Red Hat Software Collections 2.x

2.2 Release Notes

Release Notes for Red Hat Software Collections 2.2

Last Updated: 2018-12-18

Red Hat Software Collections 2.x 2.2 Release Notes

Release Notes for Red Hat Software Collections 2.2

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 2.2 Release Notes document the major features and contain important information about known problems in Red Hat Software Collections 2.2. 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 2.2

This chapter serves as an overview of the Red Hat Software Collections 2.2 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 2.2 is be available for Red Hat Enterprise Linux 7; selected new components and 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.

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.

Red Hat Developer Toolset is now 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

Red Hat Software Collections 2.2 provides recent stable versions of the tools listed in Table 1.1, “Red Hat Software Collections 2.2 Components”.

Table 1.1. Red Hat Software Collections 2.2 Components

Component	Software Collection	Description

Component	Software Collection	Description
Red Hat Developer Toolset 4.1	devtoolset-4	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**, **Eclipse** development platform, 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 5.20.1	rh-perl520	A release of Perl, a high-level programming language that is commonly used for system administration utilities and web programming. The rh-perl520 Software Collection provides additional utilities, scripts, and *database connectors for MySQL and PostgreSQL*. Also, it includes the **DateTime** Perl module and the **mod_perl** Apache httpd module, which is supported only with the httpd24 Software Collection.
PHP 5.4.40	php54	A release of PHP with **PEAR 1.9.4** and a number of additional extensions. PHP 5.4 provides a number of *language and interface improvements*. The **memcache** and **Zend OPcache** extensions are also included.
PHP 5.5.21	php55	A release of PHP with **PEAR 1.9.4** and enhanced language features including *better exception handling, generators,* and **Zend OPcache**. The **memcache** and **mongodb** extensions are also included.
PHP 5.6.5	rh-php56	A release of PHP with **PEAR 1.9.5** and enhanced language features including *constant expressions, variadic functions, arguments unpacking, and the interactive debugger*. The **memcache**, **mongo**, and **XDebug** extensions are also included.
Python 2.7.8	python27	A release of Python 2.7 with a number of additional utilities. This Python version provides various new features and enhancements, including a new ordered dictionary type, faster I/O operations, and improved forward compatibility with Python 3. The python27 Software Collections contains the *Python 2.7.8 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**.

Component	Software Collection	Description
Python 3.4.2	rh-python34	A release of Python 3 with a number of additional utilities. This Software Collection gives developers on Red Hat Enterprise Linux access to Python 3 and allows them to benefit from various advantages and new features of this version. The rh-python34 Software Collection contains *Python 3.4.2 interpreter*, a set of extension libraries useful for programming web applications and **mod_wsgi** (only supported with the httpd24 Software Collection), PostgreSQL database connector, and **numpy** and **scipy**.
Python 3.5.1[a]	rh-python35	The rh-python35 Software Collection contains *Python 3.5.1 interpreter*, a set of extension libraries useful for programming web applications and **mod_wsgi** (only supported with the httpd24 Software Collection), PostgreSQL database connector, and **numpy** and **scipy**.
Ruby 2.2.2	rh-ruby22	A release of Ruby 2.2. This version provides substantial *performance and reliability improvements, including incremental and symbol garbage collection* and many others, while maintaining source level backward compatibility with Ruby 2.0.0 and Ruby 1.9.3.
Ruby 2.3.0[a]	rh-ruby23	A release of Ruby 2.3. This version introduces *a command-line option to freeze all string literals in the source files, a safe navigation operator, and multiple performance enhancements*, while maintaining source level backward compatibility with Ruby 2.2.2, Ruby 2.0.0, and Ruby 1.9.3.
Ruby on Rails 4.1.5	rh-ror41	A release of Ruby on Rails 4.1, a web application development framework written in the Ruby language. This version provides a number of new features including *Spring application preloader, config/secrets.yml, Action Pack variants, and Action Mailer previews*. This Software Collection is supported together with the rh-ruby22 Collection.
Ruby on Rails 4.2.6[a]	rh-ror42	A release of Ruby on Rails 4.2, the latest version of the web application framework written in the Ruby language. Highlights in this release include *Active Job, asynchronous mails, Adequate Record, Web Console, and foreign key support*. This Software Collection is supported together with the rh-ruby23 and rh-nodejs4 Collections.
MariaDB 10.0.25	rh-mariadb100	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 the PAM authentication plugin to MariaDB.

Component	Software Collection	Description
MariaDB 10.1.14	rh-mariadb101	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 the Galera Cluster support.
MongoDB 2.6.9	rh-mongodb26	A release of MongoDB, a cross-platform *document-oriented database system classified as a NoSQL database*. This Software Collection includes the mongo-java-driver package version 2.14.1.
MongoDB 3.2.6[a]	rh-mongodb32	A release of MongoDB, a cross-platform *document-oriented database system classified as a NoSQL database*. This Software Collection includes the mongo-java-driver package version 3.2.1.
MongoDB 3.0.11 upgrade collection[a]	rh-mongodb30upg	A limited version of **MongoDB 3.0** is available to provide an upgrade path from **MongoDB 2.6** to **MongoDB 3.2** for customers with existing MongoDB databases.
MySQL 5.6.30	rh-mysql56	A release of MySQL, which provides a number of new features and enhancements, including improved performance.
PostgreSQL 9.4.6	rh-postgresql94	A release of PostgreSQL, which provides a new data type to store JSON more efficiently and a new SQL command for changing configuration files, reduces lock strength for some commands, allows materialized views without blocking concurrent reads, supports logical decoding of WAL data to allow stream changes in a customizable format and enable background worker processes to be dynamically registered, started, and terminated.
PostgreSQL 9.5.2	rh-postgresql95	A release of PostgreSQL, which provides a number of enhancements, including *row-level security control,* introduces replication progress tracking, improves handling of large tables with high number of columns, and improves performance for sorting and multi-CPU machines.
Node.js 0.10	nodejs010	A release of Node.js with **npm 1.4.28** and *support for the SPDY protocol version 3.1*. This Software Collection gives users of Red Hat Enterprise Linux access to this programming platform.
Node.js 4.4.2	rh-nodejs4	A release of Node.js with **npm 2.15.1** and *support for the SPDY protocol version 3.1*. This Software Collection gives users of Red Hat Enterprise Linux access to this programming platform.

Component	Software Collection	Description
rh-nginx 1.8.0	rh-nginx18	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 *back-end SSL certificate verification, logging to syslog, thread pools support for offloading I/O requests, or hash load balancing method*.
Apache httpd 2.4.18	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** module is also included.
Varnish Cache 4.0.3	rh-varnish4	A release of Varnish Cache, a *high-performance HTTP reverse proxy*. **Varnish Cache** stores files or fragments of files in memory that are used to reduce the response time and network bandwidth consumption on future equivalent requests.
Thermostat 1.4.4	thermostat1	A release of Thermostat, a monitoring and instrumentation tool for the *OpenJDK HotSpot JVM*, with support for monitoring *multiple JVM instances*. This Software Collection depends on the rh-mongodb26 and rh-java-common components.
DevAssistant 0.9.3	devassist09	A release of DevAssistant, a tool designed to assist developers with *creating and setting up basic projects* in various programming languages, installing dependencies, setting up a development environment, and working with source control. DevAssistant supports the C, C++, Java, and Python programming languages but it is able to support working with any other language, framework, or tool due to its modular architecture.
Maven 3.0.5	maven30	A release of Maven, a *software project management and comprehension tool* used primarily for Java projects. 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.
Maven 3.3.9[a]	rh-maven33	A release of Maven, a *software project management and comprehension tool* used primarily for Java projects. This version provides various enhancements, for example, *improved core extension mechanism*.

Component	Software Collection	Description
Passenger 4.0.50	rh-passenger40	A release of Phusion Passenger, a web and application server, designed to be fast, robust, and lightweight. It supports **Ruby** using the ruby193, ruby200, or rh-ruby22 Software Collections together with **Ruby on Rails** using the ror40 or rh-ror41 Collections. It can also be used with **nginx 1.6** from the nginx16 Software Collection and with **Apache httpd** from the httpd24 Software Collection.
Common Java Packages 1.1	rh-java-common	This Software Collection provides *common Java libraries and tools* used by other collections. The rh-java-common Software Collection is required by the devtoolset-4, devtoolset-3, rh-maven33, maven30, rh-mongodb32, rh-mongodb26, and thermostat1 components and it is not supposed to be installed directly by users.
V8 3.14.5.10	v8314	This Software Collection provides the *V8 JavaScript engine* and is supported only as a dependency for the mongodb24, rh-mongodb26, rh-mongodb30upg, ruby193, ror40, rh-ror41, and nodejs010 Software Collections.
[a] This Software Collection is available only for Red Hat Enterprise Linux 7		

Previously released Software Collections remain available in the same distribution channels. For example, the git19 Software Collection, which provides **Git 1.9.4**, has not been updated since Red Hat Software Collections 1.2 but still can be installed along with the Red Hat Software Collections 2.2 components or other previously released components.

All currently available Software Collections are listed in the Table 1.2, “All Available Software Collections”. For detailed information regarding components that have not been updated since Red Hat Software Collections 1, refer to the Red Hat Software Collections 1.2 Release Notes. See the Red Hat Software Collections Product Life Cycle document for information on the length of support for individual components.

