Red Hat Software Collections 3

3.4 Release Notes

Release Notes for Red Hat Software Collections 3.4

Last Updated: 2020-05-26

Red Hat Software Collections 3 3.4 Release Notes

Release Notes for Red Hat Software Collections 3.4

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

Legal Notice

Abstract

The Red Hat Software Collections 3.4 Release Notes document the major features and contain important information about known problems in Red Hat Software Collections 3.4. The Red Hat Developer Toolset collection is documented in the Red Hat Developer Toolset Release Notes and the Red Hat Developer Toolset User Guide .

Table of Contents

CHAPTER 1. RED HAT SOFTWARE COLLECTIONS 3.4

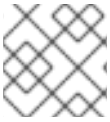
This chapter serves as an overview of the Red Hat Software Collections 3.4 content set. It provides a list of components and their descriptions, sums up changes in this version, documents relevant compatibility information, and lists known issues.

1.1. ABOUT RED HAT SOFTWARE COLLECTIONS

For certain applications, more recent versions of some software components are often needed in order to use their latest new features. **Red Hat Software Collections** is a Red Hat offering that provides a set of dynamic programming languages, database servers, and various related packages that are either more recent than their equivalent versions included in the base Red Hat Enterprise Linux system, or are available for this system for the first time.

Red Hat Software Collections 3.4 is available for Red Hat Enterprise Linux 7; selected previously released components also for Red Hat Enterprise Linux 6. For a complete list of components that are distributed as part of Red Hat Software Collections and a brief summary of their features, see Section 1.2, “Main Features”.

Red Hat Software Collections does not replace the default system tools provided with Red Hat Enterprise Linux 6 or Red Hat Enterprise Linux 7. Instead, a parallel set of tools is installed in the **/opt/** directory and can be optionally enabled per application by the user using the supplied **scl** utility. The default versions of Perl or PostgreSQL, for example, remain those provided by the base Red Hat Enterprise Linux system.



NOTE

In Red Hat Enterprise Linux 8, similar components are provided as Application Streams.

All Red Hat Software Collections components are fully supported under Red Hat Enterprise Linux Subscription Level Agreements, are functionally complete, and are intended for production use. Important bug fix and security errata are issued to Red Hat Software Collections subscribers in a similar manner to Red Hat Enterprise Linux for at least two years from the release of each major version. In each major release stream, each version of a selected component remains backward compatible. For detailed information about length of support for individual components, refer to the Red Hat Software Collections Product Life Cycle document.

1.1.1. Red Hat Developer Toolset

Red Hat Developer Toolset is a part of Red Hat Software Collections, included as a separate Software Collection. For more information about Red Hat Developer Toolset, refer to the Red Hat Developer Toolset Release Notes and the Red Hat Developer Toolset User Guide .

1.2. MAIN FEATURES

Table 1.1, “Red Hat Software Collections 3.4 Components” lists components that are supported at the time of the Red Hat Software Collections 3.4 release.

Table 1.1. Red Hat Software Collections 3.4 Components

Component	Software Collection	Description
Red Hat Developer Toolset 9.0	devtoolset-9	Red Hat Developer Toolset is designed for developers working on the Red Hat Enterprise Linux platform. It provides current versions of the **GNU Compiler Collection**, **GNU Debugger**, and other development, debugging, and performance monitoring tools. For a complete list of components, see the Red Hat Developer Toolset Components table in the *Red Hat Developer Toolset User Guide*.
Perl 5.26.3[a]	rh-perl526	A release of Perl, a high-level programming language that is commonly used for system administration utilities and web programming. The rh-perl526 Software Collection provides additional utilities, scripts, and *database connectors for MySQL and PostgreSQL*. It includes the **DateTime** Perl module and the **mod_perl** Apache httpd module, which is supported only with the httpd24 Software Collection. Additionally, it provides the **cpanm** utility for easy installation of CPAN modules. The rh-perl526 packaging is aligned with upstream; the perl526-perl package installs also core modules, while the interpreter is provided by the perl-interpreter package.
PHP 7.2.24 [a]	rh-php72	A release of PHP 7.2 with PEAR 1.10.5, APCu 5.1.12, and enhanced language features.
PHP 7.3.11 [a]	rh-php73	A release of PHP 7.3 with PEAR 1.10.9, APCu 5.1.17, and the Xdebug extension.
Python 2.7.16	python27	A release of Python 2.7 with a number of additional utilities. This Python version provides various features and enhancements, including an ordered dictionary type, faster I/O operations, and improved forward compatibility with Python 3. The python27 Software Collections contains the *Python 2.7.13 interpreter*, a set of extension libraries useful for programming web applications and **mod_wsgi** (only supported with the httpd24 Software Collection), MySQL and PostgreSQL database connectors, and **numpy** and **scipy**.
Python 3.6.9	rh-python36	The rh-python36 Software Collection contains Python 3.6.9, which introduces a number of new features, such as *f-strings, syntax for variable annotations, and asynchronous generators and comprehensions*. In addition, a set of extension libraries useful for programming web applications is included, with **mod_wsgi** (supported only together with the httpd24 Software Collection), PostgreSQL database connector, and **numpy** and **scipy**.

Component	Software Collection	Description
Ruby 2.4.6	rh-ruby24	A release of Ruby 2.4. This version provides multiple performance improvements and enhancements, for example *improved hash table, new debugging features, support for Unicode case mappings, and support for OpenSSL 1.1.0*. Ruby 2.4.0 maintains source-level backward compatibility with Ruby 2.3, Ruby 2.2, Ruby 2.0.0, and Ruby 1.9.3.
Ruby 2.5.5 [a]	rh-ruby25	A release of Ruby 2.5. This version provides multiple performance improvements and new features, for example, *simplified usage of blocks with the* ***rescue***, ***else***, *and* ***ensure*** *keywords, a new* ***yield_self*** *method, support for branch coverage and method coverage measurement, new* ***Hash#slice*** *and* ***Hash#transform_keys*** *methods*. Ruby 2.5.0 maintains source-level backward compatibility with Ruby 2.4.
Ruby 2.6.2 [a]	rh-ruby26	A release of Ruby 2.6. This version provides multiple performance improvements and new features, such as *endless ranges, the* ***Binding#source_location*** *method, and the* ***$SAFE*** *process global state*. Ruby 2.6.0 maintains source-level backward compatibility with Ruby 2.5.
Ruby on Rails 5.0.1	rh-ror50	A release of Ruby on Rails 5.0, the latest version of the web application framework written in the Ruby language. Notable new features include *Action Cable, API mode, exclusive use of rails CLI over Rake, and ActionRecord attributes*. This Software Collection is supported together with the rh-ruby24 Collection.
Scala 2.10.6 [a]	rh-scala210	A release of Scala, a general purpose programming language for the Java platform, which integrates features of object-oriented and functional languages.
MariaDB 10.2.22	rh-mariadb102	A release of MariaDB, an alternative to MySQL for users of Red Hat Enterprise Linux. For all practical purposes, MySQL is binary compatible with MariaDB and can be replaced with it without any data conversions. This version adds *MariaDB Backup, Flashback, support for Recursive Common Table Expressions, window functions, and JSON functions*.
MariaDB 10.3.13 [a]	rh-mariadb103	A release of MariaDB, an alternative to MySQL for users of Red Hat Enterprise Linux. For all practical purposes, MySQL is binary compatible with MariaDB and can be replaced with it without any data conversions. This version introduces *system-versioned tables, invisible columns, a new instant* ***ADD COLUMN*** *operation for* ***InnoDB***, *and a JDBC connector for MariaDB and MySQL*.

Component	Software Collection	Description
MongoDB 3.4.9	rh-mongodb34	A release of MongoDB, a cross-platform document-oriented database system classified as a NoSQL database. This release introduces support for new architectures, *adds message compression and support for the decimal128 type, enhances collation features* and more.
MongoDB 3.6.3 [a]	rh-mongodb36	A release of MongoDB, a cross-platform document-oriented database system classified as a NoSQL database. This release introduces *change streams, retryable writes, and JSON Schema*, as well as other features.
MySQL 8.0.17 [a]	rh-mysql80	A release of the MySQL server, which introduces a number of new *security and account management features* and enhancements.
PostgreSQL 9.6.10	rh-postgresql96	A release of PostgreSQL, which introduces parallel execution of sequential scans, joins, and aggregates, and provides enhancements to synchronous replication, full-text search, deration driver, postgres_fdw, as well as performance improvements.
PostgreSQL 10.6 [a]	rh-postgresql10	A release of PostgreSQL, which includes a significant performance improvement and a number of new features, such as *logical replication using the* ***publish*** *and* ***subscribe*** *keywords, or stronger password authentication based on the* ***SCRAM-SHA-256*** *mechanism*.
PostgreSQL 12.1 [a]	rh-postgresql12	A release of PostgreSQL, which provides *the* ***pgaudit*** *extension, various enhancements to partitioning and parallelism, support for the SQL/JSON path language,* and performance improvements.
Node.js 10.16.3 [a]	rh-nodejs10	A release of Node.js, which provides multiple API enhancements and new features, including *V8 engine version 6.6, full N-API support*, and stability improvements.
Node.js 12.10.0 [a]	rh-nodejs12	A release of Node.js, with *V8 engine version 7.6, support for ES6 modules,* and improved support for native modules.
nginx 1.10.2	rh-nginx110	A release of nginx, a web and proxy server with a focus on high concurrency, performance, and low memory usage. This version introduces a number of new features, including *dynamic module support, HTTP/2 support, Perl integration, and numerous performance improvements*.

Component	Software Collection	Description
nginx 1.14.1 [a]	rh-nginx114	A release of nginx, a web and proxy server with a focus on high concurrency, performance, and low memory usage. This version provides a number of features, such as *mirror module, HTTP/2 server push, gRPC proxy module, and numerous performance improvements*.
nginx 1.16.1 [a]	rh-nginx116	A release of nginx, a web and proxy server with a focus on high concurrency, performance, and low memory usage. This version introduces numerous *updates related to SSL, several new directives and parameters,* and various enhancements.
Apache httpd 2.4.34	httpd24	A release of the Apache HTTP Server (httpd), including a high performance *event-based processing model, enhanced SSL module and FastCGI support*. The **mod_auth_kerb**, **mod_auth_mellon**, and **ModSecurity** modules are also included.
Varnish Cache 5.2.1 [a]	rh-varnish5	A release of Varnish Cache, a high-performance HTTP reverse proxy. This version includes *the **shard** director, experimental HTTP/2 support, and improvements to Varnish configuration* through separate VCL files and VCL labels.
Varnish Cache 6.0.2 [a]	rh-varnish6	A release of Varnish Cache, a high-performance HTTP reverse proxy. This version includes *support for Unix Domain Sockets (both for clients and for back-end servers), new level of the VCL language (**vcl 4.1**), and improved HTTP/2 support*.
Maven 3.5.0 [a]	rh-maven35	A release of Maven, a software project management and comprehension tool. This release introduces support for new architectures and a number of new features, including *colorized logging*.
Maven 3.6.1 [a]	rh-maven36	A release of Maven, a software project management and comprehension tool. This release provides various enhancements and bug fixes.
Git 2.18.1 [a]	rh-git218	A release of Git, a distributed revision control system with a decentralized architecture. As opposed to centralized version control systems with a client-server model, Git ensures that each working copy of a Git repository is its exact copy with complete revision history. This version includes the *Large File Storage (LFS) extension*.
Redis 5.0.5 [a]	rh-redis5	A release of Redis 5.0, a*persistent key-value database*. Redis now provides ***redis-trib**, a cluster management tool*.

Component	Software Collection	Description
HAProxy 1.8.17 [a]	rh-haproxy18	A release of HAProxy 1.8, a reliable, high-performance *network load balancer* for TCP and HTTP-based applications.
Common Java Packages	rh-java-common	This Software Collection provides *common Java libraries and tools* used by other collections. Therh-java-common Software Collection is required by the rh-maven35 and rh-scala210 components and it is not supposed to be installed directly by users.
JDK Mission Control [a]	rh-jmc	This Software Collection includes *JDK Mission Control (JMC)*, a powerful profiler for HotSpot JVMs. JMC provides an advanced set of tools for efficient and detailed analysis of extensive data collected by the JDK Flight Recorder. JMC requires JDK version 8 or later to run. Target Java applications must run with at least OpenJDK version 11 so that JMC can access JDK Flight Recorder features. The rh-jmc Software Collection requires the rh-maven35 Software Collection.

[a] This Software Collection is available only for Red Hat Enterprise Linux 7

Previously released Software Collections remain available in the same distribution channels. All Software Collections, including retired components, are listed in the Table 1.2, "All Available Software Collections". Software Collections that are no longer supported are marked with an asterisk (*).

See the Red Hat Software Collections Product Life Cycle document for information on the length of support for individual components. For detailed information regarding previously released components, refer to the Release Notes for earlier versions of Red Hat Software Collections.

