Impact of Session 0 Isolation on Services and Drivers in Windows

November 20, 2009

Abstract

In Windows® XP, Windows Server® 2003, and earlier versions of Windows, all services run in Session 0 along with applications. This situation poses a security risk. In Windows Vista®, Windows Server 2008, and later versions of Windows, the operating system isolates services in Session 0 and runs applications in other sessions, so services are protected from attacks that originate in application code.

This paper describes changes to the way in which services are run. It provides guidelines for developers to modify application services and driver services to run in Windows Vista, Windows Server 2008, and later versions of Windows.

This information applies to the following operating systems:  
 Windows 7  
 Windows Server 2008 R2  
 Windows Server 2008  
 Windows Vista

References and resources discussed here are listed at the end of this paper.

The current version of this paper is maintained on the Web at:  
<http://www.microsoft.com/whdc/system/sysinternals/Session0Changes.mspx>

Disclaimer:

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This White Paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, email address, logo, person, place or event is intended or should be inferred.

© 2009 Microsoft Corporation. All rights reserved.

Microsoft, Win32, Windows, Windows Server, and Windows Vista are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

The names of actual companies and products mentioned herein may be the trademarks of their respective owners.

Document History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date | Change |  |  |  |

|  |  |
| --- | --- |
| November 20, 2009 | Updated to include changes to user interface |
| December 20, 2007 | Updated to include Web page link |
| March 17, 2006 | Created |

Contents

[Introduction 3](#_Toc246482657)

[Implications for Services and Service-Hosted Drivers 4](#_Toc246482658)

[What Is Affected 4](#_Toc246482659)

[Potential Issues 5](#_Toc246482660)

[Guidelines for Services and Service-Hosted Drivers 6](#_Toc246482661)

[Interactive Service Detection Service 7](#_Toc246482662)

[Resources 8](#_Toc246482663)

# Introduction

In Windows® XP, Windows Server® 2003, and earlier versions of the Windows operating system, all services run in the same session as the first user who logs on to the console. This session is called Session 0. Running services and user applications together in Session 0 poses a security risk because services run at elevated privilege and therefore are targets for malicious agents who are looking for a way to elevate their own privilege level.

In Windows Vista®, Windows Server 2008, and later versions of Windows, the operating system mitigates this security risk by isolating services in Session 0 and making Session 0 noninteractive. Only system processes and services run in Session 0. The first user logs on to Session 1, and subsequent users log on to subsequent sessions. This means that services never run in the same session as users’ applications and are therefore protected from attacks that originate in application code.

The following figures illustrate the changes. In Windows XP and Windows Server 2003, sessions are assigned as shown in Figure 1.

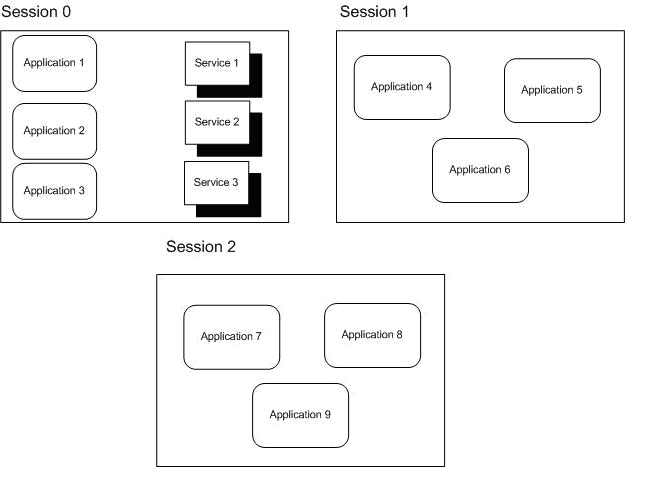


Figure 1. Sessions in Windows XP and Windows Server 2003

Figure 1 illustrates what happens with three users logged on to the system. Session 0 contains both user applications and services. In a system that is running Windows Server 2003, Session 0 is the console session and Sessions 1 and 2 represent remote users. In a system that is running Windows XP with Fast User Switching (FUS) enabled, the first user to log on is assigned to Session 0 and Sessions 1 and 2 represent other users who have logged on to the local system.

