Red Hat Software Collections 1.x

1.0 Release Notes

Release Notes for Red Hat Software Collections

Last Updated: 2017-10-11

Red Hat Software Collections 1.x 1.0 Release Notes

Release Notes for Red Hat Software Collections

Eliška Slobodová
Red Hat, Inc Engineering Content Services
eslobodo@redhat.com

Jaromír Hradílek
Red Hat, Inc Engineering Content Services
jhradilek@redhat.com

Legal Notice

Abstract

The Red Hat Software Collections 1.0 Release Notes document the major features and contains important information about known problems in Red Hat Software Collections 1.0.

Table of Contents

CHAPTER 1. RED HAT SOFTWARE COLLECTIONS 1.0

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. 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. 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.

With the notable exception of **Node.js**, 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 fixes and security errata are issued to Red Hat Software Collections subscribers in a similar manner to Red Hat Enterprise Linux for at least three years from the release of each major version. A new major version of Red Hat Software Collections is released approximately every 18 months, and in each major release stream, each version of a selected component remains backward compatible.

1.2. MAIN FEATURES

Red Hat Software Collections 1.0 provides recent versions of the tools listed in Table 1.1, “Red Hat Software Collections 1.0 Components”.

Table 1.1. Red Hat Software Collections 1.0 Components

Component	Software Collection	Description
Perl 5.16.3	perl516	A recent stable release of Perl with a number of additional utilities, scripts, and *database connectors for MySQL and PostgreSQL*. This version provides a large number of new features and enhancements, including new debugging options, improved Unicode support, and better performance.
PHP 5.4.14	php54	A recent stable release of PHP with **PEAR 1.9.4** and a number of additional utilities. This version provides new language syntax, a built-in web server for the command line, and improved performance.
Python 2.7	python27	A recent stable release of Python with a number of additional utilities and *database connectors for MySQL and PostgreSQL*. This version provides various new features and enhancements, including a new ordered dictionary type, faster I/O operations, and improved forward compatibility with Python 3.

Component	Software Collection	Description
Python 3.3	python33	A recent stable release of Python 3 with a number of additional utilities and *a database connector for PostgreSQL*. This Software Collection gives developers on Red Hat Enterprise Linux 6 access to Python 3 and allows them to benefit from various advantages and new features of this version.
Ruby 1.9.3	ruby193	A recent stable release of Ruby with **Rails 3.2.8** and *a large collection of Ruby gems*. This Software Collection gives developers on Red Hat Enterprise Linux 6 access to Ruby 1.9, which provides a number of new features and enhancements, including improved Unicode support, enhanced threading, and faster load times.
MariaDB 5.5	mariadb55	A recent stable release of MariaDB. This Software Collection gives users of Red Hat Enterprise Linux 6 an alternative to MySQL, which is binary compatible with MariaDB and can be replaced with it without any data conversions.
MySQL 5.5	mysql55	A recent stable release of MySQL. This version provides a number of new features and enhancements, including improved performance.
PostgreSQL 9.2	postgresql92	A recent stable release of PostgreSQL. This version provides a number of new features and enhancements, including cascading replication, native JSON support, improved scalability, and better performance.
Node.js 0.10[a]	nodejs010	A recent stable release of Node.js with **npm 1.2.17** and support for the **SPDY** protocol[b]. This Software Collection gives users of Red Hat Enterprise Linux 6 access to this programming platform.

[a] In Red Hat Software Collections 1.0, **Node.js** is included as a Technology Preview. For more information about Red Hat Technology Previews, see https://access.redhat.com/support/offerings/techpreview/.

[b] This feature is not fully supported on Red Hat Enterprise Linux 6.

1.3. COMPATIBILITY INFORMATION

Red Hat Software Collections 1.0 is available for the following versions of the Red Hat Enterprise Linux system on AMD64 and Intel 64 architectures:

- Red Hat Enterprise Linux 6.2 Extended Update Support
- Red Hat Enterprise Linux 6.3 Extended Update Support
- Red Hat Enterprise Linux 6.4

1.4. KNOWN ISSUES

`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.

