When I'm x64

Wes Miller

Windows XP has been available natively for 64-bit architectures for over five years. But unless you were one of the early adopters of the Intel Itanium processor (Windows XP for the Itanium shipped the same day as Windows XP), you've probably been hearing about it more recently with the availability of Windows XP and Windows Server 2003 versions for x64 systems. x64 – also referred to in the industry as x86-64 – is the generic name for AMD's AMD64 architecture and Intel's EM64T. And if you've bought a new PC within the last year or so, there's a good chance it's 64-bit capable, even if it's currently living a 32-bit life (as most likely are).

Today I work for a small startup in Austin, Texas. Because of the architecture of one of our products, we are interested in taking advantage of some very specific benefits in the x64 architecture – much like the Microsoft Exchange 2007 team, which put a stake in the ground and has begun shipping solely for the x64 architecture. Similarly, the development and test team that I manage is solely deployed on x64 versions of Windows XP and Windows Server 2003 for development workstations, laptops, servers and production servers. Furthermore, I run Windows XP Professional x64 Edition on my personal laptop in order to test and debug our product.

Interestingly, the first response I get from most people when I mention running 64-bit Windows – especially on a mobile system – is a rather confused look. And if they're familiar with 64-bit computing, it's something like disbelief meets nausea meets amazement – due to the commonly held assumption that it's difficult to round up device drivers for 64-bit systems. In this month's column, I'll explain how and why I use Windows XP and Windows Server 2003 on x64 systems in 64-bit mode, and I describe some of the deployment benefits (and hurdles) you can expect to run into. I also go into aspects of Windows Vista x64 support, migration and deployment.

A bit of history

As I noted earlier, the origin of 64-bit support in Windows really begins with support for the Intel Itanium processor. (Though Windows was available for the 64-bit Alpha processor, Windows was never actually 64-bit when running on the Alpha.) There is no support for Itanium in Windows XP and Windows Vista, so the x64 architecture currently carries the torch for 64-bit Windows client computing. A larger selection of Windows Server 2003 editions is currently available for x64 than for Itanium (which has essentially been relegated to ultrahigh-end data centre workloads) – a trend I expect to continue with Windows Server 2008 when it ships.

Support for Windows on the x64 platform became available when Windows Server 2003 Service Pack 1 (SP1) shipped. Though it's somewhat confusing, this is also when the x64 version of Windows XP became available – meaning that Windows XP 32-bit products and 64-bit products derived from different code trees within Windows. While 32-bit products currently have a second service pack available for them, the 64-bit version of Windows XP technically has no service pack level at all (or you could think of it as having SP1 for Windows Server 2003 slipstream integrated into it already).

In order to run 64-bit Windows – Windows XP, Windows Server 2003, Windows Vista or Windows Server 2008 – the requirements are effectively the same. Obviously, first you need to have a 64-bit-capable processor. In the case of AMD, this means looking for a single, dual or quad-core system that touts AMD64 compatibility (see amd.com/us-en/Processors/ProductInformation/0,,30_118,00.html). For Intel, this means looking for a single, dual or quad-core system that touts either EM64T support or the Intel 64 Architecture (see intel.com/technology/intel64). Note that the devil is in

Figure 1 **Memory address space**

Address space	64-bit Windows	32-bit Windows
Virtual memory	16Tb	4Gb
Paging file	512Tb	16Tb
Paged pool	128Gb	470Mb
Non-paged pool	128Gb	256Mb
System cache	1Tb	1Gb

the detail here. An Intel Core Duo system, for example, is not 64-bit capable. An Intel Core 2 Duo system is 64-bit capable – even though it may be referred to as simply a Centrino Duo (as in the case of my laptop).

Living with x64

Once you've determined whether your system will natively support a 64-bit version of Windows, you need to deal with device support. Regrettably, as much as Microsoft has encouraged vendors and OEMs (through at least two years of WinHEC sessions) to create and certify 64-bit drivers for their devices and systems, the largest challenge you run into when using Windows x64 is finding drivers for hardware devices or software. In my experience, you will be most successful running x64 on a server where necessary device support is limited and the benefit of x64 is the most logical.

It's hardest to find drivers for desktop and mobile systems. Generally, you'll have better luck with a major OEM provider or a system built from enterprise-class components (where the likelihood of business use leads the vendor or OEM to create and sign x64-capable drivers).

