Red Hat Software Collections 3.1 Release Notes

Release Notes for Red Hat Software Collections 3.1

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

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

Eliška Slobodová
Red Hat Customer Content Services
Abstract

# Table of Contents

**CHAPTER 1. RED HAT SOFTWARE COLLECTIONS 3.1** ........................................................................ 4  
1.1. ABOUT RED HAT SOFTWARE COLLECTIONS ........................................................................... 4  
  1.1.1. Red Hat Developer Toolset ........................................................................................................ 4  
1.2. MAIN FEATURES .................................................................................................................... 4  
1.3. CHANGES IN RED HAT SOFTWARE COLLECTIONS 3.1 ......................................................... 15  
  1.3.1. Overview .................................................................................................................................. 15  
    - Architectures ................................................................................................................................. 15  
    - New Software Collections .............................................................................................................. 15  
    - Updated Software Collections ...................................................................................................... 15  
    - Red Hat Software Collections Container Images ........................................................................ 15  
  1.3.2. Changes in Red Hat Developer Toolset ...................................................................................... 16  
  1.3.3. Changes in Perl .......................................................................................................................... 17  
  1.3.4. Changes in Ruby ....................................................................................................................... 17  
  1.3.5. Changes in MongoDB ................................................................................................................ 17  
  1.3.6. Changes in PostgreSQL ........................................................................................................... 18  
  1.3.7. Changes in Varnish Cache .......................................................................................................... 19  
  1.3.8. Changes in HAProxy ................................................................................................................ 19  
  1.3.9. Changes in PHP ........................................................................................................................ 19  
  1.3.10. Changes in MySQL ................................................................................................................ 19  
  1.3.11. Changes in Apache httpd ..................................................................................................... 20  
1.4. COMPATIBILITY INFORMATION .......................................................................................... 20  
1.5. KNOWN ISSUES .......................................................................................................................... 20  
  1.5.1. Other Notes .............................................................................................................................. 24  
1.6. DEPRECATED FUNCTIONALITY ............................................................................................ 26  

**CHAPTER 2. INSTALLATION** .................................................................................................... 27  
2.1. GETTING ACCESS TO RED HAT SOFTWARE COLLECTIONS ................................................ 27  
  2.1.1. Using Red Hat Subscription Management ............................................................................... 27  
  2.1.2. Packages from the Optional Channel ....................................................................................... 28  
2.2. INSTALLING RED HAT SOFTWARE COLLECTIONS ............................................................. 30  
  2.2.1. Installing Individual Software Collections ............................................................................. 31  
  2.2.2. Installing Optional Packages .................................................................................................. 31  
  2.2.3. Installing Debugging Information ............................................................................................ 31  
2.3. UNINSTALLING RED HAT SOFTWARE COLLECTIONS .......................................................... 32  
2.4. REBUILDING RED HAT SOFTWARE COLLECTIONS ............................................................ 32  

**CHAPTER 3. USAGE** ................................................................................................................ 33  
3.1. USING RED HAT SOFTWARE COLLECTIONS .......................................................................... 33  
  3.1.1. Running an Executable from a Software Collection ............................................................... 33  
  3.1.2. Running a Shell Session with a Software Collection as Default ........................................... 33  
  3.1.3. Running a System Service from a Software Collection ......................................................... 34  
3.2. ACCESSING A MANUAL PAGE FROM A SOFTWARE COLLECTION ...................................... 34  
3.3. DEPLOYING APPLICATIONS THAT USE RED HAT SOFTWARE COLLECTIONS .................. 34  
3.4. RED HAT SOFTWARE COLLECTIONS CONTAINER IMAGES .................................................. 35  

**CHAPTER 4. SPECIFICS OF INDIVIDUAL SOFTWARE COLLECTIONS** ....................................... 38  
4.1. RED HAT DEVELOPER TOOLSET ................................................................................................. 38  
4.2. ECLIPSE 4.6.3 ............................................................................................................................. 38  
  4.2.1. Installing Eclipse ....................................................................................................................... 41  
  4.2.2. Using Eclipse ........................................................................................................................... 41  
    4.2.2.1. Using the Red Hat Developer Toolset Toolchain ............................................................... 41  
    4.2.2.2. Using the Red Hat Enterprise Linux Toolchain ............................................................... 41
4.2.3. Additional Resources
   Installed Documentation
   See Also
4.3. RUBY ON RAILS 5.0
4.4. MONGODB 3.6
4.5. MONGODB 3.4
   MongoDB 3.4 on Red Hat Enterprise Linux 6
   MongoDB 3.4 on Red Hat Enterprise Linux 7
4.6. GIT
4.7. MAVEN
4.8. PASSENGER
4.9. DATABASE CONNECTORS

CHAPTER 5. MIGRATION
5.1. MIGRATING TO MARIADB 10.2
   5.1.1. Notable Differences Between the rh-mariadb101 and rh-mariadb102 Software Collections
   5.1.2. Upgrading from the rh-mariadb101 to the rh-mariadb102 Software Collection
5.2. MIGRATING TO MONGODB 3.6
   5.2.1. Notable Differences Between MongoDB 3.4 and MongoDB 3.6
   General Changes
   Compatibility Changes
   Backwards Incompatible Features
   5.2.2. Upgrading from the rh-mongodb34 to the rh-mongodb36 Software Collection
5.3. MIGRATING TO MONGODB 3.4
   5.3.1. Notable Differences Between MongoDB 3.2 and MongoDB 3.4
   General Changes
   Compatibility Changes
   5.3.2. Upgrading from the rh-mongodb32 to the rh-mongodb34 Software Collection
5.4. MIGRATING TO MYSQL 5.7
   5.4.1. Notable Differences Between MySQL 5.6 and MySQL 5.7
   5.4.2. Upgrading to the rh-mysql57 Software Collection
5.5. MIGRATING TO POSTGRESQL 10
   5.5.1. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 10 Software Collection
   5.5.2. Migrating from the PostgreSQL 9.6 Software Collection to the PostgreSQL 10 Software Collection
5.6. MIGRATING TO POSTGRESQL 9.6
   5.6.1. Notable Differences Between PostgreSQL 9.5 and PostgreSQL 9.6
   5.6.2. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 9.6 Software Collection
   5.6.3. Migrating from the PostgreSQL 9.5 Software Collection to the PostgreSQL 9.6 Software Collection
5.7. MIGRATING TO NGINX 1.12

CHAPTER 6. ADDITIONAL RESOURCES
6.1. RED HAT PRODUCT DOCUMENTATION
6.2. RED HAT DEVELOPERS

APPENDIX A. REVISION HISTORY
CHAPTER 1. RED HAT SOFTWARE COLLECTIONS 3.1

This chapter serves as an overview of the Red Hat Software Collections 3.1 content set. It provides a list of components and their descriptions, sums up changes in this version, documents relevant compatibility information, and lists known issues.

1.1. ABOUT RED HAT SOFTWARE COLLECTIONS

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

Red Hat Software Collections 3.1 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.

1.1.1. Red Hat Developer Toolset

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

1.2. MAIN FEATURES

Table 1.1, "Red Hat Software Collections 3.1 Components" lists all components that are supported at the time of the Red Hat Software Collections 3.1 release.

<table>
<thead>
<tr>
<th>Component</th>
<th>Software Collection</th>
<th>Description</th>
</tr>
</thead>
</table>

