Understanding Sessions in Windows

Author: Conrad Chung, 2BrightSparks

Introduction

In 2006, Microsoft released the Windows Vista operating system and incorporated several new security features. One significant change was called “Session 0 Isolation” and continues to form a part of the services enhancements first introduced in Windows Vista and supported in later versions of Windows. In this article we will focus on what sessions are and how they work.

Windows Services Overview

Windows services are computer applications which usually handle low-level tasks that operate quietly in the background. Services like networking, hardware and remote access are essential for Windows to function properly. In addition, several third-party applications like firewalls and antivirus also run partially or fully as services. For example, developers can use services to create executable applications that:

- Automatically start when the computer starts.
- May be paused or restarted.
- Run regardless whether an interactive user is logged on or not.
- Can operate in the context of a user account that is different from either the default computer or the logged-on user account.

Services are suitable in scenarios where long-running functionality is needed that does not interrupt the day to day workflow of users working on the computer. However, features like these make an attractive target for virus writers as services are usually run in a high-privilege account and usually start when the system boots up, and stop only when the operating system shuts down.

Session 0 Isolation Explained

Each user that logs on to Windows is placed in a separate session. During startup, Session 0 is created and additional sessions are created as needed. Services are always run in Session 0. However, in Windows XP and earlier versions of Windows, Session 0 can also run user applications. With Fast User

Switching enabled in Windows XP, Session 0 is assigned to the first logged-on user together with any applications that the user runs in that session. The second user is assigned to Session 1 and so on.

There are several security issues that arise when running both services and user applications in the same session. For example, a virus could wreak havoc on a user's system if it does not install itself as a service and runs from a high privilege account.

Starting from Windows Vista, Microsoft introduced two important changes in Session 0 to alleviate these issues:

- Session 0 is reserved exclusively for services and other non-interactive user applications. Users who are logged on to Windows and their user applications must run in Session 1 or higher.
- User interfaces in Session 0 are not supported. Processes running in Session 0 have no access to the graphics hardware thus user interfaces cannot be directly displayed on the monitor.

Potential Issues with Session 0 Isolation

With the changes to how sessions interact with services and service-hosted drivers, we will now turn to review potential implications that may affect applications and drivers.

Some drivers will be affected by the Session 0 changes if they are loaded within services or processes running in Session 0. Some driver classes affected include:

- Printer drivers loaded by the spooler service.
- Any drivers authored with the User-Mode Driver Framework (UMDF) because a process in Session 0 is hosting these drivers.

Application classes affected by this feature includes any service or a service-hosted driver that assumes the user is running in Session 0; these will not work correctly in Windows Vista or later versions of Windows. Some affected application classes include:

- Services which create a User Interface (UI) in Session 0. A user will not see a User Interface, such as a dialog box, created in Session 0 as he is not running in it and thus is not able to provide the input the service is looking for. The service will look like it has stopped functioning but in fact it is waiting for a response from the user that does not occur. However, services can use the WTSSendMessage function to addresses this issue. Using this function, the service can create a simple message box on the user's desktop which allows the user to see the notification and provide a simple response.
- A service that tries to communicate with an application by using window message such as SendMessage and PostMessage functions. As the application is running in a different session/message queue from the service the message will never arrive at the destination. This also applies to applications trying to communicate with services through window messages. Microsoft recommends using a client/server mechanism such as remote procedure call (RPC) or named pipes to communicate between services and applications instead.

An example of this change affecting an application within versions of Windows Vista or later would be the backup and synchronization programs SyncBackFree, SyncBackSE and SyncBackPro developed by

2BrightSparks. There is a setting in these programs to automatically close live applications that are still running before commencing the backup job. However, this auto-close setting will not work with scheduled profiles while the user has logged off from the computer. This is because processes running via the scheduler are in Session 0 and will not have access to the desktop/application processes, which are within the user logged-on sessions of 1 or greater. One workaround would be to change the scheduler settings to *run only if the user is logged on*. An issue with this setting enabled is that scheduled jobs will not run when the user is not logged on. Hence, the backup profile will not run when you want it to while you are logged off (for example, an overnight backup).

Summary

By learning the concept of Windows sessions, we better understand how versions of Windows Vista or later use session 0 to segregate services from user applications and its possible impact on applications like SyncbackFree/SE/Pro that occasionally requires circumstantial interaction with Session 0.