Advanced Memory and Storage Considerations for Provisioning Services

www.citrix.com

Contents

Introduction

As Citrix Provisioning Services has begun to play a greater role in XenApp and XenDesktop implementations, many discussions have been had on proper sizing of Provisioning Services from a network, CPU and storage perspective. However, most recommendations have left out guidelines for proper sizing of the one of the most crucial components; memory (RAM). This white paper will detail why RAM not only in the Provisioning Services servers, but RAM within the target devices is one of the most important scaling considerations for a successful Provisioning Services deployment. Additionally, this paper will discuss the impact of RAM on storage considerations and which storage options provide for optimal performance. What this white paper does not cover is storage best practices and IOPS requirements for shared target devices.

Understanding How Windows Handles Memory

The process to properly size the RAM within a server or workstation is often focused on how much committed memory a given server or workstation consumes while actively running applications. Engineers often determine this through the Performance Monitor counter for "Memory/Committed Bytes" or by viewing the Memory graph on the performance tab in Task Manager when a server or workstation is under peak load. There is a fatal flaw in using this method as it ignores the amount of memory that is required by the file cache to ensure optimal performance and limited use of costly disk resources.

To understand this better one must first understand from a high-level how Windows handles memory in general and then how Windows handles caching of file data into memory. In order to streamline this white paper, it is assumed that the reader already understands Windows memory architecture, including the difference between physical memory and virtual memory as well as the difference between various kernel memory components such as System Cache and System PTE. For a more in-depth discussion on Windows memory architecture refer to Appendix A: Windows Memory Architecture.

Windows System Cache

When compared to RAM and CPU, the hard disk within a computer is typically the slowest component within the system. In order to overcome the performance limitations of the hard disk, Windows employs virtual block level caching, a.k.a. file caching. The area of memory that Windows uses for actively mapped and cached files is called System Cache. The System Cache is often interchangeably referred to as the file cache on a Windows OS. If a file is cached in memory and can be recalled or loaded from RAM as opposed to being reread from disk then it is said to reside in the System Cache or file cache. The following diagram illustrates how the Windows file cache works when a user launches an application from the hard disk such as Microsoft Word for the first time.

