# How can I write an unkillable program, redux

devblogs.microsoft.com/oldnewthing/20160505-00

May 5, 2016

Raymond Chen

A customer wanted to know how to write an unkillable process. Specifically, they want to write a program that runs continuously in the background, and they don't want the logged-in used (who is not an administrator) to be able to terminate it. They wanted to know if there was a way to hide the program from Task Manager, so the user cannot click "End Task".

The usual way to prevent a user from killing a program is to run the program as some other user. Users can kill their own programs (by virtue of being the owner), but they cannot kill programs that are owned by others, unless they have administrator privileges, or something administrator-equivalent like debug privilege, or unless the program permits others to terminate it.

"To clarify the scenario: This program needs to display a message to the user. Therefore, it cannot run as a service, because that would open the service to a shatter attack. The message is one to tell the user that the work day is over and they need to wrap things up and go home. We found that users are killing the program to make the message go away."

Okay, so what we really have here is a social problem, not a technical one. You want to remind the user to go home. Those that want to keep working are killing the program. You think the solution is to make the program unkillable, but that really doesn't change the situation. Even if you made the program unkillable, the user can simply *ignore the message*. There's nothing you can do to make a user pay attention to a message. If they want to ignore it, then they will ignore it.

"Yes, that was my advice to the customer, but the customer isn't looking for a solution. They just want an answer. They absolutely want to write the program that displays the 'Go home' message and make it unkillable."

What you can do is have a service inject the program into the session, and let it run as the logged-on user. (You have to make it run as the logged-on user in order to avoid a shatter attack.) The service can monitor the program, and if the user terminates it, the service can log an entry to the security event log, and you can roll up those events into a report so you can see which users are killing the reminder app, then apply appropriate social pressure.

If you want to be really annoying about it, you can even relaunch the app that displays the "Go home" message. Of course, that'll just trigger Le Chatelier's Principle for complex systems: The user will write a script that kills the app as soon as it is relaunched, or attack the app so it doesn't show the message in the first place.

If you want to make sure the user goes home, you can have the service lock the workstation and set a restricted logon policy so the user cannot log back in until 9am the next morning. The service can inject a program into the session to display a countdown timer. In this case, the user can terminate the countdown timer app all they want; that's not going to stop the workstation from being locked. They're just shooting the messenger.

Raymond Chen

**Follow**