In Windows Vista, Windows Server 2008, and later versions of Windows, sessions are assigned as shown in Figure 2.

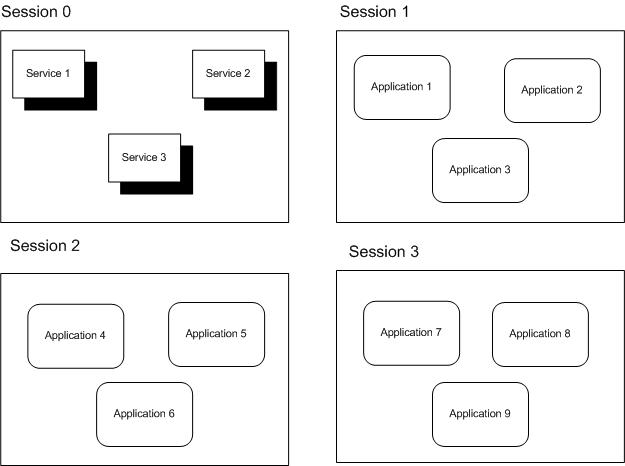


Figure 2. Sessions in Windows Vista, Windows Server 2008, and later versions

In Figure 2, again three users are logged on to the system. However, only services run in Session 0. The first user logs on to Session 1, and Sessions 2 and 3 represent subsequent users.

Because Session 0 is no longer a user session, services that are running in Session 0 do not have access to the video driver. This means that any attempt that a service makes to render graphics fails. Querying the display resolution and color depth in Session 0 reports the correct results for the system up to a maximum of 1920x1200 at 32 bits per pixel.

# Implications for Services and Service-Hosted Drivers

This section describes how these changes can affect applications and drivers.

## What Is Affected

Any applications or drivers that are installed as a service are affected by the following implications. Some drivers are loaded within operating system services or processes that are running in Session 0, and those drivers are also affected by the implications of the Session 0 changes. Specific examples of affected driver classes include:

* Printer drivers, which are loaded by the spooler service.
* All drivers that are authored by using the user-mode driver framework (UMDF) because these drivers are hosted by a process in Session 0.

## Potential Issues

Any functionality in a service or a service-hosted driver that assumes the user is running in Session 0 does not work correctly in Windows Vista, Windows Server 2008, and later versions. Some examples of places where this assumption might occur are:

* A service attempts to create a user interface (UI), such as a dialog box, in Session 0. Because the user is not running in Session 0, he or she never sees the UI and therefore cannot provide the input that the service is looking for. The service appears to stop functioning because it is waiting for a user response that does not occur.

For example, if a device installer runs in Session 0 and the installation program creates a dialog box in Session 0 that requires user input to continue, the device installation never completes because the user does not see the dialog box. From the user’s perspective, the device installer is hung because it has stopped progressing and the user has no way to resume it.

* A service tries to use window message functions such as SendMessage and PostMessage to communicate with an application. This does not work because the application is running in a different session and therefore has a different message queue. The messages never arrive at their destination. The same is true for applications that try to communicate with services through window messages.
* Because services run in Session 0, named objects that they create or open are usually in \BaseNamedObjects\. However, if a user application assumes that it is running in the same session as the service and synchronizes with the service by creating or opening objects with the Local\ prefix (or no prefix, which defaults to Local\), the application no longer works as expected. The reason is that the Local\ prefix indicates that the create or open request is specific to that session and the objects that the application creates or opens are in \Sessions\<n>\BaseNamedObjects instead of \BaseNamedObjects\. The correct way for user applications to synchronize with a service is to explicitly use the Global\ prefix when the application creates or opens objects in \BaseNamedObjects\.

These implications for services are also exposed through FUS in Windows XP because every user on a FUS-enabled machine runs in a different session. Services that assume that the user is running in Session 0 encounter the same issues when the second user logs on to a FUS-enabled machine. However, services that were not fixed to work with FUS encounter problems in Windows Vista, Windows Server 2008, and later versions even if only one user is logged on.