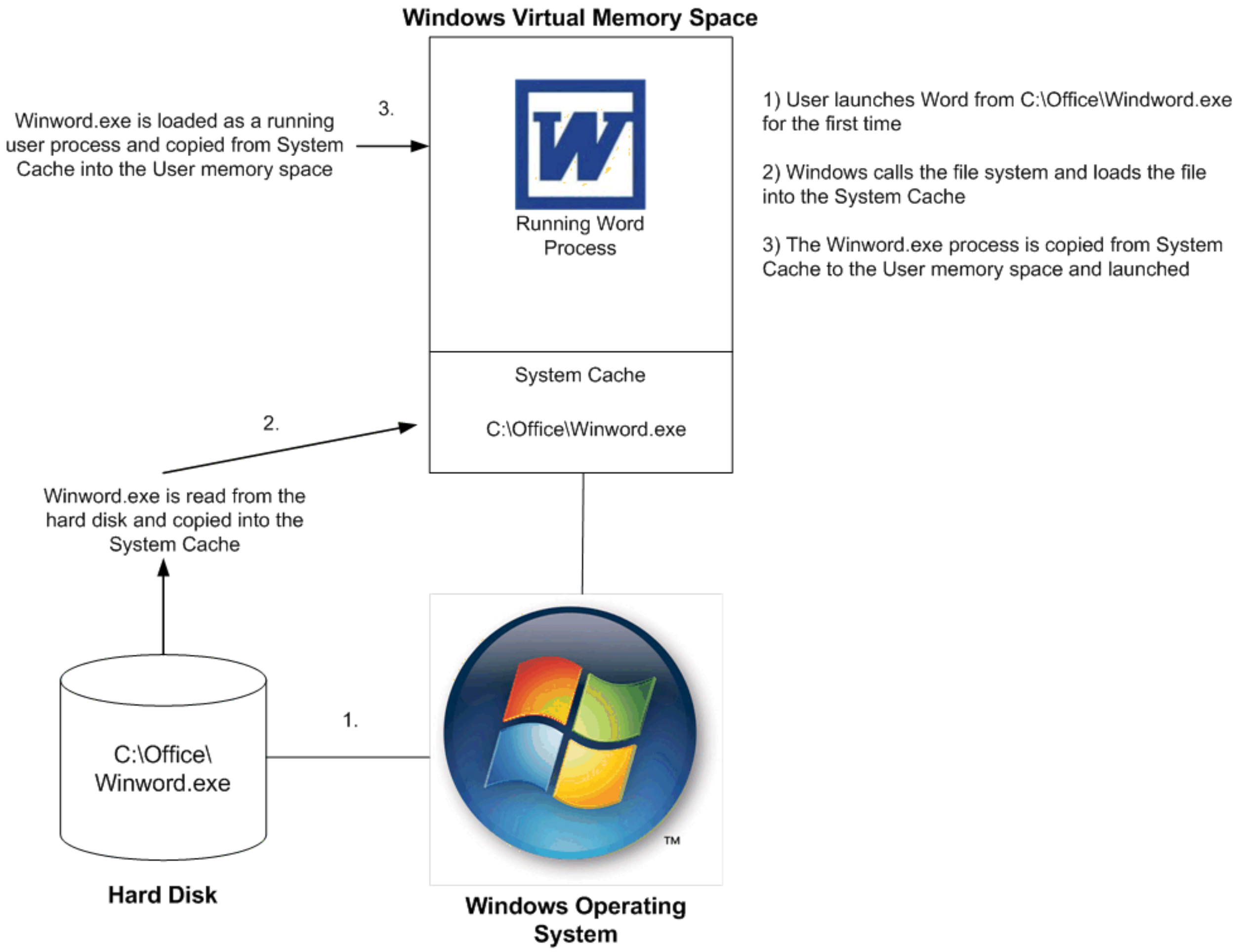

Windows Virtual Memory Space
Winword.exe is loaded as a running user process and copied from System Cache into the User memory space
3.
Running Word Process
1) User launches Word from C:\Office\Windword.exe for the first time
2) Windows calls the file system and loads the file into the System Cache
3) The Winword.exe process is copied from System Cache to the User memory space and launched
System Cache
C:\Office\Winword.exe
2.
Winword.exe is read from the hard disk and copied into the System Cache
1.
C:\Office\ Winword.exe
Hard Disk
Windows Operating System

Figure 1: First Hard Disk Read

As detailed in the diagram above, the first time a file is read from the hard disk it is copied into the System Cache. When the file gets loaded into the user memory space it is called or mapped from the System Cache space. When the user closes the application the virtual block level file data for the file is still loaded in System Cache. As long as there is enough free physical memory, the file will remain in the memory cache for an indefinite period of time. Thus, for subsequent launch or read requests for the file, Windows will load the file from memory instead of reading from the disk. This file caching feature provides a tremendous performance improvement.

The following diagram illustrates how subsequent requests to read a file even after the file has already been closed are served from the System Cache instead of being read from the hard disk.

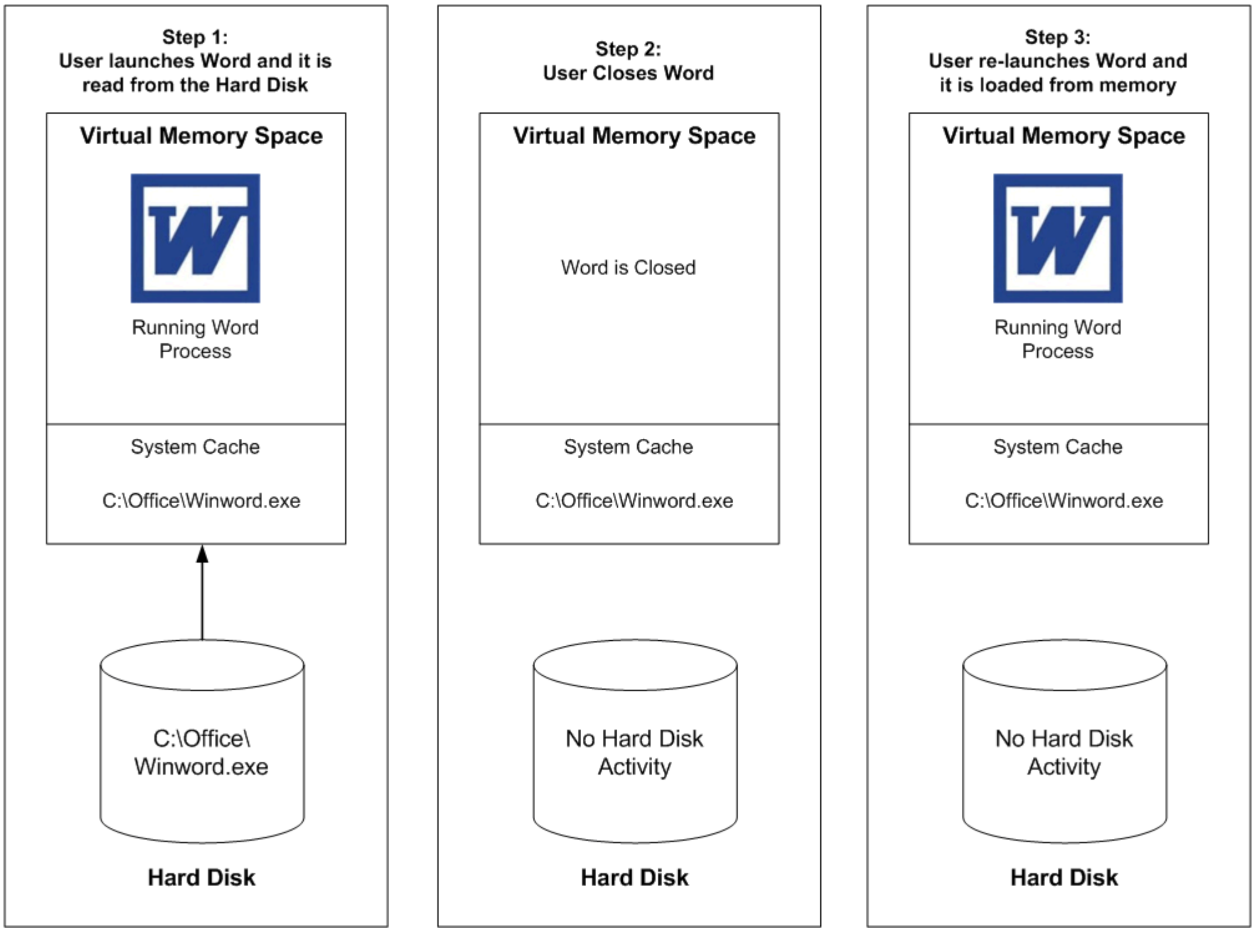


Figure 2: Serving Files from System Cache

On the 32-bit Edition of Windows Server 2003 and Windows XP the maximum size of the virtual address space allocated to the System Cache is 960 MB due to kernel memory limitations as detailed in Appendix A: Windows Memory Architecture. However, this does not mean that only 960 MB of files can be cached in RAM. This limitation only means that 960 MB of virtual memory or data can be actively mapped in the System Cache working set. Additional file cache data can still be located in RAM on Standby or Modified page lists.

