keyring

Release 12.1.1.dev0+gac4daef.d20180424

Apr 24, 2018

Contents

CHAPTER 1

Installing and Using Python Keyring Lib

Table of Contents

1.1 What is Python keyring lib?

The Python keyring lib provides an easy way to access the system keyring service from python. It can be used in any application that needs safe password storage.

The keyring library is licensed under both the MIT license and the PSF license.

These recommended keyring backends are supported by the Python keyring lib:

- macOS Keychain
- Freedesktop Secret Service supports many DE including GNOME (requires secretstorage)
- KDE4 & KDE5 KWallet (requires dbus)
- Windows Credential Locker

Other keyring implementations are available through *Third-Party Backends*.

1.2 Installation Instructions

CHAPTER 2

Install from Index

Install using your favorite installer. For example:

```
$ pip install keyring
```

On Linux, the recommended keyring relies on SecretStorage, which in turn relies on dbus-python, but dbus-python does not install correctly when using the Python installers, so dbus-python must be installed as a system package. See the SecretStorage GitHub repo for details.

The basic usage of keyring is pretty simple: just call *keyring.set_password* and *keyring.get_password*:

```
>>> import keyring
>>> keyring.set_password("system", "username", "password")
>>> keyring.get_password("system", "username")
'password'
```

CHAPTER 3

Command-line Utility

Keyring supplies a `keyring` command which is installed with the package. After installing keyring in most environments, the command should be available for setting, getting, and deleting passwords. For more information on usage, invoke with no arguments or with `--help` as so:

```
$ keyring --help
$ keyring set system username
Password for 'username' in 'system':
$ keyring get system username
password
```

The command-line functionality is also exposed as an executable package, suitable for invoking from Python like so:

```
$ python -m keyring --help
$ python -m keyring set system username
Password for 'username' in 'system':
$ python -m keyring get system username
password
```

The python keyring lib contains implementations for several backends. The library will automatically choose the keyring that is most suitable for your current environment. You can also specify the keyring you like to be used in the config file or by calling the `set_keyring()` function.

CHAPTER 4

Customize your keyring by config file

This section describes how to change your option in the config file.

The configuration of the lib is stored in a file named "keyringrc.cfg". This file must be found in a platform-specific location. To determine where the config file is stored, run the following:

```
python -c "import keyring.util.platform_; print(keyring.util.platform_.config_root())"
```

Some keyrings also store the keyring data in the file system. To determine where the data files are stored, run this command:

```
python -c "import keyring.util.platform_; print(keyring.util.platform_.data_root())"
```

To specify a keyring backend, set the **default-keyring** option to the full path of the class for that backend, such as `keyring.backends.OS_X.Keyring`.

If **keyring-path** is indicated, keyring will add that path to the Python module search path before loading the backend.

For example, this config might be used to load the SimpleKeyring from the demo directory in the project checkout:

```
[backend]
default-keyring=simplekeyring.SimpleKeyring
keyring-path=/home/kang/pyworkspace/python-keyring-lib/demo/
```

CHAPTER 5

Third-Party Backends

In addition to the backends provided by the core keyring package for the most common and secure use cases, there are additional keyring backend implementations available for other use-cases. Simply install them to make them available:

- keyrings.cryptfile - Encrypted text file storage.
- keyring_jeepney - a pure Python backend using the secret service DBus API for desktop Linux.
- keyrings.alt - "alternate", possibly-insecure backends, originally part of the core package, but available for opt-in.

CHAPTER 6

Write your own keyring backend

The interface for the backend is defined by `keyring.backend.KeyringBackend`. Every backend should derive from that base class and define a `priority` attribute and three functions: `get_password()`, `set_password()`, and `delete_password()`.

See the `backend` module for more detail on the interface of this class.

Keyring employs entry points to allow any third-party package to implement backends without any modification to the keyring itself. Those interested in creating new backends are encouraged to create new, third-party packages in the `keyrings` namespace, in a manner modeled by the keyrings.alt package. See the `setup.py` file in that project for a hint on how to create the requisite entry points. Backends that prove essential may be considered for inclusion in the core library, although the ease of installing these third-party packages should mean that extensions may be readily available.

If you've created an extension for Keyring, please submit a pull request to have your extension mentioned as an available extension.

CHAPTER 7

Set the keyring in runtime

Keyring additionally allows programmatic configuration of the backend calling the api `set_keyring()`. The indicated backend will subsequently be used to store and retrieve passwords.