Table 1.2. All Available Software Collections

Component	Software Collection	Availability
Components New in Red Hat Software Collections 2.2		
MariaDB 10.1.14	rh-mariadb101	RHEL6, RHEL7
Maven 3.3.9	rh-maven33	RHEL7
MongoDB 3.0.11 upgrade collection	rh-mongodb30upg	RHEL7
MongoDB 3.2.6	rh-mongodb32	RHEL7
Node.js 4.4.2	rh-nodejs4	RHEL6, RHEL7

Component	Software Collection	Availability
Components New in Red Hat Software Collections 2.2		
PostgreSQL 9.5.2	rh-postgresql95	RHEL6, RHEL7
Python 3.5.1	rh-python35	RHEL7
Ruby on Rails 4.2.6	rh-ror42	RHEL7
Ruby 2.3.0	rh-ruby23	RHEL7
Components Updated in Red Hat Software Collections 2.2		
Red Hat Developer Toolset 4.1	devtoolset-4	RHEL6, RHEL7
Apache httpd 2.4.18	httpd24	RHEL6, RHEL7
Python 2.7.8	python27	RHEL6, RHEL7
Common Java Packages 1.1	rh-java-common	RHEL6, RHEL7
MongoDB 2.6.9	rh-mongodb26	RHEL6, RHEL7
Thermostat 1.4.4	thermostat1	RHEL6, RHEL7
Components Not Updated since Red Hat Software Collections 2.1		
Varnish Cache 4.0.3	rh-varnish4	RHEL6, RHEL7
nginx 1.8.0	rh-nginx18	RHEL6, RHEL7
Node.js 0.10	nodejs010	RHEL6, RHEL7
Maven 3.0.5	maven30	RHEL6, RHEL7
V8 3.14.5.10	v8314	RHEL6, RHEL7
Components Not Updated since Red Hat Software Collections 2.0		
Perl 5.20.1	rh-perl520	RHEL6, RHEL7
PHP 5.6.5	rh-php56	RHEL6, RHEL7

Components Not Updated since Red Hat Software Collections 2.0		
Python 3.4.2	rh-python34	RHEL6, RHEL7
Ruby 2.2.2	rh-ruby22	RHEL6, RHEL7
Ruby on Rails 4.1.5	rh-ror41	RHEL6, RHEL7
MariaDB 10.0.25	rh-mariadb100	RHEL6, RHEL7
MySQL 5.6.30	rh-mysql56	RHEL6, RHEL7
PostgreSQL 9.4.6	rh-postgresql94	RHEL6, RHEL7
Passenger 4.0.50	rh-passenger40	RHEL6, RHEL7
PHP 5.4.40	php54	RHEL6, RHEL7
PHP 5.5.21	php55	RHEL6, RHEL7
nginx 1.6.2	nginx16	RHEL6, RHEL7
DevAssistant 0.9.3	devassist09	RHEL6, RHEL7

Components Not Updated since Red Hat Software Collections 1		
Git 1.9.4	git19	RHEL6, RHEL7
Perl 5.16.3	perl516	RHEL6, RHEL7
Python 3.3.2	python33	RHEL6, RHEL7
Ruby 1.9.3	ruby193	RHEL6, RHEL7
Ruby 2.0.0	ruby200	RHEL6, RHEL7
Ruby on Rails 4.0.2	ror40	RHEL6, RHEL7
MariaDB 5.5.44	mariadb55	RHEL6, RHEL7
MongoDB 2.4.9	mongodb24	RHEL6, RHEL7
MySQL 5.5.45	mysql55	RHEL6, RHEL7
PostgreSQL 9.2.15	postgresql92	RHEL6, RHEL7

RHEL6 — Red Hat Enterprise Linux 6

RHEL7 — Red Hat Enterprise Linux 7

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.

1.3. CHANGES IN RED HAT SOFTWARE COLLECTIONS 2.2

1.3.1. Overview

New Software Collections

Red Hat Software Collections 2.2 adds these new Software Collections:

- rh-python35 — see Section 1.3.3, “Changes in Python”
- rh-ruby23 — see Section 1.3.4, “Changes in Ruby”
- rh-ror42 — see Section 1.3.5, “Changes in Ruby on Rails”
- rh-mariadb101 — see Section 1.3.6, “Changes in MariaDB”
- rh-mongodb32 — see Section 1.3.7, “Changes in MongoDB”
- rh-mongodb30upg — see Section 1.3.7, “Changes in MongoDB”
- rh-postgresql95 — see Section 1.3.8, “Changes in PostgreSQL”
- rh-nodejs4 — see Section 1.3.9, “Changes in Node.js”
- rh-maven33 — see Section 1.3.12, “Changes in Maven”

Additionally, Red Hat Software Collections 2.2 includes the following new packages, which are not Software Collections but dependencies of other components:

- gperftools — Performance Tools is a collection of performance analysis tools, including a high-performance multi-threaded malloc() implementation that works particularly well with threads and Standard Template Library (STL), a thread-friendly heap-checker, a heap profiler, and a cpu-profiler. This is a metapackage which pulls in all of the **gperftools** (and **pprof**) binaries, libraries, and development headers. Note that this package is a dependency of the rh-mongodb32 and rh-mongodb30upg Software Collections, and it is available in the Optional channel.
- libunwind — The libunwind packages contain a C API to determine the call chain of a program. This API is necessary for compatibility with Performance Tools (**gperftools**). This package is a dependency of the rh-mongodb32 and rh-mongodb30upg Software Collections, and it is available in the Optional channel.

Updated Software Collections

The following components have been updated in Red Hat Software Collections 2.2:

- devtoolset-4 — see Section 1.3.2, “Changes in Red Hat Developer Toolset”
- python27 — see Section 1.3.3, “Changes in Python”

- rh-mongodb26 — see Section 1.3.7, “Changes in MongoDB”
- httpd24 — see Section 1.3.10, “Changes in Apache httpd”
- thermostat1 — see Section 1.3.11, “Changes in Thermostat”
- rh-java-common — see Section 1.3.13, “Changes in the Common Java Packages”

Red Hat Software Collections Container Images

Container images based on Red Hat Software Collections 2.2 are now available, in addition to the previously released Red Hat Software Collections 2.0 container images.

The available container images 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 the Knowledgebase article at https://access.redhat.com/articles/1752723. For details about images based on the Red Hat Developer Toolset components, refer to the Red Hat Developer Toolset User Guide.

The following container images are new in Red Hat Software Collections 2.2:

- rhscl/devtoolset-4-perftools-rhel7
- rhscl/mariadb-101-rhel7
- rhscl/mongodb-32-rhel7
- rhscl/nginx-18-rhel7
- rhscl/nodejs-4-rhel7
- rhscl/postgresql-95-rhel7
- rhscl/python-35-rhel7
- rhscl/ror-42-rhel7
- rhscl/ruby-23-rhel7
- rhscl/thermostat-1-agent-rhel7
- rhscl/varnish-4-rhel7

The following container images have been updated in Red Hat Software Collections 2.2:

- rhscl/devtoolset-4-toolchain-rhel7
- rhscl/httpd-24-rhel7
- rhscl/python-27-rhel7

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

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

- rhscl/nginx-16-rhel7
- rhscl/passenger-40-rhel7
- rhscl/perl-520-rhel7
- rhscl/php-56-rhel7
- rhscl/postgresql-94-rhel7
- rhscl/python-34-rhel7
- rhscl/ror-41-rhel7
- rhscl/ruby-22-rhel7
- rhscl/s2i-base-rhel7

1.3.2. Changes in Red Hat Developer Toolset

The following components have been upgraded in Red Hat Developer Toolset 4.1 compared to the previous release of Red Hat Software Collections:

- **Eclipse** to version 4.5.2
- **GCC** to version 5.3.1
- **binutils** to version 2.25.1
- **elfutils** to version 0.166
- **GDB** to version 7.11
- **SystemTap** to version 2.9
- **Valgrind** to version 3.11.0
- **Dyninst** to version 9.1.0

For detailed information on changes in Red Hat Developer Toolset 4.1, see Red Hat Developer Toolset User Guide.

1.3.3. Changes in Python

Python 2

The python27 Software Collection has been updated to a later version, which provides a number of bug fixes and enhancements over the previous version. Among others:

- The python27-PyYAML package has been added, which contains a Python YAML module. PyYAML is a YAML parser and emitter for Python; it is applicable for a broad range of tasks from complex configuration files to object serialization and persistance.
- Network security enhancements, described in the Python Enhancent Proposal 466, have been backported to the Python standard library. The security enhancements include, for example, new features in the `ssl` module, such as support for Server Name Indication (SNI) as well as support for new TLSv1.x protocols, new hash algorithms in the `hashlib` module, and much more.

- The **cert-verification.cfg** has been added. It contains new options that allow users to globally enable or disable SSL/TLS certificate verification in the HTTP clients (such as urllib, httplib, or xmlrpclib) of the Python standard library. The options are described in the Python Enhancement Proposal 493.
- The python27-python-pip package has been upgraded to version 7.1.0.
- The python27-python-virtualenv package has been upgraded to verion 13.1.0.
- The python27-python-pymongo package has been upgraded to version 3.2.1.

Python 3

The new rh-python35 Software Collection, available for Red Hat Enterprise Linux 7, includes **Python 3.5.1**. This version provides a number of security fixes, bug fixes, and enhancements. Among others:

- Security improvements
  - SSLv3 is now disabled throughout the standard library
  - HTTP cookie parsing is now stricter, to protect against potential injection attacks
- New syntax features
  - The Python Enhancement Proposal 492 has been implemented, which greatly improves support for asynchronous programming in Python by adding awaitable objects, coroutine functions, asynchronous iteration, and asynchronous context managers
  - A new @ infix operator has been added for matrix multiplication. Matrix multiplication is a notably common operation in many fields of mathematics, science, engineering, and the addition of @ allows writing cleaner code with specialized libraries like **numpy**. This new operator is described in the Python Enhancement Proposal 465
- CPython implementation improvements
  - The **.pyo** files are no longer used and have been replaced by a more flexible scheme that includes the optimization level explicitly in the **.pyc** file names; see the Python Enhancement Proposal 488 for details.