The following diagram illustrates how the file data beyond 960 MB can be cached in RAM on a 32-bit edition of Windows.

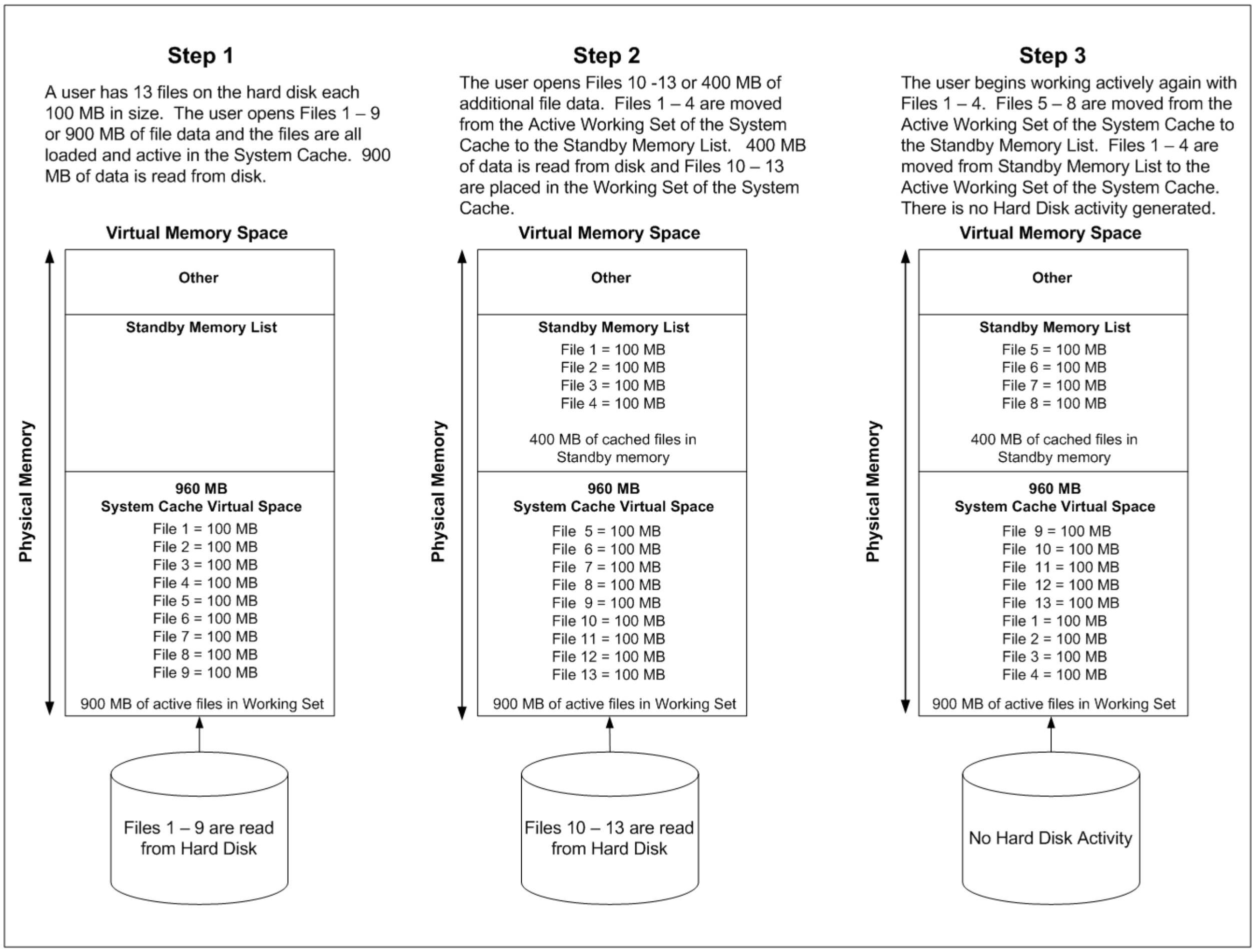
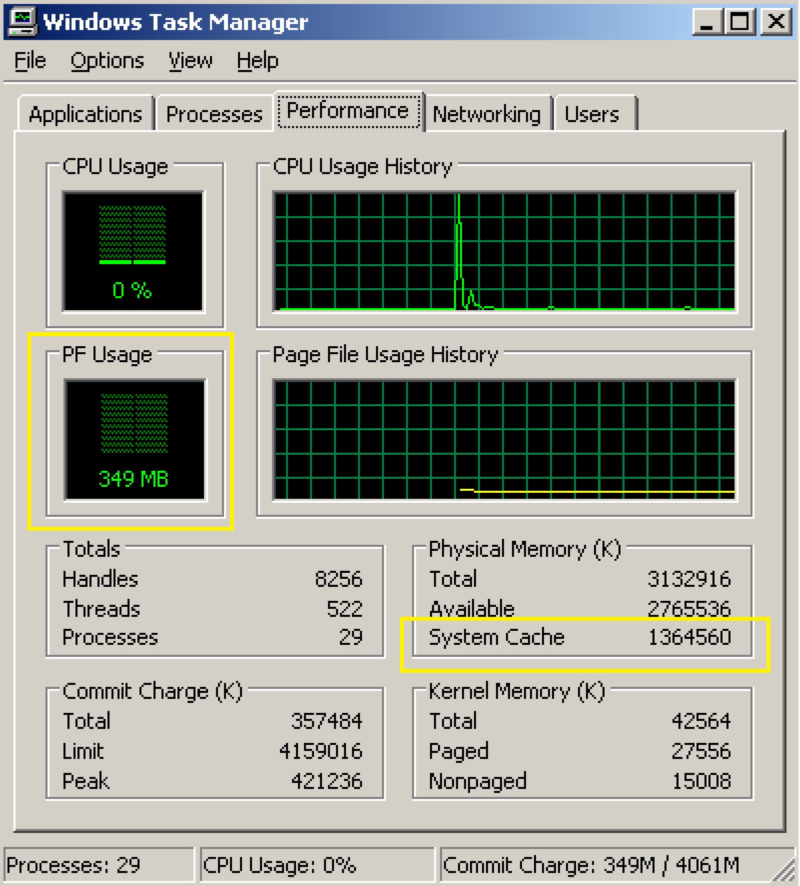

