NTFS Chkdsk Best Practices and Performance

February 18, 2011

Abstract

This paper provides best practices and guidance when sizing volumes and Chkdsk execution times. This helps design a manageable system using Microsoft® Windows® operating systems. It also provides guidelines to reduce downtime due to Chkdsk execution time in the event of volume corruption.

This information applies to the following operating systems:

* Windows® Server 2008 R2
* Windows Server 2008
* Windows°7
* Windows°Vista®

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

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.

© 2011 Microsoft Corporation. All rights reserved.

Microsoft, Windows, Windows Server, Windows Vista, and Windows XP 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.

Contents

[Introduction 3](#_Toc284709993)

[Self-Healing and Chkdsk 3](#_Toc284709994)

[How to run Chkdsk 4](#_Toc284709995)

[Chkdsk Exit Codes 8](#_Toc284709996)

[Improving General Availability of the Server 9](#_Toc284709997)

[Chkdsk performance on Windows Server 2008 R2 10](#_Toc284709998)

[Block Caching Improvements in Windows Server 2008 R2 10](#_Toc284709999)

[Effect of Volume Size on execution time of Chkdsk 10](#_Toc284710000)

[Effect of Number of files and Different OS versions on execution time of Chkdsk 11](#_Toc284710001)

[Effect of Physical Memory at different Number of files on execution time 12](#_Toc284710002)

[Effect of short file names on Chkdsk execution time 13](#_Toc284710003)

[Effect of Enabling/Disabling short file names 14](#_Toc284710004)

[Conclusion 15](#_Toc284710005)

[Call to Action 16](#_Toc284710006)

[Resources 16](#_Toc284710007)

# Introduction

NTFS is a journaling file system that uses database-like logging techniques in order to provide high-availability. Journaling file systems commit metadata changes to disk in transactions. In the event of a power failure or system crash, NTFS rolls back the uncommitted transactions and returns the file system to a consistent state. However, even journaling file systems require tools to scan the entire file system for structural errors and repair them. Windows provides an inbox file system check utility called Chkdsk that checks the file system and fixes any problems it discovers. This program can be invoked from the command prompt by running Chkdsk.exe, from the Windows Explorer under **Volume Properties**, or during system boot up as part of the Autochk process.

NTFS will first attempt to fix corruptions online using a mechanism called self-healing; when it cannot fix an issue – the volume will be marked dirty hence recommending the admin/user to run Chkdsk. By default, Autochk (Chkdsk during system boot) will run only when the volume has been marked dirty by NTFS. NTFS will mark the volume dirty if it encounters corruptions during normal operations, or if it encounters corruption when attempting to mount the volume during system boot.

It is recommended to first query the dirty bit for the volume before running Chkdsk from the command line or via the UI, this can be done using the *Fsutil* functionality provided by NTFS as shown in Figure a.

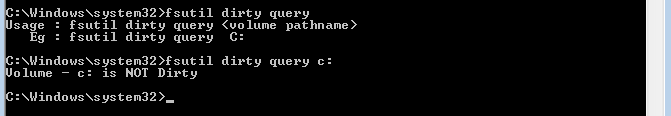


Figure : Query Dirty Bit from Fsutil in NTFS

Most of the Chkdsk execution time is spent scanning file system structures in the volume (there is a direct correlation between the number of files and the Chkdsk execution time). In Windows Vista, Chkdsk performance was improved. Chkdsk caches larger blocks of the file metadata to reduce seek time when validating cross-relationships of file system metadata (there is also a correlation between the memory size and the chkdsk execution time).

# Self-Healing and Chkdsk

The NTFS self-healing feature was introduced in Windows Vista and in Windows Server 2008 to reduce the need to run Chkdsk. Self-healing is a feature built into NTFS that attempts to fix certain corruptions encountered during normal operation. This functionality has significantly reduced the frequency of NTFS marking the volume dirty. This means that autochk will not run Chkdsk on the volume during system startup. Self-healing can fix corruptions while the volume is still online, so it does not reduce storage system availability.

Self-healing can be turned off using the *Fsutil* functionality from a command prompt, as shown in Figure b. Microsoft does not recommend turning off self-healing since this increases the overall occurrence of Chkdsk runs.

