Why is it so hard to write a program that requires UI Access privilege?

devblogs.microsoft.com/oldnewthing/20121213-00

December 13, 2012

Raymond Chen

If you want your program to have the UI Access privilege, you have to jump through a few hoops. The program needs to be digitally signed, and it needs to go into the `system32` directory. Why is it so hard to get UI Access? Because UI Access bypasses User Interface Privilege Isolation (UIPI) security measures. The more valuable the target, the more substantial the security measures. UI Access allows low-integrity programs to access and interact with the user interface of high-integrity programs. This has historically been the source of security vulnerabilities. UIPI was created in part to prevent this type of security attack. If a low-integrity program could programmatically manipulate the user interface of a high-integrity program, then all an attacker needs to do is sit and wait for the user to elevate a command prompt, and then start programmatically driving the command prompt to do whatever it wanted. If all you had to do to obtain UI Access was simply ask for it (by setting `uiaccess="true"` in your manifest), then every piece of malware would just do that, and boom, the value of UIPI has effectively vanished. (This is the sort of trap that leads to eventually, nothing is special any more.) Okay, so the digital signature requirement is there to create a barrier to entry for malware authors. It also creates some degree of accountability (since you have to identify yourself to a certificate authority, though as we've seen in the past, this relies on certificate authorities remaining trustworthy), And it allows your application's ability to obtain UI Access to be revoked by revoking the certificate. But why does the file have to be in the `system32` directory? As we saw some time ago, the directory is the application bundle. If programs with UI Access could be installed anywhere, then an attacker could exploit an insecure application directory to plant a rogue copy of a system DLL in that directory, allowing itself to be injected into the process with UI Access and thereby compromise it. The thinking here is "If the application cannot create its install directory, then the application cannot create its install directory *wrong*."

Requiring progams with UI Access to be installed into the `system32` directory is an additional *secure by default* measure. In order to compromise the application bundle, the attacker must already have compromised the `system32` directory, at which point he's already on the other side of the airtight hatchway.

Raymond Chen

Follow