Here's an example demonstrating how to invoke `set_keyring`:

```
# define a new keyring class which extends the KeyringBackend
import keyring.backend

class TestKeyring(keyring.backend.KeyringBackend):
    """A test keyring which always outputs same password
    """
    priority = 1

    def set_password(self, servicename, username, password):
        pass

    def get_password(self, servicename, username):
        return "password from TestKeyring"

    def delete_password(self, servicename, username, password):
        pass

# set the keyring for keyring lib
keyring.set_keyring(TestKeyring())

# invoke the keyring lib
try:
    keyring.set_password("demo-service", "tarek", "passexample")
    print("password stored successfully")
except keyring.errors.PasswordSetError:
    print("failed to store password")
print("password", keyring.get_password("demo-service", "tarek"))
```

CHAPTER 8

Using Keyring on Ubuntu 16.04

The following is a complete transcript for installing keyring in a virtual environment on Ubuntu 16.04. No config file was used.:

```
$ sudo apt install python3-venv libdbus-glib-1-dev
$ cd /tmp
$ pyvenv py3
$ source py3/bin/activate
$ pip install -U pip
$ pip install secretstorage dbus-python
$ pip install keyring
$ python
>>> import keyring
>>> keyring.get_keyring()
<keyring.backends.SecretService.Keyring object at 0x7f9b9c971ba8>
>>> keyring.set_password("system", "username", "password")
>>> keyring.get_password("system", "username")
'password'
```

CHAPTER 9

Using Keyring on headless Linux systems

It is possible to use the SecretService backend on Linux systems without X11 server available (only D-Bus is required). To do that, you need the following:

- Install the GNOME Keyring daemon.
- Start a D-Bus session, e.g. run `dbus-run-session -- sh` and run the following commands inside that shell.
- Run `gnome-keyring-daemon` with `--unlock` option. The description of that option says:

  > Read a password from stdin, and use it to unlock the login keyring or create it if the login keyring does not exist.

  When that command is started, enter your password into stdin and press Ctrl+D (end of data). After that the daemon will fork into background (use `--foreground` option to prevent that).
- Now you can use the SecretService backend of Keyring. Remember to run your application in the same D-Bus session as the daemon.

CHAPTER 10

API interface

The keyring lib has a few functions:

- `get_keyring()`: Return the currently-loaded keyring implementation.
- `get_password(service, username)`: Returns the password stored in the active keyring. If the password does not exist, it will return None.
- `set_password(service, username, password)`: Store the password in the keyring.
- `delete_password(service, username)`: Delete the password stored in keyring. If the password does not exist, it will raise an exception.

In all cases, the parameters (`service`, `username`, `password`) should be Unicode text. On Python 2, these parameters are accepted as simple `str` in the default encoding as they will be implicitly decoded to text. Some backends may accept `bytes` for these parameters, but such usage is discouraged.

CHAPTER 11

Exceptions

The keyring lib raises following exceptions:

- `keyring.errors.KeyringError`: Base Error class for all exceptions in keyring lib.
- `keyring.errors.InitError`: Raised when the keyring can't be initialized.
- `keyring.errors.PasswordSetError`: Raise when password can't be set in the keyring.
- `keyring.errors.PasswordDeleteError`: Raised when the password can't be deleted in the keyring.

Python keyring lib is an open community project and highly welcomes new contributors.

- Repository: https://github.com/jaraco/keyring/
- Bug Tracker: https://github.com/jaraco/keyring/issues/
- Mailing list: http://groups.google.com/group/python-keyring

CHAPTER 12

Making Releases

This project makes use of automated releases via Travis-CI. The simple workflow is to tag a commit and push it to Github. If it passes tests on a late Python version, it will be automatically deployed to PyPI.

Other things to consider when making a release:

- first ensure that tests pass (preferably on Windows and Linux)
- check that the changelog is current for the intended release

CHAPTER 13

Running Tests

Tests are continuously run using Travis-CI.

To run the tests yourself, you'll want keyring installed to some environment in which it can be tested. Recommended technique is described below.

Keyring prefers use of tox to run tests. Simply install and invoke `tox`.

This technique is the one used by the Travis-CI script.

The project was based on Tarek Ziade's idea in this post. Kang Zhang initially carried it out as a Google Summer of Code project, and Tarek mentored Kang on this project.

13.1 History

13.1.1 12.1.0

24 Apr 2018

- Unpin SecretStorage on Python 3.5+. Requires that Setuptools 17.1 be used. Note that the special handling will be unnecessary once Pip 9 can be assumed (as it will exclude SecretStorage 3 in non-viable environments.

13.1.2 12.0.2

24 Apr 2018

- Pin SecretStorage to 2.x.

13.1.3 12.0.1

05 Apr 2018

- #314: No changes except to rebuild.

13.1.4 12.0.0

19 Mar 2018

- #310: Keyring now loads all backends through entry points.