Table 1.2. All Available Software Collections

Component	Software Collection	Availability	Architectures supported on RHEL7
Components New in Red Hat Software Collections 3.4			
Red Hat Developer Toolset 9.0	devtoolset-9	RHEL7	x86_64, s390x, aarch64, ppc64, ppc64le
Node.js 12.10.0	rh-nodejs12	RHEL7	x86_64, s390x, aarch64, ppc64le
PHP 7.3.11	rh-php73	RHEL7	x86_64, s390x, aarch64, ppc64le
nginx 1.16.1	rh-nginx116	RHEL7	x86_64, s390x, aarch64, ppc64le

Component	Software Collection	Availability	Architectures supported on RHEL7
Components New in Red Hat Software Collections 3.4			
PostgreSQL 12.1	rh-postgresql12	RHEL7	x86_64, s390x, aarch64, ppc64le
Maven 3.6.1	rh-maven36	RHEL7	x86_64, s390x, aarch64, ppc64le
Components Updated in Red Hat Software Collections 3.4			
Apache httpd 2.4.34	httpd24	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
Components Last Updated in Red Hat Software Collections 3.3			
Red Hat Developer Toolset 8.1	devtoolset-8	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64, ppc64le
MariaDB 10.3.13	rh-mariadb103	RHEL7	x86_64, s390x, aarch64, ppc64le
Redis 5.0.5	rh-redis5	RHEL7	x86_64, s390x, aarch64, ppc64le
Ruby 2.6.2	rh-ruby26	RHEL7	x86_64, s390x, aarch64, ppc64le
HAProxy 1.8.17	rh-haproxy18	RHEL7	x86_64
Varnish Cache 6.0.2	rh-varnish6	RHEL7	x86_64, s390x, aarch64, ppc64le
Components Last Updated in Red Hat Software Collections 3.2			
PHP 7.2.24	rh-php72	RHEL7	x86_64, s390x, aarch64, ppc64le
MySQL 8.0.17	rh-mysql80	RHEL7	x86_64, s390x, aarch64, ppc64le
Node.js 10.16.3	rh-nodejs10	RHEL7	x86_64, s390x, aarch64, ppc64le

Components Last Updated in Red Hat Software Collections 3.2			
nginx 1.14.1	rh-nginx114	RHEL7	x86_64, s390x, aarch64, ppc64le
Git 2.18.1	rh-git218	RHEL7	x86_64, s390x, aarch64, ppc64le
JDK Mission Control	rh-jmc	RHEL7	x86_64

Components Last Updated in Red Hat Software Collections 3.1			
Red Hat Developer Toolset 7.1	devtoolset-7*	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64, ppc64le
Perl 5.26.3	rh-perl526	RHEL7	x86_64, s390x, aarch64, ppc64le
Ruby 2.5.5	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 10.6	rh-postgresql10	RHEL7	x86_64, s390x, aarch64, ppc64le
PHP 7.0.27	rh-php70*	RHEL6, RHEL7	x86_64
MySQL 5.7.24	rh-mysql57*	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le

Components Last Updated in Red Hat Software Collections 3.0			
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 3.6.9	rh-python36	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le

Components Last Updated in Red Hat Software Collections 3.0			
Maven 3.5.0	rh-maven35	RHEL7	x86_64, s390x, aarch64, ppc64le
MariaDB 10.2.22	rh-mariadb102	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
PostgreSQL 9.6.10	rh-postgresql96	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
MongoDB 3.4.9	rh-mongodb34	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
Node.js 8.11.4	rh-nodejs8*	RHEL7	x86_64, s390x, aarch64, ppc64le

Components Last Updated in Red Hat Software Collections 2.4			
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
nginx 1.10.2	rh-nginx110	RHEL6, RHEL7	x86_64
Node.js 6.11.3	rh-nodejs6*	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
Ruby 2.4.6	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 2.7.16	python27	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
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

Components Last Updated in Red Hat Software Collections 2.3			
Git 2.9.3	rh-git29*	RHEL6, RHEL7	x86_64, s390x, aarch64, ppc64le
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 2.3.8	rh-ruby23*	RHEL6, RHEL7	x86_64
PHP 5.6.25	rh-php56*	RHEL6, RHEL7	x86_64

Components Last Updated in Red Hat Software Collections 2.2			
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 upgrade collection	rh-mongodb30upg*	RHEL6, RHEL7	x86_64
Node.js 4.6.2	rh-nodejs4*	RHEL6, RHEL7	x86_64
PostgreSQL 9.5.14	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

Components Last Updated in Red Hat Software Collections 2.1			
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

Components Last Updated in Red Hat Software Collections 2.1			
Maven 3.0.5	maven30*	RHEL6, RHEL7	x86_64
V8 3.14.5.10	v8314*	RHEL6, RHEL7	x86_64

Components Last Updated in Red Hat Software Collections 2.0			
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
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

Components Last Updated in Red Hat Software Collections 1			
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

Components Last Updated in Red Hat Software Collections 1			
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

Legend:

- RHEL6 – Red Hat Enterprise Linux 6
- RHEL7 – Red Hat Enterprise Linux 7
- x86_64 – AMD64 and Intel 64 architectures
- s390x – IBM Z
- aarch64 – The 64-bit ARM architecture
- ppc64 – IBM POWER, big endian
- ppc64le – IBM POWER, little endian
- * – Retired component; this Software Collection is no longer supported

The tables above list the latest versions available through asynchronous updates.

Note that Software Collections released in Red Hat Software Collections 2.0 and later include a **rh-** prefix in their names.

Eclipse is available as a part of the Red Hat Developer Tools offering.

1.3. CHANGES IN RED HAT SOFTWARE COLLECTIONS 3.4

1.3.1. Overview

Architectures

The Red Hat Software Collections offering contains packages for Red Hat Enterprise Linux 7 running on AMD64 and Intel 64 architectures; certain earlier Software Collections are available also for Red Hat Enterprise Linux 6.

In addition, Red Hat Software Collections 3.4 supports the following architectures on Red Hat Enterprise Linux 7:

- The 64-bit ARM architecture

- IBM Z
- IBM POWER, little endian

For a full list of components and their availability, see Table 1.2, "All Available Software Collections".

New Software Collections

Red Hat Software Collections 3.4 adds the following new Software Collections:

- devtoolset-9 – see Section 1.3.2, "Changes in Red Hat Developer Toolset"
- rh-nodejs12 – see Section 1.3.3, "Changes in Node.js"
- rh-php73 – see Section 1.3.4, "Changes in PHP"
- rh-nginx116 – see Section 1.3.5, "Changes in nginx"
- rh-postgresql12 – see Section 1.3.6, "Changes in PostgreSQL"
- rh-maven36 – see Section 1.3.7, "Changes in Maven"

All new Software Collections are available only for Red Hat Enterprise Linux 7.

Updated Software Collections

The following component has been updated in Red Hat Software Collections 3.4:

- httpd24 – see Section 1.3.8, "Changes in Apache httpd"

Red Hat Software Collections Container Images

The following container images are new in Red Hat Software Collections 3.4:

- rhscl/devtoolset-9-toolchain-rhel7
- rhscl/devtoolset-9-perftools-rhel7
- rhscl/nodejs-12-rhel7
- rhscl/php-73-rhel7
- rhscl/nginx-116-rhel7
- rhscl/postgresql-12-rhel7

The following container image has been updated in Red Hat Software Collections 3.4:

- rhscl/httpd-24-rhel7

For detailed information regarding Red Hat Software Collections container images, see Section 3.4, "Red Hat Software Collections Container Images".

1.3.2. Changes in Red Hat Developer Toolset

The following components have been upgraded in Red Hat Developer Toolset 9.0 compared to the previous release of Red Hat Developer Toolset:

- **GCC** to version 9.1.1

- **binutils** to version 2.32
- **GDB** to version 8.3
- **strace** to version 5.1
- **SystemTap** to version 4.1
- **Valgrind** to version 3.15.0
- **Dyninst** to version 10.1.0

For detailed information on changes in 9.0, see the Red Hat Developer Toolset User Guide.

1.3.3. Changes in Node.js

The new rh-nodejs12 Software Collection provides **Node.js 12.10.0**. Notable enhancements in this release include:

- The V8 engine upgraded to version 7.6
- A new default HTTP parser, **llhttp** (no longer experimental)
- Integrated capability of heap dump generation
- Support for ECMAScript 2015 (ES6) modules
- Improved support for native modules
- Worker threads no longer require a flag
- A new experimental diagnostic report feature
- Improved performance

For detailed changes in Node.js 12.10.0, see the upstream release notes and upstream documentation.

1.3.4. Changes in PHP

The new rh-php73 Software Collection with **PHP 7.3.11** introduces the following notable enhancements over rh-php72:

- The Xdebug extension included for development
- Enhanced and more flexible **heredoc** and **nowdoc** syntaxes
- The PCRE extension upgraded to PCRE2
- Improved multibyte string handling
- Support for LDAP controls
- Improved FastCGI Process Manager (FPM) logging
- The ability to add a trailing comma in function calls
- Improved performance

- Several deprecations and backward incompatible changes

For more information, see Migrating from PHP 7.2.x to PHP 7.3.x .

Note that the following behavior is different from upstream:

- The rh-php73 Software Collection does not support the **Argon2** password hashing algorithm.
- The **x** (PCRE_EXTENDED) pattern modifier is always enabled in the rh-php73 Software Collection. As a result, invalid escape sequences are not interpreted as literals.

1.3.5. Changes in nginx

The new rh-nginx116 Software Collection provides **nginx 1.16.1**, which introduces a number of new features and enhancements. For example:

- Numerous updates related to SSL (loading of SSL certificates and secret keys from variables, variable support in the **ssl_certificate** and **ssl_certificate_key** directives, a new **ssl_early_data** directive)
- New **keepalive**-related directives
- A new **random** directive for distributed load balancing
- New parameters and improvements to existing directives (port ranges for the **listen** directive, a new **delay** parameter for the **limit_req** directive, which enables two-stage rate limiting)
- A new **$upstream_bytes_sent** variable
- Improvements to User Datagram Protocol (UDP) proxying

Other notable changes include:

- The **ssl** directive has been deprecated; use the **ssl** parameter for the **listen** directive instead.
- **nginx** now detects missing SSL certificates during configuration testing.
- When using a host name in the **listen** directive, **nginx** now creates listening sockets for all addresses that the host name resolves to.

For more information regarding changes in **nginx**, refer to the upstream release notes.

For migration instructions, see Section 5.8, “Migrating to nginx 1.16” .

1.3.6. Changes in PostgreSQL

The new rh-postgresql12 Software Collection includes **PostgreSQL 12.1**. This release introduces various enhancements over version 10, distributed in an earlier Software Collection, such as:

- The PostgreSQL Audit Extension, **pgaudit**, which provides detailed session and object audit logging through the standard PostgreSQL logging facility.
- Improvements to the partitioning functionality, for example, support for hash partitioning
- Enhancements to query parallelism
- Stored SQL procedures enabling transaction management

- Various performance improvements
- Enhancements to the administrative functionality
- Support for the SQL/JSON path language
- Stored generated columns
- Nondeterministic collations
- New authentication features, including encryption of TCP/IP connections when using GSSAPI authentication or multi-factor authentication.

For detailed changes, see the upstream release notes for PostgreSQL 11 and PostgreSQL 12.

Note that support for Just-In-Time (JIT) compilation, available in upstream since **PostgreSQL 11**, is not provided by the rh-postgresql12 Software Collection.

The rh-postgresql12 Software Collection includes the rh-postgresql12-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and others. After installing the rh-postgreqsl12*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-postgreqsl12* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

For information on migration, see Section 5.6, "Migrating to PostgreSQL 12" .

1.3.7. Changes in Maven

The new rh-maven36 Software Collection with **Maven 3.6.1** includes numerous bug fixes and various enhancements. For detailed changes, see the upstream release notes.

1.3.8. Changes in Apache httpd

The httpd24 Software Collection has been updated to provide several security and bug fixes.

1.4. COMPATIBILITY INFORMATION

Red Hat Software Collections 3.4 is available for all supported releases of Red Hat Enterprise Linux 7 on AMD64 and Intel 64 architectures, the 64-bit ARM architecture, IBM Z, and IBM POWER, little endian.

Certain components are available also for all supported releases of Red Hat Enterprise Linux 6 on AMD64 and Intel 64 architectures.

For a full list of available components, see Table 1.2, "All Available Software Collections".

1.5. KNOWN ISSUES

multiple components, BZ#1716378

Certain files provided by the Software Collections debuginfo packages might conflict with the corresponding debuginfo package files from the base Red Hat Enterprise Linux system or from other versions of Red Hat Software Collections components. For example, the python27-python-debuginfo package files might conflict with the corresponding files from the python-debuginfo package installed on the core system. Similarly, files from the httpd24-mod_auth_mellon-debuginfo

package might conflict with similar files provided by the base system mod_auth_mellon-debuginfo package. To work around this problem, uninstall the base system debuginfo package prior to installing the Software Collection debuginfo package.

rh-mysql80 component, BZ#1646363

The **mysql-connector-java** database connector does not work with the **MySQL 8.0** server. To work around this problem, use the **mariadb-java-client** database connector from the rh-mariadb103 Software Collection.

rh-mysql80 component, BZ#1646158

The default character set has been changed to **utf8mb4** in **MySQL 8.0** but this character set is unsupported by the **php-mysqlnd** database connector. Consequently, **php-mysqlnd** fails to connect in the default configuration. To work around this problem, specify a known character set as a parameter of the MySQL server configuration. For example, modify the **/etc/opt/rh/rh-mysql80/my.cnf.d/mysql-server.cnf** file to read:

```
[mysqld]
character-set-server=utf8
```

httpd24 component, BZ#1429006

Since **httpd 2.4.27**, the **mod_http2** module is no longer supported with the default **prefork** Multi-Processing Module (MPM). To enable HTTP/2 support, edit the configuration file at **/opt/rh/httpd24/root/etc/httpd/conf.modules.d/00-mpm.conf** and switch to the **event** or **worker** MPM.