`python-virtualenv` component

The **virtualenv** tool of version 1.7.2 does not create the python2 and python27 symlinks to the python interpreter. This causes scripts to be broken after the **`virtualenv --relocatable`** command is run. To work around this problem, the symlinks need to be created manually after running **virtualenv**.

`postgresql92` component

The postgresql92 package does 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.

`python27` and `python33` components

Due to a bug in the redhat-rpm-config package, it is not possile to rebuild the python27 and python33 packages on Red Hat Enterprise Linux 6.2 Extended Update Support.

`nodejs` component

On the Server variant, a file conflict between the ruby193-v8-debuginfo and nodejs010-v8-debuginfo packages can occur when attempting to install both debuginfo packages.

`coreutils` component

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 postgresql92 psql"
```

instead of

```
scl enable postgresql92 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`**.

`ruby`, `perl` components

When uninstalling the perl or ruby packages, some directories and files might not be removed properly and will remain on the system.

`php` component

Note that Alternative PHP Cache (APC) in Red Hat Software Collections 1.0 is provided for user data cache only.

`mariadb`, `mysql`, `postgresql` components

Red Hat Software Collections contains the MySQL 5.5, MariaDB 5.5 and PostgreSQL 9.2 databases. The core Red Hat Enterprise Linux 6 provides earlier versions of these databases (client library and daemon). Client libraries are also used in database connectors for dynamic languages, libraries etc.

Client libraries packaged in the Red Hat Software Collections database packages are not supposed to be used as they are included only for purposes of server utilities and the daemon. Users are instead expected to use the system libraries and 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 using, for example, the MySQL 5.1 client library with the MySQL 5.5 daemon works as expected.

`mariadb`, `mysql` components

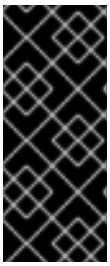
MySQL and MariaDB do not make use of the **`/opt/<provider>/<collection>/root`** prefix when creating log files. Note that log files are saved in the **`/var/log/`** directory, not **`/opt/<provider>/<collection>/root/var/log/`**.

CHAPTER 2. INSTALLATION AND USAGE

This chapter contains information related to installation and usage of Red Hat Software Collections 1.0.

2.1. GETTING ACCESS TO RED HAT SOFTWARE COLLECTIONS

Depending on the subscription management service with which you registered your Red Hat Enterprise Linux system, you can either enable Red Hat Software Collections by using Red Hat Subscription Management, or by using RHN Classic. For detailed instructions on how to enable Red Hat Software Collections using RHN Classic or Red Hat Subscription Management, see the respective section below. For information on how to register your system with one of these subscription management services, see the *Red Hat Subscription Management Guide*.



IMPORTANT

If you are running a version of Red Hat Enterprise Linux prior to 6.4, you will be unable to download Red Hat Software Collections through Red Hat Subscription Management. To obtain Red Hat Software Collections, either update to Red Hat Enterprise Linux 6.4, or register your system with RHN Classic. For more information, see https://access.redhat.com/site/solutions/129003.

2.1.1. Using RHN Classic

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

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

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

   The name of the channel depends on the specific version of Red Hat Enterprise Linux you are using and is in the **rhel-x86_64-*variant*-6-rhscl-1** format, where *variant* is the Red Hat Enterprise Linux system variant (**server** or **workstation**).

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

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

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

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

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

Once the system is subscribed, you can install Red Hat Software Collections as described in Section 2.2, “Installing Red Hat Software Collections”. For more information on how to register your system with RHN Classic, see the *Red Hat Subscription Management Guide*.

