

# Conquer Desktop Heap Problems

Michael Morales

(Reprinted From WindowsItPro Magazine)

## Executive Summary

Windows XP, Windows Server 2003, and earlier Windows OSs are prone to problems with desktop heap—memory allocated to Windows session views. Learn how to diagnose and solve desktop heap problems in XP, Windows 2003, and earlier, as well as Windows Vista and Windows Server 2008, using tools such as Desktop Heap Monitor 8.1 (Dheapmon) and registry-setting changes.

As an administrator, you've probably run into a desktop heap problem (e.g., an out of memory error message or failed application startup) and know firsthand how tough this type of problem is to solve. Part of the difficulty lies in first identifying that your symptom is related to an exhaustion of desktop heap memory. Another challenge is discovering which process or service is consuming the greatest amount of desktop heap, then determining what registry parameter to change to work around the problem. This month, I'll show how to quickly identify whether your system is running out of desktop heap. (See the box below for a list of common symptoms of a desktop heap problem.) Then I'll give you some tools and best practices you can use to resolve the problem or progress toward a solution before contacting tech support.

## Possible Symptoms of a Desktop Heap Problem

- Application startup failures (0xc0000142)
- Scheduled tasks fail to launch.
- Processes silently fail to run.
- UI elements fail to redraw properly.
- An event 243 (A desktop heap allocation failed) is logged in the system log.

## Windows Internals Background

Before starting our desktop heap troubleshooting, you'll need some Windows internals background to understand how desktop heap problems occur. Windows 2000 and later systems have a configurable area of kernel mode memory called session space. Session space represents a single user's logon environment; you can also think of session space as each user's sandbox of windows and desktops.

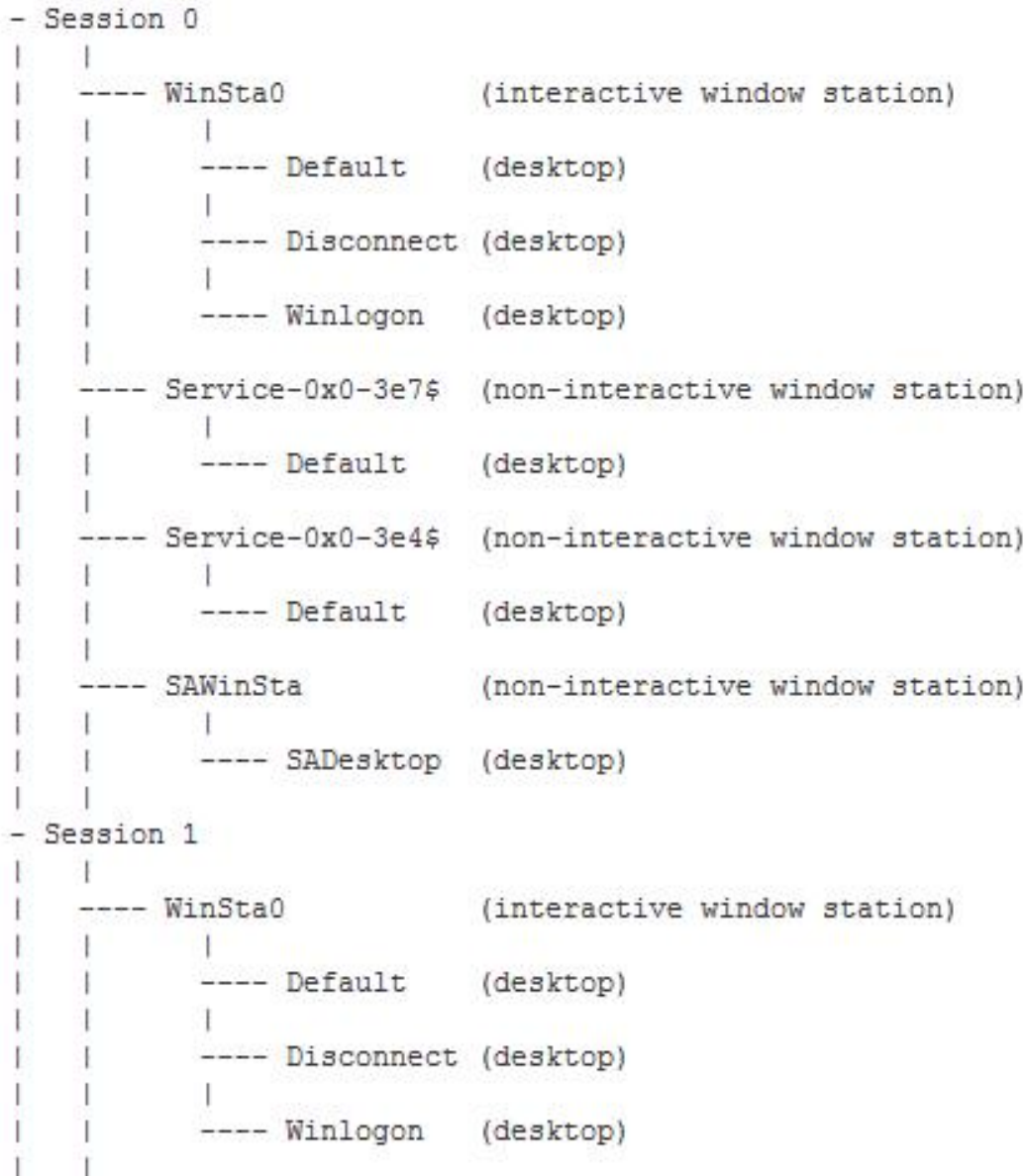
You should also know that each session (e.g., Session 0, Session 1) contains window stations that act as a security boundary for desktops. Although the term "desktop" may bring to mind the interactive desktop that each user sees when logged onto Windows, actually not every desktop interacts with the user. Each window station contains desktop objects, and every session will have one interactive window station named WinSta0 that users will see when they log into their systems.