Note that the HTTP/2 server-push feature does not work on the 64-bit ARM architecture, IBM Z, and IBM POWER, little endian.

httpd24 component, BZ#1327548

The **mod_ssl** module does not support the ALPN protocol on Red Hat Enterprise Linux 6, or on Red Hat Enterprise Linux 7.3 and earlier. Consequently, clients that support upgrading TLS connections to HTTP/2 only using ALPN are limited to HTTP/1.1 support.

httpd24 component, BZ#1224763

When using the **mod_proxy_fcgi** module with FastCGI Process Manager (PHP-FPM), **httpd** uses port **8000** for the FastCGI protocol by default instead of the correct port **9000**. To work around this problem, specify the correct port explicitly in configuration.

httpd24 component, BZ#1382706

When SELinux is enabled, the **LD_LIBRARY_PATH** environment variable is not passed through to CGI scripts invoked by **httpd**. As a consequence, in some cases it is impossible to invoke executables from Software Collections enabled in the **/opt/rh/httpd24/service-environment** file from CGI scripts run by **httpd**. To work around this problem, set **LD_LIBRARY_PATH** as desired from within the CGI script.

httpd24 component

Compiling external applications against the Apache Portable Runtime (APR) and APR-util libraries from the httpd24 Software Collection is not supported. The LD_LIBRARY_PATH environment variable is not set in httpd24 because it is not required by any application in this Software Collection.

rh-python35, rh-python36 components, BZ#1499990

The **pytz** module, which is used by **Babel** for time zone support, is not included in the rh-python35, and rh-python36 Software Collections. Consequently, when the user tries to import the dates module from **Babel**, a traceback is returned. To work around this problem, install **pytz** through the **pip** package manager from the **pypi** public repository by using the **pip install pytz** command.

rh-python36 component

Certain complex trigonometric functions provided by **numpy** might return incorrect values on the 64-bit ARM architecture, IBM Z, and IBM POWER, little endian. The AMD64 and Intel 64 architectures are not affected by this problem.

python27 component, BZ#1330489

The python27-python-pymongo package has been updated to version 3.2.1. Note that this version is not fully compatible with the previously shipped version 2.5.2. For details, see https://api.mongodb.org/python/current/changelog.html.

scl-utils component

In Red Hat Enterprise Linux 7.5 and earlier, due to an architecture-specific macro bug in the scl-utils package, the **<collection>/root/usr/lib64/** directory does not have the correct package ownership on the 64-bit ARM architecture and on IBM POWER, little endian. As a consequence, this directory is not removed when a Software Collection is uninstalled. To work around this problem, manually delete **<collection>/root/usr/lib64/** when removing a Software Collection.

ruby component

Determination of **RubyGem** installation paths is dependent on the order in which multiple Software Collections are enabled. The required order has been changed since **Ruby 2.3.1** shipped in Red Hat Software Collections 2.3 to support dependent Collections. As a consequence, **RubyGem** paths, which are used for **gem** installation during an RPM build, are invalid when the Software Collections are supplied in an incorrect order. For example, the build fails if the RPM spec file contains **scl enable rh-ror50 rh-nodejs6**. To work around this problem, enable the rh-ror50 Software Collection last, for example, **scl enable rh-nodejs6 rh-ror50**.

maven component

When the user has installed both the Red Hat Enterprise Linux system version of maven-local package and the rh-maven*-maven-local package, **XMvn**, a tool used for building Java RPM packages, run from the Maven Software Collection tries to read the configuration file from the base system and fails. To work around this problem, uninstall the maven-local package from the base Red Hat Enterprise Linux system.

perl component

It is impossible to install more than one **mod_perl.so** library. As a consequence, it is not possible to use the **mod_perl** module from more than one **Perl** Software Collection.

postgresql component

The rh-postgresql9* packages for Red Hat Enterprise Linux 6 do not provide the **sepgsql** module as this feature requires installation of libselinux version 2.0.99, which is not available in Red Hat Enterprise Linux 6.

httpd, mariadb, mongodb, mysql, nodejs, perl, php, python, ruby, and ror components, BZ#1072319

When uninstalling the httpd24, rh-mariadb*, rh-mongodb*, rh-mysql*, rh-nodejs*, rh-perl*, rh-php*, python27, rh-python*, rh-ruby*, or rh-ror* packages, the order of uninstalling can be relevant due to ownership of dependent packages. As a consequence, some directories and files might not be

removed properly and might remain on the system.

mariadb, mysql components, BZ#1194611

Since **MariaDB 10** and **MySQL 5.6**, the rh-mariadb*-mariadb-server and rh-mysql*-mysql-server packages no longer provide the **test** database by default. Although this database is not created during initialization, the grant tables are prefilled with the same values as when **test** was created by default. As a consequence, upon a later creation of the **test** or **test_*** databases, these databases have less restricted access rights than is default for new databases.

Additionally, when running benchmarks, the **run-all-tests** script no longer works out of the box with example parameters. You need to create a test database before running the tests and specify the database name in the **--database** parameter. If the parameter is not specified, **test** is taken by default but you need to make sure the **test** database exist.

mariadb, mysql, postgresql, mongodb components

Red Hat Software Collections contains the **MySQL 5.7**, **MySQL 8.0**, **MariaDB 10.2**, **MariaDB 10.3**, **PostgreSQL 9.6**, **PostgreSQL 10**, **PostgreSQL 12**, **MongoDB 3.4**, and **MongoDB 3.6** databases. The core Red Hat Enterprise Linux 6 provides earlier versions of the **MySQL** and **PostgreSQL** databases (client library and daemon). The core Red Hat Enterprise Linux 7 provides earlier versions of the **MariaDB** and **PostgreSQL** databases (client library and daemon). Client libraries are also used in database connectors for dynamic languages, libraries, and so on.

The client library packaged in the Red Hat Software Collections database packages in the **PostgreSQL** component is not supposed to be used, as it is included only for purposes of server utilities and the daemon. Users are instead expected to use the system library and the database connectors provided with the core system.

A protocol, which is used between the client library and the daemon, is stable across database versions, so, for example, using the **PostgreSQL 9.2** client library with the **PostgreSQL 9.4** or **9.5** daemon works as expected.

The core Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7 do not include the client library for **MongoDB**. In order to use this client library for your application, you should use the client library from Red Hat Software Collections and always use the **scl enable ...** call every time you run an application linked against this **MongoDB** client library.

mariadb, mysql, mongodb components

MariaDB, MySQL, and MongoDB do not make use of the **/opt/*provider*/*collection*/root** prefix when creating log files. Note that log files are saved in the **/var/opt/*provider*/*collection*/log/** directory, not in **/opt/*provider*/*collection*/root/var/log/**.

Other Notes

rh-ruby*, rh-python*, rh-php* components

Using Software Collections on a read-only NFS has several limitations.

- Ruby gems cannot be installed while the rh-ruby* Software Collection is on a read-only NFS. Consequently, for example, when the user tries to install the ab gem using the **gem install ab** command, an error message is displayed, for example:

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

The same problem occurs when the user tries to update or install gems from an external source by running the **bundle update** or **bundle install** commands.

- When installing Python packages on a read-only NFS using the Python Package Index (PyPI), running the **pip** command fails with an error message similar to this:

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

- Installing packages from PHP Extension and Application Repository (PEAR) on a read-only NFS using the **pear** command fails with the error message:

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

This is an expected behavior.

httpd component

Language modules for Apache are supported only with the Red Hat Software Collections version of **Apache httpd** and not with the Red Hat Enterprise Linux system versions of **httpd**. For example, the **mod_wsgi** module from the rh-python35 Collection can be used only with the httpd24 Collection.

all components

Since Red Hat Software Collections 2.0, configuration files, variable data, and runtime data of individual Collections are stored in different directories than in previous versions of Red Hat Software Collections.

coreutils, util-linux, screen components

Some utilities, for example, **su**, **login**, or **screen**, do not export environment settings in all cases, which can lead to unexpected results. It is therefore recommended to use **sudo** instead of **su** and set the **env_keep** environment variable in the **/etc/sudoers** file. Alternatively, you can run commands in a reverse order; for example:

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

instead of

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

When using tools like **screen** or **login**, you can use the following command to preserve the environment settings:

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

python component

When the user tries to install more than one scldevel package from the python27 and rh-python* Software Collections, a transaction check error message is returned. This is an expected behavior because the user can install only one set of the macro files provided by the packages (**%scl_python**, **%scl_*prefix*_python**).

php component

When the user tries to install more than one scldevel package from the rh-php* Software Collections, a transaction check error message is returned. This is an expected behavior because the user can install only one set of the macro files provided by the packages (**%scl_php**, **%scl_*prefix*_php**).

ruby component

When the user tries to install more than one scldevel package from the rh-ruby* Software Collections, a transaction check error message is returned. This is an expected behavior because the user can install only one set of the macro files provided by the packages (**%scl_ruby**, **%scl_*prefix*_ruby**).

perl component

When the user tries to install more than one scldevel package from the rh-perl* Software Collections, a transaction check error message is returned. This is an expected behavior because the user can install only one set of the macro files provided by the packages (**%scl_perl**, **%scl_*prefix*_perl**).

nginx component

When the user tries to install more than one scldevel package from the rh-nginx* Software Collections, a transaction check error message is returned. This is an expected behavior because the user can install only one set of the macro files provided by the packages (**%scl_nginx**, **%scl_*prefix*_nginx**).

1.6. DEPRECATED FUNCTIONALITY

httpd24 component, BZ#1434053

Previously, in an SSL/TLS configuration requiring name-based SSL virtual host selection, the **mod_ssl** module rejected requests with a **400 Bad Request** error, if the host name provided in the **Host:** header did not match the host name provided in a Server Name Indication (SNI) header. Such requests are no longer rejected if the configured SSL/TLS security parameters are identical between the selected virtual hosts, in-line with the behavior of upstream **mod_ssl**.

CHAPTER 2. INSTALLATION

This chapter describes in detail how to get access to the content set, install Red Hat Software Collections 3.4 on the system, and rebuild Red Hat Software Collections.

2.1. GETTING ACCESS TO RED HAT SOFTWARE COLLECTIONS

The Red Hat Software Collections content set is available to customers with Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7 subscriptions listed at https://access.redhat.com/solutions/472793. For information on how to register your system with Red Hat Subscription Management (RHSM), see Using and Configuring Red Hat Subscription Manager. For detailed instructions on how to enable Red Hat Software Collections using RHSM, see Section 2.1.1, "Using Red Hat Subscription Management" .

Since Red Hat Software Collections 2.2, the Red Hat Software Collections and Red Hat Developer Toolset content is available also in the ISO format at https://access.redhat.com/downloads, specifically for Server and Workstation. Note that packages that require the **Optional** channel, which are listed in Section 2.1.2, "Packages from the Optional Channel" , cannot be installed from the ISO image.

NOTE

Packages that require the **Optional** channel cannot be installed from the ISO image. A list of packages that require enabling of the **Optional** channel is provided in Section 2.1.2, "Packages from the Optional Channel".

Beta content is unavailable in the ISO format.

2.1.1. Using Red Hat Subscription Management

If your system is registered with Red Hat Subscription Management, complete the following steps to attach the subscription that provides access to the repository for Red Hat Software Collections and enable the repository:

1. Display a list of all subscriptions that are available for your system and determine the pool ID of a subscription that provides Red Hat Software Collections. To do so, type the following at a shell prompt as **root**:

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

   For each available subscription, this command displays its name, unique identifier, expiration date, and other details related to it. The pool ID is listed on a line beginning with **Pool Id**.

2. Attach the appropriate subscription to your system by running the following command as **root**:

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

   Replace *pool_id* with the pool ID you determined in the previous step. To verify the list of subscriptions your system has currently attached, type as **root**:

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

3. Display the list of available Yum list repositories to retrieve repository metadata and determine the exact name of the Red Hat Software Collections repositories. As **root**, type:

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

Or alternatively, run **yum repolist all** for a brief list.

The repository names depend on the specific version of Red Hat Enterprise Linux you are using and are in the following format:

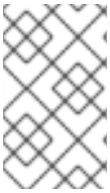
```
rhel-variant-rhscl-6-rpms
rhel-variant-rhscl-6-debug-rpms
rhel-variant-rhscl-6-source-rpms

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>
```

Replace *variant* with the Red Hat Enterprise Linux system variant, that is, **server** or **workstation**. Note that Red Hat Software Collections is supported neither on the **Client** nor on the **ComputeNode** variant.

4. Enable the appropriate repository by running the following command as **root**:

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

Once the subscription is attached to the system, you can install Red Hat Software Collections as described in Section 2.2, “Installing Red Hat Software Collections” . For more information on how to register your system using Red Hat Subscription Management and associate it with subscriptions, see Using and Configuring Red Hat Subscription Manager .



NOTE

Subscription through RHN is no longer available. For information how to migrate to RHSM, see https://access.redhat.com/products/red-hat-subscription-management/#migration.

2.1.2. Packages from the Optional Channel

Some of the Red Hat Software Collections packages require the **Optional** channel to be enabled in order to complete the full installation of these packages. For detailed instructions on how to subscribe your system to this channel, see the relevant Knowledgebase article at https://access.redhat.com/solutions/392003.

Packages from Software Collections for Red Hat Enterprise Linux that require the **Optional** channel to be enabled are listed in the tables below. Note that packages from the **Optional** channel are unsupported. For details, see the Knowledgebase article at https://access.redhat.com/articles/1150793.