For Windows Vista, this situation happily gets much better. Instead of a world where x86 is the preferred architecture and x64 is the ignored architecture with limited drivers available for the majority of hardware, the opposite is true. For Windows Vista, a device vendor must submit x64 drivers for compatibility testing. In fact, x86 drivers do not need to be included as well, but certainly can be. Conceptually, this means that some hardware – especially devices that take advantage of new Windows Vista capabilities – may actually only have x64 Windows support and not have x86 support.

The reality is that drivers are only one problem you will run into when attempting to run 64-bit Windows natively. The larger problem is getting there (or migrating) to begin with, but I'll discuss that later.

Why 64-bit?

AMD, Intel and Microsoft didn't move to a 64-bit architecture just for kicks, of course. The move to 64-bit provides several key benefits. As you can see in **Figure 1**, the most important enhancement provided by the x64 architecture is the ability to address significantly more memory – up to 16 terabytes (Tb) versus the 4Gb that 32-bit Windows can address. Note that while 64-bit pointers can address up to that 16Tb limit, applications only have access up to about 8Tb.

In addition, x64 versions of Windows provide for hardware-based Data Execution Prevention (DEP) – also available on x86 systems with NX (No Execute) support. This allows for a hardware-driven solution that Windows utilises to prevent buffer overflows. It prevents code execution from data pages in memory – see support.microsoft.com/kb/875352 for more information on this topic.

x64 versions of Windows allow for native support of multiprocessor or multicore CPUs, just as x86 versions of Windows do. But one of the largest complexities of imaging a Windows XP system is eliminated by the use of a single hardware abstraction layer (HAL) across all Windows x64 installations – no more struggling to identify the HAL in use on non-ACPI or single-CPU systems.

Basically, Windows x64 as an architecture allows you to use your existing management infrastructure and existing 32-bit software (through emulation) plus 64-bit software with the ability to access such significantly enhanced memory. In fact, the only unpleasant side effects to moving to Windows x64 are: the complete lack of support for 16-bit applications (including gotchas like 32-bit applications with 16-bit installers); problems with applications that do not run reliably when hardware DEP is enabled (though DEP can be disabled on a process-by-process basis, and the Windows Application Compatibility team shims some applications which do not run correctly with DEP enabled); and finally, the fact that in 64-bit Windows Vista, all drivers must be digitally signed.

This last item was added to help prevent tampering with the Windows kernel in memory, a technique often used by rootkits. The expectation is that since x64 is the architecture that Microsoft is encouraging vendors to develop drivers for first (in order to get drivers through Windows hardware quality testing), this isn't as scary a requirement as it might have been historically.

There are a couple of other points to note when considering Windows for x64 systems. There are no versions of Windows XP for x64 systems that contain the functionality included in Windows Media Center Edition or Windows Tablet PC Edition. However, this changes in Windows Vista, where the x64 versions of Windows have the same functionality as x86 versions.

Virtualisation

One thing I haven't mentioned is virtualisation. I'm not talking about Microsoft Virtual PC or the VMWare PC virtualisation products; I'm talking about operating system virtualisation for 32-bit executables. In 32-bit versions of Windows, 16-bit applications all run within the context of the NTVDM – a Virtual MS-DOS® Machine. As I mentioned earlier, there is no support for 16-bit applications in the 64-bit versions of Windows.

Instead, while most integrated components of Windows run natively as 64-bit binaries, some components and many third-party programs run as 32-bit applications.

Windows gives you an entirely new emulation layer called Windows On Windows (WOW) to allow for these 32-bit applications to see a different view of the operating system – effectively running on a 32-bit version of Windows. Under WOW, x86 applications see Program Files (X86) as simply Program Files. Likewise, %WINDIR%\SysWOW64 appears to x86 applications as %WINDIR%\System32. x86 applications view the registry key HKLM\SOFTWARE\Wow6432Node as they would view the HKLM\SOFTWARE node itself.

