコマンドを実行して、新しいサーバーのデータベースデータをアップグレードします。

```
scl enable rh-postgresql96 -- postgresql-setup --upgrade --upgrade-from=rh-postgresql95-postgresql
```

または、 **/opt/rh/rh-postgresql96/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql95-postgresql** コマンドを使用できます。

別のバージョンの PostgreSQL からアップグレードする場合には、**--upgrade-from** オプションを使用できます。可能なアップグレードシナリオのリストは、**--upgrade-ids** オプションを使用して利用できます。

作成された **/var/lib/pgsql/upgrade_rh-postgresql96-postgresql.log** ログファイルを読み、アップグレード中に問題が発生したかどうかを確認することを推奨します。

5. **root** で新しいサーバーを起動します。

```
service rh-postgresql96-postgresql start
```

また、以下のように **analyze_new_cluster.sh** スクリプトを実行することが推奨されます。

```
su - postgres -c 'scl enable rh-postgresql96 ~/analyze_new_cluster.sh'
```

6. 必要に応じて、システムの起動時に PostgreSQL 9.6 サーバーが自動的に起動するように設定できます。古い PostgreSQL 9.5 サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig rh-postgresql95-postgreqsql off
```

PostgreSQL 9.6 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql96-postgresql on
```

7. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

手順5.8 ダンプおよびリストアアップグレードの実行

PostgreSQL サーバーのダンプおよび復元アップグレードを実行するには、以下の手順を実行します。

1. シェルプロンプトで、**root** で以下を入力し、古い PostgreSQL サーバーが実行中であることを確認します。

```
service rh-postgresql95-postgresql start
```

2. PostgreSQL データベースの全データをスクリプトファイルにダンプします。**root** で以下のコマンドを実行します。

```
su - postgres -c 'scl enable rh-postgresql95 "pg_dumpall > ~/pgdump_file.sql"'
```

3. **root** で以下のコマンドを実行して、古いサーバーを停止します。

```
service rh-postgresql95-postgresql stop
```

4. 新規サーバーのデータディレクトリーを **root** として初期化します。

```
scl enable rh-postgresql96-postgresql -- postgresql-setup --initdb
```

5. **root** で新しいサーバーを起動します。

```
service rh-postgresql96-postgresql start
```

6. 以前に作成した SQL ファイルからデータをインポートします。

```
su - postgres -c 'scl enable rh-postgresql96 "psql -f ~/pgdump_file.sql postgres"'
```

7. 必要に応じて、システムの起動時に PostgreSQL 9.6 サーバーが自動的に起動するように設定できます。古い PostgreSQL 9.5 サーバーを無効にするには、**root** で以下のコマンドを入力します。

```
chkconfig rh-postgresql95-postgresql off
```

PostgreSQL 9.6 サーバーを有効にするには、**root** で以下を入力します。

```
chkconfig rh-postgresql96-postgresql on
```

8. 設定がデフォルトと異なる場合は、設定ファイル (特に **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** 設定ファイル) を必ず更新してください。それ以外の場合は、**postgres** ユーザーのみがデータベースにアクセスできます。

postgresql92 Software Collection から移行する必要がある場合は、Red Hat Software Collections 2.0 Release Notes を参照してください。この手順は同じですが、新しい Collection のバージョンを調整する必要があります。これは、Red Hat Software Collections 2.2 Release Notes で説明したように、rh-postgresql94 Software Collection からの移行を適用します。

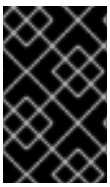
5.7. NGINX 1.12 への移行について

rh-nginx112 Software Collection は、Red Hat Enterprise Linux 7.4 以降のバージョンでのみ利用可能です。

rh-nginx112 Software Collection のルートディレクトリーは、**/opt/rh/rh-nginx112/root/** にあります。エラーログは、デフォルトでは **/var/opt/rh/rh-nginx112/log/nginx** に保存されます。

設定ファイルは、**/etc/opt/rh/rh-nginx112/nginx/** ディレクトリーに格納されます。**nginx 1.12** の設定ファイルは、以前の **nginx** Software Collections と同じ構文で、ほぼ同じ形式です。

/etc/opt/rh/rh-nginx112/nginx/default.d/ ディレクトリー内の設定ファイル (**.conf** 拡張子付き) は、ポート **80** のデフォルトのサーバーブロック設定に含まれています。



重要

nginx 1.10 から **nginx 1.12** にアップグレードする前に、**/opt/rh/nginx110/root/** ツリーにある Web ページや、**/etc/opt/rh/nginx110/nginx/** ツリーにある設定ファイルなど、すべてのデータをバックアップしてください。