Table 2.1. Packages That Require Enabling of the Optional Channel in Red Hat Enterprise Linux 7

Package from a Software Collection	Required Package from the Optional Channel
devtoolset-8-build	scl-utils-build
devtoolset-8-dyninst-testsuite	glibc-static
devtoolset-8-gcc-plugin-devel	libmpc-devel
devtoolset-9-build	scl-utils-build
devtoolset-9-dyninst-testsuite	glibc-static
devtoolset-9-gcc-plugin-devel	libmpc-devel
devtoolset-9-gdb	source-highlight
httpd24-mod_ldap	apr-util-ldap
httpd24-mod_session	apr-util-openssl
python27-python-debug	tix
python27-python-devel	scl-utils-build
python27-tkinter	tix
rh-git218-git-cvs	cvsps
rh-git218-git-svn	perl-Git-SVN, subversion
rh-git218-perl-Git-SVN	subversion-perl
rh-java-common-ant-apache-bsf	rhino
rh-java-common-batik	rhino
rh-maven35-xpp3-javadoc	java-1.7.0-openjdk-javadoc, java-1.8.0-openjdk-javadoc, java-1.8.0-openjdk-javadoc-zip, java-11-openjdk-javadoc, java-11-openjdk-javadoc-zip
rh-php72-php-pspell	aspell
rh-php73-php-devel	pcre2-devel
rh-php73-php-pspell	aspell

Package from a Software Collection	Required Package from the Optional Channel
rh-python36-python-devel	scl-utils-build
rh-python36-python-sphinx	texlive-framed, texlive-threeparttable, texlive-titlesec, texlive-wrapfig

Table 2.2. Packages That Require Enabling of the Optional Channel in Red Hat Enterprise Linux 6

Package from a Software Collection	Required Package from the Optional Channel
devtoolset-8-dyninst-testsuite	glibc-static
devtoolset-8-elfutils-devel	xz-devel
devtoolset-8-gcc-plugin-devel	gmp-devel, mpfr-devel
devtoolset-8-libatomic-devel	libatomic
devtoolset-8-libgccjit	mpfr
python27-python-devel	scl-utils-build
rh-mariadb102-boost-devel	libicu-devel
rh-mariadb102-mariadb-bench	perl-GD
rh-mongodb34-boost-devel	libicu-devel
rh-perl524-perl-devel	gdbm-devel, systemtap-sdt-devel
rh-python36-python-devel	scl-utils-build

2.2. INSTALLING RED HAT SOFTWARE COLLECTIONS

Red Hat Software Collections is distributed as a collection of RPM packages that can be installed, updated, and uninstalled by using the standard package management tools included in Red Hat Enterprise Linux. Note that a valid subscription is required to install Red Hat Software Collections on your system. For detailed instructions on how to associate your system with an appropriate subscription and get access to Red Hat Software Collections, see Section 2.1, “Getting Access to Red Hat Software Collections”.

Use of Red Hat Software Collections 3.4 requires the removal of any earlier pre-release versions, including Beta releases. If you have installed any previous version of Red Hat Software Collections 3.4, uninstall it from your system and install the new version as described in the Section 2.3, “Uninstalling Red Hat Software Collections” and Section 2.2.1, “Installing Individual Software Collections” sections.>

The in-place upgrade from Red Hat Enterprise Linux 6 to Red Hat Enterprise Linux 7 is not supported by Red Hat Software Collections. As a consequence, the installed Software Collections might not work correctly after the upgrade. If you want to upgrade from Red Hat Enterprise Linux 6 to Red Hat Enterprise Linux 7, it is strongly recommended to remove all Red Hat Software Collections packages, perform the in-place upgrade, update the Red Hat Software Collections repository, and install the Software Collections packages again. It is advisable to back up all data before upgrading.

2.2.1. Installing Individual Software Collections

To install any of the Software Collections that are listed in Table 1.1, "Red Hat Software Collections 3.4 Components", install the corresponding meta package by typing the following at a shell prompt as **root**:

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

Replace *software_collection* with a space-separated list of Software Collections you want to install. For example, to install php54 and rh-mariadb100, type as **root**:

```
~]# yum install rh-php72 rh-mariadb102
```

This installs the main meta package for the selected Software Collection and a set of required packages as its dependencies. For information on how to install additional packages such as additional modules, see Section 2.2.2, "Installing Optional Packages".

2.2.2. Installing Optional Packages

Each component of Red Hat Software Collections is distributed with a number of optional packages that are not installed by default. To list all packages that are part of a certain Software Collection but are not installed on your system, type the following at a shell prompt:

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

To install any of these optional packages, type as **root**:

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

Replace *package_name* with a space-separated list of packages that you want to install. For example, to install the rh-perl526-perl-CPAN and rh-perl526-perl-Archive-Tar, type:

```
~]# yum install rh-perl526-perl-CPAN rh-perl526-perl-Archive-Tar
```

2.2.3. Installing Debugging Information

To install debugging information for any of the Red Hat Software Collections packages, make sure that the yum-utils package is installed and type the following command as **root**:

```
debuginfo-install package_name
```

For example, to install debugging information for the rh-ruby25-ruby package, type:

```
~]# debuginfo-install rh-ruby25-ruby
```

Note that you need to have access to the repository with these packages. If your system is registered

with Red Hat Subscription Management, enable the **rhel-*variant*-rhscl-6-debug-rpms** or **rhel-*variant*-rhscl-7-debug-rpms** repository as described in Section 2.1.1, "Using Red Hat Subscription Management". For more information on how to get access to debuginfo packages, see https://access.redhat.com/solutions/9907.

2.3. UNINSTALLING RED HAT SOFTWARE COLLECTIONS

To uninstall any of the Software Collections components, type the following at a shell prompt as **root**:

```
yum remove software_collection\*
```

Replace *software_collection* with the Software Collection component you want to uninstall.

Note that uninstallation of the packages provided by Red Hat Software Collections does not affect the Red Hat Enterprise Linux system versions of these tools.

2.4. REBUILDING RED HAT SOFTWARE COLLECTIONS

<collection>-build packages are not provided by default. If you wish to rebuild a collection and do not want or cannot use the **rpmbuild --define 'scl foo'** command, you first need to rebuild the metapackage, which provides the <collection>-build package.

Note that existing collections should not be rebuilt with different content. To add new packages into an existing collection, you need to create a new collection containing the new packages and make it dependent on packages from the original collection. The original collection has to be used without changes.

For detailed information on building Software Collections, refer to the Red Hat Software Collections Packaging Guide.

CHAPTER 3. USAGE

This chapter describes the necessary steps for rebuilding and using Red Hat Software Collections 3.4, and deploying applications that use Red Hat Software Collections.

3.1. USING RED HAT SOFTWARE COLLECTIONS

3.1.1. Running an Executable from a Software Collection

To run an executable from a particular Software Collection, type the following command at a shell prompt:

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

Or, alternatively, use the following command:

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

Replace *software_collection* with a space-separated list of Software Collections you want to use and *command* with the command you want to run. For example, to execute a Perl program stored in a file named **hello.pl** with the Perl interpreter from the perl526 Software Collection, type:

```
~]$ scl enable rh-perl526 'perl hello.pl'
Hello, World!
```

You can execute any command using the **scl** utility, causing it to be run with the executables from a selected Software Collection in preference to their possible Red Hat Enterprise Linux system equivalents. For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.4 Components” .

3.1.2. Running a Shell Session with a Software Collection as Default

To start a new shell session with executables from a selected Software Collection in preference to their Red Hat Enterprise Linux equivalents, type the following at a shell prompt:

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

Replace *software_collection* with a space-separated list of Software Collections you want to use. For example, to start a new shell session with the python27 and rh-postgresql10 Software Collections as default, type:

```
~]$ scl enable python27 rh-postgresql10 bash
```

The list of Software Collections that are enabled in the current session is stored in the **$X_SCLS** environment variable, for instance:

```
~]$ echo $X_SCLS
python27 rh-postgresql10
```

For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.4 Components” .

3.1.3. Running a System Service from a Software Collection

Running a System Service from a Software Collection in Red Hat Enterprise Linux 6

Software Collections that include system services install corresponding init scripts in the **/etc/rc.d/init.d/** directory. To start such a service in the current session, type the following at a shell prompt as **root**:

```
service software_collection-service_name start
```

Replace *software_collection* with the name of the Software Collection and *service_name* with the name of the service you want to start.

To configure this service to start automatically at boot time, type the following command as **root**:

```
chkconfig software_collection-service_name on
```

For example, to start the **postgresql** service from the rh-postgresql96 Software Collection and enable it in runlevels 2, 3, 4, and 5, type as **root**:

```
~]# service rh-postgresql96-postgresql start
Starting rh-postgresql96-postgresql service:               [  OK  ]
~]# chkconfig rh-postgresql96-postgresql on
```

For more information on how to manage system services in Red Hat Enterprise Linux 6, refer to the Red Hat Enterprise Linux 6 Deployment Guide. For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.4 Components”.

Running a System Service from a Software Collection in Red Hat Enterprise Linux 7

In Red Hat Enterprise Linux 7, init scripts have been replaced by **systemd** service unit files, which end with the **.service** file extension and serve a similar purpose as init scripts. To start a service in the current session, execute the following command as **root**:

```
systemctl start software_collection-service_name.service
```

Replace *software_collection* with the name of the Software Collection and *service_name* with the name of the service you want to start.

To configure this service to start automatically at boot time, type the following command as **root**:

```
systemctl enable software_collection-service_name.service
```

For example, to start the **postgresql** service from the rh-postgresql10 Software Collection and enable it at boot time, type as **root**:

```
~]# systemctl start rh-postgresql10-postgresql.service
~]# systemctl enable rh-postgresql10-postgresql.service
```

For more information on how to manage system services in Red Hat Enterprise Linux 7, refer to the Red Hat Enterprise Linux 7 System Administrator's Guide . For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.4 Components”.

3.2. ACCESSING A MANUAL PAGE FROM A SOFTWARE COLLECTION

Every Software Collection contains a general manual page that describes the content of this component. Each manual page has the same name as the component and it is located in the **/opt/rh** directory.

To read a manual page for a Software Collection, type the following command:

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

Replace *software_collection* with the particular Red Hat Software Collections component. For example, to display the manual page for rh-mariadb102, type:

```
~]$ scl enable rh-mariadb102 "man rh-mariadb102"
```

3.3. DEPLOYING APPLICATIONS THAT USE RED HAT SOFTWARE COLLECTIONS

In general, you can use one of the following two approaches to deploy an application that depends on a component from Red Hat Software Collections in production:

- Install all required Software Collections and packages manually and then deploy your application, or
- Create a new Software Collection for your application and specify all required Software Collections and other packages as dependencies.

For more information on how to manually install individual Red Hat Software Collections components, see Section 2.2, “Installing Red Hat Software Collections” . For further details on how to use Red Hat Software Collections, see Section 3.1, “Using Red Hat Software Collections” . For a detailed explanation of how to create a custom Software Collection or extend an existing one, read the Red Hat Software Collections Packaging Guide.

3.4. RED HAT SOFTWARE COLLECTIONS CONTAINER IMAGES

Container images based on Red Hat Software Collections include applications, daemons, and databases. The images can be run on Red Hat Enterprise Linux 7 Server and Red Hat Enterprise Linux Atomic Host. For information about their usage, see Using Red Hat Software Collections 3 Container Images. For details regarding container images based on Red Hat Software Collections versions 2.4 and earlier, see Using Red Hat Software Collections 2 Container Images .

The following container images are available with Red Hat Software Collections 3.4:

- rhscl/devtoolset-9-toolchain-rhel7
- rhscl/devtoolset-9-perftools-rhel7
- rhscl/nodejs-12-rhel7
- rhscl/php-73-rhel7
- rhscl/nginx-116-rhel7
- rhscl/postgresql-12-rhel7
- rhscl/httpd-24-rhel7

The following container images are based on Red Hat Software Collections 3.3:

- rhscl/mariadb-103-rhel7
- rhscl/redis-5-rhel7
- rhscl/ruby-26-rhel7
- rhscl/devtoolset-8-toolchain-rhel7
- rhscl/devtoolset-8-perftools-rhel7
- rhscl/varnish-6-rhel7

The following container images are based on Red Hat Software Collections 3.2:

- rhscl/mysql-80-rhel7
- rhscl/nginx-114-rhel7
- rhscl/php-72-rhel7
- rhscl/nodejs-10-rhel7

The following container images are based on Red Hat Software Collections 3.1:

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

The following container images are based on Red Hat Software Collections 3.0:

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

The following container images are based on Red Hat Software Collections 2.4:

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

The following container images are based on Red Hat Software Collections 2.3:

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

The following container images are based on Red Hat Software Collections 2.2:

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

The following container images are based on Red Hat Software Collections 2.0:

- rhscl/mariadb-100-rhel7 (EOL)
- rhscl/mongodb-26-rhel7 (EOL)
- rhscl/mysql-56-rhel7 (EOL)
- rhscl/nginx-16-rhel7 (EOL)
- rhscl/passenger-40-rhel7 (EOL)
- rhscl/perl-520-rhel7 (EOL)
- rhscl/postgresql-94-rhel7 (EOL)
- rhscl/python-34-rhel7 (EOL)
- rhscl/ror-41-rhel7 (EOL)
- rhscl/ruby-22-rhel7 (EOL)
- rhscl/s2i-base-rhel7

Images marked as End of Life (EOL) are no longer supported.

CHAPTER 4. SPECIFICS OF INDIVIDUAL SOFTWARE COLLECTIONS

This chapter is focused on the specifics of certain Software Collections and provides additional details concerning these components.

4.1. RED HAT DEVELOPER TOOLSET

Red Hat Developer Toolset is designed for developers working on the Red Hat Enterprise Linux platform. Red Hat Developer Toolset provides current versions of the **GNU Compiler Collection**, **GNU Debugger**, and other development, debugging, and performance monitoring tools. Similarly to other Software Collections, an additional set of tools is installed into the /**opt**/ directory. These tools are enabled by the user on demand using the supplied **scl** utility. Similarly to other Software Collections, these do not replace the Red Hat Enterprise Linux system versions of these tools, nor will they be used in preference to those system versions unless explicitly invoked using the **scl** utility.

