

# Working with ImageX and Windows Image Files

By Mark Minasi

Microsoft's finally given the popular image tools — Acronis Snap Deploy, Ghost, Drive Image and the like — a run for their money with a tool that lets you take an entire operating system drive with all of its files, folders, settings and installed applications and crunch it down to one large file called a "Windows Image" or "WIM" file. (The file extension of Windows Image files is ".wim," which is where the short name comes from.) In this newsletter, I'll show you how to take a working Vista system and make — "capture" is the WIM terminology — that system into a single WIM file. Then we'll see how to take that image file and transfer ("apply" in WIMspeak) it to another computer. The tool we'll use to both capture and apply is the same program: ImageX.

## Getting Ready

Before we do any of that, however, we'll need to get a few things ready. First, you'll need to review some things covered in our previous newsletter #59. In it, we covered

- Downloading and installing the Windows Automated Installation Kit (WAIK), the 800+ MB ISO file that, when burned to a DVD and installed, puts Vista's deployment tools on your computer. The WAIK can be installed on XP SP2, 2003 SP1/R2, or Vista. Installing on XP or 2003 requires .NET 2.0 (it needed .NET 3.0 to work on x64 in my experience) and the MSXML parser, which is on the WAIK. **You'll need the WAIK to try out the examples in this article**, so if you haven't already read Newsletter #59, you'll find it at [www.minasi.com/newsletters/nws0701.htm](http://www.minasi.com/newsletters/nws0701.htm). In particular, you need for your WinPE CD-ROM to contain the file `imagex.exe` in its `windows\system32` folder.
- Creating and configuring Windows PE (WinPE) CD-ROM images. WinPE is a simplified, reduced-function version of Vista that you'll need to do imaging and other tasks. It lacks anything but the most rudimentary GUI and is largely controlled via command-line tools. Its beauty lies in the fact that it is much smaller than Vista — usually under 200MB, so it fits nicely on a CD-ROM — but it still knows how to work with modern storage devices and formats like NTFS (something that bootable DOS floppies, the tool of choice for imagers pre-Vista, lacks), and it knows how to load a network stack and connect to Microsoft shares (something that DOS floppies are also pretty weak at). **You'll need a WinPE CD-ROM if you'd like to try out the examples in this article**, so if you've not already created one then I recommend that you go back and create one.

You'll also need

- A working Vista system so that you've got something to image. It needn't be activated, nor does it need to run on a particularly powerful system, as we're just running ImageX through its paces.
- Finally, you'll need a place to store the image. Basic Vista images run in the 2.4-4 GB size. The two places that I've most frequently used to save a WIM file when imaging a system are either a network share, or an external USB drive.

## WIMs Examined

We met ImageX in Newsletter #59. In that article, you saw that you can use the ImageX program (which is just a file called `imagex.exe`) to view and modify the files and folders inside a WIM file, using ImageX's `/mount`, `/mountw`, and `/unmount` commands. But we didn't really discuss much of what a WIM file was in that article, so let's fill in that gap here.

As I've already said, WIMs are Microsoft's answer to Ghost images. They allow you to take a whole system and put "into cold storage," enabling you to maintain a library of pre-configured images that you can then deploy to systems quickly. Thus, for example you could imagine having built standard workstation images for managers, secretarial workers, developers, sales people, and developers. Then, when someone needs a new PC, or if that someone already *has* a PC but it's not working right,

# Working with ImageX and Windows Image Files

By Mark Minasi

then you needn't spend hours trying to fix it; instead, you just wipe it clean and install whichever standard workstation is appropriate for that person. Installing image files is usually faster than rebuilding a system from scratch, which means a big savings in time, and images can contain more than just the basic OS, which multiplies the benefits.

Of course, none of that is news to anyone who's used Ghost in the past few years. With Ghost, I could take a Vista system's C: drive and store it into one big file named (for example) markspc.gho, and with ImageX, I could take that same system and store it in one big file named (for example) markspc.wim. What's the difference? How is the WIM format different than the Ghost imaging approach? In several important ways.