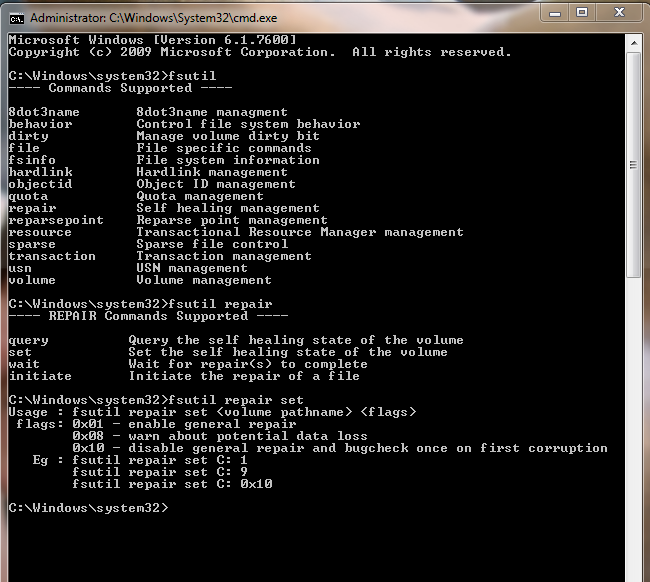


Figure : Turn Off/On Self-Healing using Fsutil

# How to run Chkdsk

You can run Chkdsk when you right-click on any of the volumes in Computer. To do this, click **Properties**, click **Tools**, and then click **Check now**, as shown in Figure c and Figure d.

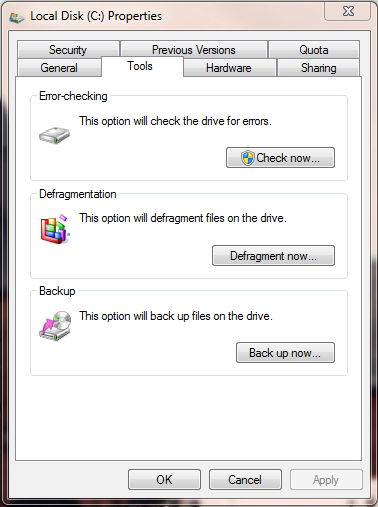


Figure : Disk Properties UI

The user is presented with two options:

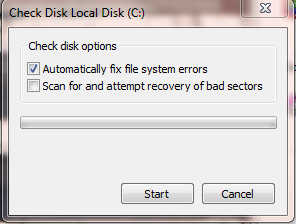


Figure : Chkdsk UI

1. Automatically fix file system errors – Chkdsk option /f (explained below)
2. Scan for and attempt recovery of bad sector – Chkdsk option /r (explained below)

Note A: Microsoft does not recommend running Chkdsk /r unless absolutely needed or there are indications that the disk(s) maybe failing. The Chkdsk /r option will significantly increase the execution time.

Chkdsk needs to perform a complete scan to discover and fix all issues. In addition, Chkdsk has several command line switches, some of which may greatly affect Chkdsk execution time:

Chkdsk [*volume*[[*path*]*filename*]]] [/F] [/V] [/R] [/X] [/I] [/C] [/L[:*size*]] [/B]

|  |  |  |
| --- | --- | --- |
| **Switch** | | **Description** |
| [volume] | | Specifies the drive letter (followed by a colon), mount point, or volume name. |
| [filename] | | FAT/FAT32 only: Specifies the files to check for fragmentation. |
| /F | | Fixes errors on the disk. Does not scan for bad sectors. |
| /V | | On FAT/FAT32: Displays the full path and name of every file on the disk. On NTFS: Displays cleanup messages if any. |
| /R | | Locates bad sectors and recovers readable information (implies /F). (see Note A above) |
| /X | | NTFS only. Forces the volume to dismount first if necessary. |
| /I | | NTFS only: Performs a less vigorous check of index entries. /I will run under any memory conditions. |
| /C | | NTFS only: Skips checking of cycles within the folder structure. /C is turned off if there is not enough memory in the system. |
| /L[:size] | | NTFS only: Changes the log file size to the specified number of kilobytes. |
| /B | NTFS only: Re-evaluates bad clusters on the volume (implies /R and /F consequently) | |

Chkdsk can run in 3 different modes:

1. **Read-only** – Chkdsk runs in read-only mode if none of the following options are specified:
   * /F
   * /R ( see note A above)
   * /X
   * /B
   * /L

In read-only mode, Chkdsk does not make any changes to the volume, and the volume is still available to other applications. By default, this is done by taking a non-persistent snapshot of the volume and running Chkdsk against the snapshot. If a snapshot cannot be created, Chkdsk runs against the original volume, but may report false positives (corruptions that don’t actually exist) due to the transient nature of a live volume. Chkdsk warns the user if this is the case. Chkdsk runs phases 1-3 in this mode (see the phase descriptions below). Chkdsk may also report that it is repairing errors when running in read only mode, but these changes are never written to disk.

**To run Chkdsk in read-only mode, follow these steps:**

1. Click **Start**, click **Run**, type **cmd.exe** in the Open box, and then click **OK**.
2. At the command prompt, type **Chkdsk [volume letter] :**   
   (for example, “**chkdsk c :**”),and then press **Enter**.

This option is used to determine if the volume has any corruptions without making the volume inaccessible.

Chkdsk execution time in this mode is related to the number of files on the volume, not the volume size.

1. **Fix-mode** – Chkdsk runs in fix mode if the /F or the /X option is specified and the /R and /B options are not specified. In this mode, Chkdsk locks the volume to prevent other applications from accessing it. Chkdsk determines if the volume has any corruptions and then fixes them. In this mode, Chkdsk runs phases 1-3 (see the phase descriptions below).