For most users, this release will be fully compatible. Some users may experience compatibility issues if entrypoints is not installed (as declared) or the metadata on which entrypoints relies is unavailable. For that reason, the package is released with a major version bump.

13.1.5 11.1.0

19 Mar 2018

- #312: Use `entrypoints` instead of pkg_resources to avoid performance hit loading pkg_resources. Adds a dependency on `entrypoints`.

13.1.6 11.0.0

29 Jan 2018

- #294: No longer expose `keyring.__version__` (added in 8.1) to avoid performance hit loading pkg_resources.

13.1.7 10.6.0

07 Jan 2018

- #299: Keyring exceptions are now derived from a base `keyring.errors.KeyringError`.

13.1.8 10.5.1

15 Dec 2017

- #296: Prevent AttributeError on import when Debian has created broken dbus installs.

13.1.9 10.5.0

12 Nov 2017

- #287: Added `--list-backends` option to command-line interface.
- Removed `logger` from `keyring`. See #291 for related request.
- #292: Set the appid for SecretService & KWallet to something meaningful.

13.1.10 10.4.0

24 Jun 2017

- #279: In Kwallet, pass mainloop to SessionBus.
- #278: Unpin pywin32-ctypes, but blacklist known incompatible versions.

13.1.11 10.3.3

04 Jun 2017

- #278: Pin to pywin32-ctypes 0.0.1 to avoid apparent breakage introduced in 0.1.0.

13.1.12 10.3.2

09 Apr 2017

- #267: More leniently unescape lowercased characters as they get re-cased by ConfigParser.

13.1.13 10.3.1

20 Mar 2017

- #266: Use private compatibity model rather than six to avoid the dependency.

13.1.14 10.3

28 Feb 2017

- #264: Implement devpi hook for supplying a password when logging in with devpi client.
- #260: For macOS, added initial API support for internet passwords.

13.1.15 10.2

10 Jan 2017

- #259: Allow to set a custom application attribute for SecretService backend.

13.1.16 10.1

04 Dec 2016

- #253: Backends now expose a '.name' attribute suitable for identifying each backend to users.

13.1.17 10.0.2

20 Oct 2016

- #247: Restored console script.

13.1.18 10.0.1

20 Oct 2016

- Update readme to reflect test recommendations.

13.1.19 10.0

20 Oct 2016

- Drop support for Python 3.2.
- Test suite now uses tox instead of pytest-runner. Test requirements are now defined in tests/requirements.txt.

13.1.20 9.3.1

14 Jul 2016

- Link to the new Gitter chat room is now in the readme.
- Issue #235: `kwallet` backend now returns string objects instead of `dbus.String` objects, for less surprising reprs.
- Minor doc fixes.

13.1.21 9.3

27 Jun 2016

- Issue #161: In SecretService backend, unlock individual entries.

13.1.22 9.2.1

26 Jun 2016

- Issue #230: Don't rely on dbus-python and instead defer to SecretStorage to describe the installation requirements.

13.1.23 9.2

26 Jun 2016

- Issue #231 via #233: On Linux, `secretstorage` is now a declared dependency, allowing recommended keyring to work simply after installation.

13.1.24 9.1

21 Jun 2016

- Issue #83 via #229: `kwallet` backend now stores the service name as a folder name in the backend rather than storing all passwords in a Python folder.

13.1.25 9.0

10 Apr 2016

- Issue #217: Once again, the OS X backend uses the Framework API for invoking the Keychain service. As a result, applications utilizing this API will be authorized per application, rather than relying on the authorization of the 'security' application. Consequently, users will be prompted to authorize the system Python executable and also new Python executables, such as those created by virtualenv. #260: No longer does the keyring honor the `store` attribute on the keyring. Only application passwords are accessible.

13.1.26 8.7

02 Apr 2016

- Changelog now links to issues and provides dates of releases.

13.1.27 8.6

02 Apr 2016

- Issue #217: Add warning in OS Keyring when 'store' is set to 'internet' to determine if this feature is used in the wild.

13.1.28 8.5.1

13 Mar 2016

- Pull Request #216: Kwallet backend now has lower priority than the preferred SecretService backend, now that the desktop check is no longer in place.

13.1.29 8.5

11 Mar 2016

- Issue #168: Now prefer KF5 Kwallet to KF4. Users relying on KF4 must use prior releases.

13.1.30 8.4

07 Feb 2016

- Pull Request #209: Better error message when no backend is available (indicating keyrings.alt as a quick workaround).
- Pull Request #208: Fix pywin32-ctypes package name in requirements.