/opt/rh/nginx110/root/ ツリーで設定ファイルの変更や Web アプリケーションのセットアップなどの特定の変更を行った場合は、その変更を新しい **/opt/rh/rh-nginx112/root/** ディレクトリーと **/etc/opt/rh/rh-nginx112/nginx/** ディレクトリーに反映します。

この手順を使用して、**nginx 1.4**、**nginx 1.6**、**nginx 1.8**、または **nginx 1.10** から **nginx 1.12** に直接アップグレードできます。この場合は、適切なパスを使用してください。

nginx の公式ドキュメントは、http://nginx.org/en/docs/ を参照してください。

第6章 関連情報

この章では、Red Hat Software Collections 3.1 および Red Hat Enterprise Linux に関するその他の関連情報源への参照を提供します。

6.1. RED HAT 製品ドキュメント

以下のドキュメントは直接的または間接的にこのガイドに関連しています。

- Red Hat Software Collections 3.1 パッケージガイド : Red Hat Software Collections の 『 パッケージガイド』 は、Software Collections の概念や **scl** ユーティリティーを説明し、カスタム Software Collection の作成方法や既存 Software Collection の拡張方法を詳細に説明しています。

- Red Hat Developer Toolset 7.1 リリースノート - Red Hat Developer Toolset の 『 リリースノート』 には、既知の問題、考えられる問題、変更点、およびこの Software Collection に関するその他の重要な情報が記載されています。

- Red Hat Developer Toolset 7.1 ユーザーガイド - Red Hat Developer Toolset の 『 ユーザーガイド』 には、この Software Collection のインストールおよび使用に関する詳細情報が記載されています。

- Using Red Hat Software Collections Container Images : 本書は、Red Hat Software Collections に基づくコンテナーイメージの使用方法に関する情報を提供します。利用可能なコンテナーイメージには、アプリケーション、デーモン、データベース、および Red Hat Developer Toolset コンテナーイメージが含まれます。イメージは、Red Hat Enterprise Linux 7 Server および Red Hat Enterprise Linux Atomic Host で実行できます。

- コンテナーのスタートガイド では、Red Hat Enterprise Linux 7 および Red Hat Enterprise Linux Atomic Host での Docker 形式のコンテナーイメージのビルドおよび使用に関する概要を包括的に説明します。

- Using and Configuring Red Hat Subscription Manager : 『Using and Configuring Red Hat Subscription Manager』 ガイドでは、Red Hat Enterprise Linux システムの登録、サブスクリプションの管理、登録システムの通知の表示方法の詳細情報を提供します。

- Red Hat Enterprise Linux 6 Deployment Guide : Red Hat Enterprise Linux 6 の 『 Deployment Guide』では、Red Hat Enterprise Linux 6 のデプロイメント、設定、および管理に関する関連情報を提供します。

- Red Hat Enterprise Linux 7 System Administrator's Guide : The 『System Administrator's Guide』 for Red Hat Enterprise Linux 7 は、このシステムのデプロイメント、設定、および管理に関する情報を提供します。

6.2. RED HAT 開発者

- Red Hat Developer Program: 『Red Hat Developers』 コミュニティーポータル

- Overview of Red Hat Software Collections on Red Hat Developers 『Red Hat Developers』 ポータルでは、さまざまな開発技術を使用してコードを開発するためのチュートリアルがいくつか紹介されています。これには、Node.js、Perl、PHP、Python、Ruby Software Collections が含まれます。

- Red Hat Enterprise Linux Developer Program 『Red Hat Enterprise Linux Developer Program 』は、業界をリードする開発者ツール、命令型リソース、デキスパートのエコシステムを提供

し、プログラマーが Linux アプリケーションの構築における生産性を最大化するのに役立ちます。

- Red Hat Developer Blog - 『Red Hat Developer Blog』 には、最新の情報、ベストプラクティス、意見、製品およびプログラムアナウンス、ならびに Red Hat の技術に基づくアプリケーションを設計および開発するユーザー向けのサンプルコードやその他のリソースへのポインターが含まれます。

付録A 更新履歴

改訂 3.1-4 **Wed Dec 20 2023** **Lenka Špačková**
壊れたリンクとアンカーを修正しました。

改訂 3.1-3 **Fri Nov 12 2021** **Lenka Špačková**
「データベースコネクター」 を更新しました。

改訂 3.1-2 **Tue Dec 18 2018** **Lenka Špačková**
PostgreSQL の移行の章で、コマンドの構文を修正しました。

改訂 3.1-1 **Thu May 03 2018** **Lenka Špačková**
Red Hat Software Collections 3.1 リリースノートの公開。

改訂 3.1-0 **Wed Mar 04 2018** **Lenka Špačková**
Red Hat Software Collections 3.1 Beta リリースノートの公開。