Global Knowledge®

Expert Reference Series of White Papers

Speeding Up File Transfers in Windows® 7

Speeding Up Windows File Transfers in Windows 7

Glenn Weadock, Global Knowledge Instructor, MCITP, MCSE, MCT, A+

Introduction

File transfer speed is a big deal. In fact, user response to Vista (pre-SP1) was negative in large part because Vista transferred files in many situations more slowly than XP, its predecessor.

The complex thing about file transfers in Windows is that they involve many factors, any one of which can present problems: which mix of operating systems you're using; Network Interface Card (NIC) settings; layered software (such as antivirus); switch settings; whether you're transferring one large file or lots of small ones; etc.

One thing is for sure: in a typical Windows network with a mix of clients and servers, file transfer performance can vary wildly. Because it's such a common operation, it's worth looking at some of the reasons and, in the process, unearthing a few tricks and tips that might make file copying snappier on your network. That's what this white paper does, and it is organized as follows.

- SMB Versions
- Pull vs. Push
- NIC Settings
- NETSH Settings
- File System Buffering
- Multithreaded Copying
- Drive Compression and Encryption
- Antivirus/Antimalware Utilities
- Local Disk Copies

SMB Versions

One of the reasons that file transfers are faster with newer versions of Windows has to do with the version of SMB (Server Message Block) that these operating systems support. The SMB file-sharing protocol (which you may also know as CIFS, the Common Internet File System) has been around since Windows for Workgroups, but it's been stuck at version 1.0 for many years.

File transfers between Vista and Windows 2008 use SMB 2.0. Transfers between Windows 7 and Server 2008 R2 use SMB 2.1. (Any transfers involving XP or Server 2003 on either end use SMB 1.0, basically the same version

that's been around for years.) The version of SMB that is used in any given file transfer is the result of a negotiation between the two computer operating systems involved.

SMB 2.x decreases overhead by allowing multiple requests to be collected into a single packet. It also supports much larger buffers (with the server controlling the size of the Maximum Transmission Unit (MTU), and permits more simultaneously open file handles. SMB 2.1 goes further by supporting "pipelined I/Os" in which a request can be sent before the previous request has been answered.

So how much of an improvement is 2.1 over 2.0? With a 2-gigabyte file size and a 10-gigabit Ethernet network, Microsoft estimates a 30% to 40% gain over SMB 2.0 (which, itself, is a big improvement over SMB 1.0). The large MTUs also let SMB 2.1 systems cache larger directories, which will make copies of folders having more than about 500 entries go even faster.

Finally, here's a nice benefit: Access to DFS (Distributed File System) shares is also accomplished via SMB 2.x messages now, so you should see improvements when copying to and from DFS shares as well as "normal" shares.

Pull vs. Push

I don't necessarily understand the underlying reasons for this particular effect, but it seems to be generally true that "push" transfers – that is, those in which the copying user is working on the system containing the file to be copied to another system – can be dramatically faster than "pull" transfers, those in which the copying user is working on the system to which the file is destined. The difference can be on the order of 50% faster in the "push" scenario!

NIC Settings

One thing to check is the link negotiation settings on all Network Interface Cards, (NICs) (see Figure 1) and switches. Some network administrators report best performance with consistent settings. I've heard frequently that "auto detect" may not work as well as an explicit setting ("1000 Mbps"). Another NIC setting to test is flow control. If the NIC supports jumbo frames, and your switches do, too, you may be able to improve file copy performance by enabling this feature. You can experiment with receive buffers also. All these settings are highly dependent on specific NICs and drivers, and on interactions between NICs and switches, so you have to do some experimenting to see what has the greatest impact.