Step 1
A user has 13 files on the hard disk each 100 MB in size. The user opens Files 1 – 9 or 900 MB of file data and the files are all loaded and active in the System Cache. 900 MB of data is read from disk.
Virtual Memory Space
Physical Memory
Other
Standby Memory List
960 MB
System Cache Virtual Space
File 1 = 100 MB
File 2 = 100 MB
File 3 = 100 MB
File 4 = 100 MB
File 5 = 100 MB
File 6 = 100 MB
File 7 = 100 MB
File 8 = 100 MB
File 9 = 100 MB
900 MB of active files in Working Set
Files 1 – 9 are read from Hard Disk
Step 2
The user opens Files 10 -13 or 400 MB of additional file data. Files 1 – 4 are moved from the Active Working Set of the System Cache to the Standby Memory List. 400 MB of data is read from disk and Files 10 – 13 are placed in the Working Set of the System Cache.
Virtual Memory Space
Physical Memory
Other
Standby Memory List
File 1 = 100 MB
File 2 = 100 MB
File 3 = 100 MB
File 4 = 100 MB
400 MB of cached files in Standby memory
960 MB
System Cache Virtual Space
File 5 = 100 MB
File 6 = 100 MB
File 7 = 100 MB
File 8 = 100 MB
File 9 = 100 MB
File 10 = 100 MB
File 11 = 100 MB
File 12 = 100 MB
File 13 = 100 MB
900 MB of active files in Working Set
Files 10 – 13 are read from Hard Disk
Step 3
The user begins working actively again with Files 1 – 4. Files 5 – 8 are moved from the Active Working Set of the System Cache to the Standby Memory List. Files 1 – 4 are moved from Standby Memory List to the Active Working Set of the System Cache. There is no Hard Disk activity generated.
Virtual Memory Space
Physical Memory
Other
Standby Memory List
File 5 = 100 MB
File 6 = 100 MB
File 7 = 100 MB
File 8 = 100 MB
400 MB of cached files in Standby memory
960 MB
System Cache Virtual Space
File 9 = 100 MB
File 10 = 100 MB
File 11 = 100 MB
File 12 = 100 MB
File 13 = 100 MB
File 1 = 100 MB
File 2 = 100 MB
File 3 = 100 MB
File 4 = 100 MB
900 MB of active files in Working Set
No Hard Disk Activity

Figure 3: System Cache in Action

In the diagram above the user has loaded 1300 MB of file data. While the active size of the System Cache on a 32-bit Windows OS is limited to 960 MB, all 1300 MB of file data can still be held in RAM through the use of the Standby Memory list. As long as the file cache data in the Standby List memory is not required for use by other processes, the file cache data will remain in memory. Whenever memory is moved from the Standby List to Active Memory, it is referred to as a soft fault. Whenever memory or file data must be retrieved from disk, it is referred to as a hard fault. Memory on the Standby List does not show as Committed Memory in Performance Monitor or Task Manager.

Let's take a closer look at Task Manager and the System Cache. The following screen shot is from a Windows Server 2003 32-bit server running Citrix Provisioning Services 5.2. The server has 3 GB of physical RAM.



Totals	
Handles	8256
Threads	522
Processes	29

Physical Memory (K)	
Total	3132916
Available	2765536
System Cache	1364560

Commit Charge (K)	
Total	357484
Limit	4159016
Peak	421236

Kernel Memory (K)	
Total	42564
Paged	27556
Nonpaged	15008

Processes: 29 | CPU Usage: 0% | Commit Charge: 349M / 4061M

Figure 4: Task Manager

Take note of the two highlighted values on the Task Manager screen shot above. The PF Usage value of 349 MB is the total amount of committed bytes or committed memory on the system. This is the same as the Performance Monitor counter of Memory/Committed Bytes. When viewing the System Cache value, it is listed as 1,364,560 bytes or 1.3 GB. This indicates that 1.3 GB worth of file data is currently cached in RAM. The System Cache is using 1.3 GB of physical memory, yet the system shows only 350 MB as committed. That is because the System Cache or file cache memory is flagged as Standby List memory or uncommitted memory. This memory could be reallocated to running processes without issue since the file cache contents could always be reloaded from disk.

Windows provides a performance monitor counter that can be used to track how effective the System Cache is functioning. The counter is Cache\Copy Read Hits %. This tracks the percentage of read operations that are serviced from the System Cache instead of being read from disk.

Sizing Memory for Provisioning Services

Now that that we understand how Windows handles memory and file caching, let's review how this affects the Citrix Provisioning Services. The following diagram illustrates the boot process for Windows XP target devices from a shared Provisioning Services vDisk. While every configuration is different, for the following example it is assumed that 200 MB of data must be read from disk by Windows XP during the boot and service startup processes.

