White Paper:
Recovering From Service Failures

Rev 2 – June 1, 2006

Lieberman Software Corporation
http://www.liebsoft.com

Abstract

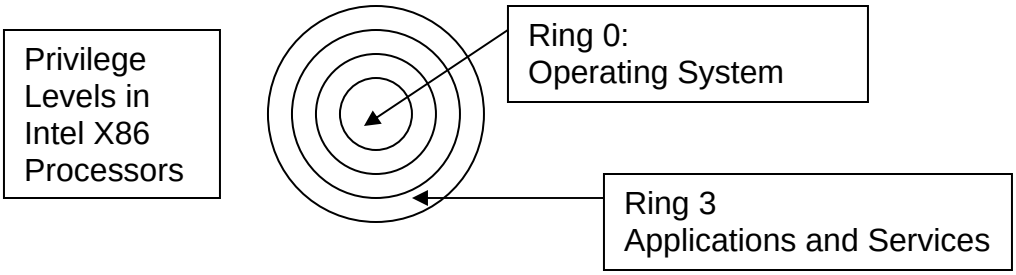
Services fail in a variety of ways. In this paper we will discuss the different types of failures and methods of recovery.

Contents

1. Privilege Levels

Windows NT and above employ hardware protection within the processor (CPU) that allows code to be classified as to privilege level. The operating system and device drivers operate at ring level 0, also known as kernel-level or system-level privilege. At this level, there are no restrictions on where a program can go or do. Consequently, you should always be concerned about the source of device drivers for machines that contain sensitive information.

The only problem at this level is that if the operating system or a device driver makes a memory or access mistake, the operating system halts with a trap screen (the famous blue screen of death, or BSOD).

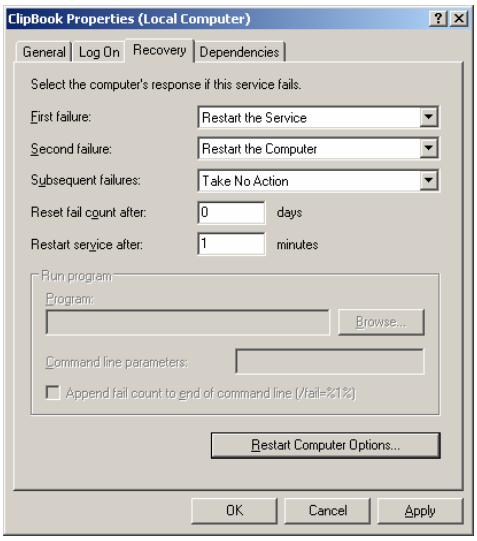


Applications and services operate at ring level 3, also known as user-level or application-level privilege. If an application or service fails at this level, a trap screen will appear (also known as a general protection fault, or GPF), that can be dismissed without the operating system caring.

The decision to have services run at the same privilege level as regular applications is based on the idea that if the service traps (blows up), the operating system should continue to operate without problems.

2. Recovery Tab in Windows 2000

Starting in Windows 2000, an automatic restart of failed services is implemented.
Microsoft added the ability in Windows 2000 to detect and recover from terminated services via the "Recover" tab in the services settings:



As long as a service terminates, and the SCM (Service Control Manager controls all services within the operating system) can detect it, this feature allows the administrator to configure the action to take. The actions include:

- Take No Action
- Restart the Service
- Run a Program
- Restart the Computer

The dialog also tracks the results of the last recovery action and can take different actions for each attempt to recover. In most recovery scenarios, the administrator will want to escalate the severity of action all the way to rebooting the system.

3. A Good Start, but…

But, all of this is of limited use depending on the nature of the failed service as some services do not exit, but stay stuck in a tight non-functional loop. Sometimes this is the result of an external attack known as a denial of service attack. In other cases it can be simply the result of poor developer programming of the service.

Another problem is the inability to gain direct keyboard/monitor/mouse access to a troubled service to see what is wrong. Services operate on a console screen that is inaccessible to everyone including the administrator. Yes, services have a keyboard, screen and mouse, but unfortunately, neither you nor I can access these peripherals since they are "virtual" or "phantom" devices to trick the services into believing that they are running at a normal console. Debuggers can connect to services, but even they cannot gain access to the peripherals the service is using.

If a service is leaking memory there is no way to cause an action to be performed when a service's memory utilization gets completely out of whack.

4. Solving Some of the Problems

From our personal experience we have managed to improve our situation with misbehaved services a couple of ways. By using Lieberman Software's **Service Account Manager™** we have configured a simple job to restart known leaking services regularly throughout the day. This is accomplished by highlighting the known problematic services on all of our systems and scheduling a service change that consists of nothing more than doing a service stop, followed by a service start. This periodic service restart causes the service to return all of its memory back to the operating system and return to normal operation.

Another trick we use is the ability to schedule the reboot of servers at regular intervals. Highlighting the machines that need reboots, then clicking on the schedule button to setup when reboots are to occur can do this.