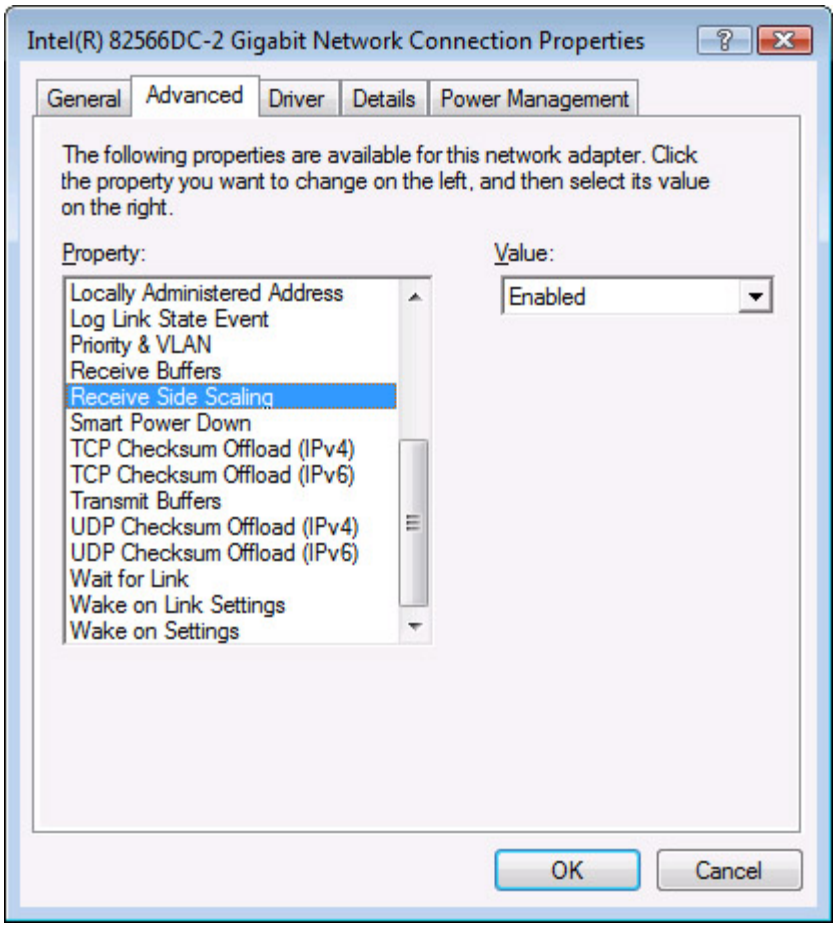

Figure 1: Viewing NIC driver settings.

NETSH Settings

The NETSH command lets you make a number of Registry settings that can have a sometimes-huge impact on file transfer performance in Windows. Here are some to consider:

TCP Chimney Offload. This is a Server 2008 technology, also supported in Server 2003 as of Service Pack 2, that moves some processing activities from the CPU to the NIC during file I/O. The NIC must support the technology or turning this setting on could actually impair performance, however. The relevant NETSH command for Server 2008 is:

```
netsh interface tcp set global chimney=enabled
```

and for Server 2003:

```
netsh interface ip set chimney ENABLED
```

Autotuning Level. Disabling automatic tuning for the TCP window size can dramatically improve network file copy performance. You can reset it to the default by replacing disabled with normal in the command below. You may also wish to experiment with the highlyrestricted option.

```
netsh interface tcp set global autotuninglevel=disabled
```

Receive Side Scaling. This technology enables packets received from a network adapter to be managed by multiple CPUs in a computer that has them. The Server 2008 command is as follows:

```
netsh interface tcp set global rss=enabled
```

Server 2003 SP2 and higher requires a Registry hack to turn on this feature. Navigate to HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters, and create (if it does not exist) the DWORD value EnableRSS, and set it to either 1 (enabled) or 0 (disabled).

NetDMA. Turning Network Direct Memory Access, or NetDMA, to off or 0 may help file copy performance with machines running other operating systems (e.g., Mac OS X). This is a Registry hack: navigate to HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters, create (if it does not already exist) a DWORD value named EnableTCPA, and set it to either 1 or 0.

After optimizing the above settings on two machines in my own office network (see Figure 2), I was able to improve file transfer speed from 16.7 MB/s to 26 MB/s between a Vista client and a Server 2003 file server. That's pretty significant!

Figure 2: Viewing TCP global parameters on a Vista system.

File System Buffering

Everybody knows what buffering can do for your file system's performance in typical use. Caching recently read blocks in memory makes them accessible more rapidly in subsequent reads. However, when you're moving a 500MB file from point A to point B, that buffering may not do you any good; in fact, it may slow things down. It's unlikely that you have any intention of re-reading that 500MB file after you've moved it to its desired location, so buffering simply imposes unnecessary overhead.

Therefore, copying large files is usually better done in an unbuffered manner, as for example is done by **ESEUtil** from Exchange Server. The tool you use may have an impact, depending on whether it performs file buffering.

Along this line, in Windows 7 and Server 2008 R2, you can specify the **/J parameter** in an **XCOPY** command to specify that you don't want to perform file buffering.