By using the Process Explorer tool (microsoft.com/technet/sysinternals/ProcessesAndThreads/ProcessExplorer.mspx), you can peer into another perspective of virtualisation. Because many 32-bit applications expect to interact with integrated 32-bit Windows binaries, many of those binaries are provided in both 64- and 32-bit versions. By adding an additional column to Process Explorer, you can easily see this. In **Figure 2**, you'll notice two instances of cmd.exe – one 32-bit version and one 64-bit version. Notice in the highlighted 32-bit instance, you'll see several wow64* DLLs loaded in the 32-bit process. These show the virtualisation taking place, which allows 32-bit applications to function under 64-bit Windows. Also note the working directory (Path) being used by the 64- and 32-bit versions of cmd.exe.

One final note of consideration on the topic of virtualisation – while Windows runs explorer.exe as a native 64-bit application, this has some pros and cons. The largest con is that any Explorer shell extension must be recompiled for x64 in order to work (the majority have not).

The opposite is true of Internet Explorer® – it runs natively with the 32-bit version as the default. This is done primarily so that the abundance of ActiveX® controls on the Internet already in use today will successfully work as-is. Without it, because 64-bit applications cannot load 32-bit DLLs or drivers in-process, there is effectively no way to load 32-bit ActiveX controls which comprise almost all currently available ActiveX controls, on a 64-bit browser – meaning quite a bit of Internet content becomes inaccessible. To see this in practice, try loading C:\Program Files\Internet Explorer\iexplore.exe on a 64-bit Windows system and visiting any site that loads ActiveX control content (even Windows Update) – the result will be varying degrees of failure.

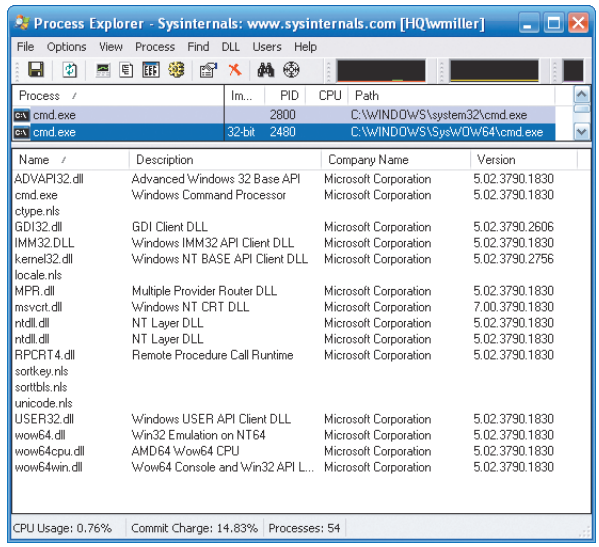

Figure 2 **Process Explorer reveals 32-bit and 64-bit versions of cmd.exe**

Windows Update (see **Figure 3**) has now updated the experience to open a 32-bit instance of Internet Explorer in this situation. In the future, surely there will be some ActiveX control content updated to take advantage of 64-bit Internet Explorer. But with the majority of Windows systems today running purely 32-bit Windows, the necessity for ActiveX control content to be updated is not critical. I don't expect this to change for quite some time.

x64 Windows at Pluck

I noted at the beginning of this article that at Pluck we have begun the transition to x64 Windows. We've done this with three different versions of Windows – Windows XP Professional x64 Edition, Windows Server 2003 Enterprise Edition x64 and Windows Vista x64.

Our organisation is small enough that as we started developing our most recent product, we could begin migrating to Windows for x64 to take advantage of its benefits. Doing so meant ensuring all new hardware we ordered was x64-capable. In fact, it is the expanded memory capabilities of x64 in concert with virtualisation that allow us to run our product the way we do. All of our physical servers run as 64-bit virtual servers on a 64-bit host server. Each host server has 16Gb of RAM divided among the virtual servers hosted on it (generally 2Gb for each staging virtual machine and 3Gb for each production virtual machine).