13.1.31 8.3

04 Feb 2016

- Issue #207: Library now requires win32ctypes on Windows systems, which will be installed automatically by Setuptools 0.7 or Pip 6 (or later).
- Actually removed QtKwallet, which was meant to be dropped in 8.0 but somehow remained.

13.1.32 8.2

27 Jan 2016

- Update readme to include how-to use with Linux non-graphical environments.

13.1.33 8.1

25 Jan 2016

- Issue #197: Add `__version__` attribute to keyring module.

13.1.34 8.0

22 Jan 2016

- Issue #117: Removed all but the preferred keyring backends for each of the major desktop platforms:
  - keyring.backends.kwallet.DBusKeyring
  - keyring.backends.OS_X.Keyring
  - keyring.backends.SecretService.Keyring
  - keyring.backends.Windows.WinVaultKeyring

  All other keyrings have been moved to a new package, keyrings.alt and backward-compatibility aliases removed. To retain availability of these less preferred keyrings, include that package in your installation (install both keyring and keyrings.alt).

  As these keyrings have moved, any keyrings indicated explicitly in configuration will need to be updated to replace "keyring.backends." with "keyrings.alt.". For example, "keyring.backends.file.PlaintextKeyring" becomes "keyrings.alt.file.PlaintextKeyring".

13.1.35 7.3.1

22 Jan 2016

- Issue #194: Redirect away from docs until they have something more than the changelog. Users seeking the changelog will want to follow the direct link.

13.1.36 7.3

14 Jan 2016

- Issue #117: Added support for filtering which backends are acceptable. To limit to only loading recommended keyrings (those with priority >= 1), call:

```
keyring.core.init_backend(limit=keyring.core.recommended)
```

13.1.37 7.2

12 Jan 2016

- Pull Request #190: OS X backend now exposes a `keychain` attribute, which if set will be used by `get_password` when retrieving passwords. Useful in environments such as when running under cron where the default keychain is not the same as the default keychain in a login session. Example usage:

```
keyring.get_keyring().keychain = '/path/to/login.keychain'
pw = keyring.get_password(...)
```

13.1.38 7.1

10 Jan 2016

- Issue #186: Removed preference for keyrings based on `XDG_CURRENT_DESKTOP` as these values are to varied to be a reliable indicator of which keyring implementation might be preferable.

13.1.39 7.0.2

10 Jan 2016

- Issue #187: Restore `Keyring` name in `kwallet` backend. Users of keyring 6.1 or later should prefer an explicit reference to DBusKeyring or QtKeyring instead.

13.1.40 7.0.1

09 Jan 2016

- Issue #183 and Issue #185: Gnome keyring no longer relies on environment variables, but instead relies on the GnomeKeyring library to determine viability.

13.1.41 7.0

09 Jan 2016

- Issue #99: Keyring now expects the config file to be located in the XDG_CONFIG_HOME rather than XDG_DATA_HOME and will fail to start if the config is found in the old location but not the new. On systems where the two locations are distinct, simply copy or symlink the config to remain compatible with older versions or move the file to work only with 7.0 and later.
- Replaced Pull Request #182 with a conditional SessionBus construction, based on subsequent discussion.

13.1.42 6.1.1

08 Jan 2016

- Pull Request #182: Prevent DBus from indicating as a viable backend when no viable X DISPLAY variable is present.

13.1.43 6.1

06 Jan 2016

- Pull Request #174: Add DBus backend for KWallet, preferred to Qt backend. Theoretically, it should be auto-detected based on available libraries and interchangeable with the Qt backend.

13.1.44 6.0

05 Jan 2016

- Drop support for Python 2.6.

13.1.45 5.7.1

12 Dec 2015

- Updated project metadata to match Github hosting and generally refreshed the metadata structure to match practices with other projects.

13.1.46 5.7

06 Dec 2015

- Issue #177: Resolve default keyring name on Gnome using the API.
- Issue #145: Add workaround for password exposure through process status for most passwords containing simple characters.

13.1.47 5.6

11 Oct 2015

- Allow keyring to be invoked from command-line with `python -m keyring`.

13.1.48 5.5.1

11 Oct 2015

- Issue #156: Fixed test failures in `pyfs` keyring related to 0.5 release.

13.1.49 5.5

11 Oct 2015

- Pull Request #176: Use recommended mechanism for checking GnomeKeyring version.

13.1.50 5.4

07 Aug 2015

- Prefer setuptools_scm to hgtools.

13.1.51 5.3

25 Feb 2015

- Prefer hgtools to setuptools_scm due to setuptools_scm '#21 <https://bitbucket.org/pypa/setuptools_scm/issue/21>'_.

