# How can I detect whether the user is running as an elevated administrator (as opposed to a natural administrator)?

devblogs.microsoft.com/oldnewthing/20241003-00

October 3, 2024


When a user with administrator privileges signs in, the User Account Control (UAC) feature signs in the administrator with a so-called “split token”, in which the user operates in Clark Kent mode: Although they have latent administrator privileges, the administrative powers in the token are disabled. To exercise those administrative powers, the user must elevate their token.

A customer wanted to discourage users from running their program elevated, but they also didn’t want to scold users who were running with UAC disabled (such as on Windows Server), since those users had no opportunity to de-elevate.

The way to inspect whether your token is split, and if so whether you have the non-administrator (“limited”) version or the administrator (“full”) version is to ask for the token’s elevation type.

| | **Standard user** | **Administrative user** |
|---|---|---|
| **UAC disabled** | TokenElevationTypeDefault | TokenElevationTypeDefault |
| **UAC enabled, not elevated** | TokenElevationTypeDefault | TokenElevationTypeLimited |
| **UAC enabled, elevated** | N/A | TokenElevationTypeFull |

Non-administrative users cannot split their token (there being no administrator privileges to split out), and administrative users cannot split their tokens if UAC is disabled.

If you are looking for “users who manually elevated”, then you can consult the table above and see that a token elevation type of `Full` exactly identifies the “Administrative user, UAC enabled, elevated” box.

```
bool IsManuallyElevatedViaUAC()
{
    TOKEN_ELEVATION_TYPE type;
    DWORD actual;
    if (!GetTokenInformation(
            GetCurrentProcessToken(),
            TokenElevationType,
            &type,
            sizeof(type),
            &actual)) {
        // insert your favorite error handling here
        throw_error(GetLastError());
    }
    return type == TokenElevationTypeFull;
}
```

We learned about `GetCurrentProcessToken()` a little while ago. This is a convenient pseudo-handle that refers to the current process. We ask for the current process's token's elevation type, and if that is `Full`, then we are in that box in the bottom right corner.

The helpers in the `token_helpers.h` header in the Windows Implementation Library (wil) turn this into a one-line function.

```
bool IsManuallyElevatedViaUAC()
{
    return wil::get_token_information<TOKEN_ELEVATION_TYPE>(
            GetCurrentProcessToken()) == TokenElevationTypeFull;
}
```