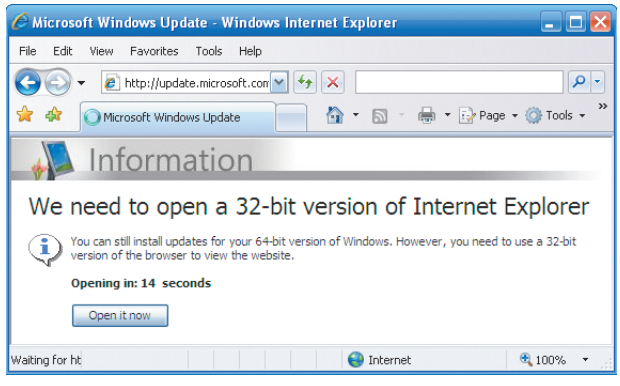

Figure 3 **32-bit ActiveX controls fail in 64-bit Internet Explorer**

By using tier 1 systems and virtualised hardware as well, we are able to achieve near 100 per cent device support. My laptop, for example, has three devices that do not have x64 drivers available. These all appear to be backplane type devices that are not critical, and the system performs fine without their presence. Interestingly, an installation of Windows Vista RC1 x64 resulted in exactly the same device support – though with my system bearing the Windows Vista Capable logo, surely even these missing drivers will arrive from the vendor in short order.

Since our use of the systems is primarily constrained to Microsoft Office, Visual Studio®, and several additional tools utilised in our development and build process, we do not have many legacy applications to cause compatibility issues in our migration. While quite a bit of our organisation does use 32-bit versions of Windows, we've begun the migration to 64-bit with new hardware and it has been relatively painless.

x64 deployment and migration

There are a few considerations that need to be taken into account regarding deployment. Microsoft provides a version of the Windows Pre-installation Environment (Windows PE) specifically for the x64 architecture. However, since there is x86 parity on x64 systems, you may well want to use x86 Windows PE to deploy – especially if you are using an imaging solution. If you are deploying Windows XP using unattended installation, Windows PE will have to be architecture-specific – as winnt32.exe must run under the same architecture it is deploying. Since Windows PE does not have a 32-bit compatibility layer, and winnt32.exe under x64 is a 64-bit application, 32-bit Windows requires 32-bit Windows PE. The same applies for 64-bit versions of Windows, which require 64-bit Windows PE.

Remote Installation Services (RIS) is different in this respect, and we use RIS at Pluck for this reason. RIS, beginning with Windows Server 2003 SP1, can deploy x64 as well as x86 or Itanium architectures of Windows. When an x64 system connects to a RIS server, the server (in default configuration) will offer up x86 and x64 RISetup or RIPrep images. RIS can be configured to offer x64 clients exclusively x86 images or exclusively x64 images. The choice was offered as the default to provide the best experience with x64 hardware.

Notice that I didn't mention upgrading yet. That's for good reason. Unfortunately, the act of upgrading an entire OS from x86 to x64 is a massive undertaking. For this reason – and given the limited deployment of Windows XP Pro x64 Edition – you cannot upgrade to Windows XP x64 from any x86 version, nor can you upgrade from any version of Windows XP (x86 or x64) to Windows Vista x64.

> 64-bit versions of Windows require 64-bit Windows PE

Windows Vista on x64 systems is, as Windows XP Pro x64 Edition before it, a clean-installation-only product. For this reason, many organisations consider migration to x64 only when a hardware refresh rolls around – a similar strategy that many organisations are expected to take with Windows Vista itself (making the dual jump at the same time potentially a logical one).

But that doesn't mean that people or organisations moving from x86 Windows to x64 Windows don't have any tools readily available – they definitely do. Versions of the User State Migration Tool (USMT) for Windows XP and for Windows Vista have both been extended to support migration (versus an in-band upgrade of the OS) from one installation of Windows to another. This is the predominant method most organisations use to deploy Windows today regardless – wipe and reload, even on the same PC – so utilising the same mechanism to move to Windows on an entirely new architecture should not present a much larger hurdle than before.

Conclusion

Though our small organisation's gradual migration to x64 may be considerably easier than the extensive migration process facing larger organisations, every IT professional must begin to consider the eventual move to the x64 architecture. The reality is that although the x86 architecture and x86 native versions of Windows will still be around well into the next decade, the x64 architecture is certainly the future of enterprise computing.

With Microsoft moving fully into x64 capabilities with Exchange Server and other products, you should begin understanding where x64 systems will fit in your architecture first, and where they may make sense in your Windows Vista migration plan as well, whether or not you are utilising Windows XP or Windows Server 2003 natively on your x64 systems today. ■

Wes Miller *is a Development Manager at Pluck (www.pluck.com) in Austin, Texas. Previously, he worked at Winternals Software and at Microsoft as a Program Manager and Product Manager for Windows. Wes can be reached at technet@getwired.com.*