It rather involved being on the other side of this airtight hatchway: Spoofing another program

devblogs.microsoft.com/oldnewthing/20231220-44

December 20, 2023

Raymond Chen

A security vulnerability report arrived that went roughly like this:

> When a desktop application registers to produce toast notifications, it provides its Application User Model ID (AUMID). An attacker can spoof any app by substituting the victim app's AUMID when registering. If the victim generates a toast notification, the desktop application will steal the click and can perform whatever action it desires. In this proof of concept, we launch Notepad.

That sounds bad when you put it that way: One app can steal clicks intended for another app.

But wait, the attacker is a desktop application that is already running code. If you can run code, then don't be surprised that you can run code: The attacker doesn't have to wait for the victim to generate a toast notification. Whatever evil things they want to do, they can just do them right away! No need to wait.

If your concern is that they can prevent the victim program from receiving clicks, well, desktop applications can already do that. Install a low-level mouse hook, and whenever you see a mouse click, see if it's on a toast from your victim program. If so, then prevent the click from being processed, and do your own evil thing instead.

Even without the benefit of low-level mouse hooks, desktop applications run at medium integrity,[1] so they have the ability to attack any of the user's programs that are running at low integrity (such as those running in the restricted UWP environment), as well as any of the user's programs that are running at medium integrity (traditional Win32 programs running non-elevated). The attacker can inject code into the victim process and patch the click handler so it does something evil instead of whatever the program intended.

Heck, they could inject code into Explorer and just patch the entire toast notification infrastructure! Generate fake notifications, suppress valid ones, alter the text in the notifications, go nuts.

Squatting on another program’s AUMID when registering for toast notifications doesn’t give desktop applications any powers beyond what they already had. Desktop applications run at medium integrity, which already gives them a great deal of power. By running a desktop app, you are trusting that they don’t abuse that power.

[1] And you can even choose to run them at high integrity if you run them elevated.