2.1.2. Using Red Hat Subscription Management

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

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

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

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

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

   ```
   subscription-manager subscribe --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, run as **root**:

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

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

   ```
   yum repolist all
   ```

   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
   ```

   Replace *variant* with the Red Hat Enterprise Linux system variant, that is, **server** or **workstation**. Note that Red Hat Software Collections is not supported on the **Client** variant.

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

   ```
   yum-config-manager --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 the *Red Hat Subscription Management Guide*.

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 the product, see Section 2.1, “Getting Access to Red Hat Software Collections”.

IMPORTANT

Some of the Red Hat Software Collections 1.0 packages require the **Optional** channel to be enabled in order to complete the full installation of these packages:

- The php54-php-imap package requires the libc-client package, which is only available in the Optional channel.
- The php54-php-recode requires the recode package, which is only available in the Optional channel.
- The perl516-perl-devel requires the gdbm-devel package, which is only available in the Optional channel.
- The mariadb55-mariadb-bench requires the perl-GD package, which is only available in the Optional channel.

For detailed instructions on how to subscribe your system to this channel, see the relevant Knowledge article on the Customer Portal.

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 1.0 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 mariadb55, type as **root**:

```
~]# yum install php54 mariadb55
```

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, run 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 perl516-perl-CPAN and perl516-perl-Archive-Tar, type:

```
~]# yum install perl516-perl-CPAN perl516-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 run the following command as **root**:

```
debuginfo-install package_name
```

For example, to install debugging information for the ruby193-ruby package, type:

```
~]# debuginfo-install ruby193-ruby
```

Note that in order to use this command, you need to have access to the repository with these packages. If your system is registered with Red Hat Subscription Management, enable the **rhel-*variant*-rhscl-6-debug-rpms** repository as described in Section 2.1.2, “Using Red Hat Subscription Management”. If your system is registered with RHN Classic, subscribe the system to the **rhel-x86_64-*variant*-6-rhscl-1-debuginfo** channel as described in Section 2.1.1, “Using RHN Classic”. For more information on how to get access to debuginfo packages, see https://access.redhat.com/site/solutions/9907.

2.3. 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. In order 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.

2.4. USING RED HAT SOFTWARE COLLECTIONS

2.4.1. Running an Executable from a Software Collection

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

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

Replace *software_collection* with a space-separated list of Software Collections you want to use and *command* with the command you want to run. For example, to execute a Perl program stored in a file named **hello.pl** with the Perl interpreter from the perl516 Software Collection, type:

```
~]$ scl enable perl516 'perl hello.pl'
Hello, World!
```

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

Collections, see Table 1.1, “Red Hat Software Collections 1.0 Components”.

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

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

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

Replace *software_collection* with a space-separated list of Software Collections you want to use. For example, to start a new shell session with the python27 and postgresql92 Software Collections as default, type:

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

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

```
~]$ echo $X_SCLS
python27 postgresql92
```

For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 1.0 Components”.

2.4.3. 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, run the following command as **root**:

```
chkconfig software_collection-service_name on
```

For example, to start the **postgresql** service from the postgresql92 Software Collection and enable it in runlevels 2, 3, 4, and 5, type as **root**:

```
~]# service postgresql92-postgresql start
Starting postgresql92-postgresql service:                  [  OK  ]
~]# chkconfig postgresql92-postgresql on
```

For more information on how to manage system services in Red Hat Enterprise Linux 6, refer to the *Red Hat Enterprise Linux 6 Deployment Guide*. For a complete list of Software Collections that are distributed with Red Hat Software Collections, see Table 1.1, “Red Hat Software Collections 1.0 Components”.

2.5. 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 a detailed explanation of how to create a custom Software Collection, read the *Red Hat Developer Toolset Software Collections Guide*.