For detailed changes, see the Python 3.5.1 documentation and Python 3.5 release highlights.

1.3.4. Changes in Ruby

The new rh-ruby23 Software Collection, available for Red Hat Enterprise Linux 7, contains **Ruby 2.3.0**. This version provides several new features, including:

- Safe navigation operator, also known as a lonely operator
- Frozen String Literal Pragma
- A new command-line option to freeze all string literals in the source files
- A new magic comment
- The **did_you_mean** gem shows the candidates on the NameError and NoMethodError exceptions to ease debugging. The **did_you_mean** gem is bundled.

For detailed changes, see the upstream documentation.

Ruby 2.3 is backward compatible with **Ruby 2.2.2**, **Ruby 2.0.0**, and **Ruby 1.9.3**. The ruby193, rh-ruby22, and ruby200 Software Collections are still available. For information about length of support for these components, refer to the Red Hat Software Collections Product Life Cycle document.

Note that the safe levels **$SAFE=2** and **$SAFE=3** are obsolete.

1.3.5. Changes in Ruby on Rails

Ruby on Rails 4.2.6, shipped in the new rh-ror42 Software Collection, provides the following major new features:

- Active Job
- Asynchronous mails
- Adequate Record
- Web Console
- Foreign key support
- GlobalID serialization

For detailed changes, see the upstream documentation.

The rh-ror42 Software Collection is supported together with the rh-ruby23 and rh-nodejs4 Collections and is available only for Red Hat Enterprise Linux 7.

1.3.6. Changes in MariaDB

The new rh-mariadb101 Software Collection includes **MariaDB 10.1.14**. It provides a number of bug fixes and new features, for example:

- Galera Cluster, a synchronous multi-master cluster, which is a standard part of **MariaDB 10.1**; see the Knowledgebase article about setting up Galera Cluster with the rh-mariadb101 Software Collection
- Table, tablespace, and log encryption
- InnoDB and XtraDB page compression
- Page compression for FusionIO

For detailed changes, refer to the upstream documentation.

For information on how to migrate to **MariaDB 10.1**, see Section 5.1, “Migrating to MariaDB 10.1”.

1.3.7. Changes in MongoDB

MongoDB 2.6

The rh-mongodb26 Software Collection contains the upgraded rh-mongodb26-mongo-java-driver package to the latest compatible version 2.14.1, which provides a number of bug fixes and includes support for functions from later versions of **MongoDB**.

MongoDB 3.0 Upgrade Collection

The new rh-mongodb30upg Collection contains a limited version of **MongoDB 3.0.11** to provide an

upgrade path from **MongoDB 2.6** to **MongoDB 3.2** for customers with existing MongoDB databases. This Software Collection includes neither MongoDB Java Driver nor MongoDB Tools and is available only for Red Hat Enterprise Linux 7.

See the Release Notes for MongoDB 3.0 for more details.

MongoDB 3.2

The new rh-mongodb32 Software Collection, available for Red Hat Enterprise Linux 7, includes **MongoDB 3.2.6**. It provides a number of bug fixes and new features, for example:

- New storage engine **WiredTiger**, which is now default; it provides better performance and document-level locking and compression
- Simplified data governance with document validation
- Support for partial indexes
- New JavaScript engine **SpiderMonkey**, which replaces **V8**
- Increased number of replica set members
- A new SCRAM-SHA-1 challenge-response user authentication mechanism.

Note that the mongo-java-driver package from the rh-mongodb32 Software Collection is incompatible with mongo-java-driver included in the previously released mongodb24 and rh-mongodb26 Collections. See details in the upstream documentation.

For detailed changes, refer to the Release Notes for MongoDB 3.2.

For information on how to migrate to **MongoDB 3.2**, see Section 5.2, “Migrating to MongoDB 3.2”.

1.3.8. Changes in PostgreSQL

The new rh-postgresql95 Software Collection includes **PostgreSQL 9.5.2**, which provides a number of enhancements. For example:

- INSERT operations that would generate constraint conflicts are now allowed to be turned into UPDATE operations or ignored (UPSERT operations)
- GROUP BY analysis features have been added: GROUPING SETS, CUBE, and ROLLUP
- Row-level security control has been added
- Mechanisms for tracking the progress of replication have been created, including methods for identifying the origin of individual changes during logical replication
- Block Range Indexes (BRIN) have been added
- Substantial performance improvements for sorting
- Substantial performance improvements for multi-CPU machines
- The rh-postgresql95-postgresql-static package includes the **`libpgport.a`** and **`libpgcommon.a`** libraries. These libraries are required by certain third-party packages to build, for example, by the pg_bulkload and pglogical packages.

For details, see the upstream documentation.

For information on how to migrate to **PostgreSQL 9.5**, see Section 5.4, “Migrating to PostgreSQL 9.5”.

1.3.9. Changes in Node.js

The new rh-nodejs4 Software Collection, includes **Node.js 4.4.2**. This version includes **npm 2.15.1** and provides numerous security and bug fixes.

For details, see the upstream release notes and upstream documentation.

Note that the rh-nodejs4 Software Collection has been made available also for Red Hat Enterprise Linux 6 through an asynchronous update.

1.3.10. Changes in Apache httpd

The httpd24 Software Collection has been upgraded to version 2.4.18, which provides numerous bug fixes and enhancements. Notable changes are as follows:

Experimental support for the HTTP/2.0 protocol has been added, provided by the new **`mod_http2`** module. HTTP/2 support can be enabled using the new **`Protocols`** directive. Upgrading to HTTP/2.0 over TLS is supported using the **`next protocol negotiation (NPN)`** TLS extension.

Version 7.47.1 of the **curl** command-line tool, with HTTP/2.0 support enabled, has been added to the Software Collection to facilitate testing.

1.3.11. Changes in Thermostat

The thermostat1 Software Collection has been upgraded to version 1.4.4, which provides numerous enhancements and bug fixes over the previous version.

- New features:
  - A new first-run setup wizard to automatically set up credentials and related information
  - The number of threads is now always recorded
  - A new tree-map visualization for viewing the heap
  - Identification of which garbage collector is being used
  - Custom prompt for the **Thermostat** shell
  - Tab completion in the **Thermostat** shell
  - Progress display in the UI for long-running operations
  - Graphical representation for Thread deadlock
  - A new manual page for **Thermostat**
- Bug fixes:
  - Various actions are now correctly disabled when the JVM they target is no longer alive
- Security-related changes:
  - Remote Method Invocation (RMI) is no longer used in the agent

    - The command channel now runs as a separate process isolating network-facing code from everything else

- Other improvements:

    - New helper classes in storage API for getting the oldest or newest POJOs

    - Efficiency improvements, such as implementation of fast-tail paths and querying less data when possible.

See the change log for details.

1.3.12. Changes in Maven

The new rh-maven33 Software Collection, available for Red Hat Enterprise Linux 7, includes **Maven 3.3.9**. This version provides numerous bug fixes and several new features. Among others:

- **Maven** now requires JDK version 1.7 or later to run
- The core extension mechanism has been improved; its usage is now simpler
- New means of passing options to **Maven** and configuring JVM on per-project basis have been added
- It is now possible to exclude all transitive dependencies using wildcards
- Maven reactor is now pluggable.

For detailed changes, see the upstream release notes for versions 3.3.1, 3.2.1, and 3.1.1.

1.3.13. Changes in the Common Java Packages

The rh-java-common Software Collection has been updated and extended to comply with the changes in the dependent components.

1.4. COMPATIBILITY INFORMATION

Red Hat Software Collections 2.2 is available for all supported releases of Red Hat Enterprise Linux 7 on AMD64 and Intel 64 architectures. 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

`rh-nodejs4` component, BZ#1334856

Multiple packages from the rh-nodej4 Software Collection for Red Hat Enterprise Linux 7 are missing license information or contain incorrect license information.

`python27` component, BZ#1330489

The python27-python-pymongo package has been updated to version 3.2.1 in Red Hat Software Collections 2.2. 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.

`httpd24` component, BZ#1327548

The **`mod_ssl`** module does not support the Application-Layer Protocol Negotiation (ALPN) protocol on Red Hat Enterprise Linux. Consequently, clients that support upgrading TLS connections to HTTP/2.0 only using ALPN are limited to HTTP/1.1 support. Clients that support the NPN protocol in addition to ALPN (such as Mozilla Firefox) are able to successfully upgrade to HTTP/2.0.

`httpd24` component, BZ#1329639

On Red Hat Enterprise Linux 7, running the **`service httpd24-httpd configtest`** command fails with an error message. To work around this problem, run the following command:

```
scl enable httpd24 'apachectl configtest'
```

`rh-maven33` component

When the user has installed both the Red Hat Enterprise Linux system version of maven-local package and the rh-maven33-maven-local package, **XMvn**, a tool used for building Java RPM packages, run from the rh-maven33 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.

`rh-nodejs4` component, BZ#1316626

The **`/opt/rh/rh-nodejs4/root/usr/share/licenses/`** directory is not owned by any package. Consequently, when the rh-nodejs4 collection is uninstalled, this directory is not removed. To work around this problem, remove the directory manually after uninstalling rh-nodejs4.

`rh-mysql56`, `rh-mariadb100`, `rh-mariadb101` components, BZ#1194611

The rh-mysql56-mysql-server, rh-mariadb100-mariadb-server, and rh-mariadb101-mariadb-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.

`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.