Another way to conceptualize the desktop tree system is that every Win32 thread belongs to a desktop. Every desktop belongs to a window station; one window station per session interacts with the user, while the rest do not. And every window station belongs to a session. The schematic in Figure 1 depicts the desktop tree on a typical system. In Figure 1, two sessions are listed: Session 0 and Session 1. Session 0 is where services typically run and also represents the console (prior to Windows Vista). Any other session such as Session 1 or Session 2 typically represents a Terminal Services or Fast User Switching session.

# Conquer Desktop Heap Problems

Michael Morales

(Reprinted From WindowsItPro Magazine)



**Figure 1**  
**Desktop Tree Schematic**

## What Is Desktop Heap Exactly?

Every desktop object requires memory to store UI objects, such as windows and menus. This memory is called desktop heap. When applications require a UI object, functions within user32.dll are called, and desktop heap memory is allocated. There is one desktop heap per desktop, and the heap memory itself is allocated from session-view space, which is a subset of session space.

# Conquer Desktop Heap Problems

Michael Morales


(Reprinted From WindowsItPro Magazine)

While this process of allocating desktop heap memory for storing windows and menus works behind the scenes, there are two primary scenarios in which failures can occur. Session-view space can become fully utilized so that no new desktops can be created. This scenario can occur when multiple services run under a nonlocal system-specific user account, creating a new desktop for every instance of the service. In the second scenario, existing desktop heaps can become fully utilized, so that threads running in that desktop can't use more desktop heap memory. This is the more common scenario and can be caused by running many instances of the same process (e.g., several instances of Internet Explorer) or by a single process that has heavy UI object usage.

## Diagnostic Tools

Let's look at some methods that make diagnosing desktop heap exhaustion issues much easier. Desktop Heap Monitor 8.1 (Dheapmon), available at [tinyurl.com/Dheapmon](http://tinyurl.com/Dheapmon), is a handy tool to use for Windows XP or Windows Server 2003 systems (Dheapmon doesn't run on Vista or Windows Server 2008). The tool's output provides a user-friendly menu describing the total number of desktops, sessions, and window stations. The output, which Figure 2 shows, displays how fully utilized each desktop is on a percentage basis.

```
C:\temp\x86>dheapmon
Desktop Heap Information Monitor Tool (Version 8.1.2925.0)
Copyright (c) Microsoft Corporation. All rights reserved.
-----
Session ID:    0 Total Desktop: ( 5312 KB - 7 desktops)
-----
WinStation\Desktop      Heap Size(KB)      Used Rate(%)
-----
WinSta0\Default         3072                27.9
WinSta0\Disconnect      64                  4.5
WinSta0\winlogon        128                  8.9
Service-0x0-3e7$\Default 512                 12.8
Service-0x0-3e4$\Default 512                  4.3
Service-0x0-3e5$\Default 512                  4.3
SAWinSta\SADesktop      512                  0.5
-----
```



**Figure 2**  
**Dheapmon Output**

The most important numbers in the Dheapmon output are those in the Used Rate (%) column, which will help you determine whether any of the desktops being monitored is becoming fully utilized (i.e., 90 percent or more). Another number to watch is the Total Desktop value—the total amount of memory allocated by all desktops. If this number approaches the total size of the session-view space, then Windows can't create any more new desktops in a session. If this is the case, you may need to change a registry value to increase the default session-view size. Table 1 shows the default session-view sizes for Win2K, XP, and Windows 2003.