2.6. MIGRATING FROM MYSQL 5.1 TO MYSQL 5.5

2.6.1. Notable Differences Between MySQL 5.1 and MySQL 5.5

The following is a list of the most important changes between MySQL 5.1 and MySQL 5.5

- Starting with MySQL 5.5, the InnoDB storage engine (formerly known as InnoDB Plugin) is the default storing engine.
- InnoDB and some other plug-ins (for example, archive, blackhole and federated) were installable plug-ins in MySQL 5.1. Starting with MySQL 5.5, these plug-ins became compiled-in storage engines, that is, they do not have to be installed nor uninstalled.
- If you used InnoDB Plugin and it was loaded using the **`plugin-load=innodb=ha_innodb_plugin.so`** configuration option, you need to remove this configuration option as it does not work in MySQL 5.5.
- In MySQL 5.1, InnoDB Plugin included a configuration variable **`innodb_file_io_threads`**. However, this variable does not exist in MySQL 5.5; new variables, **`innodb_read_io_threads`** and **`innodb_write_io_threads`**, are used instead. To ensure proper functionality, either remove the former variable from the configuration file or replace it with the current variables.
- When upgrading from MySQL 5.1 to MySQL 5.5 using the in-place upgrading method, the **`mysql.proxies_priv`** table will not exist. To create the missing table, the **mysql_upgrade** utility has to be run as soon as the new daemon is started.
- MySQL 5.5 uses latin1 for the **`stopword`** file if the **`character_set_server`** variable is ucs2, utf16 or utf32. Thus, if the table uses FULLTEXT indexes in these cases, users should repair the table using the **`REPAIR TABLE *table_name* QUICK`**.
- MySQL 5.1 used the **`language`** variable for specifying the directory which included the error message file. This option is now deprecated and has been replaced by the **`lc_messages_dir`** and **`lc_messages`** options. This also applies for configuration options. Also, error messages no longer contain mixed set of character sets and error messages are returned in the set following the **`character_set_results`** system variable instead. That is, some error messages can be different in MySQL 5.5.

Please note that the EXAMPLE plug-in is no longer distributed in Red Hat Software Collections packages.

For more information about MySQL 5.1 and MySQL 5.5, refer to the release notes available at http://dev.mysql.com/doc/relnotes/mysql/5.1/en/ and http://dev.mysql.com/doc/relnotes/mysql/5.5/en/.

2.6.2. Upgrading from MySQL 5.1 to MySQL 5.5

Before migrating from MySQL 5.1 to MySQL 5.5, back up all your data, including any MySQL databases. Because the mysql55 Software Collection does not conflict with the mysql packages from the core systems, it is possible to install the mysql55 Software Collection together with the mysql 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.

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

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

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

After upgrading, start the server and run the **`mysql_upgrade`** command. Running **`mysql_upgrade`** is necessary to check and repair internal tables. All scripts that work with a server form Software Collection, especially the **`mysql_upgrade`** script, should be run inside the **`scl enable`** environment.

The in-place upgrade method is usually faster, however, there are certain risks and known problems. For more information, refer to the MySQL 5.5 Release Notes.

In addition, once the upgrade is complete, consider changing the appropriate settings in the **`my.cnf`** file to reflect the environment.

Example 2.1. Dump and Restore Upgrade

```
~]# service mysqld start
Starting mysqld:                                           [  OK  ]
~]# mysqldump --all-databases --routines --events > dump.sql
~]# service mysqld stop
Stopping mysqld:                                           [  OK  ]
~]# service mysql55-mysqld start
Starting mysql55-mysqld:                                   [  OK  ]
~]# scl enable mysql55 'mysql' < dump.sql
~]# scl enable mysql55 'mysql_upgrade'
Looking for 'mysql' as: mysql
Looking for 'mysqlcheck' as: mysqlcheck
Running 'mysqlcheck with default connection arguments
Running 'mysqlcheck with default connection arguments
a.t1                                               OK
mysql.columns_priv                                 OK
<skipped tables list>
```

```
mysql.user                                        OK
Running 'mysql_fix_privilege_tables'...
OK
```

