



Reliably recovering evidential data from Volume Shadow Copies in Windows Vista and Windows 7

James Crabtree

Senior Forensic Computing Specialist at QCC Information Security

Gary Evans

Senior Forensic Computing Specialist at QCC Information Security

Abstract

Since the release of Windows Vista and more recently, Windows 7, analysts are encountering increased numbers of relevant artefacts in the volume shadow copy files present in the System Volume Information folder. The paucity of published data regarding the structure and architecture of these files has frustrated analysts attempting to provenance the artefacts encountered. This paper introduces a new method for reliably recovering the original files stored in the volume shadow copies which contain these artefacts. This new technique maintains the original date and time stamps and is flexible enough to extract everything in a VSS file, or right down to just one file for one specific user account. This technique is fast, proportional and employs standard tools found in any reputable digital forensics laboratory.

1. Introduction

The technology employed by Microsoft which underpins the Volume Shadow Copy system, is called “block level copy on write” and first appeared in NAS boxes and SAN systems over ten years ago. The Volume Snapshot Service (VSS) is used by Windows Vista and Windows 7 to enable restore point functionality and also for the previous versions feature, allowing single files to be rolled back to an earlier time. VSS works at the file system level and is highly integrated with NTFS.

When a disk write is about to occur to a disk which is protected with VSS, the existing 16KB sector of data is copied from the disk and written to the volume shadow copy file. Then the new data is written to the disk by NTFS. This process will copy a full 16KB block of data to the shadow file, even if only a few bytes are actually being written to the live file on disk.

The volume shadow copy file is indexed using an unknown method and this appears to be held in an external file to the shadow copy itself. Analysis of the shadow copy file indicates that it is simply a concatenation of all the 16KB blocks which have been copied out over the lifetime of that particular shadow copy file and has no other set structure.

Whilst the System Volume information folder may have several shadow copy files present, each identified by their GUID filenames, they may not be accessible to either the user or the analyst unless they have been finalised. This is the process of finalising the indexes used to reference the data held in the shadow copy. Clearly, with an allocation of up to 15% of the volume size in Vista and between 3% - 5% in Windows 7, there can be a lot of data present in shadow copy files which provide an excellent secondary source of artefacts during a forensic investigation.

2. Implementation

The Volume Snapshot Service is switched on by default and will produce shadow copy files in all SKUs of Vista and 7. However, users of Home Basic and Home Premium versions will not have access to shadow copy restore functionality. This is reserved for Professional, Business, Ultimate and Enterprise versions of the operating systems.

Shadow copies are created daily and whenever new hardware or software is installed. Users can initiate the creation of restore points manually via a GUI dialog. When a new shadow copy is created and VSS determines that space limitations will be exceeded, old shadow copies are discarded and these blocks move into the pool of unallocated space. This can explain why identical artefacts are found in unallocated as well as live shadow copies, although this is possibly better explained by multiple updates occurring to a single block.

The Registry holds two useful keys which should be considered when relying on evidence from shadow copies:

```
HKLM\SOFTWARE\Microsoft\WINDOWS
NT\CURRENTVERSION\SPP\Clients\
{09F7EDC5-294E-4180-AF6A-FB0E6A0E9513}
```

and

```
HKLM\System\CurrentControlSet\Control\
Backup Restore\FilesNotToSnapshot
```

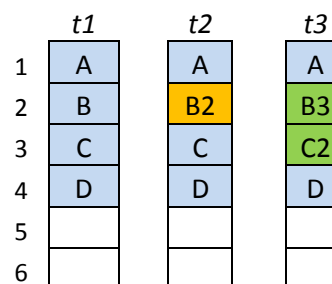
The first key records which volumes are currently monitored by VSS, whilst the second stores the names of files which should not be backed up – examples include the Windows Desktop Search database (Windows.edb), the system paging file and the hibernation file.

It is worth taking a few moments to describe in more detail the “block copy on write” theory that was mentioned in the introduction. If we consider that a brand new and empty shadow copy file has been instantiated and that all subsequent writes to the disk will be recorded in this shadow copy file, then an example can be examined that clearly sets out the process.

It must be born in mind that VSS works closely with NTFS and monitors all disk writes, both system and user, so all files can be potentially rolled back, documented exceptions notwithstanding.

The graphic in figure 1 describes the evolution of a single file, comprising four 16KB blocks over three time intervals, $t1$, $t2$, and $t3$, where $t3$ represents the state of the file system now.