## WIMs are File and Folder-Based, Not Sector-Based

Ghost doesn't know or care what sort of sector it's copying: FAT, FAT32, Linux EXT3, OS/2's HPFS, NTFS, you name it. As far as it and its brethren are concerned, disks are just partitions and partitions are just sectors. Ghost and tools like it just pull a sector off a hard disk and stuff it into an image file, then go on to the next sector, and the next until they've finished copying a partition, which is why they're called "sector-based" imaging tools. Sector-based copying is great for its flexibility — if some imaginary new operating system defines a new kind of partition then Ghost stands ready to image it — but it imposes some constraints for both modifying and deploying their image files.

In reality, it's not enough to just create a standard image like, say, "the manager's workstation" and then put it in cold storage, needing no maintenance ever again. What if want to change the image, as you essentially *must* do twelve times a year on the second Tuesday of every month; how to do that? Ghost has a tool that lets you essentially unpack and examine a Ghost file called "Ghost Explorer," but it can only let you *look* at an NTFS image, not change any files in that image. To apply a Windows patch to a Ghost image, then, you've got to first deploy that image to a computer to make the image into a functioning operating system, then apply the patch, then re-image the computer with Ghost to create a new image.

In contrast, you've already seen in the earlier article that ImageX lets you take a WIM file and "mount" it to a folder on a disk, allowing you not only to peer into the WIM file's contents, but also to modify them. (It's like you have not just a "Ghost Explorer" for NTFS images, but a "Ghost Conquistador" for them.) Mounting a file takes a lot less time than does imaging, patching, and re-imaging. But it gets even better: Microsoft says that they're designing their monthly Vista updates so that the updates can install themselves directly to a WIM, although I've not seen it happen yet. (They used to promise that we could deploy service packs directly to WIMs, but I don't see mention of that in any of the literature currently. Bummer. Also, as Vista's new I've not been able to actually *try* using a mounted volume to deploy a hotfix.)

WIMs can deploy patches because they don't store images as sector-by-sector copy operations. Instead, they see partitions not as sectors but as files and folders. As WIMs are internally laid out as files and folders, it's simpler for ImageX to let you view and modify them as files and folders.

## WIMs Install (Mostly) Non-Destructively

When you deploy a sector-based image to a hard disk, you destroy whatever partitions sit on that disk. Re-imaging a system with Ghost means wiping out a C: drive. As WIMs are file- and folder-based, however, then deploying a WIM to a system means only overwriting whatever files are in the WIM, and no others. So, for example, if you had a Vista workstation whose hard disk contained a

# Working with ImageX and Windows Image Files

By Mark Minasi

folder named "c:\olddocuments" that contained hundreds of Microsoft Word documents and you then re-applied a Vista image to that computer for some reason, then you'd have a fresh copy of the operating system that had overwritten your old one, but your c:\olddocuments folder would still be intact.

## You Can Use the WIM Format Cost-Free

All of the WIM-related tools are free downloads from Microsoft. Ghost, DriveImage and the Acronis tools all cost money. Of course, WIMs aren't the only no-cost options: you could learn a little Linux and use its dd command, which is basically an imaging tool.

## Using ImageX To Create A WIM

Enough theory; let's try this stuff out by making a working Vista box into a .WIM file. For this example, I'll have, as suggested before,