Example 2.2. In-place Upgrade

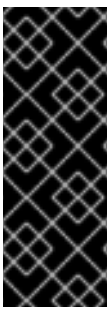
```
~]# service mysqld stop
Stopping mysqld:                                           [  OK  ]
~]# service mysql55-mysqld stop
Stopping mysql55-mysqld:                                   [  OK  ]
~]# rm -rf /opt/rh/mysql55/root/var/lib/mysql/
~]# cp -r /var/lib/mysql/ /opt/rh/mysql55/root/var/lib/
~]#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

For more information about the upgrading process, refer to MySQL 5.5 Reference Manual.

2.6.3. Using the mysql55-mysql-devel Package

Red Hat Software Collections contains the server part of MySQL 5.5 database. Red Hat Enterprise Linux 6 provides version 5.1 of this database (client library and server daemon). A protocol which is used between the client library and the daemon is stable across database versions, so using, for example, the MySQL 5.1 client library with the MySQL 5.5 daemon works as expected.

2.6.3.1. Using Database Connectors for Dynamic Languages



IMPORTANT

When a MariaDB or MySQL database contains old users created using old authentication schema, PHP using the **mysqlnd** driver will not be able to connect to the database. This is because the **old_password** setting in the **/etc/my.cnf** file is turned off by default on Red Hat Enterprise Linux 6 while it is enabled on Red Hat Enterprise Linux 5. To work around this problem, set **old_password** to 0, restart the MariaDB or MySQL service and set a new password for each user.

2.6.3.2. Building Applications for MySQL 5.5 from Red Hat Software Collections

MySQL 5.5 from Red Hat Software Collections does not include database connectors; client libraries

packaged in the MySQL 5.5 Red Hat Software Collections database packages are not supposed to be used as they are included only for purposes of server utilities and the daemon. Users are instead expected to use the system libraries and database connectors provided with the core system.

It means that users who would like to link their application against the MySQL client library should compile it and link it to the core Red Hat Enterprise Linux 6 environment, not to the MySQL 5.5 Red Hat Software Collections environment.

The only exception to this are server-side plug-ins, which are expected to be built under the MySQL 5.5 Red Hat Software Collections environment. This means the build process should be run inside the **scl enable mysql55 '...'** call.

2.7. MIGRATING FROM POSTGRESQL 8.4 TO POSTGRESQL 9.2

Red Hat Software Collections 1.0 is distributed with PostgreSQL 9.2, which can be safely installed on the same machine in parallel with PostgreSQL 8.4 from Red Hat Enterprise Linux 6. It is also possible to run both versions of PostgreSQL on one machine at the same time, but you need to use different ports or IP addresses and adjust SELinux policy.

2.7.1. Notable Differences Between PostgreSQL 8.4 and PostgreSQL 9.2

The following is a list of the most important changes between PostgreSQL 8.4 and PostgreSQL 9.2:

- The following server configuration parameters have been removed and are no longer supported: **add_missing_from**, **regex_flavor**, **silent_mode**, **wal_sender_delay**, and **custom_variable_classes**. Do not use any of these parameters in the new configuration file.

- The **unix_socket_directory** parameter has been renamed to **unix_socket_directories** and can now be used to specify more than one UNIX socket to listen on. To do so, provide a list of comma-separated directories as the value of this option. The default value remains unchanged and is **/tmp**.

- New configuration parameters **ssl_ca_file**, **ssl_cert_file**, **ssl_crl_file**, and **ssl_key_file** have been added. These configuration parameters can be used to specify the locations of server-side SSL files that were previously hard-coded as relative paths to the **root.crt**, **server.crt**, **root.crl**, and **server.key** files in the data directory.