Live File System



Shadow Files

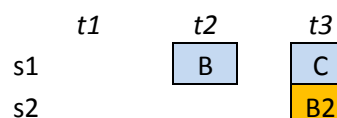


Figure 1. Simplified Block Copy on Write example

The shadow copy file s1 is empty at the point the four block file is created. At $t2$, the user alters a few bytes in the file which reside in block B. VSS intercepts the write to this block and copies block B into the shadow copy file at s1.

NTFS then carries on and writes the updated data to the users file, represented here as B2. At some point later ($t3$), the user then makes further changes to data in blocks 2 and 3 of the file. Again, VSS intercepts the writes and copies out the existing data at C and B2 to the shadow file and then allows NTFS to write the new data, B3 and C2 to the users data file.

By combining the live user data with the data held in the volume shadow copies, it is possible to now read the file as it was at any point; $t1$, $t2$ or as it is now at $t3$. This illustrates a fundamental point about VSS – which is that the backup data held in the shadow copy is not a *complete* copy of the file at any point, only the data which will be changed is kept. Thus, the existing file must be present in order for VSS to apply the deltas and effectively roll the file back.

It is for this reason that the shadow copies themselves are resistant to structural analysis, they are simply a collection of 16KB blocks of data which bear no relationship to each other. Further, the index files which appear to manage them are little more than collections of arithmetic pointers, making analysis of them a long and slow process. Microsoft have refused to release any architectural detail regarding the structure and relationships between the files to prevent 3rd party developers from creating applications that interact with VSS in an unapproved of manner.

3. The Technique

To recover data from the shadow copy file, the following tools will be used:

- EnCase with Physical Disk Emulator (PDE)
- Windows 7 Professional (live or VMWare)

For the purposes of this whitepaper, it assumed that Windows 7 is only available within a VMWare environment – although if an actual Windows 7 workstation is available it can be used just as effectively.

The process in overview is to:

1. Mount the **logical disk** containing the target shadow copies using PDE within *EnCase* (other mounting options like *Mount Image Pro* may work, but have not been tested).
2. Add this **logical** disk as a new disk to a standard *Windows 7* install in *VMWare*.
3. Boot the VM and mount the shadow copies using *VSSAdmin* and *mklink*.
4. Extract the data from the mounted shadow copies using *RoboCopy* to an evidential disk.

There are a few minor points where errors can be introduced which will cause the process to fail – these will be highlighted where necessary. This technique will operate on all of the available shadow copy files, but it works just as well for a single shadow file.

Mount the logical disk volume with PDE in *EnCase*. This step is critical – do not mount the *physical* disk, it must be the *logical* volume, typically this is represented within *EnCase* as drive letter “D”. On a standard Vista/*Windows 7* installation, *EnCase* will usually report the 100MB boot manager partition as drive “C”.

In the mounting options window presented by *EnCase*, uncheck the “disable caching” option and provide a path for the write cache as shown in the screen capture at figure 2. Take note of the physical drive number that *EnCase* allocates to this logical volume – this will become important in the next step. Once the logical disk volume has been emulated as a physical disk to the operating system by PDE, edit the settings of the *Windows 7* virtual machine that will be used to carry out the analysis.

The virtual machine is not a clone of the suspect’s machine, nor is there anything particularly special about it. It should be a standard installation of

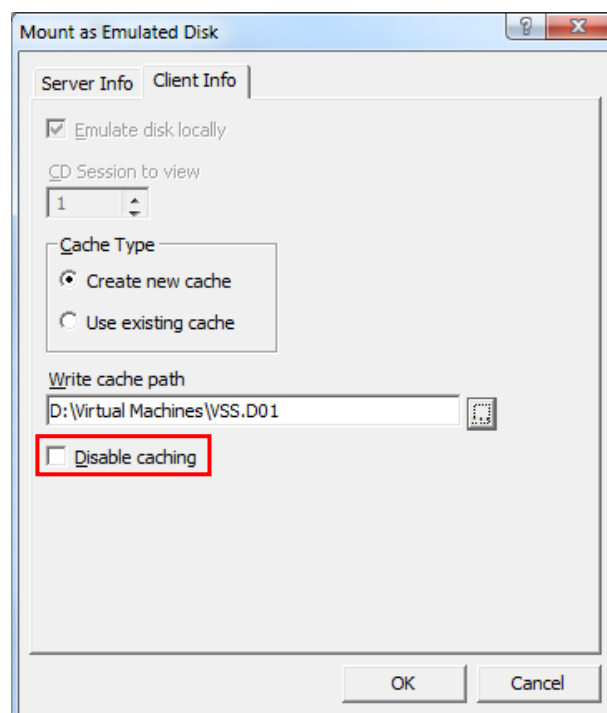


Figure 2. Enabling the write cache within *EnCase*

Windows 7 Professional so that VSS tools are available (Ultimate or Enterprise will also work) and access to an administrator privileged account is also required. Edit the settings for this virtual machine and add a new hard disk, selecting the option to “use a physical disk (for advanced users)” – click next and select the correct physical drive reported by PDE when mounting the logical disk volume within *EnCase*.