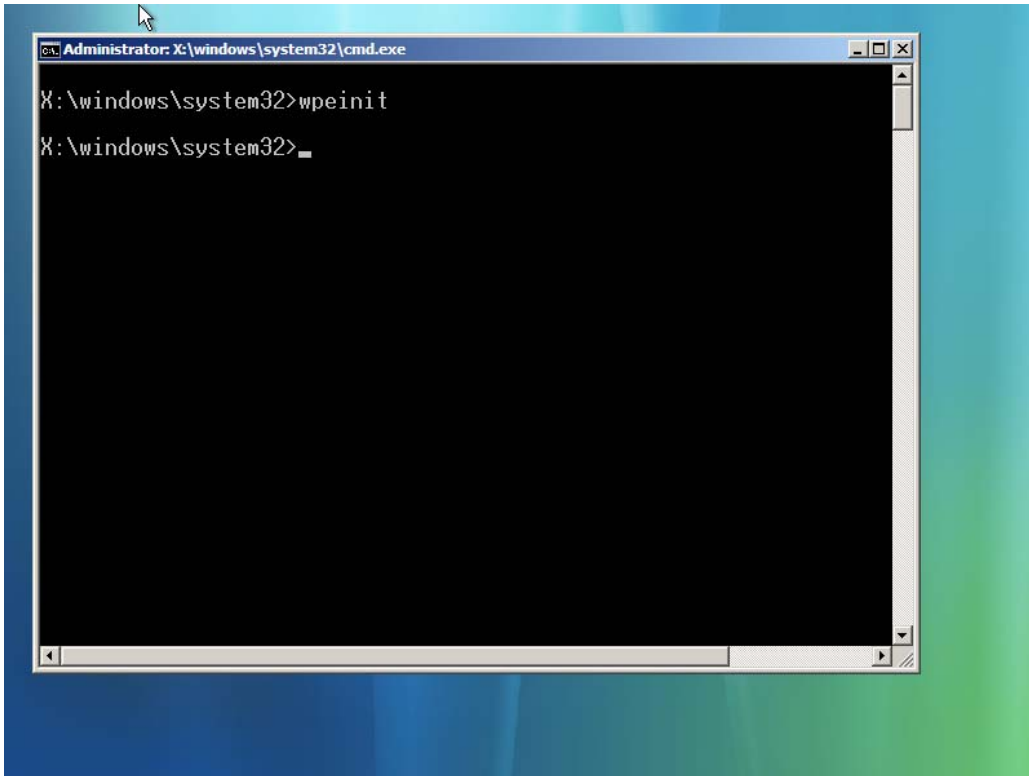
- A working WinPE CD-ROM with imagex.exe in its \windows\system32 folder
- A working test PC which has Vista running on it. To save the need for another test machine, I'm just going to create the WIM image of the computer, then I'll wipe that computer's hard disk clean, and then I'll deploy the WIM to the newly-cleared PC.
- An external USB hard disk, which has drive letter F:, or a network share mapped to F:

Start out by doing this:

1. Plug the external hard disk into your PC and verify that Vista can indeed read and write the drive. If it doesn't work out to have drive letter F:, don't worry about it — just remember to substitute that letter whenever I use an "f:" in these examples. Again, a network share works just as well, although it's a trifle slower because imagex insists on doing image verification when saved to a network share. (Which is probably a good idea.)
2. Once you're sure that the drive's ready to work, then slip the WinPE CD into the PC's optical drive and reboot the computer using the WinPE disc. You've got to do this because it's quite hard to copy all of an operating system's files while that operating system's running — sort of like trying to change an engine while a car's running. So, by booting from WinPE's very simple — but complete enough for our purposes — operating system, then we allow all of your Vista box's files to remain safely asleep while we copy them to the new WIM file.
3. Once WinPE's booted then you'll see the simple interface that we met before in the earlier WinPE article:

# Working with ImageX and Windows Image Files

By Mark Minasi



4. If you're not using an external hard disk and are instead using a network share, then now's the time to do the net use command to map that network share to F:.
5. Type: **imagex /capture c: f:\ntest.wim "Newsletter test" /compress none**

While you're waiting the half-hour or so for ImageX to pack up your system into a WIM file, let's look at those options. First, there's the "/capture" option; that's the one that says to put a computer's operating system in cold storage. ImageX is supposedly able to do this not only with computers running the Vista OS, but also XP, 2003 and 2000, although honestly I've never had time to try those options.