13.1.52 5.2

24 Feb 2015

- Prefer setuptools_scm to hgtools.

13.1.53 5.1

24 Feb 2015

- Host project at Github (repo).

13.1.54 5.0

02 Feb 2015

- Version numbering is now derived from the code repository tags via hgtools.
- Build and install now requires setuptools.

13.1.55 4.1.1

02 Feb 2015

- The entry point group must look like a module name, so the group is now "keyring.backends".

13.1.56 4.1

26 Jan 2015

- Added preliminary support for loading keyring backends through `setuptools entry points`, specifically "keyring backends".

13.1.57 4.0

31 Jul 2014

- Removed `keyring_path` parameter from `load_keyring`. See release notes for 3.0.3 for more details.
- Issue #22: Removed support for loading the config from the current directory. The config file must now be located in the platform-specific config location.

13.1.58 3.8

11 May 2014

- Issue #22: Deprecated loading of config from current directory. Support for loading the config in this manner will be removed in a future version.
- Issue #131: Keyring now will prefer pywin32-ctypes to pywin32 if available.

13.1.59 3.7

21 Mar 2014

- Gnome keyring no longer relies on the GNOME_KEYRING_CONTROL environment variable.
- Issue #140: Restore compatibility for older versions of PyWin32.

13.1.60 3.6

03 Mar 2014

- Pull Request ‘#1 (github) <https://github.com/jaraco/keyring/pull/1>‘_: Add support for packages that wish to bundle keyring by using relative imports throughout.

13.1.61 3.5

14 Feb 2014

- Issue #49: Give the backend priorities a 1.5 multiplier bump when an XDG_CURRENT_DESKTOP environment variable matches the keyring's target environment.
- Issue #99: Clarified documentation on location of config and data files. Prepared the code base to treat the two differently on Unix-based systems. For now, the behavior is unchanged.

13.1.62 3.4

01 Jan 2014

- Extracted FileBacked and Encrypted base classes.
- Add a pyinstaller hook to expose backend modules. Ref #124
- Pull request #41: Use errno module instead of hardcoding error codes.
- SecretService backend: correctly handle cases when user dismissed the collection creation or unlock prompt.

13.1.63 3.3

29 Nov 2013

- Pull request #40: KWallet backend will now honor the `KDE_FULL_SESSION` environment variable as found on openSUSE.

13.1.64 3.2.1

17 Nov 2013

- SecretService backend: use a different function to check that the backend is functional. The default collection may not exist, but the collection will remain usable in that case.

  Also, make the error message more verbose.

  Resolves https://bugs.launchpad.net/bugs/1242412.

13.1.65 3.2

27 Oct 2013

- Issue #120: Invoke KeyringBackend.priority during load_keyring to ensure that any keyring loaded is actually viable (or raises an informative exception).
- File keyring:
  - Issue #123: fix removing items.
  - Correctly escape item name when removing.
  - Use with statement when working with files.
- Add a test for removing one item in group.
- Issue #81: Added experimental support for third-party backends. See *keyring.core._load_library_extensions* for information on supplying a third-party backend.

13.1.66 3.1

20 Sep 2013

- All code now runs natively on both Python 2 and Python 3, no 2to3 conversion is required.
- Testsuite: clean up, and make more use of unittest2 methods.

13.1.67 3.0.5

18 Sep 2013

- Issue #114: Fix logic in pyfs detection.

13.1.68 3.0.4

18 Sep 2013

- Issue #114: Fix detection of pyfs under Mercurial Demand Import.

13.1.69 3.0.3

07 Sep 2013

- Simplified the implementation of `keyring.core.load_keyring`. It now uses `__import__` instead of loading modules explicitly. The `keyring_path` parameter to `load_keyring` is now deprecated. Callers should instead ensure their module is available on `sys.path` before calling `load_keyring`. Keyring still honors `keyring-path`. This change fixes Issue #113 in which the explicit module loading of keyring modules was breaking package-relative imports.

13.1.70 3.0.2

07 Sep 2013

- Renamed `keyring.util.platform` to `keyring.util.platform_`. As reported in Issue #112 and mercurial_keyring '#31 <https://bitbucket.org/Mekk/mercurial_keyring/issue/31>'_ and in Mercurial itself, Mercurial's Demand Import does not honor `absolute_import` directives, so it's not possible to have a module with the same name as another top-level module. A patch is in place to fix this issue upstream, but to support older Mercurial versions, this patch will remain for some time.

13.1.71 3.0.1

02 Sep 2013

- Ensure that modules are actually imported even in Mercurial's Demand Import environment.

13.1.72 3.0

02 Sep 2013

- Removed support for Python 2.5.
- Removed names in `keyring.backend` moved in 1.1 and previously retained for compatibility.