# Conquer Desktop Heap Problems

Michael Morales

(Reprinted From WindowsItPro Magazine)

OS	Size if no registry value configured	Default registry value
Windows 2000 *	20MB	none
Windows XP	20MB	48MB
Windows Server 2003	20MB	48MB

**Table 1**  
**Default Session-View Sizes**

Before changing registry values, you should try to identify the process(es) consuming large amounts of desktop heap so that you become aware of the conditions on your system causing the depletion of desktop heap memory. One of the easiest ways to identify a large consumer of desktop heap is by using Task Manager: On the Processes Tab, click View, Select Columns, and check USER Objects. Click the top of the column to change the sort order to descending, so you can see the application or service consuming the most desktop heap resources. Identifying the desktop heap hog on your system is important because it may indicate a problem related to the service or application requiring further investigation—and simply adjusting registry settings to work around the problem might only mask the real issue. You can also use the information in Task Manager's USER Objects column to determine which application or service is consuming the largest amount of desktop heap on a Vista or Server 2008 system.

## Session-View Space Registry Settings

For XP, Windows 2003, and Win2K, you can configure session-view space size by using the SessionViewSize registry value (REG\_DWORD). You specify the size in megabytes. Note that on Vista and later systems, this value doesn't apply because the session view space grows as needed. The values listed in Table 1 are specific to 32-bit x86 systems not booted with the /3GB switch. You must reboot your system to effect this change. You specify the value under the subkey HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management.

If you need to change the size of a specific desktop heap—that is, when Used Rate (%) approaches 90%, you have two possible ways to do so, based on whether one of two conditions exist. The first condition occurs when the information from Dheapmon reveals a high used rate for a desktop heap belonging to an interactive window station (WinSta0) and isn't the Disconnect or Winlogon desktop. In this case, you can configure the desktop's heap size using the SharedSection registry value (the second value—3072—for the SharedSection= entry in the registry listing in Figure 3). I'll explain these registry values in more detail shortly.

**Figure 3**  
**Sample Default Data For Registry Value Controlling Desktop Heap Size**

```
%SystemRoot%\system32\csrss.exe ObjectDirectory=\Windows  
SharedSection=1024,3072,512 Windows=On SubSystemType=Windows
```

# Conquer Desktop Heap Problems

Michael Morales

(Reprinted From WindowsItPro Magazine)

```
ServerDll=basesrv,1 ServerDll=winsrv:UserServerDllInitialization,3  
ServerDll=winsrv:ConServerDllInitialization,2 ProfileControl=Off  
MaxRequestThreads=16
```

The second condition occurs when the Dheapmon information reveals a high used rate for a desktop heap belonging to a non-interactive window station. In this case, you can also configure the desktop's heap size using the SharedSection registry value (the third value—512—for the SharedSection= entry in Figure 3). The size of each desktop heap allocation is controlled by the registry subkey HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Control\Session Manager\SubSystems\Windows. The default data for this registry value will look something like that in Figure 3 (it will appear on one line in your system's registry).

As mentioned, the numeric values following SharedSection= control how desktop heap is allocated. These SharedSection values are specified in kilobytes.

The first SharedSection value (1024) is the shared heap size common to all desktops. This memory isn't a desktop heap allocation, and you should not modify this value to address desktop heap problems.

The second SharedSection value (3072) is the size of the desktop heap for each desktop associated with an interactive window station (WinSta0), except for the Disconnect and Winlogon desktops.

The third SharedSection value (512) is the size of the desktop heap for each desktop associated with a non-interactive window station (usually a service). If this value isn't present, the size of the desktop heap for non-interactive window stations will be same as the size specified for interactive window stations (i.e., the second SharedSection value).

## Changes in Vista SP1 and Server 2008

In the 32-bit versions of Vista SP1 and Server 2008, session view space is now a dynamic kernel address range. As I mentioned, the SessionViewSize registry value is no longer used. This is a big improvement and one of the reasons why you might not experience as many desktop heap issues running Vista or Windows Server 2008 as with earlier Windows versions. Additionally, the second value for the SharedSection location has changed to 12,288KB, which is the value for interactive desktop heaps.

The information I've given you here can better equip you to recognize desktop heap problems and resolve such issues on your own. As always I welcome your questions or stories about your own Windows troubleshooting experiences, with desktop heap or other OS issues.

Special thanks to Matthew Justice, a Microsoft software development engineer, who contributed significantly to this article.