4
<table>
<thead>
<tr>
<th>Component</th>
<th>Software Collection</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Developer Toolset 7.1</td>
<td>devtoolset-7</td>
<td>Red Hat Developer Toolset is designed for developers working on the Red Hat Enterprise Linux platform. It provides current versions of the GNU Compiler Collection, GNU Debugger, and other development, debugging, and performance monitoring tools. For a complete list of components, see the Red Hat Developer Toolset Components table in the Red Hat Developer Toolset User Guide.</td>
</tr>
<tr>
<td>Eclipse 4.6.3[a]</td>
<td>rh-eclipse46</td>
<td>A release of the Eclipse integrated development environment that is based on the Eclipse Foundation’s Neon release train. Eclipse was previously available as a Red Hat Developer Toolset component. This Software Collection depends on the rh-java-common component.</td>
</tr>
<tr>
<td>Perl 5.24.0</td>
<td>rh-perl524</td>
<td>A release of Perl, a high-level programming language that is commonly used for system administration utilities and web programming. The rh-perl524 Software Collection provides additional utilities, scripts, and database connectors for MySQL and PostgreSQL. It includes the DateTime Perl module and the mod_perl Apache httpd module, which is supported only with the httpd24 Software Collection. Additionally, it provides the cpanm utility for easy installation of CPAN modules.</td>
</tr>
<tr>
<td>Perl 5.26.1[a]</td>
<td>rh-perl526</td>
<td>A release of Perl, a high-level programming language that is commonly used for system administration utilities and web programming. The rh-perl526 Software Collection provides additional utilities, scripts, and database connectors for MySQL and PostgreSQL. It includes the DateTime Perl module and the mod_perl Apache httpd module, which is supported only with the httpd24 Software Collection. Additionally, it provides the cpanm utility for easy installation of CPAN modules. The rh-perl526 packaging is aligned with upstream; the perl526-perl package installs also core modules, while the interpreter is provided by the perl-interpreter package.</td>
</tr>
<tr>
<td>PHP 7.0.27</td>
<td>rh-php70</td>
<td>A release of PHP 7.0 with PEAR 1.10, enhanced language features and performance improvement.</td>
</tr>
<tr>
<td>PHP 7.1.8[a]</td>
<td>rh-php71</td>
<td>A release of PHP 7.1 with PEAR 1.10, APCu 5.1.8, and enhanced language features.</td>
</tr>
<tr>
<td>Component</td>
<td>Software Collection</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Python 2.7.13</td>
<td>python27</td>
<td>A release of Python 2.7 with a number of additional utilities. This Python version provides various features and enhancements, including an ordered dictionary type, faster I/O operations, and improved forward compatibility with Python 3. The python27 Software Collections contains the Python 2.7.13 interpreter, a set of extension libraries useful for programming web applications and mod_wsgi (only supported with the httpd24 Software Collection), MySQL and PostgreSQL database connectors, and numpy and scipy.</td>
</tr>
<tr>
<td>Python 3.5.1</td>
<td>rh-python35</td>
<td>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.</td>
</tr>
<tr>
<td>Python 3.6.3</td>
<td>rh-python36</td>
<td>The rh-python36 Software Collection contains Python 3.6.3, which introduces a number of new features, such as f-strings, syntax for variable annotations, and asynchronous generators and comprehensions. In addition, a set of extension libraries useful for programming web applications is included, with mod_wsgi (supported only together with the httpd24 Software Collection), PostgreSQL database connector, and numpy and scipy.</td>
</tr>
<tr>
<td>Ruby 2.3.6</td>
<td>rh-ruby23</td>
<td>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, Ruby 2.0.0, and Ruby 1.9.3.</td>
</tr>
<tr>
<td>Ruby 2.4.3</td>
<td>rh-ruby24</td>
<td>A release of Ruby 2.4. This version provides multiple performance improvements and enhancements, for example improved hash table, new debugging features, support for Unicode case mappings, and support for OpenSSL 1.1.0. Ruby 2.4.0 maintains source-level backward compatibility with Ruby 2.3, Ruby 2.2, Ruby 2.0.0, and Ruby 1.9.3.</td>
</tr>
<tr>
<td>Ruby 2.5.0</td>
<td>rh-ruby25</td>
<td>A release of Ruby 2.5. This version provides multiple performance improvements and new features, for example, simplified usage of blocks with the rescue else and ensure keywords, a new yield_self method, support for branch coverage and method coverage measurement, new Hash#slice and Hash#transform_keys methods. Ruby 2.5.0 maintains source-level backward compatibility with Ruby 2.4.</td>
</tr>
<tr>
<td>Component</td>
<td>Software Collection</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Ruby on Rails 4.2.6</td>
<td>rh-ror42</td>
<td>A release of Ruby on Rails 4.2, a 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.</td>
</tr>
<tr>
<td>Ruby on Rails 5.0.1</td>
<td>rh-ror50</td>
<td>A release of Ruby on Rails 5.0, the latest version of the web application framework written in the Ruby language. Notable new features include Action Cable, API mode, exclusive use of rails CLI over Rake, and ActionRecord attributes. This Software Collection is supported together with the rh-ruby24 and rh-nodejs6 Collections.</td>
</tr>
<tr>
<td>Scala 2.10.6 [a]</td>
<td>rh-scala210</td>
<td>A release of Scala, a general purpose programming language for the Java platform, which integrates features of object-oriented and functional languages.</td>
</tr>
<tr>
<td>MariaDB 10.1.29</td>
<td>rh-mariadb101</td>
<td>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.</td>
</tr>
<tr>
<td>MariaDB 10.2.8</td>
<td>rh-mariadb102</td>
<td>A release of MariaDB, an alternative to MySQL for users of Red Hat Enterprise Linux. For all practical purposes, MySQL is binary compatible with MariaDB and can be replaced with it without any data conversions. This version adds MariaDB Backup, Flashback, support for Recursive Common Table Expressions, window functions, and JSON functions.</td>
</tr>
<tr>
<td>MongoDB 3.2.10</td>
<td>rh-mongodb32</td>
<td>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.</td>
</tr>
<tr>
<td>MongoDB 3.4.9</td>
<td>rh-mongodb34</td>
<td>A release of MongoDB, a cross-platform document-oriented database system classified as a NoSQL database. This release introduces support for new architectures, adds message compression and support for the decimal128 type, enhances collation features and more.</td>
</tr>
<tr>
<td>MongoDB 3.6.3 [a]</td>
<td>rh-mongodb36</td>
<td>A release of MongoDB, a cross-platform document-oriented database system classified as a NoSQL database. This release introduces change streams, retryable writes, and JSON Schema, as well as other features.</td>
</tr>
<tr>
<td>Component</td>
<td>Software Collection</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>MySQL 5.7.21</td>
<td>rh-mysql57</td>
<td>A release of MySQL, which provides a number of new features and enhancements, including improved performance.</td>
</tr>
<tr>
<td>PostgreSQL 9.5.9</td>
<td>rh-postgresql95</td>
<td>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.</td>
</tr>
<tr>
<td>PostgreSQL 9.6.5</td>
<td>rh-postgresql96</td>
<td>A release of PostgreSQL, which introduces parallel execution of sequential scans, joins, and aggregates, and provides enhancements to synchronous replication, full-text search, deration driver, postgres_fdw, as well as performance improvements.</td>
</tr>
<tr>
<td>PostgreSQL 10.3[a]</td>
<td>rh-postgresql10</td>
<td>A release of PostgreSQL, which includes a significant performance improvement and a number of new features, such as logical replication using the publish and subscribe keywords, or stronger password authentication based on the SCRAM-SHA-256 mechanism.</td>
</tr>
<tr>
<td>Node.js 4.6.2</td>
<td>rh-nodejs4</td>
<td>A release of Node.js, which provides a JavaScript runtime built on Chrome’s V8 JavaScript engine and npm 2.15.1, a package manager for JavaScript. This version includes an enhanced API, multiple security and bug fixes, and support for the SPDY protocol version 3.1</td>
</tr>
<tr>
<td>Node.js 6.11.3</td>
<td>rh-nodejs6</td>
<td>A release of Node.js, which provides multiple API enhancements, performance and security improvements, ECMAScript 2015 support, and npm 3.10.9.</td>
</tr>
<tr>
<td>Node.js 8.9.4[a]</td>
<td>rh-nodejs8</td>
<td>A release of Node.js, which provides multiple API enhancements and new features, including V8 engine version 6.0, npm 5.6.0 and npx, enhanced security, experimental N-API support, and performance improvements.</td>
</tr>
<tr>
<td>nginx 1.8.1</td>
<td>rh-nginx18</td>
<td>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.</td>
</tr>
<tr>
<td>Component</td>
<td>Software Collection</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>nginx 1.10.2</td>
<td>rh-nginx110</td>
<td>A release of nginx, a web and proxy server with a focus on high concurrency, performance and low memory usage. This version introduces a number of new features, including dynamic module support, HTTP/2 support, Perl integration, and numerous performance improvements.</td>
</tr>
<tr>
<td>nginx 1.12.1[a]</td>
<td>rh-nginx112</td>
<td>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 IP Transparency, improved TCP/UDP load balancing, enhanced caching performance, and numerous performance improvements.</td>
</tr>
<tr>
<td>Apache httpd 2.4.27</td>
<td>httpd24</td>
<td>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.</td>
</tr>
<tr>
<td>Varnish Cache 4.0.3</td>
<td>rh-varnish4</td>
<td>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.</td>
</tr>
<tr>
<td>Varnish Cache 5.2.1[a]</td>
<td>rh-varnish5</td>
<td>A release of Varnish Cache, a high-performance HTTP reverse proxy. This version includes the shard director, experimental HTTP/2 support, and improvements to Varnish configuration through separate VCL files and VCL labels.</td>
</tr>
<tr>
<td>Maven 3.3.9</td>
<td>rh-maven33</td>
<td>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.</td>
</tr>
<tr>
<td>Maven 3.5.0[a]</td>
<td>rh-maven35</td>
<td>A release of Maven, a software project management and comprehension tool. This release introduces support for new architectures and a number of new features, including colorized logging.</td>
</tr>
<tr>
<td>Git 2.9.3</td>
<td>rh-git29</td>
<td>A release of Git, a distributed revision control system with a decentralized architecture. As opposed to centralized version control systems with a client–server model, Git ensures that each working copy of a Git repository is its exact copy with complete revision history.</td>
</tr>
<tr>
<td>Redis 3.2.4</td>
<td>rh-redis32</td>
<td>A release of Redis 3.2, a persistent key-value database.</td>
</tr>
</tbody>
</table>
HAProxy 1.8.4[a]  
**rh-haproxy18**  
A release of HAProxy 1.8, a reliable, high-performance network load balancer for TCP and HTTP-based applications.

Common Java Packages  
**rh-java-common**  
This Software Collection provides common Java libraries and tools used by other collections. The rh-java-common Software Collection is required by the devtoolset-4, devtoolset-3, rh-maven33, maven30, rh-mongodb32, rh-mongodb26, thermostat1, rh-thermostat16, and rh-eclipse46 components and it is not supposed to be installed directly by users.

