# On Mac Keychains

This thread has been locked by a moderator.

1.4k

**IMPORTANT** This post is now retired in favour of TN3137 On Mac keychain APIs and implementations. I'm leaving the original post here just for the record, but you should consider the official documentation authoritative.

Greetings All

I regularly talk to folks who are confused by keychains on the Mac. This is understandable as the Mac has three keychain APIs and two keychain implementations! This post is my attempt to describe how those fit together.

Share and Enjoy
—
Quinn "The Eskimo!" @ Developer Technical Support @ Apple
`let myEmail = "eskimo" + "1" + "@" + "apple.com"`

## On Mac Keychains

macOS has three keychain *APIs*:

- Keychain (`<KeychainCore.h>`) — This originated on traditional Mac OS but was supported on Mac OS X (now macOS) as a compatibility measure.
- SecKeychain (`<Security/SecKeychain.h>`, `<Security/SecKeychainItem.h>`, `<Security/SecKeychainSearch.h>`) — This was introduced with Mac OS X [1] and was the standard macOS keychain API until recently.
- SecItem (`<Security/SecItem.h>`) — This was part of the original iOS SDK. It debuted on the Mac with macOS 10.6.

The first API is now irrelevant.

macOS has two keychain *implementations*:

- The file-based keychain
- The data protection keychain

The file-based keychain has its origins on traditional Mac OS. The data protection keychain originated on iOS and came to macOS with the advent of iCloud Keychain on macOS 10.9.

**Note** iOS and its child platforms only support the SecItem API with the data protection keychain implementation.

The Keychain and SecKeychain APIs only talk to the file-based keychain. The SecItem API talks to either implementation. Specifically, it talks to the data protection keychain if you supply either the `kSecUseDataProtectionKeychain` or the `kSecAttrSynchronizable` attribute. If not, it talks to the file-based keychain.

The file-based keychain is on the road to deprecation. It is not officially deprecated, but some of the APIs surrounding it are. For example, `SecKeychainCreate` was deprecated in macOS 12. Moreover, new features, like iCloud Keychain, are only supported by the data protection keychain.

[1] Actually, I'm not 100% sure it was part of 10.0. The earliest macOS SDK that I have easy access to is the macOS 10.2 SDK, circa 2002, and it definitely has this API.

## API Differences

The SecItem API is well aligned with the data protection keychain. When you use it to talk to the file-based keychain the API has to work through a shim. That shim has limitations. Some of those limitations are inherent to the keychain implementation. For example, the access control model of the file-based keychain is very different from that of the data protection keychain, and the shim can't make up for that. However, some limitations are just bugs. If you're porting iOS keychain code to the Mac, it's best to target the data protection keychain.

Mac Catalyst apps only have access to the data protection keychain.

iOS Apps on Mac only have access to the data protection keychain.

## Implementation Differences

Each keychain implementation uses its own access control model:

- The file-based keychain uses access control lists (`SecAccess`).
- The data protection keychain uses keychain access control groups, supplemented by an access control object (`SecAccessControl`).

The keychain access control groups available to you in the data protection keychain are determined by code signing entitlements. See Sharing Access to Keychain Items Among a Collection of Apps.

This means that the data protection keychain is only available to code that can carry an entitlement, that is, main executables like an app or an app extension. If you're building library code then your data protection keychain access is determined by the app that loads your library.

The data protection keychain is only available in a user login context. You cannot use it, for example, from a `launchd` daemon.

The data protection keychain can hold all keychain item classes (Internet password, generic password, certificate, key). Only the password items are synchronised via iCloud Keychain.

Modern keychain features are only supported by the data protection keychain, including:

- iCloud Keychain
- Protecting an item with biometrics (Touch ID and Face ID)
- Protecting a key with the Secure Enclave

## User Interface

The Keychain Access application lets you manage both file-based keychains and the data protection keychain. The keychain list includes at least two file-based keychainS (*login* and *System*) and the data protection keychain (*iCloud Keychain*, if that's enabled, otherwise *Local Items*). You can create, add, and remove keychains using commands on the File menu (New Keychain, Add Keychain, Delete Keychain).

Keychain Access displays all keychain items in file-based keychains but only password items in the data protection keychain.

The keychain support in the `security` command-line tool is primarily focused on the file-base keychain.

Security

Reply

Posted 1 year ago by eskimo

Add a Comment