How notarization works

Published: 2020-08-28 · Archived: 2026-04-05 19:09:19 UTC

In Catalina and Big Sur, notarization is no longer a bonus: for some types of software like extensions and most plug-ins, it's essential. The only two general exceptions are software supplied through Apple's App Store, and Apple's own software, including the whole of macOS itself. You can still run apps and command tools which haven't been notarized, but if they've been downloaded from the Internet or moved to your Mac using AirDrop (which also sets a quarantine flag) it's getting progressively more difficult to do so. In Big Sur, it's no longer just a matter of opening the unnotarized app in the Finder.

Notarization

When a developer notarizes their software, they have to build it to comply with Apple's rules, which include signing it fully and correctly, and 'hardening' the runtime. They can't submit the app or command tool as it stands, though: it has to be packaged in a way that's acceptable to the Notary Service. That includes disk images (in UDIF format), signed flat Installer packages (as explained here), and Zip archives (as used by Xcode to notarize apps). The notarization is then specific to the app or executable contained within that.

The Notary Service checks the submission for malware. If none is found, signatures are in order, and other requirements are met, Apple adds its cryptographic hash and other details to its notarization database, and issues a 'ticket', which the developer can download and 'staple' (attach) to the software.

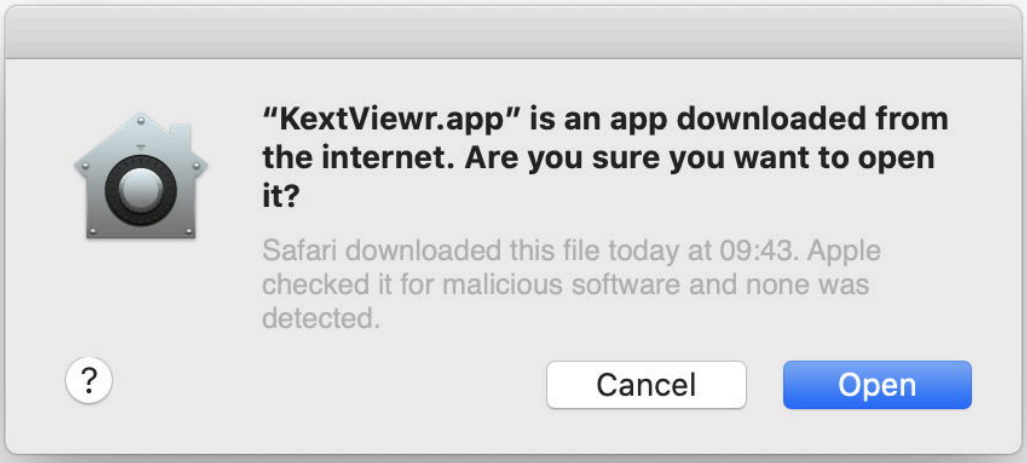
Tickets can't be stapled to single-file Mach-O executables, but they can be stapled to Installer packages containing them. They're most commonly stapled to an app or bundle in the file named CodeResources inside the bundle. However, because its details are recorded in Apple's database, the ticket doesn't have to be present for the app to be recognised as having been successfully notarized.

These procedures are straightforward for simple apps built in Xcode. The more complex the app, with helpers and plug-ins particularly, the more difficult these become. This may involve separate notarization of components, assembly into the whole, and notarization of that. For smaller developers, this is very demanding: please show understanding if they encounter problems.

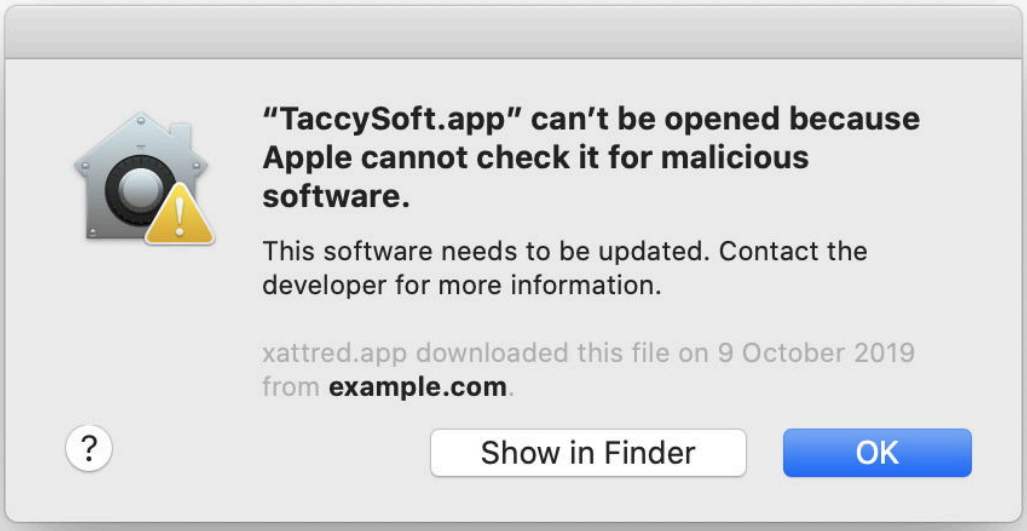
First run

Whenever you run an app, command tool, or other executable code, macOS looks to see if it has a quarantine flag set. If it finds one, Gatekeeper looks for a notarization ticket. If that's stapled to the bundle, its validity is checked by sending a cryptographic hash to Apple's servers. If there's no ticket, and no record of one in your Mac's local security database, Gatekeeper sends the cryptographic hash to Apple's servers to see if they have a ticket on record. This checking has been documented in Catalina by Jeff Johnson, even for unsigned shell scripts.

If Gatekeeper finds a valid ticket for that hash, either locally or on Apple's servers, and the signature is good, XProtect checks that the code contains no known malware, and you're invited to proceed with running the app.



If the app or code hasn't been notarized, then the normal process is stopped, and you're informed of the failure. You can then opt to bypass the notarization check if you wish.



First run notarization checks can impose significant delays on opening apps, which some users have complained about. They might complete more quickly when the notarization ticket has been stapled to the app, but the main purpose of delivering a stapled ticket with software is to enable Gatekeeper to check the ticket when Internet

access isn’t available. Any repeat checks may use cached results of previous checks recorded in the local security database, avoiding further delay.

Mystery notarization

Occasionally, you may come across old software which couldn’t have been notarized when it was released, but which now behaves as if it has been notarized: put it through app first run with a quarantine flag set, and you’ll see the same dialog as for a notarized app. This is because, since its original release, that software has been notarized, and possibly not even by its original developer, as I explained here.

Apple has operated schemes which encouraged developers to retrospectively notarize older versions of their products, and still encourages them to do so whenever possible. Because notarization can also be performed by third parties, as well as the developer, it’s quite possible that the developer isn’t even aware. There’s nothing sinister in this, however spooky it might seem at the time.

Finally, if you do encounter any problems with the notarization of software, please contact its developer, as they’re the only ones that can fix it. But appreciate that they may well have been struggling to solve their notarization problems for quite some time. Sometimes it isn’t straightforward at all.