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

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

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

### Table 1.2. All Available Software Collections

<table>
<thead>
<tr>
<th>Component</th>
<th>Software Collection</th>
<th>Availability</th>
<th>Architectures supported on RHEL7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perl 5.26.1</td>
<td>rh-perl526</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Ruby 2.5.0</td>
<td>rh-ruby25</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>MongoDB 3.6.3</td>
<td>rh-mongodb36</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Varnish Cache 5.2.1</td>
<td>rh-varnish5</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>PostgreSQL 10.3</td>
<td>rh-postgresql10</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>HAProxy 1.8.4</td>
<td>rh-haproxy18</td>
<td>RHEL7</td>
<td>x86_64</td>
</tr>
</tbody>
</table>
### Components Updated in Red Hat Software Collections 3.1

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>RHEL</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Developer Toolset 7.1</td>
<td>devtoolset-7</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64, ppc64le</td>
</tr>
<tr>
<td>PHP 7.0.27</td>
<td>rh-php70</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>MySQL 5.7.21</td>
<td>rh-mysql57</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Apache httpd 2.4.27</td>
<td>httpd24</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
</tbody>
</table>

### Components Last Updated in Red Hat Software Collections 3.0

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>RHEL</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHP 7.1.8</td>
<td>rh-php71</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>nginx 1.12.1</td>
<td>rh-nginx112</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Python 3.6.3</td>
<td>rh-python36</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Maven 3.5.0</td>
<td>rh-maven35</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>MariaDB 10.2.8</td>
<td>rh-mariadb102</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>PostgreSQL 9.6.5</td>
<td>rh-postgresql96</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>MongoDB 3.4.9</td>
<td>rh-mongodb34</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
<tr>
<td>Node.js 8.9.4</td>
<td>rh-nodejs8</td>
<td>RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64le</td>
</tr>
</tbody>
</table>

### Components Last Updated in Red Hat Software Collections 2.4

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>RHEL</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Developer Toolset 6.1</td>
<td>devtoolset-6</td>
<td>RHEL6, RHEL7</td>
<td>x86_64, s390x, aarch64, ppc64, ppc64le</td>
</tr>
<tr>
<td>Scala 2.10.6</td>
<td>rh-scala210</td>
<td>RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Component</td>
<td>Package</td>
<td>RHEL Compatibility</td>
<td>Architecture</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------</td>
<td>--------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>nginx 1.10.2</td>
<td>rh-nginx110</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Node.js 6.11.3</td>
<td>rh-nodejs6</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Ruby 2.4.3</td>
<td>rh-ruby24</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Ruby on Rails 5.0.1</td>
<td>rh-ror50</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Eclipse 4.6.3</td>
<td>rh-eclipse46</td>
<td>RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Python 2.7.13</td>
<td>python27</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Thermostat 1.6.6</td>
<td>rh-thermostat16*</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Maven 3.3.9</td>
<td>rh-maven33</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Common Java Packages</td>
<td>rh-java-common</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Package</th>
<th>RHEL Compatibility</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Git 2.9.3</td>
<td>rh-git29</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Redis 3.2.4</td>
<td>rh-redis32</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Perl 5.24.0</td>
<td>rh-perl524</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Python 3.5.1</td>
<td>rh-python35</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>MongoDB 3.2.10</td>
<td>rh-mongodb32</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>Ruby 2.3.6</td>
<td>rh-ruby23</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>PHP 5.6.25</td>
<td>rh-php56*</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Component</th>
<th>Package</th>
<th>RHEL Compatibility</th>
<th>Architecture</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Developer Toolset 4.1</td>
<td>devtoolset-4*</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>MariaDB 10.1.29</td>
<td>rh-mariadb101</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
<tr>
<td>MongoDB 3.0.11 upgrade collection</td>
<td>rh-mongodb30upg*</td>
<td>RHEL6, RHEL7</td>
<td>x86_64</td>
</tr>
</tbody>
</table>
### Components Last Updated in Red Hat Software Collections 2.2

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node.js 4.6.2</td>
<td>rh-nodejs4</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>PostgreSQL 9.5.9</td>
<td>rh-postgresql95</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Ruby on Rails 4.2.6</td>
<td>rh-ror42</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>MongoDB 2.6.9</td>
<td>rh-mongodb26*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Thermostat 1.4.4</td>
<td>thermostat1*</td>
<td>RHEL6, RHEL7</td>
</tr>
</tbody>
</table>

### Components Last Updated in Red Hat Software Collections 2.1

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Varnish Cache 4.0.3</td>
<td>rh-varnish4</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>nginx 1.8.1</td>
<td>rh-nginx18</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Node.js 0.10</td>
<td>nodejs010*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Maven 3.0.5</td>
<td>maven30*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>V8 3.14.5.10</td>
<td>v83i4*</td>
<td>RHEL6, RHEL7</td>
</tr>
</tbody>
</table>

### Components Last Updated in Red Hat Software Collections 2.0

<table>
<thead>
<tr>
<th>Component</th>
<th>Version</th>
<th>Platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red Hat Developer Toolset 3.1</td>
<td>devtoolset-3*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Perl 5.20.1</td>
<td>rh-perl520*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Python 3.4.2</td>
<td>rh-python34*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Ruby 2.2.9</td>
<td>rh-ruby22*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Ruby on Rails 4.1.5</td>
<td>rh-ror41*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>MariaDB 10.0.33</td>
<td>rh-mariadb100*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>MySQL 5.6.40</td>
<td>rh-mysql56*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>PostgreSQL 9.4.14</td>
<td>rh-postgresql94*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Passenger 4.0.50</td>
<td>rh-passenger40*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td>Components Last Updated in Red Hat Software Collections 2.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------------------------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>PHP 5.4.40</strong></td>
<td>php54*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td><strong>PHP 5.5.21</strong></td>
<td>php55*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td><strong>nginx 1.6.2</strong></td>
<td>nginx16*</td>
<td>RHEL6, RHEL7</td>
</tr>
<tr>
<td><strong>DevAssistant 0.9.3</strong></td>
<td>devassist09*</td>
<td>RHEL6, RHEL7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Components Last Updated in Red Hat Software Collections 1</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Git 1.9.4</strong></td>
</tr>
<tr>
<td><strong>Perl 5.16.3</strong></td>
</tr>
<tr>
<td><strong>Python 3.3.2</strong></td>
</tr>
<tr>
<td><strong>Ruby 1.9.3</strong></td>
</tr>
<tr>
<td><strong>Ruby 2.0.0</strong></td>
</tr>
<tr>
<td><strong>Ruby on Rails 4.0.2</strong></td>
</tr>
<tr>
<td><strong>MariaDB 5.5.53</strong></td>
</tr>
<tr>
<td><strong>MongoDB 2.4.9</strong></td>
</tr>
<tr>
<td><strong>MySQL 5.5.52</strong></td>
</tr>
<tr>
<td><strong>PostgreSQL 9.2.18</strong></td>
</tr>
</tbody>
</table>

RHEL6 – Red Hat Enterprise Linux 6

RHEL7 – Red Hat Enterprise Linux 7

x86_64 – AMD64 and Intel 64 architectures

s390x – IBM z Systems

aarch64 – The 64-bit ARM architecture

ppc64 – IBM POWER, big endian

ppc64le – IBM POWER, little endian

* Retired component – this Software Collection is no longer supported
The tables above list the latest versions available through asynchronous updates.

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

**1.3. CHANGES IN RED HAT SOFTWARE COLLECTIONS 3.1**

**1.3.1. Overview**

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

In addition, Red Hat Software Collections 3.1 supports the following architectures on Red Hat Enterprise Linux 7:

- The 64-bit ARM architecture
- IBM z Systems
- IBM POWER, little endian

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

**New Software Collections**
Red Hat Software Collections 3.1 adds these new Software Collections:

- `rh-perl526` – see Section 1.3.3, “Changes in Perl”
- `rh-ruby25` – see Section 1.3.4, “Changes in Ruby”
- `rh-mongodb36` – see Section 1.3.5, “Changes in MongoDB”
- `rh-postgresql10` – see Section 1.3.6, “Changes in PostgreSQL”
- `rh-varnish5` – see Section 1.3.7, “Changes in Varnish Cache”
- `rh-haproxy18` – see Section 1.3.8, “Changes in HAProxy”

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

**Updated Software Collections**
The following components have been updated in Red Hat Software Collections 3.1:

- `devtoolset-7` – see Section 1.3.2, “Changes in Red Hat Developer Toolset”
- `rh-php70` – see Section 1.3.9, “Changes in PHP”
- `rh-mysql57` – see Section 1.3.10, “Changes in MySQL”
- `httpd24` – see Section 1.3.11, “Changes in Apache httpd”

**Red Hat Software Collections Container Images**
The following container images are new in Red Hat Software Collections 3.1:
The following container images have been updated in Red Hat Software Collections 3.1:

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

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

1.3.2. Changes in Red Hat Developer Toolset

The following components have been upgraded in Red Hat Developer Toolset 7.1 compared to the previous release of Red Hat Developer Toolset:

- GCC to version 7.3.1

In addition, bug fix updates are available for the following components:

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

For detailed information on changes in 7.1, see the Red Hat Developer Toolset User Guide.
1.3.3. Changes in Perl

The new rh-perl526 Software Collection includes Perl 5.26.1, which provides a number of bug fixes and enhancements over the previous version. Some of the notable changes with widespread effects are:

- The current directory has been removed from the `@INC` module search path for security reasons
- The `do` statement now returns a deprecation warning when it fails to load a file because of the behavioral change described above
- Unescaped literal `{` characters in regular expression patterns are no longer permissible
- Unicode 9.0 is now supported

For detailed changes in Perl 5.26, see the upstream change log for versions 5.26.0 and 5.26.1.

In addition, the rh-perl526 packaging is aligned with upstream. The rh-perl526-perl package installs also core modules, while the `/usr/bin/perl` interpreter is provided by the rh-perl526-perl-interpreter package. In previous releases, the perl package included just a minimal interpreter, whereas the perl-core package included both the interpreter and the core modules.

1.3.4. Changes in Ruby

The new rh-ruby25 Software Collection provides Ruby 2.5.0. This version introduces performance improvements and a number of new features and changes, for example:

- The `rescue`, `else`, and `ensure` keywords can now be used directly within blocks delimited by `do` and `end` without the need to add the `begin` and `end` keywords
- A new `yield_self` method has been added to return the result of the block
- Support for branch coverage and method coverage measurement has been added
- New `Hash#slice` and `Hash#transform_keys` methods have been implemented
- `Struct` subclass constructors can accept keyword arguments
- Top-level constant look-up is no longer available

Ruby 2.5.0 maintains source-level backward compatibility with Ruby 2.4.

For detailed changes in Ruby 2.5.0, see the upstream release notes.