Chkdsk execution time in this mode is related to the number of files on the volume, not the volume size.

1. **Bad cluster recovery mode** – Chkdsk runs in this mode if the /R or /B option is specified. This mode is actually a superset of fix mode. Chkdsk performs the same actions as in fix mode above, but it also runs two additional phases, phases 4 and 5. In these additional phases, Chkdsk scans every cluster on the volume to determine whether it is bad, indicating that it cannot be read from disk due to a hardware error. It places all bad clusters that it finds on the NTFS bad cluster list. NTFS does not allocate these clusters to any file on the list.

Running Chkdsk in this mode is not recommended because modern hard drives automatically replace bad sectors. Modern hard drives also have better failure reporting than when Chkdsk was first developed and implemented. Running Chkdsk in this mode may take a significant amount of time for large volumes because it involves reading every single sector on the volume.

Chkdsk execution time in this mode is related to the number of files as well as volume size.

Note B: Running Chkdsk /r is not recommended unless absolutely necessary, or if there are indications that the disk(s) maybe failing.

There are 3 or 5 phases to a complete Chkdsk run:

Phase 1: Checking files – Chkdsk examine the volume’s master file table (MFT) and checks for consistency.

Phase 2: Checking indexes (directories) – Chkdsk checks each directory for consistency. It also verifies that the files represented in the directory match the files described in the MFT. In addition, it verifies that there are no cycles in the directory tree.

Phase 3: Checking Security Descriptors - Chkdsk checks the NTFS security descriptor stream and related data structures, verifies the security descriptor for each file, and cleans-up unused security descriptors. This phase is run only if the Chkdsk/f option is specified.

Phase 4: Verify File Data (only run when /R or /B is specified) – Chkdsk reads the data of every file to see if there are any bad clusters in the file data. Any file with bad clusters is repaired and all bad clusters are added to the NTFS bad cluster list.

Phase 5: Verify Free Space (only run when /R or /B is specified) – Chkdsk verifies every free cluster on the volume to see if there are any bad clusters. All bad clusters are added to the NTFS bad cluster list.

## Chkdsk Exit Codes

The following table lists the exit codes that **Chkdsk** returns when it exits:

| **Exit Code** | **Description** |
| --- | --- |
| 0 | No errors were found. |
| 1 | Errors were found and fixed. |
| 2 | Disk cleanup, such as garbage collection, was performed, or cleanup was not performed because **/f** was not specified. (Windows XP and earlier) |
| 3 | Could not check the disk, errors could not be fixed, or errors were not fixed because **/f** was not specified. |

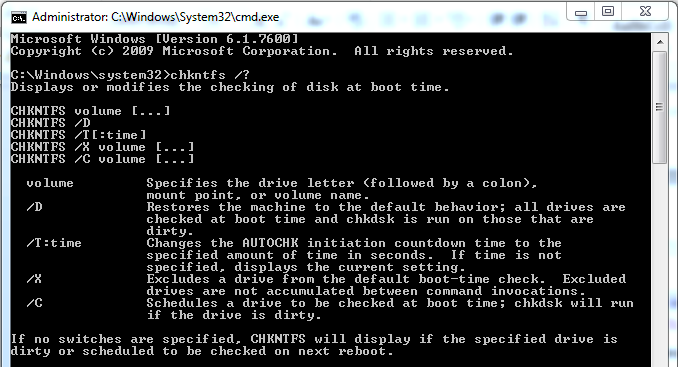
## Improving General Availability of the Server

If a volume is found to be corrupted and detected during run-time; NTFS schedules Chkdsk to run during the next reboot. This is done by invoking autochk.exe.

If the computer/server hosts multiple volumes; even if one data volume is corrupted, by default, NTFS schedules Chkdsk to run during startup.

Most reboots in servers occur due to other reasons (such as Windows Updates). This causes an inadvertent delay in starting the system. If the system volume is corrupt, it is recommended that you fix the system volume. The data volumes can be fixed after system startup. By default, the system checks and fixes all volumes that are marked dirty during start up.

Server administrators can choose to not delay the process of holding up the server at reboot by explicitly turning off checking the data volumes at reboot time. This can be done manually or using a script after the system is up and running. Use the *chkntfs* command to indicate ***not*** checking data volumes. This setting of including/not including certain volumes during startup is persistent across reboots.



***Stopping Autochk from running upon Reboot on a Volume***

***Command: chkntfs /x drive: (chkntfs /x d: )***

# Chkdsk performance on Windows Server°2008°R2

## Block Caching Improvements in Windows Server°2008°R2

In Windows°7 and Windows Server°2008°R2, Chkdsk was updated to improve how it caches metadata that is read from disk. A significant portion of the scan phase of Chkdsk requires Chkdsk to seek different locations of the disk to verify information from file system metadata objects. By caching larger blocks of the disk in RAM, Chkdsk execution time is reduced by reducing the seek time required to complete the verification.