`rh-passenger40` component, BZ#1196555

When Passenger from the rh-passenger40 Software Collection is run as a module for **httpd**, the functionality is restricted by SELinux policy. To work around this problem, switch the passenger domain to permissive mode by running the following command as **`root`**:

```
semanage permissive -a passenger_t
```

Standalone server and **nginx** integration are not affected by this issue.

`mongodb24` component

The mongodb24 Software Collection from Red Hat Software Collections 1.2 cannot be rebuilt with the rh-java-common and maven30 Software Collections shipped with Red Hat Software Collections 2.2. Additionally, the mongodb24-build and mongodb24-scldevel packages cannot be installed with Red Hat Software Collections 2.2 due to unsatisfied requires on the maven30-javapackages-tools and maven30-maven-local packages. When the mongodb24-scldevel package is installed, broken dependencies are reported and the **`yum --skip-broken`** command skips too many packages. Users are advised to update to the rh-mongodb26 Software Collection.

`perl` component

When the user tries to use the **`mod_perl`** module from both the rh-perl520 and perl516 Software Collections, a conflict in the **`/opt/rh/httpd24/root/usr/lib64/httpd/modules/mod_perl.so`** file occurs. As a consequence, it is impossible to use **`mod_perl`** from more than one **Perl** Software Collection.

`nodejs010` component

Shared libraries provided by the nodejs010 Software Collection, namely **`libcares`**, **`libhttp_parser`**, and **`libuv`**, are not properly prefixed with the Collection name. As a consequence, conflicts with the corresponding system libraries might occur.

`nodejs-hawk` component

The nodejs-hawk package uses an implementation of the SHA-1 and SHA-256 algorithms adopted from the CryptoJS project. In this release, the client-side JavaScript is obfuscated. The future fix will involve using crypto features directly from the CryptoJS library.

`postgresql` component

The postgresql92, rh-postgresql94, and rh-postgresql95 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`, `php55`, `rh-php56`, `python`, `ruby`, `ror`, `thermostat`, and `v8314` components, BZ#1072319

When uninstalling the httpd24, mariadb55, rh-mariadb100, mongodb24, rh-mongodb26, mysql55, rh-mysql56, nodejs010, perl516, rh-perl520, php55, rh-php56, python27, python33, rh-python34, ruby193, ruby200, rh-ruby22, ror40, rh-ror41, thermostat1, or v8314 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`, `postgresql`, `mongodb` components

Red Hat Software Collections 2.2 contains the **MySQL 5.6**, **MariaDB 10.0**, **MariaDB 10.1**, **PostgreSQL 9.4**, **PostgreSQL 9.5**, **MongoDB 2.6**, and **MongoDB 3.2** 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/`**.

`httpd` 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.

`python27` component

In Red Hat Enterprise Linux 7, when the user tries to install the python27-python-debuginfo package, the **`/usr/src/debug/Python-2.7.5/Modules/socketmodule.c`** file conflicts with the corresponding file from the python-debuginfo package installed on the core system. Consequently, installation of the python27-python-debuginfo fails. To work around this problem, uninstall the python-debuginfo package and then install the python27-python-debuginfo package.

`devassist` component

When the user tries to rebuild the devassist09-PyYAML package on Red Hat Enterprise Linux 6, the build fails due to a soft dependency, if the Pyrex or Cython programming languages are detected. To work around this problem, make sure the pyrex or cython packages are not installed on your system.

Other Notes

`rh-ruby22`, `rh-python34`, `rh-php56` components

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

- Ruby gems cannot be installed while the rh-ruby22 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, the following error message is displayed:

```
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.

`thermostat` component

The **thermostat1-thermostat-tomcat start** command, which starts the Thermostat web storage endpoint, can be used only on Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7.0. On Red Hat Enterprise Linux 7.1 and later versions, use **service tomcat@thermostat start** instead.

`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-python34 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
```

php54 component

Note that **Alternative PHP Cache (APC)** in Red Hat Software Collections is provided only for user data cache. For opcode cache, **Zend OPcache** is provided.

`python` component

When the user tries to install more than one scldevel package from the python27, python33, and rh-python34 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 php54, php55, and rh-php56 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 ruby193, ruby200, and rh-ruby22 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 perl516 and rh-perl520 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 nginx16 and rh-nginx18 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`**).

`nodejs` component

When installing the nodejs010 Software Collection, nodejs010 installs **GCC** in the base Red Hat Enterprise Linux system as a dependency, unless the gcc packages are already installed.

CHAPTER 2. INSTALLATION

This chapter describes in detail how to get access to the content set, install Red Hat Software Collections 2.2 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. Depending on the subscription management service with which you registered your Red Hat Enterprise Linux system, you can either enable Red Hat Software Collections by using Red Hat Subscription Management, or by using RHN Classic. For detailed instructions on how to enable Red Hat Software Collections using RHN Classic or Red Hat Subscription Management, see the respective section below. For information on how to register your system with one of these subscription management services, see Using and Configuring Red Hat Subscription Manager.

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 2.1.3, “Packages from the Optional Channel”, cannot be installed from the ISO image.

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.

2.1.2. Using RHN Classic

If your system is registered with RHN Classic, complete the following steps to subscribe to Red Hat Software Collections:

1. Display a list of all channels that are available to you and determine the exact name of the Red Hat Software Collections channel. To do so, type the following at a shell prompt as **root**:

```
rhn-channel --available-channels
```

The name of the channel depends on the specific version of Red Hat Enterprise Linux you are using and is in the following format, where *variant* is the Red Hat Enterprise Linux system variant (**server** or **workstation**):

```
rhel-x86_64-variant-6-rhscl-1

rhel-x86_64-server-6.5.z-rhscl-1
rhel-x86_64-server-6.6.z-rhscl-1

rhel-x86_64-variant-7-rhscl-1

rhel-x86_64-server-7.1.eus-rhscl-1
```

Red Hat Enterprise Linux 7 channels are accessible only through Red Hat Satellite instances.

NOTE

Red Hat Software Collections 2.x are distributed in the same channels as Red Hat Software Collections 1.x.

2. Subscribe the system to the Red Hat Software Collections channel by running the following command as **root**:

```
rhn-channel --add --channel=channel_name
```

Replace *channel_name* with the name you determined in the previous step.

3. Verify the list of channels you are subscribed to. As **root**, type:

```
rhn-channel --list
```

When the system is subscribed, 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 with RHN Classic, see Using and Configuring Red Hat Subscription Manager.

2.1.3. Packages from the Optional Channel

Some of the Red Hat Software Collections 2.2 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 articles on Red Hat Customer Portal: https://access.redhat.com/solutions/392003 for Red Hat Subscription Management or https://access.redhat.com/solutions/70019 if your system is registered with RHN Classic.

Packages from Software Collections for Red Hat Enterprise Linux 6 that require the **Optional** channel to be enabled are listed in the following table.

Table 2.1. Packages Requiring Enabling of the Optional Channel in Red Hat Enterprise Linux 6

Package from a Software Collection	Required Package from the Optional Channel
devtoolset-4-dyninst-testsuite	glibc-static
devtoolset-4-elfutils-devel	xz-devel
devtoolset-4-gcc-plugin-devel	mpfr
devtoolset-4-libgccjit	mpfr
git19-git-all	cvsps, subversion-perl
git19-git-cvs	cvsps

Package from a Software Collection	Required Package from the Optional Channel
git19-git-svn	perl-YAML, subversion-perl
git19-perl-Git-SVN	perl-YAML, subversion-perl
mariadb55-mariadb-bench	perl-GD
mysql55-mysql-bench	perl-GD
php54-php-imap	libc-client
php54-php-recode	recode
php55-php-imap	libc-client
php55-php-recode	recode
rh-mariadb100-mariadb-bench	perl-GD
rh-mariadb101-boost-devel	libicu-devel
rh-mariadb101-boost-examples	libicu-devel
rh-mariadb101-boost-static	libicu-devel
rh-mariadb101-mariadb-bench	perl-GD
rh-mysql56-mysql-bench	perl-GD
rh-php56-php-imap	libc-client
rh-php56-php-recode	recode

Software Collections packages that require the **`Optional`** channel in Red Hat Enterprise Linux 7 are listed in the table below.

Table 2.2. Packages Requiring Enabling of the Optional Channel in Red Hat Enterprise Linux 7

Package from a Software Collection	Required Package from the Optional Channel
devtoolset-4-dyninst-testsuite	glibc-static
devtoolset-4-gcc-plugin-devel	libmpc-devel

Package from a Software Collection	Required Package from the Optional Channel
git19-git-all	cvsps, subversion-perl
git19-git-cvs	cvsps
git19-git-svn	subversion-perl
git19-perl-Git-SVN	subversion-perl
httpd24-mod_ldap	apr-util-ldap
nodejs010	c-ares-devel
nodejs010-node-gyp	c-ares-devel
nodejs010-nodejs-columnify	c-ares-devel
nodejs010-nodejs-devel	c-ares-devel
nodejs010-nodejs-npmconf	c-ares-devel
nodejs010-nodejs-wcwidth	c-ares-devel
nodejs010-npm	c-ares-devel
rh-mongodb30upg	gperftools, gperftools-devel, gperftools-libs, libunwind, pprof
rh-mongodb32	gperftools, gperftools-devel, gperftools-libs, libunwind, pprof
rh-perl520-perl-Pod-Perldoc	groff

Note that packages from the **`Optional`** channel are not supported. For details, see the Knowledgebase article https://access.redhat.com/articles/1150793.