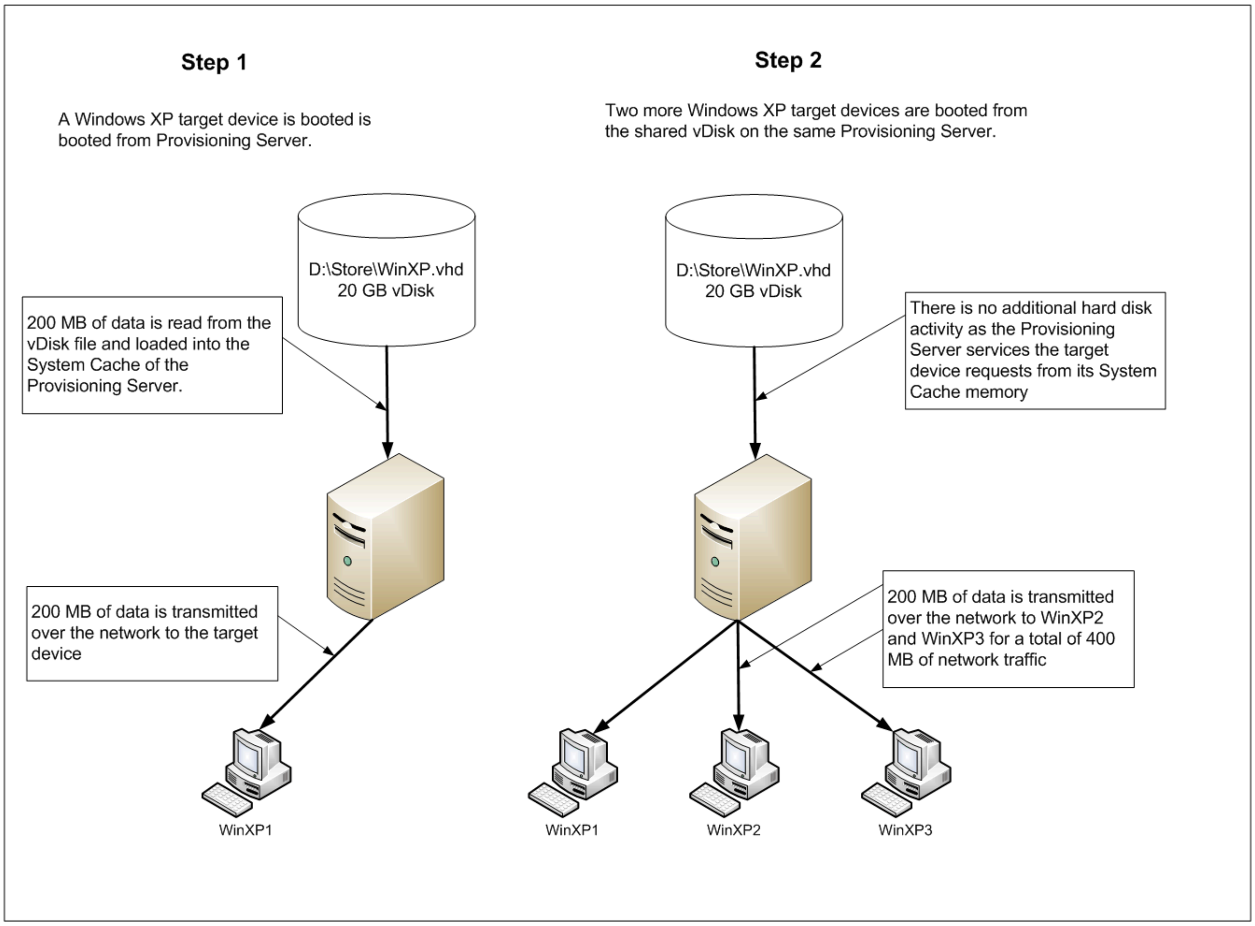

Step 1
A Windows XP target device is booted is booted from Provisioning Server.
D:\Store\WinXP.vhd 20 GB vDisk
200 MB of data is read from the vDisk file and loaded into the System Cache of the Provisioning Server.
200 MB of data is transmitted over the network to the target device
WinXP1
Step 2
Two more Windows XP target devices are booted from the shared vDisk on the same Provisioning Server.
D:\Store\WinXP.vhd 20 GB vDisk
There is no additional hard disk activity as the Provisioning Server services the target device requests from its System Cache memory
200 MB of data is transmitted over the network to WinXP2 and WinXP3 for a total of 400 MB of network traffic
WinXP1
WinXP2
WinXP3

Figure 5: Provisioning Services and System Cache

As shown in the diagram above, the server will only need to read from the physical disk one time for the first target device. Then, it will be able to cache the vDisk contents in System Cache memory and service all additional target devices from RAM as opposed to disk. As long as there is enough memory in the server, when using a shared vDisk model (as is typically used with XenApp and XenDesktop), there are few read IOPS that get serviced by the underlying physical disk subsystem of the Provisioning Services server.