For an overview of features, refer to the Features section of the *Red Hat Developer Toolset Release Notes*.

4.2. RUBY ON RAILS 5.0

Red Hat Software Collections 3.4 provides the rh-ruby24 Software Collection together with the rh-ror50 Collection.

To install **Ruby on Rails 5.0**, type the following command as **root**:

```
yum install rh-ror50
```

Installing any package from the rh-ror50 Software Collection automatically pulls in rh-ruby24 and rh-nodejs6 as dependencies.

The rh-nodejs6 Collection is used by certain gems in an asset pipeline to post-process web resources, for example, **sass** or **coffee-script** source files. Additionally, the **Action Cable** framework uses rh-nodejs6 for handling **WebSockets** in Rails.

To run the **rails s** command without requiring rh-nodejs6, disable the **coffee-rails** and **uglifier** gems in the **Gemfile**.

To run **Ruby on Rails** without **Node.js**, run the following command, which will automatically enable rh-ruby24:

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

To run **Ruby on Rails** with all features, enable also the rh-nodejs6 Software Collection:

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

The rh-ror50 Software Collection is supported together with the rh-ruby24 and rh-nodejs6 components.

4.3. MONGODB 3.6

The rh-mongodb36 Software Collection is available only for Red Hat Enterprise Linux 7. See Section 4.4, "MongoDB 3.4" for instructions on how to use **MongoDB 3.4** on Red Hat Enterprise Linux 6.

To install the rh-mongodb36 collection, type the following command as **root**:

```
yum install rh-mongodb36
```

To run the **MongoDB** shell utility, type the following command:

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

NOTE

The rh-mongodb36-mongo-cxx-driver package has been built with the **-std=gnu++14** option using **GCC** from Red Hat Developer Toolset 6. Binaries using the shared library for the MongoDB C++ Driver that use C++11 (or later) features have to be built also with Red Hat Developer Toolset 6 or later. See C++ compatibility details in the Red Hat Developer Toolset 6 User Guide.

To start the **MongoDB** daemon, type the following command as **root**:

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

To start the **MongoDB** daemon on boot, type this command as **root**:

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

To start the **MongoDB** sharding server, type the following command as **root**:

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

To start the **MongoDB** sharding server on boot, type this command as **root**:

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

Note that the **MongoDB** sharding server does not work unless the user starts at least one configuration server and specifies it in the **mongos.conf** file.

4.4. MONGODB 3.4

To install the rh-mongodb34 collection, type the following command as **root**:

```
yum install rh-mongodb34
```

To run the **MongoDB** shell utility, type the following command:

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

NOTE

The rh-mongodb34-mongo-cxx-driver package has been built with the **-std=gnu++14** option using **GCC** from Red Hat Developer Toolset 6. Binaries using the shared library for the MongoDB C++ Driver that use C++11 (or later) features have to be built also with Red Hat Developer Toolset 6. See C++ compatibility details in the Red Hat Developer Toolset 6 User Guide.

MongoDB 3.4 on Red Hat Enterprise Linux 6

If you are using Red Hat Enterprise Linux 6, the following instructions apply to your system.

To start the **MongoDB** daemon, type the following command as **root**:

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

To start the **MongoDB** daemon on boot, type this command as **root**:

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

To start the **MongoDB** sharding server, type this command as **root**:

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

To start the **MongoDB** sharding server on boot, type the following command as **root**:

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

Note that the **MongoDB** sharding server does not work unless the user starts at least one configuration server and specifies it in the **mongos.conf** file.

MongoDB 3.4 on Red Hat Enterprise Linux 7

When using Red Hat Enterprise Linux 7, the following commands are applicable.

To start the **MongoDB** daemon, type the following command as **root**:

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

To start the **MongoDB** daemon on boot, type this command as **root**:

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

To start the **MongoDB** sharding server, type the following command as **root**:

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

To start the **MongoDB** sharding server on boot, type this command as **root**:

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

Note that the **MongoDB** sharding server does not work unless the user starts at least one configuration server and specifies it in the **mongos.conf** file.

4.5. MAVEN

The rh-maven35 Software Collection, available only for Red Hat Enterprise Linux 7, provides a software project management and comprehension tool. Based on the concept of a project object model (POM), **Maven** can manage a project's build, reporting, and documentation from a central piece of information.

To install the rh-maven36 Collection, type the following command as **root**:

```
yum install rh-maven36
```

To enable this collection, type the following command at a shell prompt:

```
scl enable rh-maven36 bash
```

Global Maven settings, such as remote repositories or mirrors, can be customized by editing the **/opt/rh/rh-maven36/root/etc/maven/settings.xml** file.

For more information about using Maven, refer to the Maven documentation. Usage of plug-ins is described in this section; to find documentation regarding individual plug-ins, see the index of plug-ins.

4.6. PASSENGER

The rh-passenger40 Software Collection provides **Phusion Passenger**, a web and application server designed to be fast, robust and lightweight.

The rh-passenger40 Collection supports multiple versions of **Ruby**, particularly the ruby193, ruby200, and rh-ruby22 Software Collections together with **Ruby on Rails** using the ror40 or rh-ror41 Collections. Prior to using **Passenger** with any of the **Ruby** Software Collections, install the corresponding package from the rh-passenger40 Collection: the rh-passenger-ruby193, rh-passenger-ruby200, or rh-passenger-ruby22 package.

The rh-passenger40 Software Collection can also be used with **Apache httpd** from the httpd24 Software Collection. To do so, install the rh-passenger40-mod_passenger package. Refer to the default configuration file **/opt/rh/httpd24/root/etc/httpd/conf.d/passenger.conf** for an example of **Apache httpd** configuration, which shows how to use multiple **Ruby** versions in a single **Apache httpd** instance.

Additionally, the rh-passenger40 Software Collection can be used with the **nginx 1.6** web server from the nginx16 Software Collection. To use **nginx 1.6** with rh-passenger40, you can run **Passenger** in Standalone mode using the following command in the web appplication's directory:

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

Alternatively, edit the nginx16 configuration files as described in the upstream Passenger documentation.

4.7. DATABASE CONNECTORS

Database connector packages provide the database client functionality, which is necessary for local or remote connection to a database server. Table 4.1, “Interoperability Between Languages and Databases” lists Software Collections with language runtimes that include connectors for certain database servers.

Table 4.1. Interoperability Between Languages and Databases

Language (Software Collection)	Database				
	MariaDB	MongoDB	MySQL	PostgreSQL	Redis
rh-nodejs4	✗	✓	✗	✗	✗
rh-nodejs6	✗	✓	✗	✗	✗
rh-nodejs8	✗	✓	✗	✗	✗
rh-nodejs10	✗	✓	✗	✗	✗
rh-nodejs12	✗	✓	✗	✗	✗
rh-perl520	✓	✗	✓	✓	✗
rh-perl524	✓	✗	✓	✓	✗
rh-perl526	✓	✗	✓	✓	✗
rh-php56	✓	✓	✓	✓	✗
rh-php70	✓	✗	✓	✓	✗
rh-php71	✓	✗	✓	✓	✗
rh-php72	✓	✗	✓	✓	✗
rh-php73	✓	✗	✓	✓	✗
python27	✓	✓	✓	✓	✗

Language (Software Collection)	Database				
	MariaDB	MongoDB	MySQL	PostgreSQL	Redis
rh-python34	✗	✓	✗	✓	✗
rh-python35	✓	✓	✓	✓	✗
rh-python36	✓	✓	✓	✓	✗
rh-ror41	✓	✓	✓	✓	✗
rh-ror42	✓	✓	✓	✓	✗
rh-ror50	✓	✓	✓	✓	✗
rh-ruby25	✓	✓	✓	✓	✗
rh-ruby26	✓	✓	✓	✓	✗
✓	Supported		✗	Unsupported	

CHAPTER 5. MIGRATION

This chapter provides information on migrating to versions of components included in Red Hat Software Collections 3.4.

5.1. MIGRATING TO MARIADB 10.3

The rh-mariadb103 Software Collection is available for Red Hat Enterprise Linux 7, which includes **MariaDB 5.5** as the default MySQL implementation.

The rh-mariadb103 Software Collection does not conflict with the mysql or mariadb packages from the core systems. Unless the *-syspaths packages are installed (see below), it is possible to install the rh-mariadb103 Software Collection together with the mysql or mariadb packages. It is also possible to run both versions at the same time, however, the port number and the socket in the **my.cnf** files need to be changed to prevent these specific resources from conflicting. Additionally, it is possible to install the rh-mariadb103 Software Collection while the rh-mariadb102 Collection is still installed and even running.

The rh-mariadb103 Software Collection includes the rh-mariadb103-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mariadb103*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-mariadb103* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mariadb102 and rh-mysql80 Software Collections. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

The recommended migration path from **MariaDB 5.5** to **MariaDB 10.3** is to upgrade to **MariaDB 10.0** first, and then upgrade by one version successively. For details, see instructions in earlier Red Hat Software Collections Release Notes: Migrating to MariaDB 10.0, Migrating to MariaDB 10.1, and Migrating to MariaDB 10.2.

NOTE

The rh-mariadb103 Software Collection supports neither mounting over NFS nor dynamical registering using the **scl register** command.

5.1.1. Notable Differences Between the rh-mariadb102 and rh-mariadb103 Software Collections

- The mariadb-bench subpackage has been removed.
- The default allowed level of the plug-in maturity has been changed to one level less than the server maturity. As a result, plug-ins with a lower maturity level that were previously working, will no longer load.

For more information regarding **MariaDB 10.3**, see the upstream documentation about changes and about upgrading.

5.1.2. Upgrading from the rh-mariadb102 to the rh-mariadb103 Software Collection

IMPORTANT

Prior to upgrading, back up all your data, including any MariaDB databases.

1. Stop the rh-mariadb102 database server if it is still running.

   Before stopping the server, set the **innodb_fast_shutdown** option to **0**, so that **InnoDB** performs a slow shutdown, including a full purge and insert buffer merge. Read more about this option in the upstream documentation. This operation can take a longer time than in case of a normal shutdown.

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

   Stop the rh-mariadb102 server.

   ```
   systemctl stop rh-mariadb102-mariadb.service
   ```

2. Install the rh-mariadb103 Software Collection, including the subpackage providing the **mysql_upgrade** utility.

   ```
   yum install rh-mariadb103-mariadb-server rh-mariadb103-mariadb-server-utils
   ```

   Note that it is possible to install the rh-mariadb103 Software Collection while the rh-mariadb102 Software Collection is still installed because these Collections do not conflict.

3. Inspect configuration of rh-mariadb103, which is stored in the **/etc/opt/rh/rh-mariadb103/my.cnf** file and the **/etc/opt/rh/rh-mariadb103/my.cnf.d/** directory. Compare it with configuration of rh-mariadb102 stored in **/etc/opt/rh/rh-mariadb102/my.cnf** and **/etc/opt/rh/rh-mariadb102/my.cnf.d/** and adjust it if necessary.

4. All data of the rh-mariadb102 Software Collection is stored in the **/var/opt/rh/rh-mariadb102/lib/mysql/** directory unless configured differently. Copy the whole content of this directory to **/var/opt/rh/rh-mariadb103/lib/mysql/**. You can move the content but remember to back up your data before you continue to upgrade. Make sure the data are owned by the **mysql** user and SELinux context is correct.

5. Start the rh-mariadb103 database server.

   ```
   systemctl start rh-mariadb103-mariadb.service
   ```

6. Perform the data migration. Note that running the **mysql_upgrade** command is required due to upstream changes introduced in MDEV-14637.

   ```
   scl enable rh-mariadb103 mysql_upgrade
   ```

   If the **root** user has a non-empty password defined (it should have a password defined), it is necessary to call the **mysql_upgrade** utility with the **-p** option and specify the password.

   ```
   scl enable rh-mariadb103 -- mysql_upgrade -p
   ```

   Note that when the rh-mariadb103*-syspaths packages are installed, the **scl enable** command is not required. However, the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mariadb102 and rh-mysql80 Software Collections.

5.2. MIGRATING TO MARIADB 10.2

Red Hat Enterprise Linux 6 contains **MySQL 5.1** as the default **MySQL** implementation. Red Hat Enterprise Linux 7 includes **MariaDB 5.5** as the default **MySQL** implementation. **MariaDB** is a community-developed drop-in replacement for **MySQL**. **MariaDB 10.1** has been available as a Software Collection since Red Hat Software Collections 2.2; Red Hat Software Collections 3.4 is distributed with **MariaDB 10.2**.

The rh-mariadb102 Software Collection, available for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7, does not conflict with the mysql or mariadb packages from the core systems. Unless the *-syspaths packages are installed (see below), it is possible to install the rh-mariadb102 Software Collection together with the mysql or mariadb packages. It is also possible to run both versions at the same time, however, the port number and the socket in the **my.cnf** files need to be changed to prevent these specific resources from conflicting. Additionally, it is possible to install the rh-mariadb102 Software Collection while the rh-mariadb101 Collection is still installed and even running.

The recommended migration path from **MariaDB 5.5** to **MariaDB 10.3** is to upgrade to **MariaDB 10.0** first, and then upgrade by one version successively. For details, see instructions in earlier Red Hat Software Collections Release Notes: Migrating to MariaDB 10.0 and Migrating to MariaDB 10.1.