13.1.73 2.1.1

31 Aug 2013

- Restored Python 2.5 compatibility (lost in 2.0).

13.1.74 2.1

31 Aug 2013

- Issue #10: Added a 'store' attribute to the OS X Keyring, enabling custom instances of the KeyringBackend to use another store, such as the 'internet' store. For example:

```
keys = keyring.backends.OS_X.Keyring()
keys.store = 'internet'
keys.set_password(system, user, password)
keys.get_password(system, user)
```

  The default for all instances can be set in the class:

```
keyring.backends.OS_X.Keyring.store = 'internet'
```

- GnomeKeyring: fix availability checks, and make sure the warning message from pygobject is not printed.
- Fixes to GnomeKeyring and SecretService tests.

13.1.75 2.0.3

30 Aug 2013

- Issue #112: Backend viability/priority checks now are more aggressive about module presence checking, requesting `__name__` from imported modules to force the demand importer to actually attempt the import.

13.1.76 2.0.2

26 Aug 2013

- Issue #111: Windows backend isn't viable on non-Windows platforms.

13.1.77 2.0.1

24 Aug 2013

- Issue #110: Fix issues with `Windows.RegistryKeyring`.

13.1.78 2.0

17 Aug 2013

- Issue #80: Prioritized backend support. The primary interface for Keyring backend classes has been refactored to now emit a 'priority' based on the current environment (operating system, libraries available, etc). These priorities provide an indication of the applicability of that backend for the current environment. Users are still welcome to specify a particular backend in configuration, but the default behavior should now be to select the most appropriate backend by default.

13.1.79 1.6.1

23 Jul 2013

- Only include pytest-runner in 'setup requirements' when ptr invocation is indicated in the command-line (Issue #105).

13.1.80 1.6

13 Jul 2013

- GNOME Keyring backend:
  - Use the same attributes (`username` / `service`) as the SecretService backend uses, allow searching for old ones for compatibility.
  - Also set `application` attribute.

  - Correctly handle all types of errors, not only `CANCELLED` and `NO_MATCH`.
  - Avoid printing warnings to stderr when GnomeKeyring is not available.
- Secret Service backend:
  - Use a better label for passwords, the same as GNOME Keyring backend uses.

13.1.81 1.5

23 Jun 2013

- SecretService: allow deleting items created using previous python-keyring versions.

  Before the switch to secretstorage, python-keyring didn't set "application" attribute. Now in addition to supporting searching for items without that attribute, python-keyring also supports deleting them.
- Use `secretstorage.get_default_collection` if it's available.

  On secretstorage 1.0 or later, python-keyring now tries to create the default collection if it doesn't exist, instead of just raising the error.
- Improvements for tests, including fix for Issue #102.

13.1.82 1.4

05 Jun 2013

- Switch GnomeKeyring backend to use native libgnome-keyring via GObject Introspection, not the obsolete python-gnomekeyring module.

13.1.83 1.3

25 May 2013

- Use the SecretStorage library to implement the Secret Service backend (instead of using dbus directly). Now the keyring supports prompting for and deleting passwords. Fixes #69, #77, and #93.
- Catch *gnomekeyring.IOError* per the issue reported in Nova client.
- Issue #92 Added support for delete_password on Mac OS X Keychain.

13.1.84 1.2.3

25 May 2013

- Fix for Encrypted File backend on Python 3.
- Issue #97 Improved support for PyPy.

13.1.85 1.2.2

10 Feb 2013

- Fixed handling situations when user cancels kwallet dialog or denies access for the app.

13.1.86 1.2.1

07 Feb 2013

- Fix for kwallet delete.
- Fix for OS X backend on Python 3.
- Issue #84: Fix for Google backend on Python 3 (use of raw_input not caught by 2to3).

13.1.87 1.2

03 Jan 2013

- Implemented delete_password on most keyrings. Keyring 2.0 will require delete_password to implement a Keyring. Fixes #79.

13.1.88 1.1.2

02 Jan 2013

- Issue #78: pyfilesystem backend now works on Windows.

13.1.89 1.1.1

02 Jan 2013

- Fixed MANIFEST.in so .rst files are included.

13.1.90 1.1

01 Jan 2013

This is the last build that will support installation in a pure-distutils mode. Subsequent releases will require setuptools/distribute to install. Python 3 installs have always had this requirement (for 2to3 install support), but starting with the next minor release (1.2+), setuptools will be required.

Additionally, this release has made some substantial refactoring in an attempt to modularize the backends. An attempt has been made to maintain 100% backward-compatibility, although if your library does anything fancy with module structure or clasess, some tweaking may be necessary. The backward-compatible references will be removed in 2.0, so the 1.1+ releases represent a transitional implementation which should work with both legacy and updated module structure.