1.3.5. Changes in MongoDB

The new rh-mongodb36 Software Collection includes MongoDB 3.6.3, which provides a number of bug fixes and enhancements over the previous version. The most notable new features are:

- Change streams, which enable applications to access real-time data changes and immediately react to them
- Retryable writes, which enable MongoDB drivers to automatically retry certain write operations, for example, in case of a network error
- JSON Schema, which enables users to define their own criteria for validating JSON documents
The following subpackages have also been upgraded:

- mongo-c-driver to version 1.9.2
- mongo-cxx-driver to version 3.2.0
- mongo-tools to version 3.6.3
- mongo-java-driver to version 3.6.3

For detailed changes in **MongoDB 3.6**, refer to the [upstream release notes](#).

**NOTE**

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

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

For instructions regarding migration, see Section 5.2, “Migrating to MongoDB 3.6”.

### 1.3.6. Changes in PostgreSQL

The new rh-postgresql10 Software Collection provides **PostgreSQL 10.3**. Major enhancements in this version include:

- Logical replication using the `publish` and `subscribe` keywords
- Stronger password authentication based on the `SCRAM-SHA-256` mechanism
- Declarative table partitioning
- Improved query parallelism
- Significant general performance improvements
- Improved monitoring and control

For detailed changes in **PostgreSQL 10.3**, see the [upstream documentetation](#).

The rh-postgresql10 Software Collection includes the rh-postgresql10-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-postgresql10*-syspaths packages, users are not required to use the `scl enable` command for correct functioning of the binaries and scripts provided by the rh-postgresql10* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system. To find out more about syspaths, see the [Red Hat Software Collections Packaging Guide](#).

For information on migration, see Section 5.5, “Migrating to PostgreSQL 10”.
1.3.7. Changes in Varnish Cache

The new rh-varnish5 Software Collection includes Varnish Cache 5.2.1, which provides a number of bug fixes and enhancements over the previous version. Notably:

- Improvements to configuration through separate Varnish Configuration Language (VCL) files and VCL labels, which can reduce complexity in configuration
- Experimental HTTP/2 support
- The shard director
- Improved stability

For detailed changes in Varnish Cache 5.2.1, refer to the upstream change log. See also the upstream documentation and upgrading notes.

1.3.8. Changes in HAProxy

The new rh-haproxy18 Software Collection provides HAProxy 1.8.4. The most notable changes in this version include:

- HTTP request and response captures
- Multiprocess peers and stick-tables
- Server IP resolution using DNS at runtime
- Email alerts
- Seamless server states
- HTTP/2 support (experimental)
- Multithreading (experimental)
- Seamless reloads
- DNS for service discovery
- Improved Runtime API

For detail changes in HAProxy since version 1.5, included in Red Hat Enterprise Linux 7, see the HAProxy Configuration Manual.

1.3.9. Changes in PHP

The rh-php70 Software Collection has been upgraded to version 7.0.27, which provides a number of bug fixes and enhancements over the version released in Red Hat Software Collections 2.3.

For detailed information on bug fixes and enhancements in this release, see the upstream change log for version 7.0.27 and earlier.

1.3.10. Changes in MySQL

This update adds support for the following architectures to the rh-mysql57 Software Collection:
The 64-bit ARM architecture

IBM z Systems

IBM POWER, little endian

1.3.11. Changes in Apache httpd

This update adds the httpd24-mod_auth_mellon package to the httpd24 Software Collection. The mod_auth_mellon module for the Apache HTTP Server is an authentication service that implements the SAML 2.0 federation protocol. The module grants access based on the attributes received in assertions generated by an Identity Provider (IdP) server.

1.4. COMPATIBILITY INFORMATION

Red Hat Software Collections 3.1 is available for all supported releases of Red Hat Enterprise Linux 7 on AMD64 and Intel 64 architectures, the 64-bit ARM architecture, IBM z Systems, and IBM POWER, little endian.

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

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

1.5. KNOWN ISSUES

httpd24 component, BZ#1429006
Since httpd 2.4.27, the mod_http2 module is no longer supported with the default prefork Multi-Processing Module (MPM). To enable HTTP/2 support, edit the configuration file at /opt/rh/httpd24/root/etc/httpd/conf.modules.d/00-mpm.conf and switch to the event or worker MPM.

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

httpd24 component, BZ#1327548
The mod_ssl module does not support the ALPN protocol on Red Hat Enterprise Linux 6, or on Red Hat Enterprise Linux 7.3 and earlier. Consequently, clients that support upgrading TLS connections to HTTP/2 only using ALPN are limited to HTTP/1.1 support.

httpd24 component, BZ#1224763
When using the mod_proxy_fcgi module with FastCGI Process Manager (PHP-FPM), httpd uses port 8000 for the FastCGI protocol by default instead of the correct port 9000. To work around this problem, specify the correct port explicitly in configuration.

httpd24 component, BZ#1382706
When SELinux is enabled, the LD_LIBRARY_PATH environment variable is not passed through to CGI scripts invoked by httpd. As a consequence, in some cases it is impossible to invoke executables from Software Collections enabled in the /opt/rh/httpd24/service-environment file from CGI scripts run by httpd. To work around this problem, set LD_LIBRARY_PATH as desired from within the CGI script.
httpd24 component

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

rh-python34, rh-python35, rh-python36 components, BZ#1499990

The pytz module, which is used by Babel for time zone support, is not included in the rh-python34, rh-python35, and rh-python36 Software Collections. Consequently, when the user tries to import the dates module from Babel, a traceback is returned. To work around this problem, install pytz through the pip package manager from the pypi public repository by using the pip install pytz command.

rh-python36 component

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

python27 component, BZ#1330489

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

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.

scl-utils component

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

rh-ruby24, rh-ruby23 components

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

rh-maven35, rh-maven33 components

When the user has installed both the Red Hat Enterprise Linux system version of maven-local package and the rh-maven35-maven-local package or rh-maven33-maven-local package, XMvn, a tool used for building Java RPM packages, run from the rh-maven35 or 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.

perl component

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

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.

rh-mysql57, rh-mysql56, rh-mariadb100, rh-mariadb101 components, BZ#1194611

The rh-mysql57-mysql-server, 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.

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 3.1. Additionally, the mongodb24-build and mongodb24-scldevel packages cannot be installed with Red Hat Software Collections 3.1 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.

**mariadb, mysql, postgresql, mongodb components**

Red Hat Software Collections 3.1 contains the MySQL 5.6, MySQL 5.7, MariaDB 10.0, MariaDB 10.1, MariaDB 10.2, PostgreSQL 9.4, PostgreSQL 9.5, PostgreSQL 9.6, PostgreSQL 10, MongoDB 2.6, MongoDB 3.2, MongoDB 3.4, and MongoDB 3.6 databases. The core Red Hat Enterprise Linux 6 provides earlier versions of the MySQL and PostgreSQL databases (client library and daemon). The core Red Hat Enterprise Linux 7 provides earlier versions of the MariaDB and PostgreSQL databases (client library and daemon). Client libraries are also used in database connectors for dynamic languages, libraries, and so on.

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

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

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

**mariadb, mysql, mongodb components**

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

**rh-eclipse46 component**

When a plug-in from a third-party update site is installed, Eclipse sometimes fails to start with a `NullPointerException` in the workspace log file. To work around this problem, restart Eclipse with the `-clean` option. For example:

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

**rh-eclipse46 component**

The Eclipse Docker Tooling introduces a Dockerfile editor with syntax highlighting and a basic command auto-completion. When the Build Image Wizard is open and the Edit Dockerfile button is pressed, the Dockerfile editor opens the file in a detached editor window. However, this window does not contain the Cancel and Save buttons. To work around this problem, press Ctrl+S to save your changes or right-click in the editor to launch a context menu, which offers the Save option. To cancel your changes, close the window.

**rh-eclipse46 component**

On Red Hat Enterprise Linux 7.2, a bug in the perf tool, which is used to populate the Perf Profile View in Eclipse, causes some of the items in the view not to be properly linked to their respective positions in the Eclipse Editor. While the profiling works as expected, it is not possible to navigate to related positions in the Editor by clicking on parts of the Perf Profile View.
**rh-thermostat16 component**

Due to typos in the desktop application file, users are unable to launch *Thermostat* using the desktop icon. To work around this problem, modify the `/usr/share/applications/rh-thermostat16-thermostat.desktop` file from:

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

To:

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

Alternatively, run *Thermostat* from command line:

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

**Other Notes**


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

- Ruby gems cannot be installed while the rh-ruby22 or rh-ruby23 Software Collection is on a read-only NFS. Consequently, for example, when the user tries to install the ab gem using the `gem install ab` command, an error message is displayed, for example:

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

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

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

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

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

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

**httpd component**

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

**all components**

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

**coreutils, util-linux, screen components**

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

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

instead of

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

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

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

**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, rh-python34, and rh-python35 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, rh-php56, and rh-php70 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, rh-ruby22, and rh-ruby23 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, rh-perl520, and rh-perl524 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.

**rh-eclipse46 component**

The Eclipse SWT graphical library on Red Hat Enterprise Linux 7 uses GTK 3.x. Eclipse Dark Theme is not yet fully stable on GTK 3.x, so this theme is considered a Technology Preview and not supported. For more information about Red Hat Technology Previews, see https://access.redhat.com/support/offerings/techpreview/.

### 1.6. DEPRECATED FUNCTIONALITY

**httpd24 component, BZ\#1434053**

Previously, in an SSL/TLS configuration requiring name-based SSL virtual host selection, the mod_ssl module rejected requests with a **400 Bad Request** error, if the host name provided in the Host: header did not match the host name provided in a Server Name Indication (SNI) header. Such requests are no longer rejected if the configured SSL/TLS security parameters are identical between the selected virtual hosts, in-line with the behavior of upstream mod_ssl.
CHAPTER 2. INSTALLATION

This chapter describes in detail how to get access to the content set, install Red Hat Software Collections 3.1 on the system, and rebuild Red Hat Software Collections.

2.1. GETTING ACCESS TO RED HAT SOFTWARE COLLECTIONS

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

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

NOTE

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

Beta content is unavailable in the ISO format.