For more information about **MariaDB 10.2**, see the upstream documentation about changes in version 10.2 and about upgrading.

NOTE

The rh-mariadb102 Software Collection supports neither mounting over NFS nor dynamical registering using the **scl register** command.

5.2.1. Notable Differences Between the rh-mariadb101 and rh-mariadb102 Software Collections

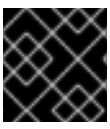
Major changes in **MariaDB 10.2** are described in the Red Hat Software Collections 3.0 Release Notes.

Since **MariaDB 10.2**, behavior of the **SQL_MODE** variable has been changed; see the upstream documentation for details.

Multiple options have changed their default values or have been deprecated or removed. For details, see the Knowledgebase article Migrating from MariaDB 10.1 to the MariaDB 10.2 Software Collection .

The rh-mariadb102 Software Collection includes the rh-mariadb102-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mariadb102*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-mariadb102* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mysql80 Software Collection. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

5.2.2. Upgrading from the rh-mariadb101 to the rh-mariadb102 Software Collection



IMPORTANT

Prior to upgrading, back up all your data, including any MariaDB databases.

1. Stop the rh-mariadb101 database server if it is still running.

   Before stopping the server, set the **innodb_fast_shutdown** option to **0**, so that **InnoDB**

performs a slow shutdown, including a full purge and insert buffer merge. Read more about this option in the upstream documentation. This operation can take a longer time than in case of a normal shutdown.

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

Stop the rh-mariadb101 server.

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

2. Install the rh-mariadb102 Software Collection.

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

Note that it is possible to install the rh-mariadb102 Software Collection while the rh-mariadb101 Software Collection is still installed because these Collections do not conflict.

3. Inspect configuration of rh-mariadb102, which is stored in the **/etc/opt/rh/rh-mariadb102/my.cnf** file and the **/etc/opt/rh/rh-mariadb102/my.cnf.d/** directory. Compare it with configuration of rh-mariadb101 stored in **/etc/opt/rh/rh-mariadb101/my.cnf** and **/etc/opt/rh/rh-mariadb101/my.cnf.d/** and adjust it if necessary.

4. All data of the rh-mariadb101 Software Collection is stored in the **/var/opt/rh/rh-mariadb101/lib/mysql/** directory unless configured differently. Copy the whole content of this directory to **/var/opt/rh/rh-mariadb102/lib/mysql/**. You can move the content but remember to back up your data before you continue to upgrade. Make sure the data are owned by the **mysql** user and SELinux context is correct.

5. Start the rh-mariadb102 database server.

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

6. Perform the data migration.

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

If the **root** user has a non-empty password defined (it should have a password defined), it is necessary to call the **mysql_upgrade** utility with the **-p** option and specify the password.

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

Note that when the rh-mariadb102*-syspaths packages are installed, the **scl enable** command is not required. However, the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mysql80 Software Collection.

5.3. MIGRATING TO MYSQL 8.0

The rh-mysql80 Software Collection is available for Red Hat Enterprise Linux 7, which includes **MariaDB 5.5** as the default **MySQL** implementation.

The rh-mysql80 Software Collection conflicts neither with the mysql or mariadb packages from the core systems nor with the rh-mysql* or rh-mariadb* Software Collections, unless the *-syspaths packages are installed (see below). It is also possible to run multiple versions at the same time; however,

the port number and the socket in the **my.cnf** files need to be changed to prevent these specific resources from conflicting.

Note that it is possible to upgrade to **MySQL 8.0** only from **MySQL 5.7**. If you need to upgrade from an earlier version, upgrade to **MySQL 5.7** first. For instructions, see Migration to MySQL 5.7.

5.3.1. Notable Differences Between MySQL 5.7 and MySQL 8.0

Differences Specific to the rh-mysql80 Software Collection

- The **MySQL 8.0** server provided by the rh-mysql80 Software Collection is configured to use **mysql_native_password** as the default authentication plug-in because client tools and libraries in Red Hat Enterprise Linux 7 are incompatible with the **caching_sha2_password** method, which is used by default in the upstream **MySQL 8.0** version.

  To change the default authentication plug-in to **caching_sha2_password**, edit the **/etc/opt/rh/rh-mysql80/my.cnf.d/mysql-default-authentication-plugin.cnf** file as follows:

  ```
  [mysqld]
  default_authentication_plugin=caching_sha2_password
  ```

  For more information about the **caching_sha2_password** authentication plug-in, see the upstream documentation.

- The rh-mysql80 Software Collection includes the rh-mysql80-syspaths package, which installs the rh-mysql80-mysql-config-syspaths, rh-mysql80-mysql-server-syspaths, and rh-mysql80-mysql-syspaths packages. These subpackages provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mysql80*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-mysql80* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mariadb102 and rh-mariadb103 Software Collections. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

General Changes in MySQL 8.0

- Binary logging is enabled by default during the server startup. The **log_bin** system variable is now set to **ON** by default even if the **--log-bin** option has not been specified. To disable binary logging, specify the **--skip-log-bin** or **--disable-log-bin** option at startup.

- For a **CREATE FUNCTION** statement to be accepted, at least one of the **DETERMINISTIC**, **NO SQL**, or **READS SQL DATA** keywords must be specified explicitly, otherwise an error occurs.

- Certain features related to account management have been removed. Namely, using the **GRANT** statement to modify account properties other than privilege assignments, such as authentication, SSL, and resource-limit, is no longer possible. To establish the mentioned properties at account-creation time, use the **CREATE USER** statement. To modify these properties, use the **ALTER USER** statement.

- Certain SSL-related options have been removed on the client-side. Use the **--ssl-mode=REQUIRED** option instead of **--ssl=1** or **--enable-ssl**. Use the **--ssl-mode=DISABLED** option instead of **--ssl=0**, **--skip-ssl**, or **--disable-ssl**. Use the **--ssl-mode=VERIFY_IDENTITY** option instead of **--ssl-verify-server-cert** options. Note that these option remains unchanged on the server side.

- The default character set has been changed from **latin1** to **utf8mb4**.

- The **utf8** character set is currently an alias for **utf8mb3** but in the future, it will become a reference to **utf8mb4**. To prevent ambiguity, specify **utf8mb4** explicitly for character set references instead of **utf8**.
- Setting user variables in statements other than **SET** has been deprecated.
- The **log_syslog** variable, which previously configured error logging to the system logs, has been removed.
- Certain incompatible changes to spatial data support have been introduced.
- The deprecated **ASC** or **DESC** qualifiers for **GROUP BY** clauses have been removed. To produce a given sort order, provide an **ORDER BY** clause.

For detailed changes in **MySQL 8.0** compared to earlier versions, see the upstream documentation: What Is New in MySQL 8.0 and Changes Affecting Upgrades to MySQL 8.0.

5.3.2. Upgrading to the rh-mysql80 Software Collection

IMPORTANT

Prior to upgrading, back-up all your data, including any MySQL databases.

1. Install the rh-mysql80 Software Collection.

```
yum install rh-mysql80-mysql-server
```

2. Inspect the configuration of rh-mysql80, which is stored in the **/etc/opt/rh/rh-mysql80/my.cnf** file and the **/etc/opt/rh/rh-mysql80/my.cnf.d/** directory. Compare it with the configuration of rh-mysql57 stored in **/etc/opt/rh/rh-mysql57/my.cnf** and **/etc/opt/rh/rh-mysql57/my.cnf.d/** and adjust it if necessary.
3. Stop the rh-mysql57 database server, if it is still running.

```
systemctl stop rh-mysql57-mysqld.service
```

4. All data of the rh-mysql57 Software Collection is stored in the **/var/opt/rh/rh-mysql57/lib/mysql/** directory. Copy the whole content of this directory to **/var/opt/rh/rh-mysql80/lib/mysql/**. You can also move the content but remember to back up your data before you continue to upgrade.
5. Start the rh-mysql80 database server.

```
systemctl start rh-mysql80-mysqld.service
```

6. Perform the data migration.

```
scl enable rh-mysql80 mysql_upgrade
```

If the **root** user has a non-empty password defined (it should have a password defined), it is necessary to call the **mysql_upgrade** utility with the **-p** option and specify the password.

```
scl enable rh-mysql80 -- mysql_upgrade -p
```

Note that when the rh-mysql80*-syspaths packages are installed, the **scl enable** command is not required. However, the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system and from the rh-mariadb102 and rh-mariadb103 Software Collections.

5.4. MIGRATING TO MONGODB 3.6

Red Hat Software Collections 3.4 is released with **MongoDB 3.6**, provided by the rh-mongodb36 Software Collection and available only for Red Hat Enterprise Linux 7.

The rh-mongodb36 Software Collection includes the rh-mongodb36-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mongodb36*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-mongodb36* packages. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

5.4.1. Notable Differences Between MongoDB 3.4 and MongoDB 3.6

General Changes

The rh-mongodb36 Software Collection introduces the following significant general change:

- On Non-Uniform Access Memory (NUMA) hardware, it is possible to configure **systemd** services to be launched using the **numactl** command; see the upstream recommendation. To use **MongoDB** with the **numactl** command, you need to install the numactl RPM package and change the **/etc/opt/rh/rh-mongodb36/sysconfig/mongod** and **/etc/opt/rh/rh-mongodb36/sysconfig/mongos** configuration files accordingly.

Compatibility Changes

MongoDB 3.6 includes various minor changes that can affect compatibility with previous versions of **MongoDB**:

- **MongoDB** binaries now bind to **localhost** by default, so listening on different IP addresses needs to be explicitly enabled. Note that this is already the default behavior for **systemd** services distributed with **MongoDB** Software Collections.
- The MONGODB-CR authentication mechanism has been deprecated. For databases with users created by **MongoDB** versions earlier than 3.0, upgrade authentication schema to SCRAM.
- The HTTP interface and REST API have been removed
- Arbiters in replica sets have priority **0**
- Master-slave replication has been deprecated

For detailed compatibility changes in **MongoDB 3.6**, see the upstream release notes.

Backwards Incompatible Features

The following **MongoDB 3.6** features are backwards incompatible and require the version to be set to 3.6 using the **featureCompatibilityVersion** command:

- UUID for collections
- **$jsonSchema** document validation
- Change streams

- Chunk aware secondaries
- View definitions, document validators, and partial index filters that use version 3.6 query features
- Sessions and retryable writes
- Users and roles with **authenticationRestrictions**

For details regarding backward incompatible changes in **MongoDB 3.6**, see the upstream release notes.

5.4.2. Upgrading from the rh-mongodb34 to the rh-mongodb36 Software Collection

IMPORTANT

Before migrating from the rh-mongodb34 to the rh-mongodb36 Software Collection, back up all your data, including any **MongoDB** databases, which are by default stored in the **/var/opt/rh/rh-mongodb34/lib/mongodb/** directory. In addition, see the Compatibility Changes to ensure that your applications and deployments are compatible with **MongoDB 3.6**.

To upgrade to the rh-mongodb36 Software Collection, perform the following steps.

1. To be able to upgrade, the rh-mongodb34 instance must have **featureCompatibilityVersion** set to **3.4**. Check **featureCompatibilityVersion**:

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

   If the **mongod** server is configured with enabled access control, add the **--username** and **--password** options to the **mongo** command.

2. Install the **MongoDB** servers and shells from the rh-mongodb36 Software Collections:

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

3. Stop the **MongoDB 3.4** server:

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

4. Copy your data to the new location:

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

5. Configure the **rh-mongodb36-mongod** daemon in the **/etc/opt/rh/rh-mongodb36/mongod.conf** file.

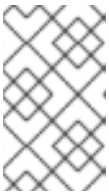
6. Start the **MongoDB 3.6** server:

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

7. Enable backwards incompatible features:

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

If the **mongod** server is configured with enabled access control, add the **--username** and **--password** options to the **mongo** command.



NOTE

After upgrading, it is recommended to run the deployment first without enabling the backwards incompatible features for a burn-in period of time, to minimize the likelihood of a downgrade.

For detailed information about upgrading, see the upstream release notes.

For information about upgrading a Replica Set, see the upstream MongoDB Manual.

For information about upgrading a Sharded Cluster, see the upstream MongoDB Manual.

5.5. MIGRATING TO MONGODB 3.4

The rh-mongodb34 Software Collection, available for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7, provides **MongoDB 3.4**.

5.5.1. Notable Differences Between MongoDB 3.2 and MongoDB 3.4

General Changes

The rh-mongodb34 Software Collection introduces various general changes. Major changes are listed in the Knowledgebase article Migrating from MongoDB 3.2 to MongoDB 3.4 . For detailed changes, see the upstream release notes.

In addition, this Software Collection includes the rh-mongodb34-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mongodb34*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-mongodb34* packages. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

Compatibility Changes

MongoDB 3.4 includes various minor changes that can affect compatibility with previous versions of **MongoDB**. For details, see the Knowledgebase article Migrating from MongoDB 3.2 to MongoDB 3.4 and the upstream documentation.

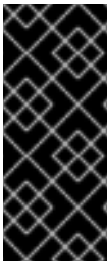
Notably, the following **MongoDB 3.4** features are backwards incompatible and require that the version is set to **3.4** using the **featureCompatibilityVersion** command:

- Support for creating read-only views from existing collections or other views
- Index version **v: 2**, which adds support for collation, decimal data and case-insensitive indexes
- Support for the **decimal128** format with the new **decimal** data type

For details regarding backward incompatible changes in **MongoDB 3.4**, see the upstream release notes.

5.5.2. Upgrading from the rh-mongodb32 to the rh-mongodb34 Software Collection

Note that once you have upgraded to **MongoDB 3.4** and started using new features, cannot downgrade to version 3.2.7 or earlier. You can only downgrade to version 3.2.8 or later.