Before one calculates the proper amount of RAM to allocate to the System Cache for vDisk read requests, one must first identify how much data is typically read from the shared vDisk by the target devices. For example, consider a Windows XP image that resides on a 20 GB vDisk with 12 GB of space consumed by files within the vDisk. One might assume that 12 GB of RAM should be allocated to potential System Cache use so that the entire contents of the vDisk could theoretically reside in RAM. However, this would be an incorrect assumption as rarely, if ever, will a Windows operating system actually read and load every single one of its files. Most files on the operating system will be read rarely, if at all. While every use case is unique, a typical Windows XP session will read 1 – 3 GB of files from the hard disk. A typical session could involve the following:

- Windows XP boots to logon screen and starts all services (200 MB)
- A user logs on and loads their profile and sets up their desktop environment (50 MB)
- User runs Office and other applications throughout a typical work day (600 GB)
- User logs off and/or shuts down Windows XP (150 MB)

Given the above workflow, approximately 1.0 GB of data would be read from the vDisk. Since this is a shared vDisk, many of the same files would be read regardless of the user. However, since all users may run different applications, it would be safe to assume that no more than 2 GB of unique file data would typically be read by the various users of the shared vDisk. Given this type of workflow, 2 GB would be the amount of System Cache RAM that the Provisioning server would require in order to keep the contents of the vDisk in RAM and eliminate the need for the Provisioning Server to read from physical disk to service target device read requests.

In addition to the amount of extra RAM required to support caching of the vDisks in System Cache, the System Cache should also be provided enough RAM for caching of the core files used by the Windows OS and services as well as the software running on the server such as the services of Provisioning Services. Typically, 512 MB should be considered an effective amount of memory for base caching purposes. In order to calculate the RAM required for proper sizing of Provisioning Services, the following formula can be used to determine the amount of extra RAM that should be in the server to support the overall System Cache:

$$\boldsymbol{System\ Cache\ RAM\ =\ 512\ MB\ +\ \#\ active\ vDisks\ *\ Avg.data\ read\ from\ vDisk}$$

The amount of RAM required by the System Cache for vDisk usage should be added to the total amount of committed bytes that the server shows under load. The formula for the total amount of RAM required for the server would be:

$$\boldsymbol{Total\ RAM\ =\ Committed\ Bytes\ under\ load\ +\ System\ Cache\ RAM}$$

For example, assume the following configuration:

- A running Provisioning Services server under load showing 2 GB of committed memory

- 4 unique vDisk files hosting Windows XP for various target devices
- 2 GB of data read on average from each vDisk
- System Cache RAM = 512 MB + (4 * 2 GB) = 8.5 GB
- Total RAM = 2 GB + 8.5 GB = 10.5 GB

Using the previous scenario, in order to properly size the server to minimize physical disk usage, at least 10.5 GB of RAM should be allocated to the Provisioning Services server.

Given how System Cache and physical memory are of critical importance to the performance of Provisioning Services, it is highly recommended that an x64 edition of Windows be used for the Provisioning Services operating system.

Calculating IOPS for Provisioning Services Servers

One of the key items of concern in a Provisioning Services deployment, especially for large XenDesktop implementations, is the IOPS required for servers and target devices. As discussed in many best practice documents, the write cache for target devices should typically be located on a storage device seen as local to the target rather than hosting it on the Provisioning Services server. When implementing Provisioning Services using the shared vDisk model where the write cache is not stored on server, the IOPS required by Provisioning Services server are negligible so long as the proper amount of memory has been allocated to the server as discussed in the previous section of this document.

As long as the server has enough RAM to cache the frequently requested blocks of data from the vDisks, the server will only generate IOPS one time for each vDisk data block. For example, consider the following scenario:

- A Windows 2003 x64 server with 16 GB RAM running Provisioning Services
- The server under load shows 2 GB of committed bytes
- The server hosts two unique 30 GB Windows 7 shared vDisks
- On average 3 GB of data is actually read from each of the shared vDisks by target devices
- Each day the server hosts 1000+ Windows 7 targets on the shared vDisks
- The server is rebooted once per month
- How many IOPS are required by the server to host the shared vDisks?

Using the previous formulas, the total RAM required for the server would be:

$$2GB\ Committed\ Bytes\ +\ (512\ MB\ +\ 6GB)\ System\ Cache\ =\ 8.5\ GB\ Total\ RAM$$

Given the previous scenario, on the first day after it is rebooted the server will need to read 6 GB of data from the two vDisks in order to service the target devices (3 GB per vDisk). Once this data is read one time during the first day of use, for the remaining days prior to reboot, the server will generate almost zero IOPS for vDisk activity. This is due to the fact that the server has enough memory to read the vDisk contents one time and then serve all subsequent target device requests from System Cache. Thus, SAN or local disk IOPS for Provisioning Services server hosting shared vDisks are of little concern so long as the server has sufficient RAM. Deciding whether to host the vDisk store on local storage or attached to a SAN is determined more by operations and cost and not by performance.

Storage Considerations for the vDisk Store of Provisioning Services

There are two primary options for hosting the vDisk store for Provisioning Services.

- **Block Level Storage.** This option includes any storage that is seen as a block level device by the Provisioning Server. Examples include SATA, SCSI, iSCSI, and Fibre Channel.

- **Network Storage.** This storage option involves the use of a network redirector on Provisioning Server to access shared storage. Examples include CIFS, NFS and Netware.

Provisioning Services should always host shared vDisks on block level storage devices. When multiple Provisioning Services servers uses a network redirector such as CIFS to access the vDisk store, they not able to take advantage of the System Cache for caching the contents of the vDisk. This creates a significant network burden that will double the total network traffic and reduce scalability. For example, if a target device boots and reads 200 MB of data from the vDisk, this would be a 200 MB network transfer from server to the target device. However, if the vDisk was on non-block level network attached storage to the server, the server would first have to read 200 MB from the network storage and then transmit 200 MB to the target device for a total of 400 MB of network traffic. Since Provisioning Services cannot cache the vDisk contents hosted on network storage, the server will have to download 200 MB from the network storage for every target device that connects to it, thus doubling all network operations.