2.1.1. Using Red Hat Subscription Management

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

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

   ```shell
   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:

   ```shell
   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:

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

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

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

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

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

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

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

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

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

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

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

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

**NOTE**

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

### 2.1.2. Packages from the Optional Channel

Some of the Red Hat Software Collections 3.1 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 at https://access.redhat.com/solutions/392003 for Red Hat Subscription Management or at 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 That Require Enabling of the Optional Channel in Red Hat Enterprise Linux 6**
<table>
<thead>
<tr>
<th>Package from a Software Collection</th>
<th>Required Package from the Optional Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>devtoolset-6-dyninst-testsuite</td>
<td>glibc-static</td>
</tr>
<tr>
<td>devtoolset-7-dyninst-testsuite</td>
<td>glibc-static</td>
</tr>
<tr>
<td>rh-git29-git-all</td>
<td>cvsp, perl-Net-SMTP-SSL</td>
</tr>
<tr>
<td>rh-git29-git-cvs</td>
<td>cvsp</td>
</tr>
<tr>
<td>rh-git29-git-email</td>
<td>perl-Net-SMTP-SSL</td>
</tr>
<tr>
<td>rh-git29-perl-Git-SVN</td>
<td>perl-YAML, subversion-perl</td>
</tr>
<tr>
<td>rh-mariadb101-boost-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mariadb101-boost-examples</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mariadb101-boost-static</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb30upg-boost-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb30upg-boost-examples</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb30upg-boost-static</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb32-boost-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb32-boost-examples</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb32-boost-static</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb32-yaml-cpp-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb34-boost-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb34-boost-examples</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb34-boost-static</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-mongodb34-yaml-cpp-devel</td>
<td>libc-devel</td>
</tr>
<tr>
<td>rh-php56-php-imap</td>
<td>libc-client</td>
</tr>
</tbody>
</table>
Software Collections packages that require the Optional channel in Red Hat Enterprise Linux 7 are listed in the table below.

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

<table>
<thead>
<tr>
<th>Package from a Software Collection</th>
<th>Required Package from the Optional Channel</th>
</tr>
</thead>
<tbody>
<tr>
<td>devtoolset-7-dyninst-testsuite</td>
<td>glibc-static</td>
</tr>
<tr>
<td>devtoolset-7-gcc-plugin-devel</td>
<td>libmpc-devel</td>
</tr>
<tr>
<td>httpd24-mod_ldap</td>
<td>apr-util-ldap</td>
</tr>
<tr>
<td>rh-eclipse46</td>
<td>ruby-doc</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-dltk-ruby</td>
<td>ruby-doc</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-dltk-sdk</td>
<td>ruby-doc</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-dltk-tests</td>
<td>ruby-doc</td>
</tr>
<tr>
<td>rh-git29-git-all</td>
<td>cvspsp</td>
</tr>
<tr>
<td>rh-git29-git-cvs</td>
<td>cvspsp</td>
</tr>
<tr>
<td>rh-git29-perl-Git-SVN</td>
<td>subversion-perl</td>
</tr>
<tr>
<td>rh-perl520-perl-Perldoc</td>
<td>groff</td>
</tr>
</tbody>
</table>

Note that packages from the Optional channel are not supported. For details, see the Knowledgebase article at 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 3.1 requires the removal of any earlier pre-release versions, including Beta releases. If you have installed any previous version of Red Hat Software Collections 3.1, uninstall it from your system and install the new version as described in the Section 2.3, “Uninstalling Red Hat Software Collections” and Section 2.2.1, “Installing Individual Software Collections” sections.

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

2.2.1. Installing Individual Software Collections

To install any of the Software Collections that are listed in Table 1.1, “Red Hat Software Collections 3.1 Components”, install the corresponding meta package by typing the following at a shell prompt as root:

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

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

```
~# yum install rh-php56 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-perl524-perl-CPAN rh-perl524-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 you need to have access to the repository with these packages. If your system is registered with Red Hat Subscription Management, enable the `rhel-variant-rhscl-6-debug-rpms` or `rhel-variant-rhscl-7-debug-rpms` repository as described in Section 2.1.1, “Using Red Hat Subscription Management”. For more information on how to get access to debuginfo packages, see https://access.redhat.com/solutions/9907.

2.3. UNINSTALLING RED HAT SOFTWARE COLLECTIONS

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

```
yum remove software_collection
```

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

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

2.4. REBUILDING RED HAT SOFTWARE COLLECTIONS

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

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

For detailed information on building Software Collections, refer to the Red Hat Software Collections Packaging Guide.
CHAPTER 3. USAGE

This chapter describes the necessary steps for rebuilding and using Red Hat Software Collections 3.1, 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:

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

Or, alternatively, use the following command:

```bash
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 rh-perl524 'perl hello.pl'
Hello, World!
```

You can execute any command using the `scl` utility, causing it to be run with the executables from a selected Software Collection in preference to their possible Red Hat Enterprise Linux system equivalents. For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.1 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:

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

Replace `software_collection` with a space-separated list of Software Collections you want to use. For example, to start a new shell session with the python27 and `rh-postgresql95` Software Collections as default, type:

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

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

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

For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.1 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 `rh-postgresql95` Software Collection and enable it in runlevels 2, 3, 4, and 5, type as `root`:

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

For more information on how to manage system services in Red Hat Enterprise Linux 6, refer to the Red Hat Enterprise Linux 6 Deployment Guide. For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 3.1 Components”.

3.2. ACCESSING A MANUAL PAGE FROM A SOFTWARE COLLECTION

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

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

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

Replace `software_collection` with the particular Red Hat Software Collections component. For example, to display the manual page for `rh-mariadb101`, type:

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

3.3. DEPLOYING APPLICATIONS THAT USE RED HAT SOFTWARE COLLECTIONS

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

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

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

3.4. RED HAT SOFTWARE COLLECTIONS CONTAINER IMAGES

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

The following container images are available with Red Hat Software Collections 3.1:

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

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

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

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

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

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

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

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

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

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

Images marked as End of Life (EOL) are no longer supported.
CHAPTER 4. SPECIFICS OF INDIVIDUAL SOFTWARE COLLECTIONS

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

4.1. RED HAT DEVELOPER TOOLSET

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

For an overview of features, refer to the 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. ECLIPSE 4.6.3

The rh-eclipse46 Software Collection, available for Red Hat Enterprise Linux 7, includes Eclipse 4.6.3, which is based on the Eclipse Foundation’s Neon release train. This integrated development environment was previously available as a part of Red Hat Developer Toolset. Note that the rh-eclipse46 Software Collection requires the rh-java-common Collection.

NOTE

A new version of Eclipse is now available as the rh-eclipse47 component of the Red Hat Developer Tools offering. For more information, see Using Eclipse.

Eclipse is a powerful development environment that provides tools for each phase of the development process. It integrates a variety of disparate tools into a unified environment to create a rich development experience, provides a fully configurable user interface, and features a pluggable architecture that allows for an extension in a variety of ways. For instance, the Valgrind plug-in allows programmers to perform memory profiling, otherwise performed on the command line, through the Eclipse user interface.
Figure 4.1. Sample Eclipse Session

Eclipse provides a graphical development environment alternative to traditional interaction with command line tools and as such, it is a welcome alternative to developers who do not want to use the command line interface. The traditional, mostly command line-based Linux tools suite (such as gcc or gdb) and Eclipse offer two distinct approaches to programming.

Note that if you intend to develop applications for Red Hat JBoss Middleware or require support for OpenShift Tools, it is recommended that you use Red Hat JBoss Developer Studio.

Table 4.1. Eclipse Components Included in the rh-eclipse46 Software Collection

<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>rh-eclipse46-eclipse-cdt</td>
<td>The C/C++ Development Tooling (CDT), which provides features and plug-ins for development in C and C++.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-changelog</td>
<td>The ChangeLog plug-in, which allows you to create and maintain changelog files.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-egit</td>
<td>EGit, a team provider for Eclipse that provides features and plug-ins for interaction with Git repositories.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-emf</td>
<td>The Eclipse Modeling Framework (EMF), which allows you to build applications based on a structured data model.</td>
</tr>
<tr>
<td>Package</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-epp-logging</td>
<td>The Eclipse error reporting tool.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-gcov</td>
<td>The GCov plug-in, which integrates the GCov test coverage program with Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-gef</td>
<td>The Graphical Editing Framework (GEF), which allows you to create a rich graphical editor from an existing application model.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-gprof</td>
<td>The Gprof plug-in, which integrates the Gprof performance analysis utility with Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-jdt</td>
<td>The Eclipse Java development tools (JDT) plug-in.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-jgit</td>
<td>JGit, a Java implementation of the Git revision control system.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-manpage</td>
<td>The Man Page plug-in, which allows you to view manual pages in Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-mpc</td>
<td>The Eclipse Marketplace Client.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-mylyn</td>
<td>Mylyn, a task management system for Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-oprofile</td>
<td>The OProfile plug-in, which integrates OProfile with Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-pde</td>
<td>The Plugin Development Environment for developing Eclipse plugins.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-perf</td>
<td>The Perf plug-in, which integrates the perf tool with Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-ptp</td>
<td>A subset of the PTP project providing support for synchronized projects.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-pydev</td>
<td>A full featured Python IDE for Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-remote</td>
<td>The Remote Services plug-in, which provides an extensible remote-services framework.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-rpm-editor</td>
<td>The Eclipse Spec File Editor, which allows you to maintain RPM spec files.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-rse</td>
<td>The Remote System Explorer (RSE) framework, which allows you to work with remote systems from Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-systemtap</td>
<td>The SystemTap plug-in, which integrates SystemTap with Eclipse.</td>
</tr>
<tr>
<td>rh-eclipse46-eclipse-valgrind</td>
<td>The Valgrind plug-in, which integrates Valgrind with Eclipse.</td>
</tr>
</tbody>
</table>
### 4.2.1. Installing Eclipse

The **Eclipse** development environment is provided as a collection of RPM packages. To install the **rh-eclipse46** Software Collection, type the following command as root:

```
yum install rh-eclipse46
```

For a list of available components, see Table 4.1, “Eclipse Components Included in the **rh-eclipse46** Software Collection”.

**NOTE**

The **rh-eclipse46** Software Collection fully supports C, C++, and Java development, but does not provide support for the Fortran programming language.

### 4.2.2. Using Eclipse

To start the **rh-eclipse46** Software Collection, either select **Applications → Programming → Red Hat Eclipse** from the panel, or type the following at a shell prompt:

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

During its startup, **Eclipse** prompts you to select a workspace, that is, a directory in which you want to store your projects. You can either use `~/.workspace/`, which is the default option, or click the **Browse** button to browse your file system and select a custom directory. Additionally, you can select the **Use this as the default and do not ask again** check box to prevent **Eclipse** from displaying this dialog box the next time you run this development environment. When you are done, click the **OK** button to confirm the selection and proceed with the startup.

#### 4.2.2.1. Using the Red Hat Developer Toolset Toolchain

To use the **rh-eclipse46** Software Collection with support for the **GNU Compiler Collection** and **binutils** from Red Hat Developer Toolset, make sure that the **devtoolset-7-toolchain** package is installed and run the application as described in Section 4.2.2, "Using Eclipse”. The **rh-eclipse46** Collection uses the Red Hat Developer Toolset toolchain by default.

For detailed instructions on how to install the devtoolset-7-toolchain package in your system, see the **Red Hat Developer Toolset User Guide**.

**IMPORTANT**

If you are working on a project that you previously built with the Red Hat Enterprise Linux version of the **GNU Compiler Collection**, make sure that you discard all previous build results. To do so, open the project in **Eclipse** and select **Project → Clean** from the menu.

#### 4.2.2.2. Using the Red Hat Enterprise Linux Toolchain
To use the rh-eclipse46 Software Collection with support for the toolchain distributed with Red Hat Enterprise Linux, change the configuration of the project to use absolute paths to the Red Hat Enterprise Linux system versions of gcc, g++, and as.

To configure Eclipse to explicitly use the Red Hat Enterprise Linux system versions of the tools for the current project, complete the following steps:

1. In the C/C++ perspective, choose Project → Properties from the main menu bar to open the project properties.

2. In the menu on the left-hand side of the dialog box, click C/C++ Build → Settings.

3. Select the Tool Settings tab.

4. If you are working on a C project:
   1. select GCC C Compiler or Cross GCC Compiler and change the value of the Command field to:

          /usr/bin/gcc

   2. select GCC C Linker or Cross GCC Linker and change the value of the Command field to:

          /usr/bin/gcc

   3. select GCC Assembler or Cross GCC Assembler and change the value of the Command field to:

          /usr/bin/as

If you are working on a C++ project:

1. select GCC C++ Compiler or Cross G++ Compiler and change the value of the Command field to:

          /usr/bin/g++

2. select GCC C Compiler or Cross GCC Compiler and change the value of the Command field to:

          /usr/bin/gcc

3. select GCC C++ Linker or Cross G++ Linker and change the value of the Command field to:

          /usr/bin/g++

4. select GCC Assembler or Cross GCC Assembler and change the value of the Command field to:

          /usr/bin/as

5. Click the OK button to save the configuration changes.
4.2.3. Additional Resources

A detailed description of Eclipse and all its features is beyond the scope of this book. For more information, see the resources listed below.

Installed Documentation

- Eclipse includes a built-in Help system, which provides extensive documentation for each integrated feature and tool. This greatly decreases the initial time investment required for new developers to become fluent in its use. The use of this Help section is detailed in the Red Hat Enterprise Linux Developer Guide linked below.

See Also

- Using Eclipse describing usage of the rh-eclipse47 component of Red Hat Developer Tools.
- The Red Hat Developer Toolset chapter in the Red Hat Developer Toolset User Guide provides an overview of Red Hat Developer Toolset and more information on how to install it on your system.
- The GNU Compiler Collection (GCC) chapter in the Red Hat Developer Toolset User Guide provides information on how to compile programs written in C, C++, and Fortran on the command line.

4.3. RUBY ON RAILS 5.0

Red Hat Software Collections 3.1 provides the rh-ruby24 Software Collection together with the rh-ror50 Collection.

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

```
yum install rh-ror50
```

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

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

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

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

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

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

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

The rh-ror50 Software Collection is supported together with the rh-ruby24 and rh-nodejs6 components.
4.4. MONGODB 3.6

The rh-mongodb36 Software Collection is available only for Red Hat Enterprise Linux 7. See Section 4.5, “MongoDB 3.4” for instructions on how to use MongoDB 3.4 on Red Hat Enterprise Linux 6.

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

```bash
yum install rh-mongodb36
```

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

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

**NOTE**

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

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

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

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

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

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

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

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

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

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

4.5. MONGODB 3.4

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

```bash
yum install rh-mongodb34
```

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

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

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

MongoDB 3.4 on Red Hat Enterprise Linux 6
If you are using Red Hat Enterprise Linux 6, the following instructions apply to your system.

To start the MongoDB daemon, type the following command as \texttt{root}:

\begin{verbatim}
  service rh-mongodb34-mongod start
\end{verbatim}

To start the MongoDB daemon on boot, type this command as \texttt{root}:

\begin{verbatim}
  chkconfig rh-mongodb34-mongod on
\end{verbatim}

To start the MongoDB sharding server, type this command as \texttt{root}:

\begin{verbatim}
  service rh-mongodb34-mongos start
\end{verbatim}

To start the MongoDB sharding server on boot, type the following command as \texttt{root}:

\begin{verbatim}
  chkconfig rh-mongodb34-mongos on
\end{verbatim}

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

MongoDB 3.4 on Red Hat Enterprise Linux 7
When using Red Hat Enterprise Linux 7, the following commands are applicable.

To start the MongoDB daemon, type the following command as \texttt{root}:

\begin{verbatim}
  systemctl start rh-mongodb34-mongod.service
\end{verbatim}

To start the MongoDB daemon on boot, type this command as \texttt{root}:

\begin{verbatim}
  systemctl enable rh-mongodb34-mongod.service
\end{verbatim}

To start the MongoDB sharding server, type the following command as \texttt{root}:

\begin{verbatim}
  systemctl start rh-mongodb34-mongos.service
\end{verbatim}

To start the MongoDB sharding server on boot, type this command as \texttt{root}:

\begin{verbatim}
  systemctl enable rh-mongodb34-mongos.service
\end{verbatim}

Note that the MongoDB sharding server does not work unless the user starts at least one configuration server and specifies it in the \texttt{mongos.conf} file.
4.6. GIT

Git is a distributed revision control system with a decentralized architecture. As opposed to centralized version control systems with a client-server model, Git ensures that each working copy of a Git repository is an exact copy with complete revision history. This not only allows you to work on and contribute to projects without the need to have permission to push your changes to their official repositories, but also makes it possible for you to work with no network connection. For detailed information, see the Git chapter in the Red Hat Enterprise Linux 7 Developer Guide.

4.7. MAVEN

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

To install the rh-maven35 Collection, type the following command as root:

```
yum install rh-maven35
```

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

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

Global Maven settings, such as remote repositories or mirrors, can be customized by editing the `/opt/rh/rh-maven35/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.

4.8. 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 application’s directory:

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

Alternatively, edit the nginx16 configuration files as described in the upstream Passenger documentation.
### 4.9. DATABASE CONNECTORS

Database connector packages provide the database client functionality, which is necessary for local or remote connection to a database server. Table 4.2, “Interoperability Between Languages and Databases” lists Software Collections with language runtimes that include connectors for certain database servers.

**Table 4.2. Interoperability Between Languages and Databases**

<table>
<thead>
<tr>
<th>Language (Software Collection)</th>
<th>MariaDB</th>
<th>MongoDB</th>
<th>MySQL</th>
<th>PostgreSQL</th>
<th>Redis</th>
</tr>
</thead>
<tbody>
<tr>
<td>rh-nodejs4</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>rh-nodejs6</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>rh-nodejs8</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>rh-perl520</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-perl524</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-perl526</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-php56</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-php70</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-php71</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>python27</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-python34</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>rh-python35</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>Language (Software Collection)</td>
<td>MariaDB</td>
<td>MongoDB</td>
<td>MySQL</td>
<td>PostgreSQL</td>
<td>Redis</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------</td>
<td>---------</td>
<td>-------</td>
<td>------------</td>
<td>-------</td>
</tr>
<tr>
<td>rh-python36</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>rh-ror41</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>rh-ror42</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>rh-ror50</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>rh-ruby25</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
<tr>
<td>Supported</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>❌</td>
<td></td>
</tr>
<tr>
<td>Unsupported</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
</tr>
</tbody>
</table>
CHAPTER 5. MIGRATION

This chapter provides information on migrating to versions of components included in Red Hat Software Collections 3.1.

5.1. MIGRATING TO MARIADB 10.2

Red Hat Enterprise Linux 6 contains MySQL 5.1 as the default MySQL implementation. Red Hat Enterprise Linux 7 includes MariaDB 5.5 as the default MySQL implementation. MariaDB is a community-developed drop-in replacement for MySQL. MariaDB 10.1 has been available as a Software Collection since Red Hat Software Collections 2.2; Red Hat Software Collections 3.1 is distributed with MariaDB 10.2.

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

Note that if you are using MariaDB 5.5 or MariaDB 10.0, it is necessary to upgrade to the rh-mariadb101 Software Collection first, which is described in the Red Hat Software Collections 2.4 Release Notes.

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

NOTE

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

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

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

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

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

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

5.1.2. Upgrading from the rh-mariadb101 to the rh-mariadb102 Software Collection
IMPORTANT

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

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

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

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

   Stop the rh-mariadb101 server.

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

2. Install the rh-mariadb102 Software Collection.

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

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


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

5. Start the rh-mariadb102 database server.

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

6. Perform the data migration.

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

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

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

5.2. MIGRATING TO MONGODB 3.6

Red Hat Software Collections 3.1 is released with MongoDB 3.6, provided by the rh-mongodb36 Software Collection and available only for Red Hat Enterprise Linux 7.
The rh-mongodb36 Software Collection includes the rh-mongodb36-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-mongodb36*-syspaths packages, users are not required to use the `scl enable` command for correct functioning of the binaries and scripts provided by the rh-mongodb36* packages. To find out more about syspaths, see the Red Hat Software Collections Packaging Guide.

5.2.1. Notable Differences Between MongoDB 3.4 and MongoDB 3.6

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

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

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

- MongoDB binaries now bind to `localhost` by default, so listening on different IP addresses needs to be explicitly enabled. Note that this is already the default behavior for `systemd` services distributed with MongoDB Software Collections.

- The MONGODB-CR authentication mechanism has been deprecated. For databases with users created by MongoDB versions earlier than 3.0, upgrade authentication schema to SCRAM.

- The HTTP interface and REST API have been removed

- Arbiters in replica sets have priority 0

- Master-slave replication has been deprecated

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

Backwards Incompatible Features
The following MongoDB 3.6 features are backwards incompatible and require the version to be set to 3.6 using the `featureCompatibilityVersion` command:

- UUID for collections

- `$jsonSchema` document validation

- Change streams

- Chunk aware secondaries

- View definitions, document validators, and partial index filters that use version 3.6 query features

- Sessions and retryable writes

- Users and roles with `authenticationRestrictions`

For details regarding backward incompatible changes in MongoDB 3.6, see the upstream release notes.
5.2.2. Upgrading from the rh-mongodb34 to the rh-mongodb36 Software Collection

**IMPORTANT**

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

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

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

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

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

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

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

3. Stop the MongoDB 3.4 server:

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

4. Copy your data to the new location:

   ```
   ```


6. Start the MongoDB 3.6 server:

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

7. Enable backwards incompatible features:

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

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

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

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

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

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

5.3. MIGRATING TO MONGODB 3.4

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

5.3.1. Notable Differences Between MongoDB 3.2 and MongoDB 3.4

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

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

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

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

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

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

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

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

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

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

1. Install the MongoDB servers and shells from the rh-mongodb34 Software Collections:
   
   ```
   ~]# yum install rh-mongodb34
   ```

2. Stop the MongoDB 3.2 server:
   
   ```
   ~]# systemctl stop rh-mongodb32-mongod.service
   ```
   Use the `service rh-mongodb32-mongodb stop` command on a Red Hat Enterprise Linux 6 system.

3. Copy your data to the new location:
   
   ```
   ```


5. Start the MongoDB 3.4 server:
   
   ```
   ~]# systemctl start rh-mongodb34-mongod.service
   ```
   On Red Hat Enterprise Linux 6, use the `service rh-mongodb34-mongodb start` command instead.

6. Enable backwards-incompatible features:
   
   ```
   ~]# scl enable rh-mongodb34 'mongo --host localhost --port 27017 admin' --eval 'db.adminCommand( { setFeatureCompatibilityVersion: "3.4" } )'
   ```
   If the `mongod` server is configured with enabled access control, add the `--username` and `--password` options to `mongo` command.

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

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

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

For information about upgrading a Sharded Cluster, see the upstream MongoDB Manual.
5.4. MIGRATING TO MYSQL 5.7

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 5.6 has been available as a Software Collection for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7 since Red Hat Software Collections 2.0.

The rh-mysql57 Software Collection, available for both Red Hat Enterprise Linux 6 and Red Hat Enterprise Linux 7, conflicts neither with the mysql or mariadb packages from the core systems nor with the rh-mysql56 Software Collection, so it is possible to install the rh-mysql57 Software Collection together with the mysql, mariadb, or rh-mysql56 packages. It is also possible to run multiple versions at the same time; however, the port number and the socket in the my.cnf files need to be changed to prevent these specific resources from conflicting.

Note that it is possible to upgrade to MySQL 5.7 only from MySQL 5.6. If you need to upgrade from an earlier version, upgrade to MySQL 5.6 first. Instructions how to upgrade to MySQL 5.6 are available in the Red Hat Software Collections 2.2 Release Notes.

5.4.1. Notable Differences Between MySQL 5.6 and MySQL 5.7

- The mysql-bench subpackage is not included in the rh-mysql57 Software Collection.

- Since MySQL 5.7.7, the default SQL mode includes NO_AUTO_CREATE_USER. Therefore it is necessary to create MySQL accounts using the CREATE USER statement because the GRANT statement no longer creates a user by default. See the upstream documentation for details.

To find out about more detailed changes in MySQL 5.7 compared to earlier versions, see the upstream documentation: What Is New in MySQL 5.7 and Changes Affecting Upgrades to MySQL 5.7.

5.4.2. Upgrading to the rh-mysql57 Software Collection

**IMPORTANT**

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

1. Install the rh-mysql57 Software Collection.

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

2. Inspect the configuration of rh-mysql57, which is stored in the /etc/opt/rh/rh-mysql57/my.cnf file and the /etc/opt/rh/rh-mysql57/my.cnf.d/ directory. Compare it with the configuration of rh-mysql56 stored in /etc/opt/rh/rh-mysql56/my.cnf and /etc/opt/rh/rh-mysql56/my.cnf.d/ and adjust it if necessary.

3. Stop the rh-mysql56 database server, if it is still running.

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

4. All data of the rh-mysql56 Software Collection is stored in the /var/opt/rh/rh-mysql56/lib/mysql/ directory. Copy the whole content of this directory to /var/opt/rh/rh-mysql57/lib/mysql/. You can also move the content but remember to back up your data before you continue to upgrade.
5. Start the rh-mysql57 database server.

   service rh-mysql57-mysqld start

6. Perform the data migration.

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

5.5. MIGRATING TO POSTGRESQL 10

Red Hat Software Collections 3.1 is distributed with PostgreSQL 10, available only for Red Hat Enterprise Linux 7. The rh-postgresql10 Software Collection can be safely installed on the same machine in parallel with the base Red Hat Enterprise Linux system version of PostgreSQL or any PostgreSQL Software Collection. It is also possible to run more than one version of PostgreSQL on a machine at the same time, but you need to use different ports or IP addresses and adjust SELinux policy. See Section 5.6, "Migrating to PostgreSQL 9.6" for instructions how to migrate to an earlier version or when using Red Hat Enterprise Linux 6.

The rh-postgresql10 Software Collection includes the rh-postgresql10-syspaths package, which installs packages that provide system-wide wrappers for binaries, scripts, manual pages, and other. After installing the rh-postgresql10*-syspaths packages, users are not required to use the scl enable command for correct functioning of the binaries and scripts provided by the rh-postgresql10* packages. Note that the *-syspaths packages conflict with the corresponding packages from the base Red Hat Enterprise Linux system. To find out more about sypaths, see the Red Hat Software Collections Packaging Guide.

**IMPORTANT**

Before migrating to PostgreSQL 10, see the upstream compatibility notes.

The following table provides an overview of different paths in a Red Hat Enterprise Linux 7 system version of PostgreSQL provided by the postgresql package, and in the rh-postgresql96 and rh-postgresql10 Software Collections.

<table>
<thead>
<tr>
<th>Table 5.1. Differences in the PostgreSQL paths</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Content</strong></td>
</tr>
<tr>
<td>Executables</td>
</tr>
<tr>
<td>Content</td>
</tr>
<tr>
<td>--------------------------</td>
</tr>
<tr>
<td>Source</td>
</tr>
<tr>
<td>Development Headers</td>
</tr>
</tbody>
</table>

5.5.1. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 10 Software Collection

Red Hat Enterprise Linux 7 is distributed with PostgreSQL 9.2. To migrate your data from a Red Hat
Enterprise Linux system version of PostgreSQL to the rh-postgresql10 Software Collection, you can either perform a fast upgrade using the `pg_upgrade` tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method.

**IMPORTANT**

Before migrating your data from a Red Hat Enterprise Linux system version of PostgreSQL to PostgreSQL 10, 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`:

   ```bash
   systemctl stop postgresql.service
   ```

   To verify that the server is not running, type:

   ```bash
   systemctl status postgresql.service
   ```

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

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

   and back up your data.

3. Verify that the new data directory `/var/opt/rh/rh-postgresql10/lib/pgsql/data/` does not exist:

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

   If you are running a fresh installation of PostgreSQL 10, this directory should not be present in your system. If it is, back it up by running the following command as `root`:

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

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

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

   Alternatively, you can use the `/opt/rh/rh-postgresql10/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-postgresql10-postgresql.log` log file to find out if any problems occurred during the upgrade.
5. Start the new server as **root**:

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

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

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

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

   ```bash
   chkconfig postgresql off
   ```

   To enable the PostgreSQL 10 server, type as **root**:

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

7. If your configuration differs from the default one, make sure to update configuration files, especially the `/var/opt/rh/rh-postgresql10/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**:

   ```bash
   systemctl start postgresql.service
   ```

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

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

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

   ```bash
   systemctl stop postgresql.service
   ```

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

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

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

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

6. Import data from the previously created SQL file:

   ```bash
   su - postgres -c 'scl enable rh-postgresql10 "psql -f ~/pgdump_file.sql postgres"'
   ```
7. Optionally, you can configure the PostgreSQL 10 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 10 server, type as **root**:

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

8. If your configuration differs from the default one, make sure to update configuration files, especially the `/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf` configuration file. Otherwise only the `postgres` user will be allowed to access the database.

### 5.5.2. Migrating from the PostgreSQL 9.6 Software Collection to the PostgreSQL 10 Software Collection

To migrate your data from the `rh-postgresql96` Software Collection to the `rh-postgresql10` Collection, you can either perform a fast upgrade using the `pg_upgrade` tool (recommended), or dump the database data into a text file with SQL commands and import it in the new database. Note that the second method is usually significantly slower and may require manual fixes; see the PostgreSQL documentation for more information about this upgrade method.

**IMPORTANT**

Before migrating your data from **PostgreSQL 9.6** to **PostgreSQL 10**, 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-postgresql96/lib/pgsql/data/` directory.

**Procedure 5.3. Fast Upgrade Using the `pg_upgrade` Tool**

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

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

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

To verify that the server is not running, type:

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

2. Verify that the old directory `/var/opt/rh/rh-postgresql96/lib/pgsql/data/` exists:

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

and back up your data.

3. Verify that the new data directory `/var/opt/rh/rh-postgresql10/lib/pgsql/data/` does not exist:

```
file /var/opt/rh/rh-postgresql10/lib/pgsql/data/
```
If you are running a fresh installation of PostgreSQL 10, 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-postgresql10/lib/pgsql/data{,-scl-backup}
```

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

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

Alternatively, you can use the `/opt/rh/rh-postgresql10/root/usr/bin/postgresql-setup --upgrade --upgrade-from=rh-postgresql96-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-postgresql10-postgresql.log` log file to find out if any problems occurred during the upgrade.