IMPORTANT

Before migrating from the rh-mongodb32 to the rh-mongodb34 Software Collection, back up all your data, including any **MongoDB** databases, which are by default stored in the **/var/opt/rh/rh-mongodb32/lib/mongodb/** directory. In addition, see the compatibility changes to ensure that your applications and deployments are compatible with **MongoDB 3.4**.

To upgrade to the rh-mongodb34 Software Collection, perform the following steps.

1. Install the **MongoDB** servers and shells from the rh-mongodb34 Software Collections:

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

2. Stop the **MongoDB 3.2** server:

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

   Use the **service rh-mongodb32-mongodb stop** command on a Red Hat Enterprise Linux 6 system.

3. Copy your data to the new location:

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

4. Configure the **rh-mongodb34-mongod** daemon in the **/etc/opt/rh/rh-mongodb34/mongod.conf** file.

5. Start the **MongoDB 3.4** server:

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

   On Red Hat Enterprise Linux 6, use the **service rh-mongodb34-mongodb start** command instead.

6. Enable backwards-incompatible features:

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

   If the **mongod** server is configured with enabled access control, add the **--username** and **--password** options to **mongo** command.

   Note that it is recommended to run the deployment after the upgrade without enabling these features first.

For detailed information about upgrading, see the upstream release notes.

For information about upgrading a Replica Set, see the upstream MongoDB Manual.

For information about upgrading a Sharded Cluster, see the upstream MongoDB Manual.

5.6. MIGRATING TO POSTGRESQL 12

Red Hat Software Collections 3.4 is distributed with **PostgreSQL 12**, available only for Red Hat Enterprise Linux 7. The rh-postgresql12 Software Collection can be safely installed on the same machine in parallel with the base Red Hat Enterprise Linux system version of **PostgreSQL** or any **PostgreSQL** Software Collection. It is also possible to run more than one version of **PostgreSQL** on a machine at the same time, but you need to use different ports or IP addresses and adjust SELinux policy. See Section 5.7, "Migrating to PostgreSQL 9.6" for instructions how to migrate to an earlier version or when using Red Hat Enterprise Linux 6.

The rh-postgresql12 Software Collection includes the rh-postgresql12-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-postgreqsl12*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-postgreqsl12* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

IMPORTANT

Before migrating to **PostgreSQL 12**, see the upstream compatibility notes for PostgreSQL 11 and PostgreSQL 12.

In case of upgrading the **PostgreSQL** database in a container, see the container-specific instructions.

The following table provides an overview of different paths in a Red Hat Enterprise Linux 7 system version of **PostgreSQL** provided by the postgresql package, and in the rh-postgresql10 and rh-postgresql12 Software Colections.

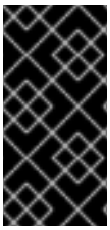
Table 5.1. Diferences in thePostgreSQL paths

Content	postgresql	rh-postgresql10	rh-postgresql12
Executables	/usr/bin/	/opt/rh/rh-postgresql10/root/usr/bin/	/opt/rh/rh-postgresql12/root/usr/bin/
Libraries	/usr/lib64/	/opt/rh/rh-postgresql10/root/usr/lib64/	/opt/rh/rh-postgresql12/root/usr/lib64/
Documentation	/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql12/root/usr/share/doc/postgresql/html/

Content	postgresql	rh-postgresql10	rh-postgresql12
PDF documentation	/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql12/root/usr/share/doc/postgresql-docs/
Contrib documentation	/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql10/root/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql12/root/usr/share/doc/postgresql-contrib/
Source	not installed	not installed	not installed
Data	/var/lib/pgsql/data/	/var/opt/rh/rh-postgresql10/lib/pgsql/data/	/var/opt/rh/rh-postgresql12/lib/pgsql/data/
Backup area	/var/lib/pgsql/backups/	/var/opt/rh/rh-postgresql10/lib/pgsql/backups/	/var/opt/rh/rh-postgresql12/lib/pgsql/backups/
Templates	/usr/share/pgsql/	/opt/rh/rh-postgresql10/root/usr/share/pgsql/	/opt/rh/rh-postgresql12/root/usr/share/pgsql/
Procedural Languages	/usr/lib64/pgsql/	/opt/rh/rh-postgresql10/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql12/root/usr/lib64/pgsql/
Development Headers	/usr/include/pgsql/	/opt/rh/rh-postgresql10/root/usr/include/pgsql/	/opt/rh/rh-postgresql12/root/usr/include/pgsql/
Other shared data	/usr/share/pgsql/	/opt/rh/rh-postgresql10/root/usr/share/pgsql/	/opt/rh/rh-postgresql12/root/usr/share/pgsql/
Regression tests	/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/rh-postgresql10/root/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/rh-postgresql12/root/usr/lib64/pgsql/test/regress/ (in the -test package)

5.6.1. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 12 Software Collection

Red Hat Enterprise Linux 7 is distributed with **PostgreSQL 9.2**. To migrate your data from a Red Hat Enterprise Linux system version of **PostgreSQL** to the rh-postgresql12 Software Collection, you can either perform a fast upgrade using the **pg_upgrade** tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method.



IMPORTANT

Before migrating your data from a Red Hat Enterprise Linux system version of PostgreSQL to PostgreSQL 12, make sure that you back up all your data, including the PostgreSQL database files, which are *by default* located in the **/var/lib/pgsql/data/** directory.

Procedure 5.1. Fast Upgrade Using the pg_upgrade Tool

To perform a fast upgrade of your PostgreSQL server, complete the following steps:

1. Stop the old PostgreSQL server to ensure that the data is not in an inconsistent state. To do so, type the following at a shell prompt as **root**:

   ```
   systemctl stop postgresql.service
   ```

   To verify that the server is not running, type:

   ```
   systemctl status postgresql.service
   ```

2. Verify that the old directory **/var/lib/pgsql/data/** exists:

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

   and back up your data.

3. Verify that the new data directory **/var/opt/rh/rh-postgresql12/lib/pgsql/data/** does not exist:

   ```
   file /var/opt/rh/rh-postgresql12/lib/pgsql/data/
   ```

   If you are running a fresh installation of **PostgreSQL 12**, this directory should not be present in your system. If it is, back it up by running the following command as **root**:

   ```
   mv /var/opt/rh/rh-postgresql12/lib/pgsql/data{,-scl-backup}
   ```

4. Upgrade the database data for the new server by running the following command as **root**:

   ```
   scl enable rh-postgresql12 -- postgresql-setup --upgrade
   ```

   Alternatively, you can use the **/opt/rh/rh-postgresql12/root/usr/bin/postgresql-setup --upgrade** command.

   Note that you can use the **--upgrade-from** option for upgrade from different versions of **PostgreSQL**. The list of possible upgrade scenarios is available using the **--upgrade-ids** option.

   It is recommended that you read the resulting **/var/lib/pgsql/upgrade_rh-postgresql12-postgresql.log** log file to find out if any problems occurred during the upgrade.

5. Start the new server as **root**:

   ```
   systemctl start rh-postgresql12-postgresql.service
   ```

   It is also advised that you run the **analyze_new_cluster.sh** script as follows:

```
su - postgres -c 'scl enable rh-postgresql12 ~/analyze_new_cluster.sh'
```

6. Optionally, you can configure the PostgreSQL 12 server to start automatically at boot time. To disable the old system PostgreSQL server, type the following command as **root**:

```
chkconfig postgresql off
```

To enable the PostgreSQL 12 server, type as **root**:

```
chkconfig rh-postgresql12-postgresql on
```

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql12/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

Procedure 5.2. Performing a Dump and Restore Upgrade

To perform a dump and restore upgrade of your PostgreSQL server, complete the following steps:

1. Ensure that the old PostgreSQL server is running by typing the following at a shell prompt as **root**:

```
systemctl start postgresql.service
```

2. Dump all data in the PostgreSQL database into a script file. As **root**, type:

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

3. Stop the old server by running the following command as **root**:

```
systemctl stop postgresql.service
```

4. Initialize the data directory for the new server as **root**:

```
scl enable rh-postgresql12 -- postgresql-setup initdb
```

5. Start the new server as **root**:

```
systemctl start rh-postgresql12-postgresql.service
```

6. Import data from the previously created SQL file:

```
su - postgres -c 'scl enable rh-postgresql12 "psql -f ~/pgdump_file.sql postgres"'
```

7. Optionally, you can configure the PostgreSQL 12 server to start automatically at boot time. To disable the old system PostgreSQL server, type the following command as **root**:

```
chkconfig postgresql off
```

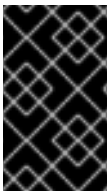
To enable the PostgreSQL 12 server, type as **root**:

```
chkconfig rh-postgresql12-postgresql on
```

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql12/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

5.6.2. Migrating from the PostgreSQL 10 Software Collection to the PostgreSQL 12 Software Collection

To migrate your data from the rh-postgresql10 Software Collection to the rh-postgresql12 Collection, you can either perform a fast upgrade using the **pg_upgrade** tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method.



IMPORTANT

Before migrating your data from **PostgreSQL 10** to **PostgreSQL 12**, make sure that you back up all your data, including the PostgreSQL database files, which are by default located in the **/var/opt/rh/rh-postgresql10/lib/pgsql/data/** directory.

Procedure 5.3. Fast Upgrade Using the pg_upgrade Tool

To perform a fast upgrade of your PostgreSQL server, complete the following steps:

1. Stop the old PostgreSQL server to ensure that the data is not in an inconsistent state. To do so, type the following at a shell prompt as **root**:

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

   To verify that the server is not running, type:

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

2. Verify that the old directory **/var/opt/rh/rh-postgresql10/lib/pgsql/data/** exists:

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

   and back up your data.

3. Verify that the new data directory **/var/opt/rh/rh-postgresql12/lib/pgsql/data/** does not exist:

   ```
   file /var/opt/rh/rh-postgresql12/lib/pgsql/data/
   ```

   If you are running a fresh installation of **PostgreSQL 12**, this directory should not be present in your system. If it is, back it up by running the following command as **root**:

   ```
   mv /var/opt/rh/rh-postgresql12/lib/pgsql/data{,-scl-backup}
   ```

4. Upgrade the database data for the new server by running the following command as **root**:

```
scl enable rh-postgresql12 -- postgresql-setup --upgrade --upgrade-from=rh-postgresql10-postgresql
```

Alternatively, you can use the **/opt/rh/rh-postgresql12/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql10-postgresql** command.

Note that you can use the **--upgrade-from** option for upgrading from different versions of **PostgreSQL**. The list of possible upgrade scenarios is available using the **--upgrade-ids** option.

It is recommended that you read the resulting **/var/lib/pgsql/upgrade_rh-postgresql12-postgresql.log** log file to find out if any problems occurred during the upgrade.

5. Start the new server as **root**:

```
systemctl start rh-postgresql12-postgresql.service
```

It is also advised that you run the **analyze_new_cluster.sh** script as follows:

```
su - postgres -c 'scl enable rh-postgresql12 ~/analyze_new_cluster.sh'
```

6. Optionally, you can configure the PostgreSQL 12 server to start automatically at boot time. To disable the old PostgreSQL 10 server, type the following command as **root**:

```
chkconfig rh-postgresql10-postgreqsql off
```

To enable the PostgreSQL 12 server, type as **root**:

```
chkconfig rh-postgresql12-postgresql on
```

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql12/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

Procedure 5.4. Performing a Dump and Restore Upgrade

To perform a dump and restore upgrade of your PostgreSQL server, complete the following steps:

1. Ensure that the old PostgreSQL server is running by typing the following at a shell prompt as **root**:

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

2. Dump all data in the PostgreSQL database into a script file. As **root**, type:

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

3. Stop the old server by running the following command as **root**:

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

4. Initialize the data directory for the new server as **root**:

```
scl enable rh-postgresql12 -- postgresql-setup initdb
```

5. Start the new server as **root**:

```
systemctl start rh-postgresql12-postgresql.service
```

6. Import data from the previously created SQL file:

```
su - postgres -c 'scl enable rh-postgresql12 "psql -f ~/pgdump_file.sql postgres"'
```

7. Optionally, you can configure the PostgreSQL 12 server to start automatically at boot time. To disable the old PostgreSQL 10 server, type the following command as **root**:

```
chkconfig rh-postgresql10-postgresql off
```

   To enable the PostgreSQL 12 server, type as **root**:

```
chkconfig rh-postgresql12-postgresql on
```

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql12/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

5.7. MIGRATING TO POSTGRESQL 9.6

PostgreSQL 9.6 is available for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7 and it can be safely installed on the same machine in parallel with **PostgreSQL 8.4** from Red Hat Enterprise Linux 6, **PostgreSQL 9.2** from Red Hat Enterprise Linux 7, or any version of **PostgreSQL** released in previous versions of Red Hat Software Collections. It is also possible to run more than one version of **PostgreSQL** on a machine at the same time, but you need to use different ports or IP addresses and adjust SELinux policy.

IMPORTANT

In case of upgrading the **PostgreSQL** database in a container, see the container-specific instructions. Note that it is currently impossible to upgrade **PostgreSQL** from 9.5 to 9.6 in a container in an OpenShift environment that is configured with Gluster file volumes.

5.7.1. Notable Differences Between PostgreSQL 9.5 and PostgreSQL 9.6

The most notable changes between **PostgreSQL 9.5** and **PostgreSQL 9.6** are described in the upstream release notes.