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 2.2 requires the removal of any earlier pre-release versions, including Beta releases. If you have installed any previous version of Red Hat Software Collections 2.2, 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 2.2 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 php54 rh-mariadb100
```

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-perl520-perl-CPAN and rh-perl520-perl-Archive-Tar, type:

```
~]# yum install rh-perl520-perl-CPAN rh-perl520-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-ruby22-ruby package, type:

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

Note that in order to use this command, 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”. If your system is registered with RHN Classic, subscribe the system to the **rhel-x86_64-*variant*-6-rhscl-1-debuginfo** or **rhel-x86_64-*variant*-7-rhscl-1-debuginfo** channel as described in Section 2.1.2, “Using RHN Classic”. 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 2.2, 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 perl516 Software Collection, type:

```
~]$ scl enable perl516 '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 2.2 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 postgresql92 Software Collections as default, type:

```
~]$ scl enable python27 postgresql92 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 postgresql92
```

For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 2.2 Components”.

3.1.3. Running a System Service from a Software Collection

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 postgresql92 Software Collection and enable it in runlevels 2, 3, 4, and 5, type as **root**:

```
~]# service postgresql92-postgresql start
Starting postgresql92-postgresql service:                  [  OK  ]
~]# chkconfig postgresql92-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 2.2 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 mariadb55, type:

```
~]$ scl enable mariadb55 "man mariadb55"
```

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. DOCKERFILES FOR RED HAT SOFTWARE COLLECTIONS

Red Hat Software Collections is shipped with Dockerfiles for the following Software Collections:

- httpd24
- mariadb55
- mongodb24
- mysql55
- nginx16
- nodejs010
- perl516
- php54
- php55
- postgresql92
- python27
- python33
- rh-mariadb100
- rh-mongodb26
- rh-mysql56
- rh-passenger40
- rh-perl520
- rh-php56
- rh-postgresql94
- rh-python34
- rh-ror41
- rh-ruby22
- ror40
- ruby193

- ruby200

The Dockerfiles are included in the rhscl-dockerfiles package distributed with Red Hat Software Collections. Dockerfiles are text files that define how a Docker image is created. Note that the rhscl-dockerfiles package has not been updated since Red Hat Software Collections 2.0.

NOTE

The docker package, which contains the **`Docker`** daemon, command line tool, and other necessary components for building and using docker-formatted container images, is currently only available for the Server variant of the Red Hat Enterprise Linux 7 product. Red Hat Software Collections Dockerfiles are distributed for Red Hat Enterprise Linux 6 as well, but the images built using them can only be deployed on Red Hat Enterprise Linux 7 Server.

Each Dockerfile creates a minimal Docker image from Red Hat Enterprise Linux 6 or Red Hat Enterprise Linux 7 plus the Software Collection. Each Dockerfile will create an image which:

- Installs the basic set of packages from each Software Collection,
- Exposes some TCP ports; for example, port **`80`** and **`443`** for the httpd24 collection.

The Dockerfiles are provided as examples, using which customers can build more complex containers.

Dockerfiles are available also for previously released Software Collections. For detailed information about them, refer to the Red Hat Software Collections documentation and the Red Hat Software Collections Product Life Cycle document.

3.4.1. Installation and Usage

To install the rhscl-dockerfiles package, type the following command as **`root`**:

```
yum install rhscl-dockerfiles
```

Use these Dockerfiles to create Docker images for the covered Software Collections.

For more information about building an image from a Dockerfile, see the Get Started with Docker Formatted Container Images chapter in the *Getting Started with Containers* documentation.

3.4.2. Deploying Software Collections Dependent on the Red Hat Software Collections Docker Images

You can use a Red Hat Software Collections Docker image as a base image and create your own containerized Software Collection on top of it as a separate image.

For more information about creating a new Docker image, see the Creating Docker Images section in the *Getting Started with Containers* documentation.

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**, **Eclipse** development platform, 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 Main Features section of the *Red Hat Developer Toolset Release Notes*.

For a complete list of components, see the Red Hat Developer Toolset Components table in the *Red Hat Developer Toolset User Guide*.

Note that since Red Hat Developer Toolset 3.1, Red Hat Developer Toolset requires the rh-java-common Software Collection.

4.2. THERMOSTAT 1

The **Thermostat** Software Collection provides a monitoring and instrumentation tool for the OpenJDK HotSpot JVM, with support for monitoring multiple JVM instances. The system is made up of two components: an `Agent`, which collects data, and a `Client`, which allows users to visualize collected data. These components communicate via a storage layer: either directly via **MongoDB** or indirectly via a Web layer for increased security. A pluggable agent and GUI framework allows for collection and visualization of performance data beyond what is included out of the box.

To install the thermostat1 collection, type the following command as `root`:

```
yum install thermostat1
```

Note that since Red Hat Software Collections 2.0, the thermostat1 Software Collection requires the rh-java-common Collection.

To enable the thermostat1 collection, type the following command at a shell prompt:

```
scl enable thermostat1 bash
```

For more information, refer to the Thermostat User Guide. In order to deploy Thermostat securely, see the Configuration and Administration Guide.

4.3. RUBY ON RAILS 4.2

Red Hat Software Collections 2.2 adds the rh-ruby23 Software Collection together with the rh-ror42 Collection.

To install **Ruby on Rails 4.2**, type the following command as **root**:

```
yum install rh-ror42
```

Installing any package from the rh-ror42 Software Collection automatically pulls in rh-ruby23 and rh-nodejs4 as dependencies.

The rh-nodejs4 Collection is used by certain gems in an asset pipeline to post-process web resources, for example, **sass** or **coffee-script** source files. To run the **rails s** command without requiring rh-nodejs4, disable the **coffee-rails** and **uglifier** gems in the **Gemfile**.

The **Ruby on Rails** Collection can be enabled by the following command, which will automatically enable rh-ruby23:

```
scl enable rh-ror42 bash
```

These two Collections are supported together and available only for Red Hat Enterprise Linux 7. On Red Hat Enterprise Linux 6, use the rh-ruby22 and rh-ror41 Software Collections.

4.4. MONGODB 3.2

To install the rh-mongodb32 collection, which is available for Red Hat Enterprise Linux 7, type the following command as **root**:

```
yum install rh-mongodb32
```

Note that the rh-mongodb32 Software Collection requires the rh-java-common Collection.

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

```
scl enable rh-mongodb32 'mongo'
```

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

```
systemctl start rh-mongodb32-mongod.service
```

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

```
systemctl enable rh-mongodb32-mongod.service
```

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

```
systemctl start rh-mongodb32-mongos.service
```

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

```
systemctl enable rh-mongodb32-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. DEVASSISTANT

DevAssistant is a tool designed to assist developers with creating and setting up basic projects in various programming languages, installing dependencies, setting up a development environment, and working with source control. The devassist09 Software Collection supports several programming languages, namely C, C++, Java, and Python. Additionally, DevAssistant is able to support working with any other language, framework, or tool due to its modular architecture.

DevAssistant is a framework that runs plug-ins called *assistants*. Each assistant can have several subassistants.

4.5.1. Getting Started with DevAssistant

To install the devassist09 Software Collection, type the following command as **root**:

```
yum install devassist09
```

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

```
scl enable devassist09 bash
```

To get help for DevAssistant, use the following command:

```
devassistant --help
```

or the shorter variant of the same command:

```
da -h
```

It is advisable to use the **--help** option on each level to list your possible next steps, until you reach the level of an executable subassistant (see Example 4.1, “Creating a New Python Library Project”).

To access the graphical user interface, type this command at a shell prompt:

```
devassistant-gui
```

or the shortened variant:

```
da-gui
```

Please note that the GUI is available only if you install the devassist09 Software Collection on Red Hat Enterprise Linux 7. The functionalities and procedures are the same as when using the command line interface.

Note that the **devassistant** and **da** commands are equal. Further in the text, we will use only the shorter variant, the **da** command.

4.5.2. Running Assistants

DevAssistant provides the following functionalities: **create**, **modify**, **prepare**, and **task**. To run an assistant, use the following command:

```
da [--debug] {create,modify,prepare,task} [assistant [arguments]] ...
```

The four basic commands and descriptions related to these functionalities are listed in the following table:

Table 4.1. Functionalities of DevAssistant

Command	Shortened Command	Description
`da create`	`da crt`	Creating a new project from scratch
`da modify`	`da mod`	Working with an existing project
`da prepare`	`da prep`	Preparing a development environment for an upstream project
`da task`		Performing a custom task not related to a specific project

The devassist09 Software Collection does not include any assistants for the **`modify`**, **`prepare`**, and **`task`** functionalities. These categories are available for users who want to create their own assistants.

4.5.3. Creating Projects with DevAssistant

The devassist09 Software Collection includes the following assistants for creating projects:

Table 4.2. Assistants for Creating Projects

Assistant	Subassistant	Description
c	app	An application in C
	lib	A dynamically linked library in C
cpp	app	An application in C++
	lib	A dynamically linked library in C++
java	maven	A simple project using Maven
python	lib	A simple library for Python

The following example demonstrates creating a new Python library project by following instructions displayed by the **`--help`** option.

Example 4.1. Creating a New Python Library Project

To create a new Python library project, complete the following steps:

1. Enable the devassist09 Software Collection by running this command:

```
~]$ scl enable devassist09 bash
```

2. Display help about DevAssistant by using the **--help** option:

```
~]$ da --help
You can either run assistants with:
da [--debug] {create,modify,prepare,task} [ASSISTANT [ARGUMENTS]]
...

Where:
create   used for creating new projects
modify   used for working with existing projects
prepare  used for preparing environment for upstream projects
task     used for performing custom tasks not related to a
specific project
You can shorten "create" to "crt", "modify" to "mod" and "prepare"
to "prep".

Or you can run a custom action:
da [--debug] [ACTION] [ARGUMENTS]

Available actions:
help     Print detailed help
version  Print version
```

3. List the possible next steps for creating a project by typing:

```
~]$ da create --help
usage:  create [-h] [--deps-only] {c,cpp,java,python} ...