# Guidelines for Services and Service-Hosted Drivers

In Windows Vista, Windows Server 2008, and later versions of Windows, drivers that are hosted within a service should follow these guidelines to work properly:

* Use a client/server mechanism such as remote procedure call (RPC) or named pipes rather than window messages to communicate with applications.
* Implement any necessary user interface for the service as follows:

Use the WTSSendMessage function to create a simple message box on the user’s desktop. This allows the service to give the user a notification and request a simple response.

For more complex interactions, developers should move their UI code into an agent that runs in the user’s session and handles all UI requirements. The agent communicates with the service through RPC or named pipes. If the user initiates the UI interaction by using Control Panel, Internet Explorer, or a similar UI experience, that UI experience should start the agent. The agent then handles all UI interactions. If UI is required but is not initiated by the user, the service must request the agent to start any required UI, instead of attempting to launch that UI by itself. In the rare situation where the service must initiate a user interaction and the agent is not already running, the service should call the **CreateProcessAsUser** API to start the agent. The agent can then initiate all UI interactions. It is important for developers to carefully review all possible usage scenarios and consider moving all UI code into an agent that runs in the user session.

* Query display properties in the user’s session, not in Session 0, because the resolution and color depth that are reported in Session 0 are unlikely to reflect the actual display properties.
* Explicitly choose either the Local\ or Global\ namespace for any named objects, such as events or mapped memory, that the service makes available. If an object must be accessible to user applications, it must be created in the Global\ namespace to be accessible to other sessions. The following Microsoft Win32® functions all accept named objects: OpenEvent, OpenMutex, OpenSemaphore, OpenWaitableTimer, OpenJobObject, and OpenFileMapping. Care should be taken when using these functions to ensure that the named object is accessible within the current session.
* Test the driver in Windows Vista, Windows 2008, or a later version to ensure that it runs properly. If that is not possible, test the driver in Windows XP with FUS enabled and multiple users logged on. If the driver works correctly for second and subsequent logged-on users, it is not likely to be affected by the Session 0 changes. The only issues that this test does not detect are those related to the absence of the video driver in Session 0.

# Interactive Service Detection Service

Windows Vista, Windows Server 2008, and later versions provide the option to enable the Interactive Service Detection Service for customers who have legacy services that send user interaction dialog boxes to Session 0 instead of the corresponding user’s session. This support might be removed from a future Windows release, at which time all applications and drivers must handle Session 0 isolation properly.

Starting with Windows Vista Beta 2, the service is demand start by default and starts only when a visible dialog box that is not a command window is detected. If the service is started, then users are notified when a dialog box or window (including a command window) appears in Session 0. If more information is shown, information about each of the last ten dialog boxes appears in turn if more information is shown. This helps ensure that testers are aware of legacy services in their environment and have the opportunity to contact the vendors for updated services.

The service detects these visible dialog boxes or windows and sends a notification to the user. Users may choose to:

* Respond to the dialog box immediately by clicking a button to switch to Session 0, interact with the task dialog box, and then return to their session.
* Be reminded again in 5 minutes. They continue to be reminded until the dialog box closes.

Sessions on the glass—at the physical system—always receive notification as long as the feature is not disabled. In client SKUs of Windows Vista and later versions, the remote desktop session is notified when the user is remote instead of on the glass. In server SKUs of Windows Server 2008 and later versions, the remote administration sessions are notified if they are in use. When a Microsoft Terminal Services application server role is on the system, only the administrative sessions are notified and regular user sessions are never notified.

After Interactive Services Detection Service is disabled, users no longer receive notifications when the devices or services send dialog boxes to Session 0.

# Resources

#### Platform SDK

Services  
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/services.asp>

Synchronization   
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/synchronization.asp>

Making a Remote Procedure Call   
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/rpc/rpc/making_a_remote_procedure_call.asp>

Client/Server Applications   
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/termserv/termserv/client_server_applications.asp>

CreateProcessAsUser Function  
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dllproc/base/createprocessasuser.asp>

Test Your Application with Fast User Switching  
<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/apcompat/apcompat/test_your_application_with_fast_user_switching.asp>

#### Windows Driver Kit

Associating Services, Driver Packages, and Applications  
<http://msdn.microsoft.com/en-us/library/ms790301.aspx>

INF AddService Directive  
<http://msdn.microsoft.com/en-us/library/ms794559.aspx>