The rh-postgresql96 Software Collection includes the rh-postgresql96-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-postgreqsl96*-syspaths packages, users are not required to use the **scl enable** command for correct functioning of the binaries and scripts provided by the rh-postgreqsl96* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

The following table provides an overview of different paths in a Red Hat Enterprise Linux system version

of **PostgreSQL** (postgresql) and in the postgresql92, rh-postgresql95, and rh-postgresql96 Software Collections. Note that the paths of **PostgreSQL 8.4** distributed with Red Hat Enterprise Linux 6 and the system version of **PostgreSQL 9.2** shipped with Red Hat Enterprise Linux 7 are the same; the paths for the rh-postgresql94 Software Collection are analogous to rh-postgresql95.

Table 5.2. Diferences in the PostgreSQL paths

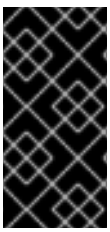
Content	postgresql	postgresql92	rh-postgresql95	rh-postgresql96
Executables	/usr/bin/	/opt/rh/postgresql92/root/usr/bin/	/opt/rh/rh-postgresql95/root/usr/bin/	/opt/rh/rh-postgresql96/root/usr/bin/
Libraries	/usr/lib64/	/opt/rh/postgresql92/root/usr/lib64/	/opt/rh/rh-postgresql95/root/usr/lib64/	/opt/rh/rh-postgresql96/root/usr/lib64/
Documentation	/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/
PDF documentation	/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 documentation	/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/
Source	not installed	not installed	not installed	not installed
Data	/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/
Backup area	/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/
Templates	/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/
Procedural Languages	/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/

Content	postgresql	postgresql92	rh-postgresql95	rh-postgresql96
Development Headers	/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/
Other shared data	/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/
Regression tests	/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/postgresql92/root/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/rh-postgresql95/root/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/rh-postgresql96/root/usr/lib64/pgsql/test/regress/ (in the -test package)

For changes between **PostgreSQL 8.4** and **PostgreSQL 9.2**, refer to the Red Hat Software Collections 1.2 Release Notes. Notable changes between **PostgreSQL 9.2** and **PostgreSQL 9.4** are described in Red Hat Software Collections 2.0 Release Notes . For differences between **PostgreSQL 9.4** and **PostgreSQL 9.5**, refer to Red Hat Software Collections 2.2 Release Notes.

5.7.2. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 9.6 Software Collection

Red Hat Enterprise Linux 6 includes **PostgreSQL 8.4**, Red Hat Enterprise Linux 7 is distributed with **PostgreSQL 9.2**. To migrate your data from a Red Hat Enterprise Linux system version of **PostgreSQL** to the rh-postgresql96 Software Collection, you can either perform a fast upgrade using the **pg_upgrade** tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method. The following procedures are applicable for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7 system versions of **PostgreSQL**.



IMPORTANT

Before migrating your data from a Red Hat Enterprise Linux system version of PostgreSQL to PostgreSQL 9.6, make sure that you back up all your data, including the PostgreSQL database files, which are *by default* located in the **/var/lib/pgsql/data/** directory.

Procedure 5.5. Fast Upgrade Using the pg_upgrade Tool

To perform a fast upgrade of your PostgreSQL server, complete the following steps:

1. Stop the old PostgreSQL server to ensure that the data is not in an inconsistent state. To do so, type the following at a shell prompt as **root**:

   ```
   service postgresql stop
   ```

   To verify that the server is not running, type:

```
service postgresql status
```

2. Verify that the old directory **/var/lib/pgsql/data/** exists:

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

and back up your data.

3. Verify that the new data directory **/var/opt/rh/rh-postgresql96/lib/pgsql/data/** does not exist:

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

If you are running a fresh installation of **PostgreSQL 9.6**, this directory should not be present in your system. If it is, back it up by running the following command as **root**:

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

4. Upgrade the database data for the new server by running the following command as **root**:

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

Alternatively, you can use the **/opt/rh/rh-postgresql96/root/usr/bin/postgresql-setup --upgrade** command.

Note that you can use the **--upgrade-from** option for upgrade from different versions of **PostgreSQL**. The list of possible upgrade scenarios is available using the **--upgrade-ids** option.

It is recommended that you read the resulting **/var/lib/pgsql/upgrade_rh-postgresql96-postgresql.log** log file to find out if any problems occurred during the upgrade.

5. Start the new server as **root**:

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

It is also advised that you run the **analyze_new_cluster.sh** script as follows:

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

6. Optionally, you can configure the PostgreSQL 9.6 server to start automatically at boot time. To disable the old system PostgreSQL server, type the following command as **root**:

```
chkconfig postgresql off
```

To enable the PostgreSQL 9.6 server, type as **root**:

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

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

Procedure 5.6. Performing a Dump and Restore Upgrade

To perform a dump and restore upgrade of your PostgreSQL server, complete the following steps:

1. Ensure that the old PostgreSQL server is running by typing the following at a shell prompt as **root**:

```
service postgresql start
```

2. Dump all data in the PostgreSQL database into a script file. As **root**, type:

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

3. Stop the old server by running the following command as **root**:

```
service postgresql stop
```

4. Initialize the data directory for the new server as **root**:

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

5. Start the new server as **root**:

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

6. Import data from the previously created SQL file:

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

7. Optionally, you can configure the PostgreSQL 9.6 server to start automatically at boot time. To disable the old system PostgreSQL server, type the following command as **root**:

```
chkconfig postgresql off
```

   To enable the PostgreSQL 9.6 server, type as **root**:

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

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

5.7.3. Migrating from the PostgreSQL 9.5 Software Collection to the PostgreSQL 9.6 Software Collection

To migrate your data from the rh-postgresql95 Software Collection to the rh-postgresql96 Collection, you can either perform a fast upgrade using the **pg_upgrade** tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method.

IMPORTANT

Before migrating your data from **PostgreSQL 9.5** to **PostgreSQL 9.6**, make sure that you back up all your data, including the PostgreSQL database files, which are by default located in the **/var/opt/rh/rh-postgresql95/lib/pgsql/data/** directory.

Procedure 5.7. Fast Upgrade Using the pg_upgrade Tool

To perform a fast upgrade of your PostgreSQL server, complete the following steps:

1. Stop the old PostgreSQL server to ensure that the data is not in an inconsistent state. To do so, type the following at a shell prompt as **root**:

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

   To verify that the server is not running, type:

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

2. Verify that the old directory **/var/opt/rh/rh-postgresql95/lib/pgsql/data/** exists:

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

   and back up your data.

3. Verify that the new data directory **/var/opt/rh/rh-postgresql96/lib/pgsql/data/** does not exist:

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

   If you are running a fresh installation of **PostgreSQL 9.6**, this directory should not be present in your system. If it is, back it up by running the following command as **root**:

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

4. Upgrade the database data for the new server by running the following command as **root**:

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

   Alternatively, you can use the **/opt/rh/rh-postgresql96/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql95-postgresql** command.

   Note that you can use the **--upgrade-from** option for upgrading from different versions of **PostgreSQL**. The list of possible upgrade scenarios is available using the **--upgrade-ids** option.

   It is recommended that you read the resulting **/var/lib/pgsql/upgrade_rh-postgresql96-postgresql.log** log file to find out if any problems occurred during the upgrade.

5. Start the new server as **root**:

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

   It is also advised that you run the **analyze_new_cluster.sh** script as follows:

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

6. Optionally, you can configure the PostgreSQL 9.6 server to start automatically at boot time. To disable the old PostgreSQL 9.5 server, type the following command as **root**:

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

To enable the PostgreSQL 9.6 server, type as **root**:

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

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

Procedure 5.8. Performing a Dump and Restore Upgrade

To perform a dump and restore upgrade of your PostgreSQL server, complete the following steps:

1. Ensure that the old PostgreSQL server is running by typing the following at a shell prompt as **root**:

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

2. Dump all data in the PostgreSQL database into a script file. As **root**, type:

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

3. Stop the old server by running the following command as **root**:

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

4. Initialize the data directory for the new server as **root**:

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

5. Start the new server as **root**:

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

6. Import data from the previously created SQL file:

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

7. Optionally, you can configure the PostgreSQL 9.6 server to start automatically at boot time. To disable the old PostgreSQL 9.5 server, type the following command as **root**:

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

To enable the PostgreSQL 9.6 server, type as **root**:

```
-
```

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

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql96/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

If you need to migrate from the postgresql92 Software Collection, refer to Red Hat Software Collections 2.0 Release Notes; the procedure is the same, you just need to adjust the version of the new Collection. The same applies to migration from the rh-postgresql94 Software Collection, which is described in Red Hat Software Collections 2.2 Release Notes.

5.8. MIGRATING TO NGINX 1.16

The root directory for the rh-nginx116 Software Collection is located in **/opt/rh/rh-nginx116/root/**. The error log is stored in **/var/opt/rh/rh-nginx116/log/nginx** by default.

Configuration files are stored in the **/etc/opt/rh/rh-nginx116/nginx/** directory. Configuration files in **nginx 1.16** have the same syntax and largely the same format as previous **nginx** Software Collections.

Configuration files (with a **.conf** extension) in the **/etc/opt/rh/rh-nginx116/nginx/default.d/** directory are included in the default server block configuration for port **80**.

IMPORTANT

Before upgrading from **nginx 1.14** to **nginx 1.16**, back up all your data, including web pages located in the **/opt/rh/nginx114/root/** tree and configuration files located in the **/etc/opt/rh/nginx114/nginx/** tree.

If you have made any specific changes, such as changing configuration files or setting up web applications, in the **/opt/rh/nginx114/root/** tree, replicate those changes in the new **/opt/rh/rh-nginx116/root/** and **/etc/opt/rh/rh-nginx116/nginx/** directories, too.

You can use this procedure to upgrade directly from **nginx 1.8**, **nginx 1.10**, **nginx 1.12**, or **nginx 1.14** to **nginx 1.16**. Use the appropriate paths in this case.

For the official **nginx** documentation, refer to http://nginx.org/en/docs/.

5.9. MIGRATING TO REDIS 5

Redis 3.2, provided by the rh-redis32 Software Collection, is mostly a strict subset of **Redis 4.0**, which is mostly a strict subset of **Redis 5.0**. Therefore, no major issues should occur when upgrading from version 3.2 to version 5.0.

To upgrade a **Redis** Cluster to version 5.0, a mass restart of all the instances is needed.

Compatibility Notes

- The format of RDB files has been changed. **Redis 5** is able to read formats of all the earlier versions, but earlier versions are incapable of reading the **Redis 5** format.
- Since version 4.0, the **Redis** Cluster bus protocol is no longer compatible with **Redis 3.2**.
- For minor non-backward compatible changes, see the upstream release notes for version 4.0 and version 5.0.

CHAPTER 6. ADDITIONAL RESOURCES

This chapter provides references to other relevant sources of information about Red Hat Software Collections 3.4 and Red Hat Enterprise Linux.

6.1. RED HAT PRODUCT DOCUMENTATION

The following documents are directly or indirectly relevant to this book:

- Red Hat Software Collections 3.4 Packaging Guide — The *Packaging Guide* for Red Hat Software Collections explains the concept of Software Collections, documents the **scl** utility, and provides a detailed explanation of how to create a custom Software Collection or extend an existing one.
- Red Hat Developer Toolset 9.0 Release Notes — The *Release Notes* for Red Hat Developer Toolset document known problems, possible issues, changes, and other important information about this Software Collection.
- Red Hat Developer Toolset 9.0 User Guide — The *User Guide* for Red Hat Developer Toolset contains more information about installing and using this Software Collection.
- Using Red Hat Software Collections Container Images — This book provides information on how to use container images based on Red Hat Software Collections. The available container images include applications, daemons, databases, as well as the Red Hat Developer Toolset container images. The images can be run on Red Hat Enterprise Linux 7 Server and Red Hat Enterprise Linux Atomic Host.
- Getting Started with Containers — This guide contains a comprehensive overview of information about building and using container images on Red Hat Enterprise Linux 7 and Red Hat Enterprise Linux Atomic Host.
- Using and Configuring Red Hat Subscription Manager — The *Using and Configuring Red Hat Subscription Manager* book provides detailed information on how to register Red Hat Enterprise Linux systems, manage subscriptions, and view notifications for the registered systems.
- Red Hat Enterprise Linux 6 Deployment Guide — The *Deployment Guide* for Red Hat Enterprise Linux 6 provides relevant information regarding the deployment, configuration, and administration of this system.
- Red Hat Enterprise Linux 7 System Administrator's Guide — The *System Administrator's Guide* for Red Hat Enterprise Linux 7 provides information on deployment, configuration, and administration of this system.

6.2. RED HAT DEVELOPERS

- Red Hat Developer Program — The *Red Hat Developers* community portal.
- Overview of Red Hat Software Collections on Red Hat Developers — The *Red Hat Developers* portal provides a number of tutorials to get you started with developing code using different development technologies. This includes the Node.js, Perl, PHP, Python, and Ruby Software Collections.
- Red Hat Developer Blog — The *Red Hat Developer Blog* contains up-to-date information, best practices, opinion, product and program announcements as well as pointers to sample code and

other resources for those who are designing and developing applications based on Red Hat technologies.

APPENDIX A. REVISION HISTORY

Revision 3.4-2 **Tue Mar 17 2020** **Lenka Špačková**
Added a reference to container-specific upgrading instructions for **PostgreSQL**.

Revision 3.4-1 **Tue Dec 10 2019** **Lenka Špačková**
Release of Red Hat Software Collections 3.4 Release Notes.

Revision 3.4-0 **Thu Nov 07 2019** **Lenka Špačková**
Release of Red Hat Software Collections 3.4 Beta Release Notes.