The Block Cache feature also reduces the need to re-access data from the disk; this reduces the total device I/O time, and therefore, the overall Chkdsk run-time.

This may cause Chkdsk to consume a significant amount of memory (this is apparent only when Chkdsk is run in read-only mode on the system volume, or when Chkdsk is run in fix mode on non-system volumes), but reduces the amount of disk I/O and therefore significantly improves performance. The net result is greatly reduced execution time in common scenarios.

Microsoft recently ran internal benchmarks in order to characterize Chkdsk performance. The primary measurement used to evaluate Chkdsk performance is the execution time.

For these tests, Microsoft selected a standard server configuration with memory between 8GB – 32GB of RAM. Data sets were chosen to simulate common Windows file server scenarios.

Key elements that affect the execution time of Chkdsk are:

1. Number of files in the Volume (option: /F, /R, /X, /B)
2. Available physical memory (option: /R, /F, /X, /B)
3. Size of Volume (option: /R, /B)

Effect of Volume Size on execution time of Chkdsk

In this test, Windows Server°2008°R2 is configured in a file server role to show the effect of volume size on Chkdsk execution time. All Chkdsk execution times in this test were measured on a volume with 10 million files each of which are 4KB. (Note that this test was run on a different system than specified above due to the need for large volume area.)

The results show that the volume size has no effect on the execution time of Chkdsk.

|  |  |  |
| --- | --- | --- |
| **Volume size (in TB's)** | **Time (sec)** | **Time (mins)** |
| 5 | 384.823 | 6.4 |
| 10 | 407.798 | 6.7 |
| 15 | 417.008 | 6.9 |

**Note that this test was done on the following configuration:**

* System Manufacturer: Dell Inc.
* System Model: PowerEdge 2970
* System Type: x64-based PC
* Processor: Quad-Core AMD Opteron(tm) Processor 2360 SE, 2493 Mhz, 4 Core(s), 4 Logical Processor(s)
* Installed Physical Memory (RAM): 8.00 GB
* Total Physical Memory: 7.99 GB
* Available Physical Memory: 7.05 GB
* Disk Details:
* Description: Disk drive
* Manufacturer: (Standard disk drives)
* Model: DELL PERC 6/E Adapter SCSI Disk Device
* Size: 25.46 TB (27,997,816,734,720 bytes)