Kickstart new projects easily with DevAssistant.

optional arguments:
  -h, --help           show this help message and exit
  --deps-only          Only install dependencies

subassistants:
  Following subassistants will help you with setting up your
project.

  {c,cpp,java,python}
```

4. Display help on the **python** assistant by typing at a shell prompt:

```
~]$ da create python --help
usage: create python [-h] {lib} ...

This is a base Python assistant, you have to select a
subassistant.

optional arguments:
  -h, --help  show this help message and exit

subassistants:
```

```
  Following subassistants will help you with setting up your
project.

  {lib}
```

5. List your choices for the only **python** subassistant, **lib**, by running this command:

```
~]$ da create python lib --help
usage: create python lib [-h] [-e [ECLIPSE]] -n NAME

Scaffolds a simple Python library project.

optional arguments:
  -h, --help            show this help message and exit
  -e [ECLIPSE], --eclipse [ECLIPSE]
                        Configure as Eclipse project (uses
~/workspace or
                        specified directory)
  -n NAME, --name NAME  Name of project to create
```

6. Run the assistant to create your new Python library project named **mypythonlib** by using the following command:

```
~]$ da create python lib -n mypythonlib
```

To get more information about the upstream version of **DevAssistant**, refer to the DevAssistant User Documentation. Please note that though the basic concept of the upstream application is the same as in the devassist09 Software Collection, individual plug-ins and their functionalities might differ.

4.5.4. Backward Compatibility in DevAssistant

The updated version of **DevAssistant** can cause incompatibility in assistants that have not been provided by the devassist09-devassistant-assistants-dts package, that is, in your own assistants.

- Since **DevAssistant 0.9.3**, the variable names in the assistant files are no longer derived from the argument flags but from the argument names. In the following example, the **$foo** variable is initialized instead of the **$bar** variable:

```
args:
   foo:
     ...
     flags: [-b, --bar]
     ...
```

- Unknown attributes in the arguments section in the assistant file are no longer allowed. Since **DevAssistant 0.9.3**, an error message is returned in the following example because the **unknown_attribute** is not known to the parser:

```
args:
   foo:
     ...
```

```
unknown_attribute: foo bar baz
...
```

4.6. MAVEN

The rh-maven33 Software Collection 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-maven33 Collection, type the following command as **root**:

```
yum install rh-maven33
```

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

```
scl enable rh-maven33 bash
```

Global Maven settings, such as remote repositories or mirrors, can be customized by editing the **/opt/rh/rh-maven33/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.

4.7. 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.

CHAPTER 5. MIGRATION

This chapter provides information on migrating to versions of components included in Red Hat Software Collections 2.2.

5.1. MIGRATING TO MARIADB 10.1

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 10.0** has been available as a Software Collection since Red Hat Software Collections 2.0; Red Hat Software Collections 2.2 is distributed with **MariaDB 10.1**.

The rh-mariadb101 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, so it is possible to install the rh-mariadb101 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-mariadb101 Software Collection while the rh-mariadb100 Collection is still installed and even running.

Note that if you are using an **MariaDB 5.5**, it is necessary to upgrade to the rh-mariadb100 Software Collection first, which is described in the Red Hat Software Collections 2.0 Release Notes.

For more information about **MariaDB 10.1**, see the upstream documentation about changes in version 10.1 and about upgrading.

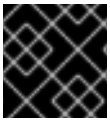
NOTE

The rh-mariadb101 Software Collection supports neither mounting over NFS nor dynamical registering using the `scl register` command.

5.1.1. Notable Differences Between the mariadb100 and rh-mariadb101 Software Collections

- Galera Cluster, a synchronous multi-master cluster, which is a standard part of MariaDB 10.1. See the Knowledgebase article about setting up Galera Cluster with the rh-mariadb101 Software Collection.
- Since **MariaDB 10.1.7**, the `SQL_MODE` variable is by default set to `NO_ENGINE_SUBSTITUTION,NO_AUTO_CREATE_USER` while in earlier versions of **MariaDB** no default was set. Consequently, the `GRANT` statement does not create a user by default. The setting of the `SQL_MODE` variable can be changed in the configuration file. See the upstream documentation for details.

5.1.2. Upgrading from the rh-mariadb100 to the rh-mariadb101 Software Collection



IMPORTANT

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

1. Install the rh-mariadb101 Software Collection.

   -

```
yum install rh-mariadb101-mariadb-server
```

2. Inspect the configuration of rh-mariadb101, which is stored in the **/etc/opt/rh/rh-mariadb101/my.cnf** file and the **/etc/opt/rh/rh-mariadb101/my.cnf.d/** directory. Compare it with the configuration of rh-mariadb100 stored in **/etc/opt/rh/rh-mariadb100/my.cnf** and **/etc/opt/rh/rh-mariadb100/my.cnf.d/** and adjust it if necessary.

3. Stop the rh-mariadb100 database server, if it is still running.

```
service rh-mariadb100-mariadb stop
```

4. All the data of the rh-mariadb100 Software Collection is stored in the **/var/opt/rh/rh-mariadb100/lib/mysql/** directory. Copy the whole content of this directory to **/var/opt/rh/rh-mariadb101/lib/mysql/**. You can also move the content but remember to back up your data before you continue to upgrade.

5. Start the rh-mariadb101 database server.

```
service rh-mariadb101-mariadb start
```

6. Perform the data migration.

```
scl enable rh-mariadb101 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-mariadb101 -- mysql_upgrade -p
```

5.2. MIGRATING TO MONGODB 3.2

Red Hat Software Collections 2.2 is shipped with **MongoDB 3.2**, provided by the rh-mongodb32 Software Collection and available for Red Hat Enterprise Linux 7. Previously released **MongoDB** Software Collections, mongodb24 and rh-mongodb26 are available for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7. See the Red Hat Software Collections 2.0 Release Notes if you need to upgrade to **MongoDB 2.6**.

Note that when migrating from the rh-mongodb26 to the rh-mongodb32 Software Collection, it is necessary to first upgrade to the 3.0-series release of **MongoDB**, which is provided by the rh-mongodb30upg Collection.

5.2.1. Notable Differences Between MongoDB 2.6 and MongoDB 3.2

General Changes

The rh-mongodb32 Software Collection introduces several general changes listed below.

- **MongoDB** now ships configuration files in the YAML format
- **MongoDB** server and tools are no longer shipped in a single package; **MongoDB** tools are packaged in rh-mongodb32-mongo-tools

- Improved **MongoDB** testsuite provided by the rh-mongodb32-mongodb-test package. For more information about usage, install this package and read the **`/opt/rh/rh-mongodb32/root/usr/share/mongodb-test/README`** file.

Compatibility Changes

MongoDB 3.2 includes various minor changes that can affect compatibility with previous versions of **MongoDB**.

Compatibility Changes in MongoDB 3.0

- Configuration file options changes due to inclusion of additional storage engines
- Data files must correspond to the configured storage engine; if files in the **`dbPath`** directory were created by a storage engine other than the current one, an error is returned
- Changes due to using the **WiredTiger** storage engine: **`oplog`** entries generated by versions of **MongoDB** earlier than 2.2.1 are overwritten
- Replica set configuration validation
- The **`w:majority`** semantics has been changed so that the **`w:majority`** value is satisfied when a majority of the voting members replicates a write operation
- The **`local.slaves`** collection has been removed
- The FATAL replica set state no longer exists
- The **`mongodump`**, **`mongorestore`**, **`mongoexport`**, **`mongoimport`**, **`mongofiles`**, and **`mongooplog`** tools must connect to a running **MongoDB** instance
- The **MongoDB 2.4** user model has been removed
- The localhost exception has been changed so that it allows to create only the first user on the admin database
- The **`db.addUser()`** function has been removed; use **`db.createUser()`** and **`db.updateUser()`** instead
- TLS/SSL changes
- The **`mongo`** shell versions earlier than 3.0 are not compatible with 3.0 deployments of **MongoDB**
- Index changes
- Direct access to the **`system.indexes`** and **`system.namespaces`** collections has been deprecated
- The following commands have been deprecated: **`closeAllDatabases`**, **`getoptime`**, **`text`**, **`indexStats`**, **`db.collection.getIndexStats()`**, and **`db.collection.indexStats()`**
- The **`Date`** and **`Timestamp`** data types are no longer equivalent for comparison purposes

For details regarding compatibility changes in **Mongodb 3.0**, refer to the upstream release notes.

Compatibility Changes in MongoDB 3.2

- The **WiredTiger** storage engine is now the default one

- The JavaScript engine has been changed from **V8** to **SpiderMonkey**
- Creation of version 0 indexes is now disallowed
- Aggregation compatibility changes

For detailed compatibility changes in **MongoDB 3.2**, see the upstream release notes.

5.2.2. Upgrading from the rh-mongodb26 to the rh-mongodb32 Software Collection

Note that once you have upgraded to **MongoDB 3.2** and started using new features, you cannot downgrade to any earlier version.

IMPORTANT

Before migrating from the rh-mongodb26 to the rh-mongodb32 Software Collection, back up all your data, including any **MongoDB** databases, which are by default stored in the **`/var/opt/rh/rh-mongodb26/lib/mongodb/`** directory.

To upgrade to the rh-mongodb32 Software Collection, perform the following steps.

1. Install the **MongoDB** servers and shells from the rh-mongodb30upg and rh-mongodb32 Software Collections:

```
~]# yum install rh-mongodb30upg rh-mongodb30upg-mongodb rh-mongodb32 rh-mongodb32-mongodb
```

2. Connect the **`mongo`** shell from the rh-mongodb26 Collection to your **MongoDB 2.6** server (for example, running on **`localhost`**, port **`27017`**).