DynCache, or The Microsoft Windows Dynamic Cache Service (supported on Server 2003, Vista, and Server 2008. but not 2008 R2), is a free download from Microsoft that dynamically manages the working set size of the Windows file cache. This is especially useful for 64-bit systems in which it is possible for the read cache to grow until it occupies the entirety of physical memory in the machine (this is a bad thing). Be advised, however, that DynCache is categorized by Microsoft as "experimental." Also be advised that you will be doing some Registry hacking with this tool, given the fact that there is no graphical user interface for its operating parameters.

Multithreaded Copying

Most of the copy utilities we use daily are single-threaded: they copy one file before moving to the next one. That means you're looking at pretty awful performance when you have lots of small files. In such a scenario, the metadata (file name, etc.) becomes a higher percentage of the total bits transferred than when you're copying a smaller number of large files.

The version of Robocopy provided with Windows 7 and Server 2008 R2 supports full multithreaded operation, which is a nice improvement over previous iterations of Robocopy, which were all synchronous (that is, non-parallel). But what about shops that haven't yet moved to Windows 7?

One tool you may want to check out is **RichCopy** (see Figure 3), by Microsoft's own Ken Tamaru. It's not supported, but that may not stop the server admin who's hunting for better file copy performance. (FYI, RichCopy also lets you turn off file system buffering.) Another tool to investigate is the **Robocopy GUI** written by Microsoft employee Derk Benisch. This tool lets you create Robocopy scripts that you can then execute in parallel.

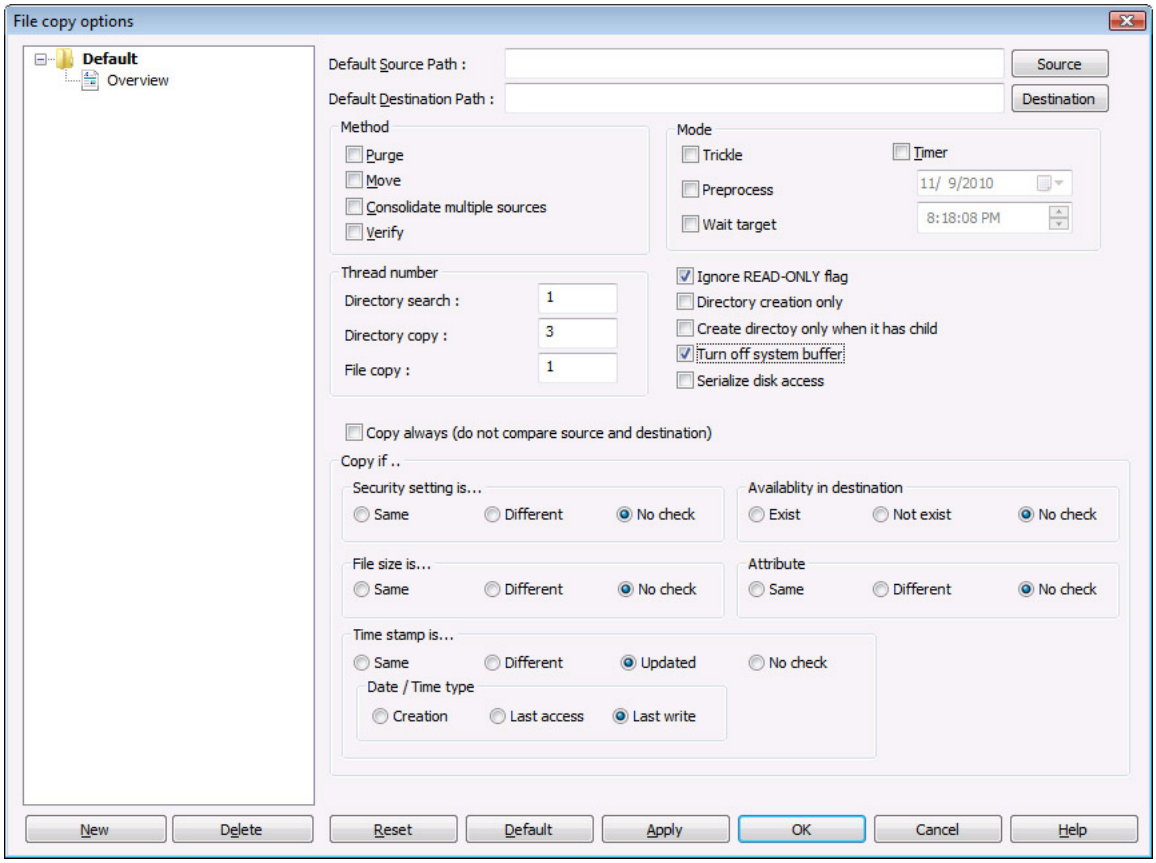

Figure 3: RichCopy is rich with options, including disabling the file buffer.