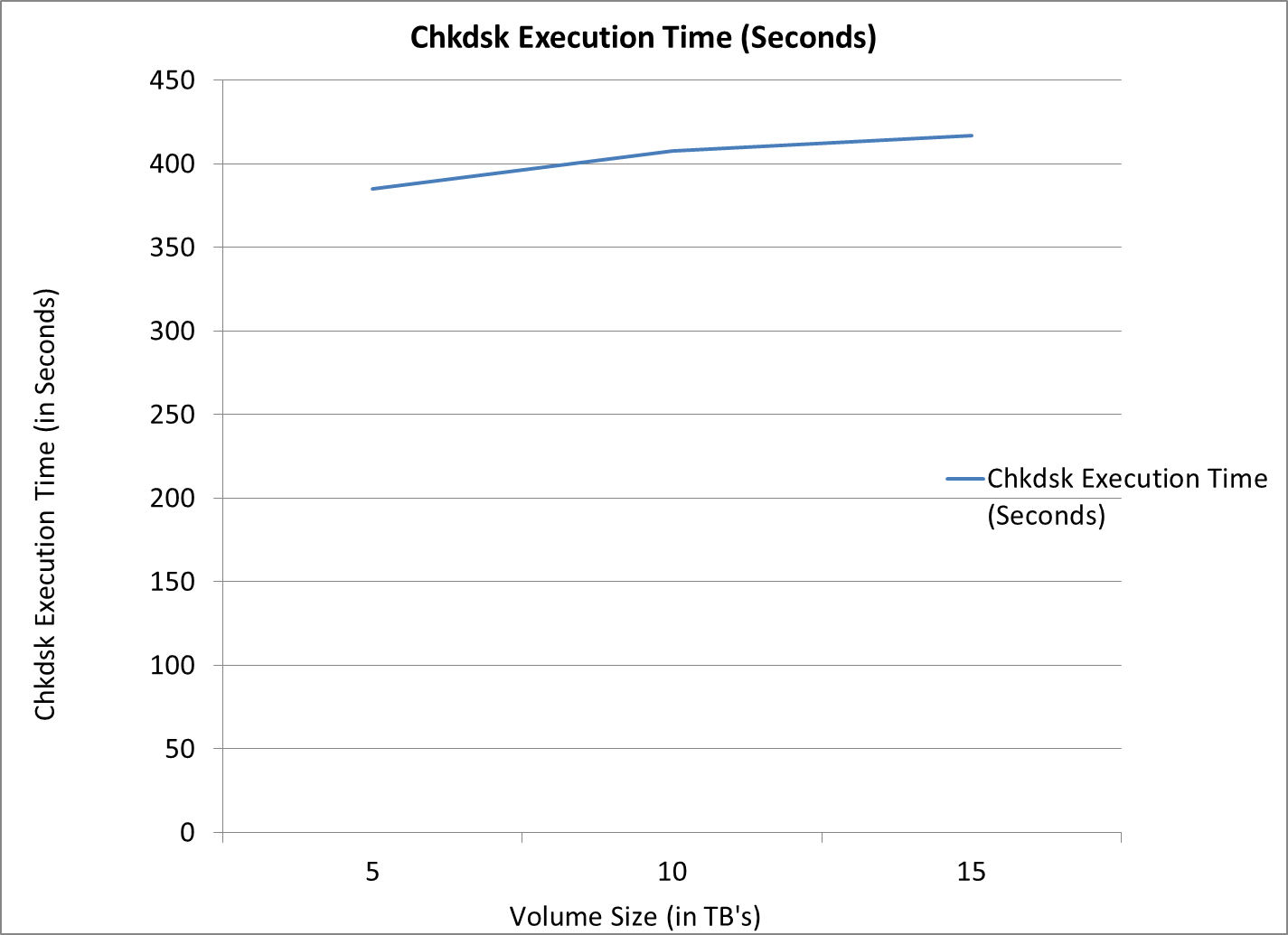


Figure : Chkdsk execution time vs. Volume Size

Effect of Number of files and Different OS versions on execution time of Chkdsk

In this test, several different versions of the Windows Server are configured in a file server role to show the effect of the number of files and different OS versions on Chkdsk execution time. In this configuration, tests are run with the number of files varying from 100 million to 300 million.

The results show that in cases that tested the Chkdsk available in Windows Server°2008°R2, Chkdsk run-time is faster than the Chkdsk available in Windows Server°2008.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Number of Files** | | |
| **OS Type** | **100M** | **200M** | **300M** |
| Windows Server 2008 | 4:14:00 | 8:34:00 | 24:54:00 |
| Windows Server 2008 R2 | 1:49:00 | 4:52:00 | 6:16:00 |