It should be noted that the network redirector for protocols such as CIFS can make use of and benefit from the System Cache for certain file operations. However, the System Cache does not cache CIFS reads performed on vDisks by Provisioning Services. There are new file locking and caching methods that have been introduced in Windows 2008 with SMB 2.0 that may address this issue in a future release of Provisioning Services; however, at this time all editions of Provisioning Services 5.6 and earlier will not cache vDisk content from a CIFS share when more than one server attaches to the vDisk store.

Sizing Memory on Target Devices

Assigning the proper amount of memory to a target device is just as important as sizing the memory correctly on the Provisioning Services server. From an architectural perspective, the same memory and System Cache functions discussed in the previous sections of this document also apply to target devices.

Since the target device is conducting read requests and its hard drive is served from the Provisioning Services server, the read IOPS are streaming over the network from the server. If the operating system on the target device did not cache these read requests, the network traffic generated would be overwhelming. However, as long as the target device has enough free RAM to support caching of the files in System Cache, the files are only read or transferred over the network one time until the target OS is rebooted or runs low on RAM. This intelligent caching of content by the Windows operating system is what allows Provisioning Services server to serve a shared hard disk over the network without generating an unsupportable amount of network traffic.

For shared target devices, one is also concerned with IOPS generated by write operations. As a best practice, these write operations are typically redirected to a local storage device assigned to the target device. It is important to note that write operations are cached in the System Cache in same manner as read operations. For example, if a 10 MB temporary file is written to the hard drive, there is 10MB of IO that must be passed to the disk subsystem in order to write the file to disk. However, when the contents of the file are to be recalled or reread by an application, the contents will be served from the System Cache so long as there is sufficient RAM available. This means there will be no read IOPS generated to the disk subsystem.

While every use case is unique, the formula below provides a good starting baseline for calculating the RAM that should be assigned to a shared target device. Without conducting a detailed analysis of the number of files that are typically read by the OS and the applications launched by the user from a shared vDisk, allocating 256 - 512 MB of RAM for System Cache use is a safe baseline.

$$\boldsymbol{Target\ Device\ RAM\ =\ Maximum\ Committed\ Bytes\ +\ System\ Cache\ RAM}$$

For example, consider the following shared target device:

- Windows 7 hosting Office Productivity Applications
- Maximum committed bytes for the typical user shows 1.0 GB
- 1.0 GB + 512 MB = 1.5 GB

In the previous example, assuming that 512 MB is an adequate amount for the System Cache, the target device should be allocated a minimum of 1.5 GB RAM in order to run the applications and have enough RAM to cache hard disk data in the System Cache.

Conclusion

Memory and storage are critical components that need to be taken into consideration when designing a Citrix Provisioning Services infrastructure that is used to host shared vDisks for target devices. As long as the proper amount of RAM has been allocated to both the Provisioning Services server and target devices, read based IOPS from the vDisk are of little concern. In summary, here are a few best practices to keep in mind:

Provisioning Services Considerations

- Use an x64 Edition of Windows.
- Allocate the proper amount of RAM based upon the formulas discussed in this paper. For most Provisioning Services servers this will be 8 - 32 GB RAM.
- Use block level storage for the Provisioning Services vDisk Stores (Do NOT use CIFS).

Target Device Considerations

- Allocate the proper amount of RAM to target devices based upon the formulas discussed in this paper. For a typical Windows desktop operating system this will be total committed bytes + 256 - 512 MB for system cache.

Appendix A: Windows Memory Architecture

The first thing one must understand is the difference between the virtual memory addressable by the OS and the physical amount of RAM installed in a system. A 32-bit Windows Operating System can directly address 4 GB of virtual memory regardless of the amount of physical RAM. The Enterprise and Datacenter Editions of Windows Server 2003 32-bit are capable of addressing up 32 and 64 GB RAM respectively, due to the fact that most 32-bit Processors since the Pentium Pro line are actually capable of 36-bit addressing with the use of Physical Address Extension (PAE). PAE allows the CPU and Operating system to make use of physical RAM beyond 4GB; however, from a kernel and process perspective, the core kernel of 32-bit Windows still operates in a 32-bit or 4 GB virtual address space.

The following diagram illustrates how the 32-bit Edition of Windows Server 2003 splits the 4 GB of addressable virtual memory into two distinct halves: Kernel and User space.