The "c:" and "f:\ntest.wim" tell ImageX what drive the OS to image is on, and where to store the resulting WIM file, as well as what to name it. One of the neat side effects of working with an image technology that works on files and folders rather than sectors is that you could, if you wanted to, have typed

**imagex /capture c: c:\ntest.wim "Newsletter test" /compress none**

Notice the difference: in that case, we'd actually be telling ImageX to image the C: drive onto itself! The "Newsletter test" is required and is a descriptive blurb about the image. You can see that information by typing (once the imaging's done)

**imagex /info f:/ntest.wim**

# Working with ImageX and Windows Image Files

By Mark Minasi

That will cause ImageX to spit out a bunch of XML describing the image, one of the first parts of which will be

```
<NAME>Newsletter test</NAME>
```

Finally, the /compress none says not to apply any compression algorithms to the WIM, something that I did mainly to speed up the whole process. You can also specify fast and maximum instead of none.

## Using ImageX To Apply a WIM

Congratulations, you've created your first WIM! But where to deploy it? Well, inasmuch as I don't have another test machine around, I'm just going to wipe my test machine's hard disk clean, which will enable it to look pretty much like an empty test machine.

You'll still have that WinPE command prompt window open — if not, then just re-boot the test machine from the WinPE CD-ROM — so you need only type

```
format c: /q /y
```

And press Enter, and in a minute or two your test machine's hard disk will be wiped clean. Or, just for the sake of completeness, let's do what we'd normally do with a new system that we were about to wipe and re-image. Again, we'd start from WinPE, but now let's re-partition and format. (Let me repeat that. We're about to wipe all of the data off of the hard disk of your test machine or, rather, make that data very, very hard to recover. If you've got some information on there that you can't part with, then please either back it up and then do this, or just wait until you've got a hard disk that you don't mind nuking.) With the WinPE command prompt up, type

```
diskpart
select disk 0
clean
create partition primary
assign letter=c:
active
exit
format c: /q /y
```

Diskpart is the Vista version of what used to be FDISK and appeared, if memory serves, back in Windows 2000. You must select a disk even if it's the only one that you have because, um, I guess Diskpart's not that smart. "Clean" wipes the master boot record, deleting any pointers to any existing partitions and making getting to any data that was on the hard disk very, very hard to do. "Create partition primary" and "assign letter=c:" should be self-explanatory. "Active" marks the new partition as the one to boot from, and "exit" exits Diskpart and returns us to the C:\> prompt. Once there, we can format the newly-partitioned drive C:.

Once that's done, we can then move our ntest.wim image onto our newly-cleaned C: drive with this command:

```
imagex /apply f:\ntest.wim 1 c:\
```

# Working with ImageX and Windows Image Files

By Mark Minasi

After about 10 minutes and some very interesting messages — my favorite is "SACL is going away" — the command prompt will return. You can then reboot the system and your machine now runs Vista once again.

We've seen, then, how to use imagex to create ("capture") and deploy ("apply") Windows images. ImageX works well, but clearly there's more to the story. We wouldn't be very successful deploying ntest.wim to more than one computer because we'd end up with a lot of computers all bearing the same SIDs. How to address that?

The answer is to add a step before imaging the computer, one familiar to anyone who's done sizeable rollouts before: Sysprep. As with earlier versions of Windows, Sysprep rips off the SIDs, machine name and similar attributes of an operating system, leaving one that has been (in Sysprep terms) "generalized." Once generalized, we can then ImageX the computer and deploy that generalized WIM to any number of machines. There are, however, a few wrinkles in making that work and unfortunately I'm out of space for this newsletter. In the next one, I'll discuss running Sysprep — which hasn't changed all that much from XP SP2's Sysprep — and how to automate the "mini-Setup" that Sysprepped machines require — which *has* changed quite a bit. See you then!