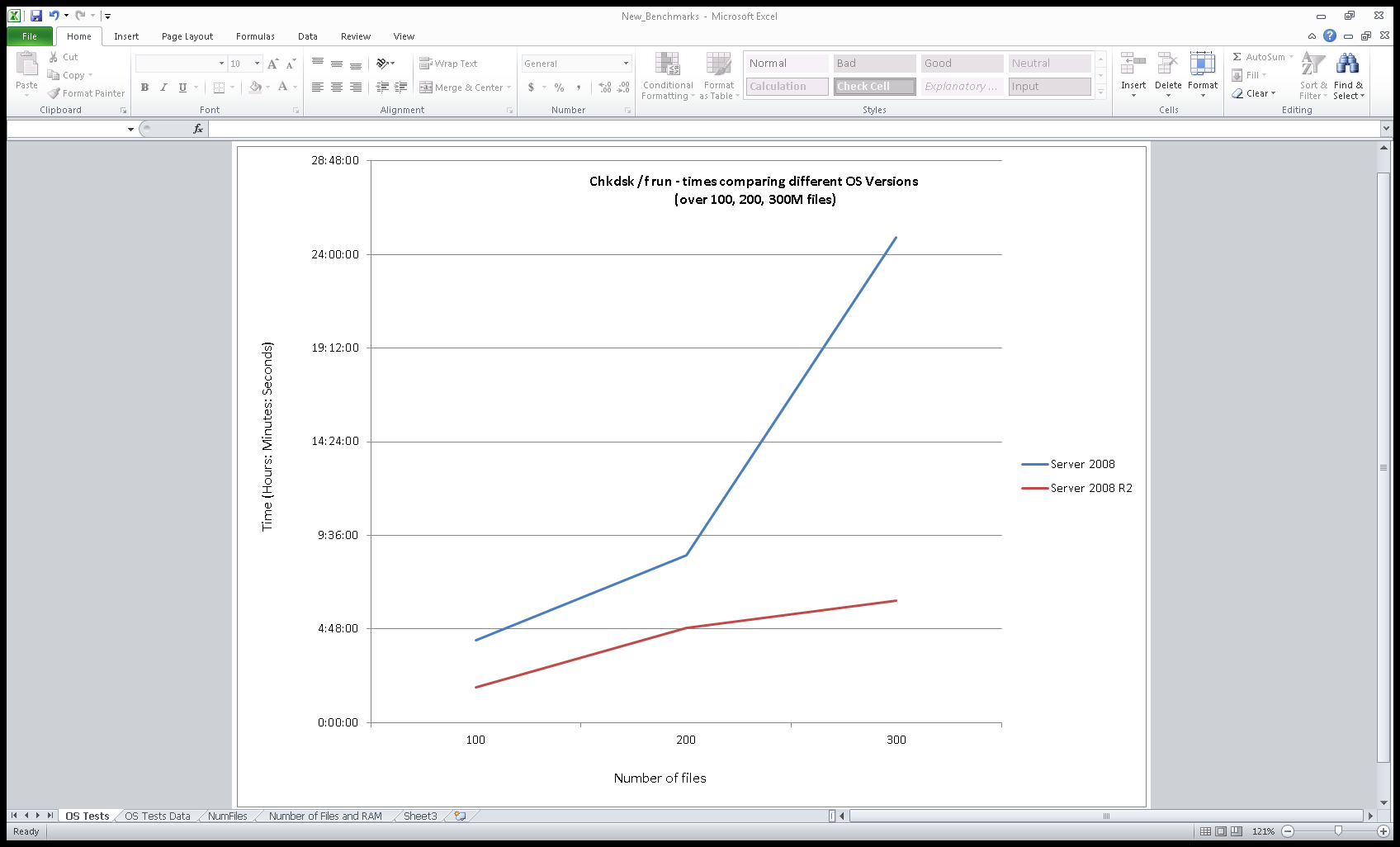


Figure : Chkdsk execution time over different OS'es

Effect of Physical Memory at different Number of files on execution time of Chkdsk /f

In this test, Windows Server°2008°R2 is configured in a file server role to show the effect of number of files on Chkdsk /f execution time.In this configuration, tests are run with a varying number of files between 100 and 500 million files.

The results show that execution time of running Chkdsk /f is directly correlated with the number (millions) of files in the volume.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Ram Size** | | |
| **Num of Files** | **32GB** | **16GB** | **8GB** |
| 100M | 1:53:00 | 1:48:00 | 2:04:00 |
| 250M | 5:16:00 | 6:53:00 | 5:41:00 |
| 500M | 19:25:00 | 76:07:00 | 99:08:00 |