5. Start the new server as `root`:

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

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

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

6. Optionally, you can configure the PostgreSQL 10 server to start automatically at boot time. To disable the old PostgreSQL 9.6 server, type the following command as `root`:

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

To enable the PostgreSQL 10 server, type as `root`:

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

7. If your configuration differs from the default one, make sure to update configuration files, especially the `/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf` configuration file. Otherwise only the `postgres` user will be allowed to access the database.

**Procedure 5.4. Performing a Dump and Restore Upgrade**

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

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

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

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

```
su - postgres -c 'scl enable rh-postgresql96 "pg_dumpall > ~/pgdump_file.sql"'
```
3. Stop the old server by running the following command as root:

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

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

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

5. Start the new server as root:

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

6. Import data from the previously created SQL file:

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

7. Optionally, you can configure the PostgreSQL 10 server to start automatically at boot time. To disable the old PostgreSQL 9.6 server, type the following command as root:

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

To enable the PostgreSQL 10 server, type as root:

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

8. If your configuration differs from the default one, make sure to update configuration files, especially the `/var/opt/rh/rh-postgresql10/lib/pgsql/data/pg_hba.conf` configuration file. Otherwise only the `postgres` user will be allowed to access the database.

### 5.6. MIGRATING TO POSTGRESQL 9.6

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

#### 5.6.1. Notable Differences Between PostgreSQL 9.5 and PostgreSQL 9.6

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

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

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

Table 5.2. Differences in the PostgreSQL paths

<table>
<thead>
<tr>
<th>Content</th>
<th>postgresql</th>
<th>postgresql92</th>
<th>rh-postgresql95</th>
<th>rh-postgresql96</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executables</td>
<td>/usr/bin/</td>
<td>/opt/rh/postgresql92/root/usr/bin/</td>
<td>/opt/rh/rh-postgresql95/root/usr/bin/</td>
<td>/opt/rh/rh-postgresql96/root/usr/bin/</td>
</tr>
<tr>
<td>Source</td>
<td>not installed</td>
<td>not installed</td>
<td>not installed</td>
<td>not installed</td>
</tr>
</tbody>
</table>
5.6.2. Migrating from a Red Hat Enterprise Linux System Version of PostgreSQL to the PostgreSQL 9.6 Software Collection

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

IMPORTANT

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

Procedure 5.5. Fast Upgrade Using the pg_upgrade Tool

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

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

   ```
   service postgresql stop
   ```

To verify that the server is not running, type:
service postgresql status

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

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

   and back up your data.

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

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

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

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

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

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

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

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

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

5. Start the new server as `root`:

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

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

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

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

   ```
   chkconfig postgresql off
   ```

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

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

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

Procedure 5.6. Performing a Dump and Restore Upgrade
To perform a dump and restore upgrade of your PostgreSQL server, complete the following steps:

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

2. Dump all data in the PostgreSQL database into a script file. As root, type:
   ```
   su - postgres -c 'pg_dumpall > ~/pgdump_file.sql'
   ```

3. Stop the old server by running the following command as root:
   ```
   service postgresql stop
   ```

4. Initialize the data directory for the new server as root:
   ```
   scl enable rh-postgresql96-postgresql -- postgresql-setup --initdb
   ```

5. Start the new server as root:
   ```
   service rh-postgresql96-postgresql start
   ```

6. Import data from the previously created SQL file:
   ```
   su - postgres -c 'scl enable rh-postgresql96 "psql -f ~/pgdump_file.sql postgres"'
   ```

7. Optionally, you can configure the PostgreSQL 9.6 server to start automatically at boot time. To disable the old system PostgreSQL server, type the following command as root:
   ```
   chkconfig postgresql off
   ```
   To enable the PostgreSQL 9.6 server, type as root:
   ```
   chkconfig rh-postgresql96-postgresql on
   ```

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

### 5.6.3. Migrating from the PostgreSQL 9.5 Software Collection to the PostgreSQL 9.6 Software Collection

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

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

Procedure 5.7. Fast Upgrade Using the pg_upgrade Tool

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

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

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

   To verify that the server is not running, type:

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

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

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

   and back up your data.

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

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

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

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

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

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

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

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

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

5. Start the new server as root:

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

   It is also advised that you run the analyze_new_cluster.sh script as follows:
su - postgres -c 'scl enable rh-postgresql96 ~/analyze_new_cluster.sh'

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

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

To enable the PostgreSQL 9.6 server, type as root:

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

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

Procedure 5.8. Performing a Dump and Restore Upgrade

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

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

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

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

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

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

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

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

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

5. Start the new server as root:

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

6. Import data from the previously created SQL file:

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

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

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

To enable the PostgreSQL 9.6 server, type as root:
chkconfig rh-postgresql96-postgresql on

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

If you need to migrate from the postgresq92 Software Collection, refer to Red Hat Software Collections 2.0 Release Notes; the procedure is the same, you just need to adjust the version of the new Collection. The same applies to migration from the rh-postgresql94 Software Collection, which is described in Red Hat Software Collections 2.2 Release Notes.

5.7. MIGRATING TO NGINX 1.12

The rh-nginx112 Software Collection is available only for Red Hat Enterprise Linux 7.4 and later versions.

The root directory for the rh-nginx112 Software Collection is located in /opt/rh/rh-nginx112/root/. The error log is stored in /var/opt/rh/rh-nginx112/log/nginx by default.

Configuration files are stored in the /etc/opt/rh/rh-nginx112/nginx/ directory. Configuration files in nginx 1.12 have the same syntax and largely the same format as previous nginx Software Collections.

Configuration files (with a .conf extension) in the /etc/opt/rh/rh-nginx112/nginx/default.d/ directory are included in the default server block configuration for port 80.

IMPORTANT

Before upgrading from nginx 1.10 to nginx 1.12, back up all your data, including web pages located in the /opt/nginx110/root/ tree and configuration files located in the /etc/nginx110/nginx/ tree.

If you have made any specific changes, such as changing configuration files or setting up web applications, in the /opt/nginx110/root/ tree, replicate those changes in the new /opt/nginx112/root/ and /etc/nginx112/nginx/ directories, too.

You can use this procedure to upgrade directly from nginx 1.4, nginx 1.6, nginx 1.8, or nginx 1.10 to nginx 1.12. Use the appropriate paths 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 3.1 and Red Hat Enterprise Linux.

6.1. RED HAT PRODUCT DOCUMENTATION

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

- **Red Hat Software Collections 3.1 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.


- **Using Red Hat Software Collections Container Images** — This book provides information on how to use container images based on Red Hat Software Collections. The available container images include applications, daemons, databases, as well as the Red Hat Developer Toolset container images. The images can be run on Red Hat Enterprise Linux 7 Server and Red Hat Enterprise Linux Atomic Host.

- **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.

6.2. RED HAT DEVELOPERS

- **Red Hat Developer Program** — The Red Hat Developers community portal.

- **Overview of Red Hat Software Collections on Red Hat Developers** — The Red Hat Developers portal provides a number of tutorials to get you started with developing code using different development technologies. This includes the Node.js, Perl, PHP, Python, and Ruby Software Collections.
CHAPTER 6. ADDITIONAL RESOURCES

- **Red Hat Enterprise Linux Developer Program** – The *Red Hat Enterprise Linux Developer Program* delivers industry-leading developer tools, instructional resources, and an ecosystem of experts to help programmers maximize productivity in building Linux applications.

- **Red Hat Developer Blog** – The *Red Hat Developer Blog* contains up-to-date information, best practices, opinion, product and program announcements as well as pointers to sample code and other resources for those who are designing and developing applications based on Red Hat technologies.
### APPENDIX A. REVISION HISTORY

<table>
<thead>
<tr>
<th>Revision 3.1-3</th>
<th>Fri Nov 12 2021</th>
<th>Lenka Špačková</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updated Section 4.9, &quot;Database Connectors&quot;.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Revision 3.1-2</th>
<th>Tue Dec 18 2018</th>
<th>Lenka Špačková</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed syntax of a command in the PostgreSQL migration chapter.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Revision 3.1-1</th>
<th>Thu May 03 2018</th>
<th>Lenka Špačková</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release of Red Hat Software Collections 3.1 Release Notes.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Revision 3.1-0</th>
<th>Wed Mar 04 2018</th>
<th>Lenka Špačková</th>
</tr>
</thead>
<tbody>
<tr>
<td>Release of Red Hat Software Collections 3.1 Beta Release Notes.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>