Drive Compression and Encryption

NTFS compression has been around for many years in Windows and will normally degrade file operations. NTFS-compressed files are expanded prior to being copied over the network. The compression and decompression requires more processor overhead than copying uncompressed files, inducing a performance drag that will be more noticeable on CPU-bound systems. Microsoft does not recommend using NTFS compression for folders that are frequently read from or written to.

The well-known BitLocker full volume encryption technology has been well engineered from the speed standpoint, and users tell me that they notice essentially no difference when performing file copies over the network to or from a BitLocker-encrypted drive. I have measured a BitLocker overhead of 1% to 2% in my own informal testing. Other encryption products should be tested to see what their performance overhead might be.

Antivirus/Antimalware Utilities

Antivirus and antimalware utilities generally include real-time analyzers that can kick in whenever a user copies a file from one place to another. These products vary wildly in terms of their impact on system performance. For

example, Microsoft's own client-side tools (Defender, Security Essentials [see Fig. 4]) seem to have a fairly light "drag factor" on the operating system. Other tools have been known to impact performance in a big way.

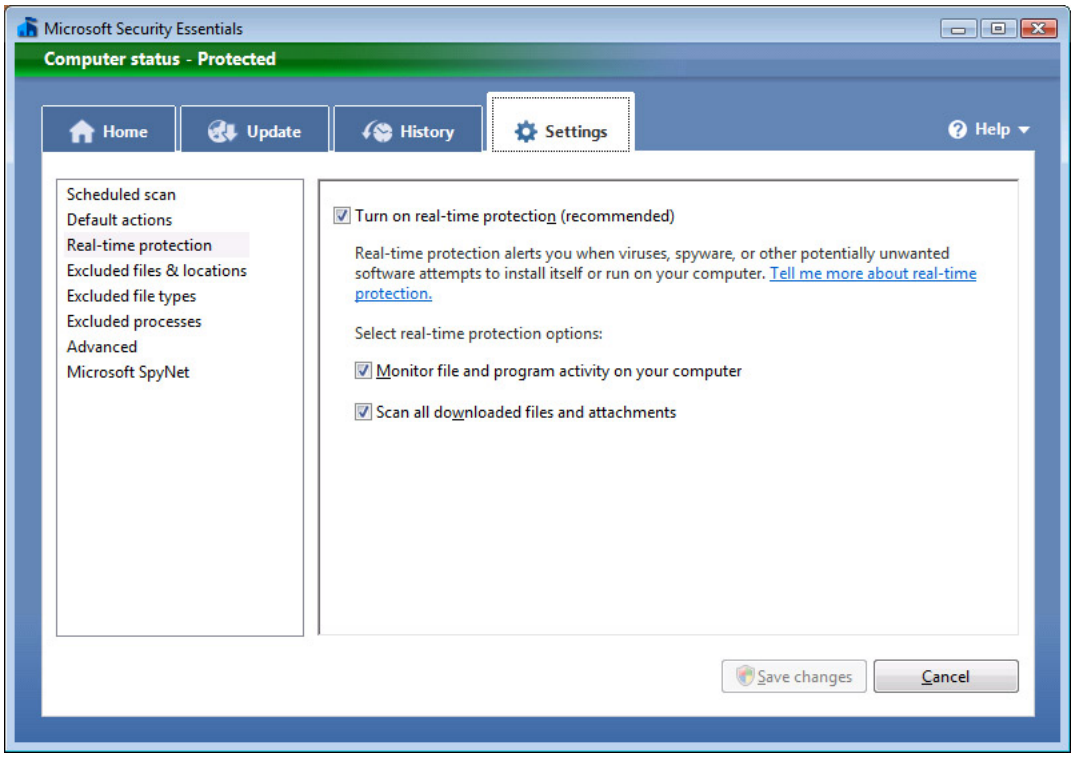

Figure 4: Microsoft Security Essentials includes a realtime file scanner.

When you are testing the effect of some of the tips and tricks outlined above, you may want to test both with and without antivirus and antimalware tools active. You may discover that although you can document a significant difference with the tools off, the difference is dwarfed by the overhead imposed with the tools on.

For many years, I've been advising companies and seminar students that the impact on system performance should be an important criterion when organizations perform comparative evaluations of antivirus and antimalware tools. File transfer speed is certainly one aspect of that impact and worth some analysis when those utility software licenses come up for renewal.