```
~]$ scl enable rh-mongodb26 'mongo --host localhost --port 27017 admin'
```

3. In the **`mongo`** shell, check your data set for compatibility issues mentioned above and fix the ones that affect your application.

4. Stop the **MongoDB 2.6** server:

```
~]# systemctl stop rh-mongodb26-mongod.service
```

5. Copy your data to the new location:

```
~]# cp -a /var/opt/rh/rh-mongodb26/lib/mongodb/* /var/opt/rh/rh-mongodb32/lib/mongodb
```

6. Change the **`dbpath`** variable in the **`/etc/opt/rh/rh-mongodb30upg/mongod.conf`** file to **`/var/opt/rh/rh-mongodb32/lib/mongodb/`**.

7. Start the **MongoDB** server from the rh-mongodb30upg Software Collection:

```
~]# systemctl start rh-mongodb30upg-mongod.service
```

8. Connect the **mongo** shell from the rh-mongodb32 Collection to your **MongoDB 3.0** server (for example, running on **localhost**, port **27017**).

```
~]$ scl enable rh-mongodb30upg 'mongo --host localhost --port 27017 admin'
```

9. In the **mongo** shell, check your data set for compatibility issues mentioned above and fix the ones that affect your application.

10. Stop the **MongoDB 3.0** server.

```
~]# systemctl stop rh-mongodb30upg-mongod.service
```

11. Configure the **rh-mongodb32-mongod** daemon in the **/etc/opt/rh/rh-mongodb32/mongod.conf** file.

12. **MongoDB 3.2** has the new default storage engine, **WiredTiger**, which introduces performance improvements. To be able to run the **MongoDB** server with old data, configure the **rh-mongodb32-mongod** daemon to use the old storage engine. Uncomment the **engine** property in the **storage** section in the **/etc/opt/rh/rh-mongodb32/mongod.conf** file and change its value to **mmapv1**.

13. Start the **MongoDB 3.2** server.

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

14. If you want to use the **WiredTiger** storage engine, you have to perform additional migration steps described in the MongoDB documentation.

```
~]# yum install rh-mongodb32-mongo-tools
~]$ scl enable rh-mongodb32 'mongodump --out ~/mongodb.dump'
~]# systemctl stop rh-mongodb32-mongod.service
~]# rm -rf /var/opt/rh/rh-mongodb32/lib/mongodb/*
```

Change the **engine** property in the **/etc/opt/rh/rh-mongodb32/mongod.conf** to **wiredTiger**.

```
~]# systemctl start rh-mongodb32-mongod.service
~]$ scl enable rh-mongodb32 'mongorestore ~/mongodb.dump'
```

For detailed information about upgrading, see the upstream MongoDB 3.0 and MongoDB 3.2 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.

5.3. MIGRATING TO MYSQL 5.6

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. In addition to these basic versions, **MySQL 5.5** has been available as a Software Collection for Red Hat Enterprise Linux 6

since Red Hat Software Collections 1.0 and for Red Hat Enterprise Linux 7 since Red Hat Software Collections 1.1.

The rh-mysql56 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, so it is possible to install the rh-mysql56 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.

Note that it is possible to upgrade to **MySQL 5.6** only from **MySQL 5.5**. If you need to upgrade from an earlier version, upgrade to **MySQL 5.5** first. Instructions how to upgrade to **MySQL 5.5** are available in the Red Hat Software Collections 1.2 Release Notes.

5.3.1. Notable Differences Between MySQL 5.5 and MySQL 5.6

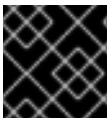
The rh-mysql56 Software Collection introduces the following notable changes:

- The service has been renamed to **rh-mysql56-mysqld** in both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7
- The **test** database is no longer created by default
- Configuration files for the rh-mysql56 Software Collection are the **/etc/opt/rh/rh-mysql56/my.cnf** file and in the **/etc/opt/rh/rh-mysql56/my.cnf.d/** directory
- Variable files including the database files for the rh-mysql56 Software Collection are located in the **/var/opt/rh/rh-mysql56/lib/** directory
- The log file for the MySQL daemon is **/var/opt/rh/rh-mysql56/log/mysql/mysqld.log**
- The pid file for the daemon is **/var/run/rh-mysql56-mysqld/mysqld.pid**

Note that the rh-mysql56 Software Collection supports neither mounting over NFS nor dynamical registering using the **scl register** command.

For detailed changes, refer to the MySQL documentation.

5.3.2. Upgrading to the rh-mysql56 Software Collection



IMPORTANT

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

Upgrading can be performed either by using the **mysqldump** and **mysqlimport** utilities or using an in-place upgrade.

- In the former scenario, the whole dump of all databases from one database is generated and **mysql** is run with the dump file as an input using the **mysqlimport** or **LOAD DATA INFILE SQL** command within the other database. At the same time, the appropriate daemons have to be running during both dumping and restoring. You can use the **--all-databases** option in the **mysqldump** call to include all databases in the dump. The **--routines**, **--triggers**, and **--events** options can also be used if needed.

- During the in-place upgrade, the data files are copied from one database directory to another database directory. The daemons must not be running at the time of copying. Set appropriate permissions and SELinux context for the copied files.

After upgrading, start the server and run the **mysql_upgrade** command. Running **mysql_upgrade** is necessary to check and repair internal tables.

In case 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.

Service names and paths bellow depend on which version you are upgrading from.

Example 5.1. Dump and Restore Upgrade

1. Create a backup from the mysql55 Software Collection:

```
~]# service mysql55-mysqld start
Starting mysql55-mysqld:                                   [  OK
]
~]# scl enable mysql55 -- mysqldump --all-databases --routines --
events > dump.sql
~]# service mysql55-mysqld stop
Stopping mysql55-mysqld:                                   [  OK
]
```

   For upgrading from the mariadb55 Software Collection in Red Hat Enterprise Linux 6, use **mariadb55-mysqld** as the service name.

   For upgrading from the mariadb55 Software Collection in Red Hat Enterprise Linux 7, use **mariadb55-mariadb** as the service name.

   For upgrading from **MariaDB 5.5** from base Red Hat Enterprise Linux 7, use **mariadb** as the service name and do not use **scl enable mysql55 --** when creating the dump.

2. Import the dumped database into the rh-mysql56 Software Collection:

```
~]# service rh-mysql56-mysqld start
Starting rh-mysql56-mysqld:                                [  OK
]
~]# scl enable rh-mysql56 'mysql' < dump.sql
~]# scl enable rh-mysql56 'mysql_upgrade -u root -p'
Enter password:
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                               OK
mysql.columns_priv                                 OK
<skipped tables list>
mysql.user                                         OK
Running 'mysql_fix_privilege_tables'...
OK
```

Example 5.2. In-place Upgrade from MySQL 5.5

If you are upgrading from the mysql55 Software Collection, perform the upgrade as shown in the following example:

```
~]# service mysql55-mysqld stop
Stopping mysql55-mysqld                                    [  OK  ]
~]# service rh-mysql56-mysqld stop
Stopping rh-mysql56-mysqld:                                [  OK  ]
~]# rm -rf /var/opt/rh/rh-mysql56/lib/mysql/
~]# cp -r /opt/rh/mysql55/root/var/lib/mysql/ /var/opt/rh/rh-
mysql56/lib/mysql/
~]# chown -R mysql:mysql /var/opt/rh/rh-mysql56/lib/mysql/
~]# restorecon -R /var/opt/rh/rh-mysql56/lib/mysql/
~]# service rh-mysql56-mysqld start
Starting rh-mysql56-mysqld:                                [  OK  ]
~]# scl enable rh-mysql56 'mysql_upgrade -u root -p'
Enter password:
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                               OK
mysql.columns_priv                                 OK
<skipped tables list>
mysql.user                                         OK
Running 'mysql_fix_privilege_tables'...
OK
```

For upgrading from the mariadb55 Software Collection, use the **`/opt/rh/mariadb55/root/var/lib/mysql/`** as a source when copying the data.

For upgrading from **MariaDB 5.5** from base Red Hat Enterprise Linux 7, use the **`/var/lib/mysql/`** as a source when copying the data.

For more details about migration to **MySQL 5.6**, refer to the MySQL documentation.

5.4. MIGRATING TO POSTGRESQL 9.5

Red Hat Software Collections 2.2 is distributed with **PostgreSQL 9.5**, which 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 Red Hat Software Collections 1, or **PostgreSQL 9.4** from Red Hat Software Collections 2. 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.

5.4.1. Notable Differences Between PostgreSQL 9.4 and PostgreSQL 9.5

The most notable changes between **PostgreSQL 9.4** and **PostgreSQL 9.5** are desribed in the upstream release notes for versions 9.5, 9.5.1, and 9.5.2.

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-postgresql94, and rh-postgresql95 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.

Table 5.1. Diferences in the PostgreSQL paths

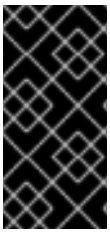
Content	postgresql	postgresql92	rh-postgresql94	rh-postgresql95
Executables	/usr/bin/	/opt/rh/postgresql92/root/usr/bin/	/opt/rh/rh-postgresql94/root/usr/bin/	/opt/rh/rh-postgresql95/root/usr/bin/
Libraries	/usr/lib64/	/opt/rh/postgresql92/root/usr/lib64/	/opt/rh/rh-postgresql94/root/usr/lib64/	/opt/rh/rh-postgresql95/root/usr/lib64/
Documentation	/usr/share/doc/postgresql/html/	/opt/rh/postgresql92/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql94/root/usr/share/doc/postgresql/html/	/opt/rh/rh-postgresql95/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-postgresql94/root/usr/share/doc/postgresql-docs/	/opt/rh/rh-postgresql95/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-postgresql94/root/usr/share/doc/postgresql-contrib/	/opt/rh/rh-postgresql95/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-postgresql94/lib/pgsql/data/	/var/opt/rh/rh-postgresql95/lib/pgsql/data/
Backup area	/var/lib/pgsql/backups/	/opt/rh/postgresql92/root/var/lib/pgsql/backups/	/var/opt/rh/rh-postgresql94/lib/pgsql/backups/	/var/opt/rh/rh-postgresql95/lib/pgsql/backups/
Templates	/usr/share/pgsql/	/opt/rh/postgresql92/root/usr/share/pgsql/	/opt/rh/rh-postgresql94/root/usr/share/pgsql/	/opt/rh/rh-postgresql95/root/usr/share/pgsql/
Procedural Languages	/usr/lib64/pgsql/	/opt/rh/postgresql92/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql94/root/usr/lib64/pgsql/	/opt/rh/rh-postgresql95/root/usr/lib64/pgsql/
Development Headers	/usr/include/pgsql/	/opt/rh/postgresql92/root/usr/include/pgsql/	/opt/rh/rh-postgresql94/root/usr/include/pgsql/	/opt/rh/rh-postgresql95/root/usr/include/pgsql/

Content	postgresql	postgresql92	rh-postgresql94	rh-postgresql95
Other shared data	/usr/share/pgsql/	/opt/rh/postgresql92/root/usr/share/pgsql/	/opt/rh/rh-postgresql94/root/usr/share/pgsql/	/opt/rh/rh-postgresql95/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-postgresql94/root/usr/lib64/pgsql/test/regress/ (in the -test package)	/opt/rh/rh-postgresql95/root/usr/lib64/pgsql/test/regress/ (in the -test package)

For detailed changes, see the upstream PostgreSQL 9.5 Release Notes. 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.

5.4.2. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 9.5 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-postgresql95 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 9.5, 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`**:

   ```
   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-postgresql95/lib/pgsql/data/** does not exist:

