Why a bash script ran directly doesn't need user permissions, but a .app file running the script needs them?

Asked 6 years ago Modified 6 years ago Viewed 242 times

4

I have a bash script `myscript` that loads some applescript to prompt a graphical interface to the user, to ask for his password.

- If I run the bash file directly, outside of a .app bundle, it can do it without the system prompt `do you want to allow this program to access...[stuff]?`
- But if I put the same bash file inside a .app bundle (with the same name as the script), and double click the .app file, then it will first show the system prompt to ask for user authorisation. I have added no special entitlements to the app.

Is there a link that explains how a .app security is different for executing a script as opposed from the command line?

bash macos entitlements

Share

Improve this question

Follow

edited Dec 16, 2018 at 16:37

bmike
975 ● 2 ● 24 ● 42

asked Dec 16, 2018 at 16:29

Johnny Pralo
55 ● 4

1 How do you create the .app bundle? – nohillside Dec 16, 2018 at 16:30

new folder > rename it to myscript.app > show packages content > add my script 'myscript' inside it. – Johnny Pralo Dec 16, 2018 at 16:31

1 Answer

Sorted by: Highest score (default)

5

In both cases the script is being launched by an app. In one case that app is Terminal, and it has the required permissions. Your app doesn't at this point.

This is a change in 10.14, described in the release notes:

> Sending Apple events from an app—including script applets—now requires user approval. The list of currently approved apps can be viewed and edited in the Automation category in the Privacy tab in System Preferences > Security & Privacy. If an event is blocked because the user didn’t approve that app, the event will fail with the error code: -1743 ('': Not authorized to send Apple events to ''). An event can be preflighted

> using AEDeterminePermissionToAutomateTarget(::::).

Mojave has introduced very stringent controls around Apple Events (the core of AppleScript), which has greatly complicated automation. Michael Tsai has aggregated a lot of helpful discussion of the issues.

See also: com.apple.iTunes AEDeterminePermissionToAutomateTarget is always return -600

Share Improve this answer Follow

answered Dec 16, 2018 at 16:43

Rob Napier
299k ● 36 ● 481 ● 645

So you when I double click on a .app file, the bash script is not ran inside a terminal? – Johnny Pralo Dec 16, 2018 at 19:28

I'm not certain what you're picturing here, but definitely not. First, there's a difference between "a terminal" (i.e. a process tied to pty) and Terminal.app (a Cocoa application that provides a UI tied to a pty). But neither of them are involved when you invoke `exec` inside of a program (which is how ultimately you're going to run that bash script, and how ultimately that bash script is going to run the apple script). – Rob Napier Dec 16, 2018 at 19:47

Hmm what I was picturing is : When I double-click a script, you said it will open it with Terminal.app, so that's why it already has the permissions. And so when I double-click a .app containing the script, it doesn't run it with the Terminal.app, right? – Johnny Pralo Dec 16, 2018 at 20:25

No, there's no implicit relationship between an application you've built and Terminal. If you want there to be a relationship, you'd have to launch Terminal yourself in your app. (Terminal.app isn't the only "terminal app" out there. I use an app called iTerm for instance. There isn't one particular "terminal" that applications could pick even if that were something they did.) – Rob Napier Dec 16, 2018 at 20:27