How can I detect that Windows is running in S-Mode?

devblogs.microsoft.com/oldnewthing/20250807-00/?p=111444

August 7, 2025

Windows S-Mode is a mode of Windows which restricts programs to those which came from the Microsoft Store. You might have a program that is available from the Microsoft Store (therefore is allowed to run in S-Mode), and you want to know whether to suggest that the user install a companion program that is not allowed in S-Mode. How can a program detect whether the system is running in S-Mode?

You can detect S-Mode by using the `Windows.System.Profile.WindowsIntegrityPolicy` class.

```
// C#
using Windows.System.Profile;

if (WindowsIntegrityPolicy.IsEnabled) {
    // System is in S-Mode
    if (WindowsIntegrityPolicy.CanDisable) {
        // System is in S-Mode but can be taken out of S-Mode
        suggestCompanion = true;
    } else {
        // System is locked into S-Mode
        suggestCompanion = false;
    }
} else {
    // System is not in S-Mode
    suggestCompanion = true;
}
```

```
// C++/WinRT
#include <winrt/Windows.System.Profile.h>

namespace winrt
{
    using namespace winrt::Windows::System::Profile;
}

if (winrt::WindowsIntegrityPolicy::IsEnabled()) {
    // System is in S-Mode
    if (winrt::WindowsIntegrityPolicy::CanDisable()) {
        // System is in S-Mode but can be taken out of S-Mode
        suggestCompanion = true;
    } else {
        // System is locked into S-Mode
        suggestCompanion = false;
    }
} else {
    // System is not in S-Mode
    suggestCompanion = true;
}

// JavaScript
let WindowsIntegrityPolicy = Windows.System.Profile.WindowsIntegrityPolicy;

if (WindowsIntegrityPolicy.isEnabled) {
    // System is in S-Mode
    if (WindowsIntegrityPolicy.canDisable) {
        // System is in S-Mode but can be taken out of S-Mode
        suggestCompanion = true;
    } else {
        // System is locked into S-Mode
        suggestCompanion = false;
    }
} else {
    // System is not in S-Mode
    suggestCompanion = true;
}

// C++/CX
using namespace Windows::System::Profile;

if (WindowsIntegrityPolicy::IsEnabled) {
    // System is in S-Mode
    if (WindowsIntegrityPolicy::CanDisable) {
        // System is in S-Mode but can be taken out of S-Mode
        suggestCompanion = true;
    } else {
        // System is locked into S-Mode
        suggestCompanion = false;
    }
} else {
    // System is not in S-Mode
    suggestCompanion = true;
}
```

```
// C++/WRL
#include <wrl/client.h>
#include <wrl/wrappers/corewrappers.h>
#include <Windows.System.Profile.h>
#include <wil/result_macros.h>

namespace WRL
{
    using namespace Microsoft::WRL;
    using namespace Microsoft::WRL::Wrappers;
}
namespace ABI
{
    using namespace ABI::Windows::System::Profile;
}

WRL::ComPtr<ABI::IWindowsIntegrityPolicyStatics> statics;
THROW_IF_FAILED(::RoGetActivationFactory(

WRL::HStringReference(RuntimeClass_Windows_System_Profile_WindowsIntegrityPolicy).G

    IID_PPV_ARGS(&statics)));

boolean isEnabled;
THROW_IF_FAILED(statics->get_IsEnabled(&isEnabled));
if (isEnabled) {
    // System is in S-Mode
    boolean canDisable;
    THROW_IF_FAILED(statics->get_CanDisable(&canDisable));
    if (canDisable) {
        // System is in S-Mode but can be taken out of S-Mode
        suggestCompanion = true;
    } else {
        // System is locked into S-Mode
        suggestCompanion = false;
    }
} else {
    // System is not in S-Mode
    suggestCompanion = true;
}
```