- Added a console-script 'keyring' invoking the command-line interface.
- Deprecated _ExtensionKeyring.
- Moved PasswordSetError and InitError to an *errors* module (references kept for backward-compatibility).
- Moved concrete backend implementations into their own modules (references kept for backward compatibility):
  - OSXKeychain -> backends.OS_X.Keyring
  - GnomeKeyring -> backends.Gnome.Keyring
  - SecretServiceKeyring -> backends.SecretService.Keyring
  - KDEKWallet -> backends.kwallet.Keyring

  - BasicFileKeyring -> backends.file.BaseKeyring
  - CryptedFileKeyring -> backends.file.EncryptedKeyring
  - UncryptedFileKeyring -> backends.file.PlaintextKeyring
  - Win32CryptoKeyring -> backends.Windows.EncryptedKeyring
  - WinVaultKeyring -> backends.Windows.WinVaultKeyring
  - Win32CryptoRegistry -> backends.Windows.RegistryKeyring
  - select_windows_backend -> backends.Windows.select_windows_backend
  - GoogleDocsKeyring -> backends.Google.DocsKeyring
  - Credential -> keyring.credentials.Credential
  - BaseCredential -> keyring.credentials.SimpleCredential
  - EnvironCredential -> keyring.credentials.EnvironCredential
  - GoogleEnvironCredential -> backends.Google.EnvironCredential
  - BaseKeyczarCrypter -> backends.keyczar.BaseCrypter
  - KeyczarCrypter -> backends.keyczar.Crypter
  - EnvironKeyczarCrypter -> backends.keyczar.EnvironCrypter
  - EnvironGoogleDocsKeyring -> backends.Google.KeyczarDocsKeyring
  - BasicPyfilesystemKeyring -> backends.pyfs.BasicKeyring
  - UnencryptedPyfilesystemKeyring -> backends.pyfs.PlaintextKeyring
  - EncryptedPyfilesystemKeyring -> backends.pyfs.EncryptedKeyring
  - EnvironEncryptedPyfilesystemKeyring -> backends.pyfs.KeyczarKeyring
  - MultipartKeyringWrapper -> backends.multi.MultipartKeyringWrapper

- Officially require Python 2.5 or greater (although unofficially, this requirement has been in place since 0.10).

13.1.91 1.0

29 Nov 2012

This backward-incompatible release attempts to remove some cruft from the codebase that's accumulated over the versions.

- Removed legacy file relocation support. *keyring* no longer supports loading configuration or file-based backends from ~. If upgrading from 0.8 or later, the files should already have been migrated to their new proper locations. If upgrading from 0.7.x or earlier, the files will have to be migrated manually.
- Removed CryptedFileKeyring migration support. To maintain an existing CryptedFileKeyring, one must first upgrade to 0.9.2 or later and access the keyring before upgrading to 1.0 to retain the existing keyring.
- File System backends now create files without group and world permissions. Fixes #67.

13.1.92 0.10.1

29 Nov 2012

- Merged 0.9.3 to include fix for #75.

13.1.93 0.10

12 Nov 2012

- Add support for using Keyczar to encrypt keyrings. Keyczar is "an open source cryptographic toolkit designed to make it easier and safer for developers to use cryptography in their applications."
- Added support for storing keyrings on Google Docs or any other filesystem supported by pyfilesystem.
- Fixed issue in Gnome Keyring when unicode is passed as the service name, username, or password.
- Tweaked SecretService code to pass unicode to DBus, as unicode is the preferred format.
- Issue #71 - Fixed logic in CryptedFileKeyring.
- Unencrypted keyring file will be saved with user read/write (and not group or world read/write).

13.1.94 0.9.3

29 Nov 2012

- Ensure migration is run when get_password is called. Fixes #75. Thanks to Marc Deslauriers for reporting the bug and supplying the patch.

13.1.95 0.9.2

11 Jun 2012

- Keyring 0.9.1 introduced a whole different storage format for the CryptedFileKeyring, but this introduced some potential compatibility issues. This release incorporates the security updates but reverts to the INI file format for storage, only encrypting the passwords and leaving the service and usernames in plaintext. Subsequent releases may incorporate a new keyring to implement a whole-file encrypted version. Fixes #64.
- The CryptedFileKeyring now requires simplejson for Python 2.5 clients.