  Note that the PostgreSQL server no longer reads the **root.crt** and **root.crl** files by default. To load these files, change the corresponding parameters to non-default values.

- The **=>** operator has been removed and users are now advised to use the **hstore(*text*, *text*)** function.

- The default value of the **standard_conforming_strings** configuration parameter is now **on**. This configuration parameter controls if ordinary string literals (strings enclosed in single quotes) treat backslashes literally as specified in the SQL standard.

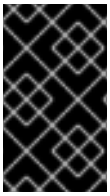
- A new configuration parameter, **backslash_quote**, has been added. This configuration parameter can be used to control whether a single quotation mark can be represented by **\'** in string literals. The default value is **safe_encoding**, which permits the use of **\'** only when the client encoding does not allow ASCII backslashes in multi-byte characters. As a consequence, **\'** can now be interpreted differently only in specific cases and only in string literals that do not conform to standards, including escape string syntax, **E'*value*'**.

- PostgreSQL 9.0 introduced access privileges for large objects. Consequently, a new configuration parameter, **lo_compat_privileges**, has been added to allow you to disable security checks related to the large objects affected by this change. To disable these security checks, change the value of this configuration parameter to **on**. The default value is **off**.

For a detailed list of known compatibility issues with earlier versions, see the official notes for PostgreSQL 9.0, PostgreSQL 9.1, and PostgreSQL 9.2. For an in-depth list of changes in behavior, see the upstream Release Notes.

2.7.2. Upgrading from PostgreSQL 8.4 to PostgreSQL 9.2

To migrate your data from PostgreSQL 8.4 that is distributed with Red Hat Enterprise Linux 6 to PostgreSQL 9.2 that is included in Red Hat Software Collections 1.0, you can either perform an in-place upgrade, 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 official documentation for more information about this upgrade method.



IMPORTANT

Before migrating your data from PostgreSQL 8.4 to PostgreSQL 9.2, make sure that you back up all your data, including the PostgreSQL database files that are by default located in the **/var/lib/pgsql/data/** directory.

Procedure 2.1. Performing In-place Upgrade

To perform an in-place 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 new data directory located in **/opt/rh/postgresql92/root/var/lib/pgsql/data/** does not exist:

   ```
   file /opt/rh/postgresql92/root/var/lib/pgsql/data/
   ```

   If you are running a fresh installation of PostgreSQL 9.2, this directory should not be present in your system. If it is, back it up by running the following command as **root**:

   ```
   mv /opt/rh/postgresql92/root/var/lib/pgsql/data{,-scl-backup}
   ```

3. Copy the old database data to the new location by typing the following at a shell prompt as **root**:

   ```
   cp -ra /var/lib/pgsql/data/ /opt/rh/postgresql92/root/var/lib/pgsql/
   ```

4. Open the **/opt/rh/postgresql92/root/var/lib/pgsql/data/pg_hba.conf**

configuration file and verify that the **postgres** user is allowed to connect to the PostgreSQL server from **localhost** without a password. If not, you can edit this file and temporarily set the authentication method for the **postgres** user to **trust** or **ident**. For a detailed description of the **pg_hba.conf** file and a complete list of available configuration options, see the official documentation.

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