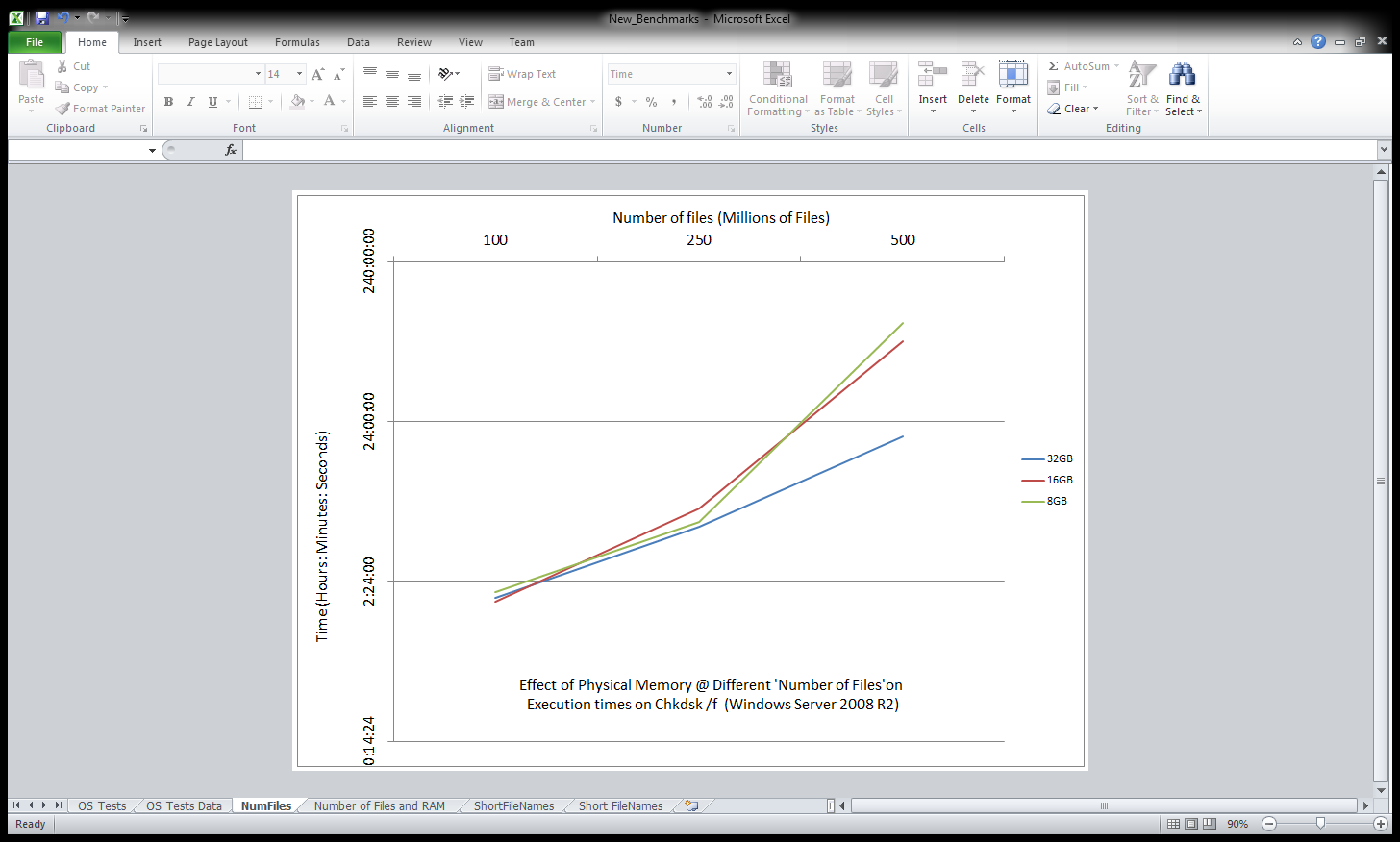


Figure : Chkdsk /f execution times varying physical memory over number of files (Millions of Files)

Effect of short file names on Chkdsk execution time

When you save a file with a long file name to an NTFS drive, by default, NTFS creates a second file directory entry with a short file name that conforms to the 8.3 convention.

When NTFS enumerates files in a directory, it has to look up the 8.3 names associated with the long file names. Because an NTFS directory is maintained in a sorted state, the corresponding long file names and 8.3 names are generally next to one another in the directory listing. However, in some cases, for many files with the same prefix, the corresponding long file names can be spread out. As the number of files in the directory increases, there is a direct performance impact on the creation of new files in that directory.

There are similar effects on the execution time of Chkdsk with short file names when there are tens of thousands of files present in a directory.

8.3 filenames can be turned off using the *Fsutil* functionality from a command prompt as shown in Figure h. If possible, you should disable 8.3 file name creation since this increases the overall execution time of Chkdsk.

Starting with Windows°7 and Windows Server°2008°R2, ‘8dot3name’ creation is can be set on a per volume basis versus one global setting as in earlier versions of Windows.



Figure : Fsutil option to turn off 8.3 file names

When disabled, NTFS no longer creates short file names (8.3 names) when files are created with long file names. This reduces the scan time needed during Chkdsk runs. Note that existing short file names will continue to function as before.

Effect of Enabling/Disabling short file names on execution time of Chkdsk /f

In this test, Windows Server°2008°R2 is configured as a file server role to show the effect of enabling and disabling short file names on Chkdsk /f execution time. In this configuration, tests are run with a varying number of files between 100 and 300 million files.

The results show that execution time of running Chkdsk /f is more when short file names are enabled in the volume.

|  |  |  |
| --- | --- | --- |
| **Short File Names** | **Number of files** | **Time in Seconds** |
| Enabled | 100M | 7716 |
| Disabled | 100M | 6495 |