13.1.96 0.9.1

03 Jun 2012

- Fix for issue where SecretServiceBackend.set_password would raise a UnicodeError on Python 3 or when a unicode password was provided on Python 2.
- CryptedFileKeyring now uses PBKDF2 to derive the key from the user's password and a random hash. The IV is chosen randomly as well. All the stored passwords are encrypted at once. Any keyrings using the old format will be automatically converted to the new format (but will no longer be compatible with 0.9 and earlier). The user's password is no longer limited to 32 characters. PyCrypto 2.5 or greater is now required for this keyring.

13.1.97 0.9

20 Apr 2012

- Add support for GTK 3 and secret service D-Bus. Fixes #52.
- Issue #60 - Use correct method for decoding.

13.1.98 0.8.1

05 Mar 2012

- Fix regression in keyring lib on Windows XP where the LOCALAPPDATA environment variable is not present.

13.1.99 0.8

21 Feb 2012

- Mac OS X keyring backend now uses subprocess calls to the *security* command instead of calling the API, which with the latest updates, no longer allows Python to invoke from a virtualenv. Fixes issue #13.
- When using file-based storage, the keyring files are no longer stored in the user's home directory, but are instead stored in platform-friendly locations (*%localappdata%Python Keyring* on Windows and according to the freedesktop.org Base Dir Specification (*$XDG_DATA_HOME/python_keyring* or *$HOME/.local/share/python_keyring*) on other operating systems). This fixes #21.

Backward Compatibility Notice

Due to the new storage location for file-based keyrings, keyring 0.8 supports backward compatibility by automatically moving the password files to the updated location. In general, users can upgrade to 0.8 and continue to operate normally. Any applications that customize the storage location or make assumptions about the storage location will need to take this change into consideration. Additionally, after upgrading to 0.8, it is not possible to downgrade to 0.7 without manually moving configuration files. In 1.0, the backward compatibility will be removed.

13.1.100 0.7.1

21 Jan 2012

- Removed non-ASCII characters from README and CHANGES docs (required by distutils if we're to include them in the long_description). Fixes #55.

13.1.101 0.7

28 Dec 2011

- Python 3 is now supported. All tests now pass under Python 3.2 on Windows and Linux (although Linux backend support is limited). Fixes #28.
- Extension modules on Mac and Windows replaced by pure-Python ctypes implementations. Thanks to Jerome Laheurte.
- WinVaultKeyring now supports multiple passwords for the same service. Fixes #47.
- Most of the tests don't require user interaction anymore.
- Entries stored in Gnome Keyring appears now with a meaningful name if you try to browser your keyring (for ex. with Seahorse)
- Tests from Gnome Keyring no longer pollute the user own keyring.
- *keyring.util.escape* now accepts only unicode strings. Don't try to encode strings passed to it.

13.1.102 0.6.2

03 Aug 2011

- fix compiling on OSX with XCode 4.0

13.1.103 0.6.1

02 Aug 2011

- Gnome keyring should not be used if there is no DISPLAY or if the dbus is not around (https://bugs.launchpad.net/launchpadlib/+bug/752282).
- Added *keyring.http* for facilitating HTTP Auth using keyring.
- Add a utility to access the keyring from the command line.

13.1.104 0.5.1

07 Jan 2011

- Remove a spurious KDE debug message when using KWallet
- Fix a bug that caused an exception if the user canceled the KWallet dialog (https://bitbucket.org/kang/python-keyring-lib/issue/37/user-canceling-of-kde-wallet-dialogs).

13.1.105 0.5

05 Nov 2010

- Now using the existing Gnome and KDE python libs instead of custom C++ code.
- Using the getpass module instead of custom code

13.1.106 0.4

12 Sep 2010

- Fixed the setup script (some subdirs were not included in the release.)

13.1.107 0.3

07 Jul 2010

- Fixed keyring.core when the user doesn't have a cfg, or is not properly configured.
- Fixed escaping issues for usernames with non-ascii characters

13.1.108 0.2

06 Sep 2009

- Add support for Python 2.4+ http://bitbucket.org/kang/python-keyring-lib/issue/2
- Fix the bug in KDE Kwallet extension compiling http://bitbucket.org/kang/python-keyring-lib/issue/3

keyring.**set_keyring**(*keyring*)
Set current keyring backend.

keyring.**get_keyring**()
Get current keyring backend.

keyring.**set_password**(*service_name*, *username*, *password*)
Set password for the user in the specified service.

keyring.**get_password**(*service_name*, *username*)
Get password from the specified service.

keyring.**delete_password**(*service_name*, *username*)
Delete the password for the user in the specified service.

keyring.**get_pass_get_password**(*prompt='Password: '*, *stream=None*, *service_name='Python'*, *username=None*)

CHAPTER 14

Indices and tables

- genindex
- modindex
- search

Python Module Index

k

Index