Local Disk Copies

If you're doing a lot of file copying between local disks, as opposed to across the network, you may want to check out the TeraCopy program (see Figure 5). It makes impressive speed claims, and it has a variety of user-friendly features, such as auto-continue, resume of a failed copy, notification of failed copy operations, and copy verification.

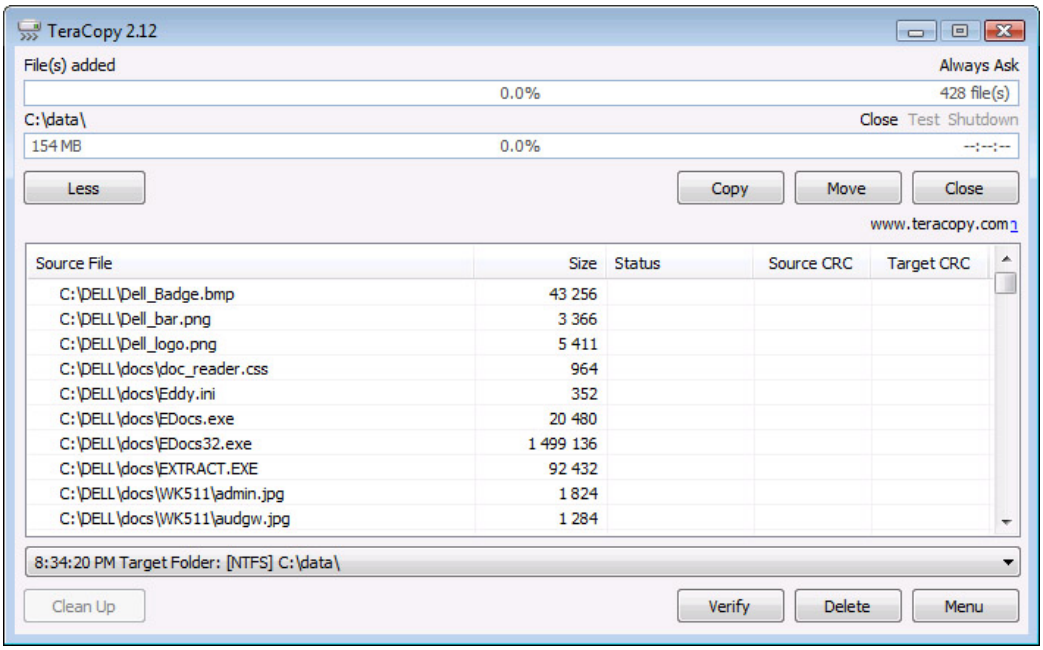

Figure 5: TeraCopy is well suited for non-network file transfers.

Conclusion

You wouldn't think that a function as basic as file copying could be as complicated as it is under Windows clients and servers. Some of the factors that determine file transfer speed may not be easily modifiable: the version(s) of Windows that you're running, for example, or the network interfaces and switches in which your organization has invested. Other factors, however, can be changed: the programs you use for copying, the specific antivirus and anti-malware tools you employ, the NETSH settings for tuning the TCP/IP stack, and so on.

File copying is something many administrators and users do every day, sometimes dozens or hundreds of times a day. This white paper has identified a few techniques that could improve speeds in your Windows network. A little experimentation in this area can pay big dividends. So get out the stopwatch (or, more likely, the stopwatch app on your smartphone!) and get to work. I think you'll be surprised at the improvements that are possible.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge. Check out the following Global Knowledge courses:

Administering and Maintaining Windows 7 (M50292)

Configuring, Managing, and Maintaining Server 2008 R2 (6419)

Managing and Maintaining Windows Server 2008 Network Infrastructure Servers (M6431)

For more information or to register, visit **www.globalknowledge.com** or call **1-800-COURSES** to speak with a sales representative.

Our courses and enhanced, hands-on labs and exercises offer practical skills and tips that you can immediately put to use. Our expert instructors draw upon their experiences to help you understand key concepts and how to apply them to your specific work situation. Choose from our more than 1,200 courses, delivered through Classrooms, e-Learning, and On-site sessions, to meet your IT and business training needs.

About the Author

Glenn Weadock is a longtime instructor for Global Knowledge and teaches Windows 7, Vista, Server 2008, and Active Directory. He has recently co-developed with Mark Wilkins two advanced Server 2008 classes in the Microsoft Official Curriculum. Glenn also consults through his Colorado-based company Independent Software, Inc.