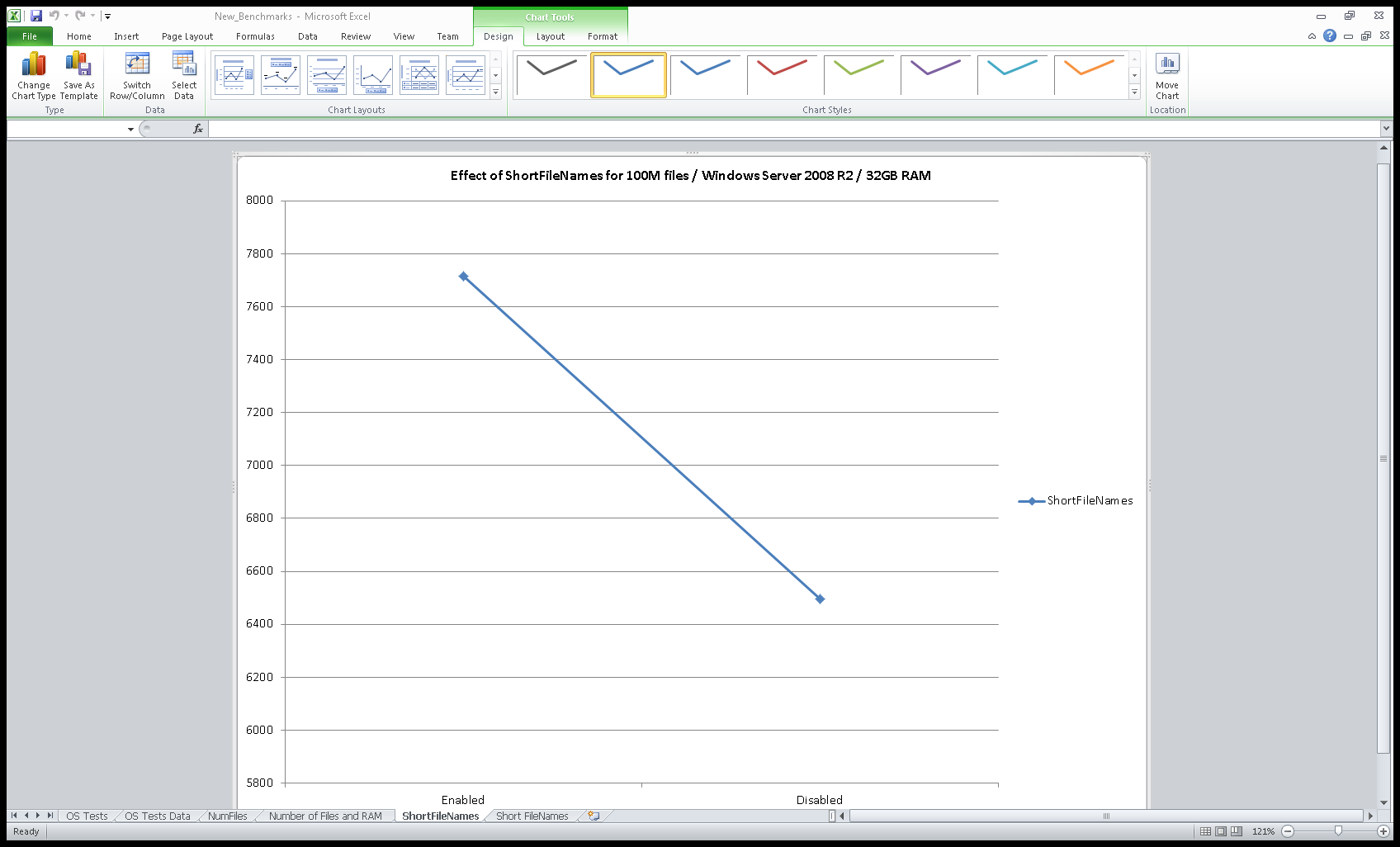


Figure : Chkdsk /f execution time varying the short filename file setting

Conclusion

Microsoft continues to invest in improving Chkdsk, including the additions in Windows°7/Windows Server°2008°R2 (such as caching and other improvements) that reduces Chkdsk execution time.

Chkdsk /R can take a significant amount of time to run, and therefore, is only recommended when there is doubt that the disk is failing and this type of scan is needed.

Chkdsk /I /C can fix several file system related errors and can finish quickly.

Server availability can be significantly increased by eliminating the need to check data volumes at boot time. This allows other volumes to run while the one corrupted data volume is being checked.

# Call to Action

Microsoft strongly recommends that administrators set up a typical scenario for their server and measure typical Chkdsk execution times, but only use the performance numbers in this paper as an initial data point.

You should add more memory if the Chkdsk execution time is too long for your scenario. The data in this paper provides some guidance to determine the amount of memory needed for your scenario.

Remove data volumes from the boot time check. This will help increase overall server and system availability. You will need to create scripts to check and fix issues in data volumes after startup.

Do not run Chkdsk on a volume unless it has been marked ‘dirty’ by the file system. Microsoft recommends querying the file system for whether it is ‘dirty’ before running Chkdsk.

Running Chkdsk in read-only mode can help predict server or system down-time before running the full Chkdsk offline.

Microsoft recommends turning off short file names at a volume level to reduce Chkdsk execution time. Note that some applications may require short file names. You should check your applications before making this change or test with this change to avoid unintentional behavior of the system.

# Resources

For more information about Chkdsk, please refer to the following publications and Microsoft Knowledge Base articles:

* *Inside Microsoft Windows 2003, Third Edition* from Microsoft Press®
* [187941](http://support.microsoft.com/default.aspx?scid=kb;en-us;187941) “An Explanation of Chkdsk and the New **/c** and **/i** Parameters”
* [191603](http://support.microsoft.com/default.aspx?scid=kb;en-us;191603) “Modifying the Autochk.exe Time-out Value”
* [160963](http://support.microsoft.com/default.aspx?scid=kb;en-us;160963) “CHKNTFS.EXE: What You Can Use It For”
* [280353](http://support.microsoft.com/default.aspx?scid=kb;en-us;280353) “How to Change Quorum Disk Designation”
* [265533](http://support.microsoft.com/default.aspx?scid=kb;en-us;265533) “Explanation of Chkdsk Status Codes in Cluster Log”
* [272244](http://support.microsoft.com/default.aspx?scid=kb;en-us;272244) “Location of the Chkdsk Results for Windows Clustering Resources”
* [176970](http://support.microsoft.com/default.aspx?scid=kb;en-us;176970) “How to Run the Chkdsk **/f** Command on a Shared Cluster Disk”
* [218461](http://support.microsoft.com/default.aspx?scid=kb;en-us;218461) “Enhanced Chkdsk, Autochk, and Chkntfs Tools in Windows 2003”
* [314835](http://support.microsoft.com/default.aspx?scid=kb;EN-US;314835) “An Explanation of the New /C and /I Switches That Are Available to Use with Chkdsk.exe (Windows XP)”
* [903650](http://support.microsoft.com/default.aspx?scid=kb;EN-US;903650) “Extended maintenance mode functionality for cluster physical disk resources in Windows Server 2003”