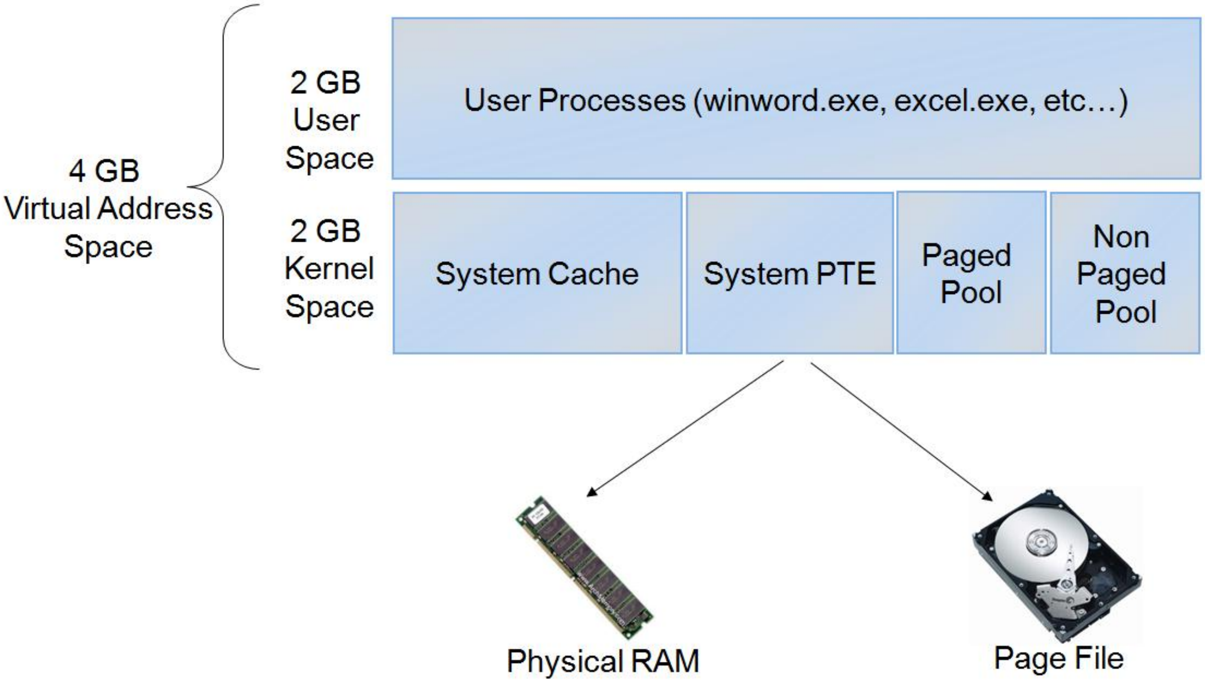


Figure 6: Windows Virtual Memory Space

As illustrated by the diagram above, the kernel memory of Windows is given 2 GB of address space and is broken into four key areas:

- System Cache: This is the area of virtual memory where files are mapped in and out of memory. The active caching of open and in use files occurs in this address space. The maximum amount of virtual address space that can be allocated to System cache is 960 MB.

- System Page Table Entries (PTE): This address space serves as the index that keeps track of the physical location where all virtual memory is mapped. For example, if Winword.exe is running and some of its virtual memory is located in physical RAM and some in the page file on disk, it is System PTE records that detail which part of the processes memory is stored where. The maximum amount of virtual memory that can be allocated to System PTE is 1.3 GB.

- Paged Pool: This address space contains kernel level system an application process virtual memory that may be paged to disk. The maximum amount of virtual memory that can be allocated to Paged Pool is 650 MB.

- Non Paged Pool: This address space contains kernel level process and driver information that must always reside in physical RAM and cannot be paged to disk. The maximum amount of virtual memory that can be allocated to Non Paged Pool is 256 MB.

There are a few other small kernel memory divisions such as Session View, Session Pool and Hyperspace, which are beyond the scope of this document. For more information on these kernel memory areas refer to Microsoft articles 840342 and 294418.

http://support.microsoft.com/kb/840342

http://support.microsoft.com/kb/294418

The sum of all kernel virtual memory allocations must not exceed the 2 GB of virtual address space allocated to the kernel. Thus, each kernel area cannot be simultaneously allocated its respective maximum amount as that would exceed the 2 GB total. These kernel allocations are set at system boot and are not modifiable until restart. Thus, if System PTE runs low, but Paged Pool has plenty of free memory, the Paged Pool size cannot be reduced to give more allocation to System PTE. This limitation has been removed to some degree in Windows Server 2008.

For x64 Editions of Windows, the Windows memory architecture is similar in that it still splits user and kernel memory space; however the limits are much higher than on a 32-bit OS. The chart below details the kernel memory limits of Windows Server 2008 x64.

Physical Memory	Kernel Memory	System Cache	System PTE	Paged Pool	Non Paged Pool
2 TB	8 TB	1 TB	128 GB	128 GB	128 GB

Revision History

Revision	Change Description	Updated By	Date
0.1	Initial Draft Creation	Dan Allen	3/29/2010
0.2	Editorial Revisions	Kevin Bacon, Dan Feller, Nick Rintalan	4/21/2010
0.3	Final Revisions	Dan Allen	5/10/2010
0.4	Minor Updates	Dan Allen	5/13/2010

Citrix Consulting - Americas
Citrix Systems, Inc.
851 W Cypress Creek Road
Fort Lauderdale, FL 33309 USA
Phone: 1-800-393-1888 Option 55
Phone: 1-954-229-6170

Citrix Consulting - Pacific
Citrix Systems Singapore Pte Ltd

8, Temasek Boulevard
#34-02, Suntec Tower 3
Singapore, 038988
Singapore

Citrix Consulting - Europe
Citrix Systems UK, Ltd.
Chalfont House
Chalfont Park, Gerrards Cross
Bucks SL9 0BG
Phone: +44 1753-276-200

Citrix Systems Japan KK

Kasumigaseki Common Gate West Tower 24F
3-2-1, Kasumigaseki, Chiyoda-ku
Tokyo, 100-0013 Japan

About Citrix

Citrix Systems, Inc. (NASDAQ:CTXS) is the leading provider of virtualization, networking and software as a service technologies for more than 230,000 organizations worldwide. Its Citrix Delivery Center, Citrix Cloud Center (C3) and Citrix Online Services product families radically simplify computing for millions of users, delivering applications as an on-demand service to any user, in any location on any device. Citrix customers include the world's largest Internet companies, 99 percent of Fortune Global 500 enterprises, and hundreds of thousands of small businesses and prosumers worldwide. Citrix partners with over 10,000 companies worldwide in more than 100 countries. Founded in 1989, annual revenue in 2008 was $1.6 billion.