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

If you are running a fresh installation of **PostgreSQL 9.5**, 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-postgresql95/lib/pgsql/data{,-scl-backup}
```

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

```
scl enable rh-postgresql95 -- postgresql-setup --upgrade
```

Alternatively, you can use the **/opt/rh/rh-postgresql95/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-postgresql95-postgresql.log** log file to find out if any problems occurred during the upgrade.

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

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

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

```
su - postgres -c 'scl enable rh-postgresql95
~/analyze_new_cluster.sh'
```

6. Optionally, you can configure the PostgreSQL 9.5 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 9.5 server, type as **root**:

```
chkconfig rh-postgresql95-postgresql on
```

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql95/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**:

   ```
   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-postgresql95-postgresql -- postgresql-setup --initdb
   ```

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

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

6. Import data from the previously created SQL file:

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

7. Optionally, you can configure the PostgreSQL 9.5 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 9.5 server, type as **root**:

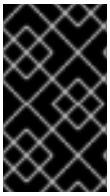
   ```
   chkconfig rh-postgresql95-postgresql on
   ```

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql95/lib/pgsql/data/pg_hba.conf** configuration file. Otherwise only the **postgres** user will be allowed to access the database.

5.4.3. Migrating from the PostgreSQL 9.4 Software Collection to the PostgreSQL 9.5 Software Collection

To migrate your data from the rh-postgresql94 Software Collection to the rh-postgresql95 Collection included in Red Hat Software Collections 2.2, 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 9.4** to **PostgreSQL 9.5**, 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-postgresql94/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`**:

   ```
   service rh-postgresql94-postgresql stop
   ```

   To verify that the server is not running, type:

   ```
   service rh-postgresql94-postgresql status
   ```

2. Verify that the old directory **`/var/opt/rh/rh-postgresql94/lib/pgsql/data/`** exists:

   ```
   file /var/opt/rh/rh-postgresql94/lib/pgsql/data/
   ```

   and back up your data.

3. Verify that the new data directory **`/var/opt/rh/rh-postgresql95/lib/pgsql/data/`** does not exist:

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

   If you are running a fresh installation of **PostgreSQL 9.5**, 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-postgresql95/lib/pgsql/data{,-scl-backup}
   ```

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

   ```
   scl enable rh-postgresql95 -- postgresql-setup --upgrade --upgrade-
   from=rh-postgresql94-postgresql
   ```

   Alternatively, you can use the **`/opt/rh/rh-postgresql95/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql94-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-postgresql95-postgresql.log** log file to find out if any problems occurred during the upgrade.

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

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

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

```
su - postgres -c 'scl enable rh-postgresql95
~/analyze_new_cluster.sh'
```

6. Optionally, you can configure the PostgreSQL 9.5 server to start automatically at boot time. To disable the old PostgreSQL 9.4 server, type the following command as **root**:

```
chkconfig rh-postgresql94-postgreqsql off
```

To enable the PostgreSQL 9.5 server, type as **root**:

```
chkconfig rh-postgresql95-postgresql on
```

7. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql95/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**:

```
service rh-postgresql94-postgresql start
```

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

```
su - postgres -c 'scl enable rh-postgresql94 "pg_dumpall >
~/pgdump_file.sql"'
```

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

```
service rh-postgresql94-postgresql stop
```

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

```
scl enable rh-postgresql95-postgresql -- postgresql-setup --initdb
```

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

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

6. Import data from the previously created SQL file:

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

7. Optionally, you can configure the PostgreSQL 9.5 server to start automatically at boot time. To disable the old PostgreSQL 9.4 server, type the following command as **root**:

```
chkconfig rh-postgresql94-postgresql off
```

   To enable the PostgreSQL 9.5 server, type as **root**:

```
chkconfig rh-postgresql95-postgresql on
```

8. If your configuration differs from the default one, make sure to update configuration files, especially the **/var/opt/rh/rh-postgresql95/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.

5.5. MIGRATING TO NGINX 1.8

The root directory for the rh-nginx18 Software Collection is located in **/opt/rh/rh-nginx18/root/**. The error log is stored in **/var/opt/rh/rh-nginx18/log/nginx** by default, and the init script is called **rh-nginx18-nginx**.

Configuration files are now stored in the **/etc/opt/rh/rh-nginx18/nginx/** directory. Configuration files in **nginx 1.8** have the same format as in the previous versions and they are compatible among versions 1.4, 1.6, and 1.8.

IMPORTANT

Before upgrading from **nginx 1.6** to **nginx 1.8**, back up all your data, including web pages and configuration files located in the **/opt/rh/nginx16/root/** tree.

If you have made any specific changes, such as changing configuration files or setting up web applications, in the **/opt/rh/nginx16/root/** tree, replicate those changes in the new **/opt/rh/rh-nginx18/root/** and **/etc/opt/rh/rh-nginx18/nginx/** directories, too.

You can use this procedure to upgrade directly from **nginx 1.4** to **nginx 1.8**. Use the appropriate paths for **nginx 1.4** in this case.

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

CHAPTER 6. ADDITIONAL RESOURCES

This chapter provides references to other relevant sources of information about Red Hat Software Collections 2.2 and Red Hat Enterprise Linux.

6.1. RED HAT ENTERPRISE LINUX DEVELOPER PROGRAM GROUP

Users of Red Hat Software Collections can access the Red Hat Enterprise Linux Developer Program Group in the Red Hat Customer Portal to get developer related information for the development tools available for Red Hat Enterprise Linux. In addition, users can find developer related papers and videos on topics that are of interest to developers, for example RPM building, threaded programming, performance tuning, debugging, and so on.

To visit the Red Hat Enterprise Linux Developer Program Group, log in to the Red Hat Customer Portal, click **Products & Services** at the top of the page, choose **Services**, and then **Red Hat Enterprise Linux Developer Program** from the list.

6.2. RED HAT PRODUCT DOCUMENTATION

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

- Red Hat Software Collections 2.2 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 4.1 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 4.1 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 article provides information on how to use container images based on Red Hat Software Collections. The available container images include applications, daemons, and databases. The images can be run on Red Hat Enterprise Linux 7 Server and Red Hat Enterprise Linux Atomic Host.
- Get Started with Docker Formatted Container Images — This guide contains a comprehensive overview of information about building and using docker-formatted 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.

6.3. RED HAT DEVELOPER BLOG

Red Hat Developer Blog content is directed to designers and developers of applications based on Red Hat technologies. It contains links to product team blogs and other relevant internal and external resources. Its goal is to inform and engage the developer community with up-to-date information, best practices, opinion, product and program announcements as well as pointers to sample code and other resources.

APPENDIX A. REVISION HISTORY

Revision 2.2-14 **Tue Dec 18 2018** **Lenka Špačková**

Fixed syntax of a command in the PostgreSQL migration chapter.

Revision 2.2-13 **Tue Apr 04 2017** **Lenka Špačková**

Fixed a container image name.

Revision 2.2-12 **Fri Jul 01 2016** **Lenka Špačková**

Added a link to a Knowledgebase article about setting up Galera Cluster with MariaDB 10.1.

Revision 2.2-10 **Tue Jun 21 2016** **Lenka Špačková**

The rh-nodejs4 Software Collection is now available also for Red Hat Enterprise Linux 6.

Revision 2.2-9 **Thu Jun 09 2016** **Lenka Špačková**

Updated a link to content in the ISO format.

Revision 2.2-8 **Tue May 31 2016** **Lenka Špačková**

Release of Red Hat Software Collections 2.2 Release Notes.

Revision 2.2-3 **Thu May 05 2016** **Lenka Špačková**

Release of Red Hat Software Collections 2.2 Beta Release Notes.