```
service postgresql92-postgresql upgrade
```

It is recommended that you read the resulting **/opt/rh/postgresql92/root/var/lib/pgsql/pgupgrade.log** log file to see if there were any problems with the upgrade.

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

```
service postgresql92-postgresql start
```

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

```
su - postgres -c 'scl enable postgresql92 ~/analyze_new_cluster.sh'
```

7. Optionally, you can configure the PostgreSQL 9.2 server to start automatically at boot time. To disable the old PostgreSQL 8.4 server, run the following command as **root**:

```
chkconfig postgresql off
```

To enable the PostgreSQL 9.2 server, type as **root**:

```
chkconfig postgresql92-postgresql on
```

Procedure 2.2. Performing a Dump and Restore Upgrade

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

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

```
service postgresql start
```

2. Dump all data in the PostgreSQL database into an SQL 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**:

```
service postgresql92-postgresql initdb
```

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

```
service postgresql92-postgresql start
```

6. Import data from the previously created SQL file:

```
su - postgres -c 'scl enable postgresql92 "psql -f ~/pgdump_file.sql
postgres"'
```

7. Optionally, you can configure the PostgreSQL 9.2 server to start automatically at boot time. To disable the old PostgreSQL 8.4 server, run the following command as **root**:

```
chkconfig postgresql off
```

To enable the PostgreSQL 9.2 server, type as **root**:

```
chkconfig postgresql92-postgresql on
```

2.8. REPORTING BUGS IN RED HAT SOFTWARE COLLECTIONS

Bug reports are essential in making Red Hat Software Collections reliable. Reporting a bug can also help you bring a solution to your problem. Nevertheless, the main function of a bug report is to help the community by improving the next version of Red Hat Software Collections.

Procedure 2.3. File a Bug using Bugzilla

1. To report a bug, navigate to Bugzilla, the Red Hat bug-tracking system. Note that you need to be logged in, or you will be prompted to do so.

2. On the top bar, click on **New**, and on the next page, choose **Red Hat**.

3. This brings you to a page with a list of Red Hat Products. Click on **Red Hat Software Collections**.

4. **Component** and **Summary** are required fields; your report cannot be submitted successfully if you do not fill these fields. The **Description** field provides a simple template which can help you describe the bug. It is important to provide as much information about the problem as possible.

5. Additionally, you can add an attachment, which can be a screenshot of the problem, a patch or similar.

6. Once you have completed your bug report, press the **Submit Bug** button. Your report will be assigned a Bugzilla number so you can get back to it later.

If you wish to use the **Automatic Bug Reporting Tool** (ABRT) utility, see Automatic Bug Reporting Tool chapter in the *Red Hat Enterprise Linux 6 Deployment Guide*.

For detailed information on bug reporting, refer to Bugilla's help.

CHAPTER 3. ADDITIONAL RESOURCES

For more information about Red Hat Software Collections 1.0 and Red Hat Enterprise Linux, refer to the resources listed below.

3.1. RED HAT ENTERPRISE LINUX DEVELOPER PROGRAM GROUP

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

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

3.2. RED HAT PRODUCT DOCUMENTATION

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

- Red Hat Developer Toolset Software Collections Guide — The *Software Collections Guide* for Red Hat Developer Toolset explains the concept of Software Collections and documents the `scl` utility.
- Red Hat Subscription Management Guide — The *Subscription Management Guide* provides detailed information on how to manage subscriptions on Red Hat Enterprise Linux.
- Red Hat Enterprise Linux 6 Developer Guide — The *Developer Guide* for Red Hat Enterprise Linux 6 provides more information for developers on the Red Hat Enterprise Linux platform.
- 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.

APPENDIX A. REVISION HISTORY

Revision 1.1-25 **Thu 09 Jun 2016** **Lenka Špačková**

Fixed a path in the MySQL in-place upgrade example.

Revision 1.0-8 **Mon 10 Sep 2013** **Eliška Slobodová**

Release of Red Hat Software Collections 1.0 GA Release Notes.

Revision 1.0-7 **Tue 13 Aug 2013** **Eliška Slobodová**

Release of Red Hat Software Collections 1.0 Beta-2 Release Notes.

Revision 1.0-6 **Mon 08 Jul 2013** **Jaromír Hradílek**

Updated the description of Red Hat Software Collections components.

Revision 1.0-4 **Thu Jun 26 2013** **Eliška Slobodová**

Added information on Rebuilding Red Hat Software Collections.

Revision 1.0-3 **Thu Jun 6 2013** **Eliška Slobodová**

Removed a fixed known issue.

Revision 1.0-2 **Tue Jun 4 2013** **Eliška Slobodová**

Republished the book with a known issue.

Revision 1.0-1 **Tue Jun 4 2013** **Eliška Slobodová**

Release of Red Hat Software Collections 1.0 Beta-1 Release Notes.