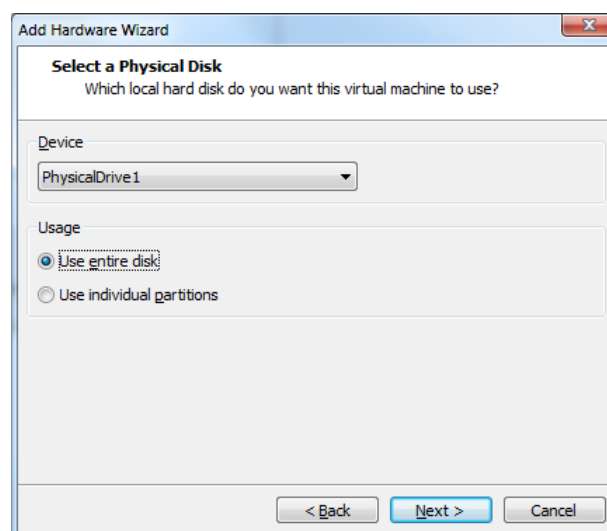


Figure 3. Adding the physical disk to *VMWare*

Now the *Windows 7* virtual machine can be booted as normal. Login using an account which has administrator privileges and open a command prompt window. Start *Windows Explorer* and note

the drive letter that Windows 7 has allocated to the emulated logical volume of the suspects computer. Using the *vssadmin* command, the full listing of shadow copies available for our emulated drive can be examined. This will provide a long listing of all the volume shadow copies present on

```
c:\>vssadmin list shadows /for=d:
```

the drive letter described in the “/for=” parameter. A typical listing showing a single VSS file is shown in figure 4, below:

```
c:\>vssadmin list shadows /for=d:
Contents of shadow copy set ID: {1472c511-1ca0-480a-bd67-18a36f177b86}
Contained 1 shadow copies at creation time: 10/03/2010 22:40:08
Shadow Copy ID: {454d0cef-af5e-4dac-a779-73c2ffe51ca2}
Original Volume: (C:)\\?\Volume{de70417a-2bb7-11df-8864-806e6f6e6963}\
Shadow Copy Volume: \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1
Originating Machine: Windows7.forensics.qccis.com
Service Machine: Windows7.forensics.qccis.com
Provider: 'Microsoft Software Shadow Copy provider 1.0'
Type: ClientAccessibleWriters
Attributes: Persistent, Client-accessible, No auto release, Differential, Auto
recovered
```

Figure 4. The output from the “vssadmin list shadows” command

There are a number of very useful items of information provided by this output, these have been highlighted and are described below:

At the top, is the date and time that this shadow copy was created. This is very useful for identifying exactly which shadow copy file is of interest when many are available. Directly below is the “Shadow Copy ID” – this unique number authoritatively identifies the shadow copy file and can be verified within EnCase. Select the shadow copy file, and in hex view, swipe 16 bytes at file offset 144. Bookmark these as a Windows GUID and the shadow copy ID for that file will be visible.

Note in the example above, that the command lists the shadows for the “d:” drive, yet in line four of the listing, the original volume is reported as “c:”. This reflects the drive letter assignment at the time the shadow copy was created and is exactly as expected. Lastly, we have the shadow copy number. In this example it is 1, however it is rare that the numbering displayed begins at 1, simply because the Windows 7 machine being used to display the shadows on the mounted disk will also be maintaining its own shadow copies. This can be verified by executing the *vssadmin list shadows* command without the drive qualifier on the end, this will list all shadows on the system, from all drives.

Scroll down to the details of the last shadow copy from the initial listing carried out using /for=d: and

note the shadow copy number. On the test system used for the screen captures for this paper, it is 23, however it will vary greatly depending upon the configuration of the suspects drive and the Windows 7 computer being used to analyse it. Note also that analysing the same suspects drive on different occasions is highly likely to produce differing start and end numbers – this is again because the Volume Shadow Service running on the analysis computer will be maintaining its own shadows and these numbers may have changed since the computer was last used to examine a mounted disk.

With the beginning and end numbers of live shadow copy files from the mounted suspects disk available, it is possible to now mount these using the *mklink* command. This Windows command allows a hard link, like a reparse point or directory junction to be created. These are very different from shortcut files, or soft links which are files in their own right, which point to a target file or folder. By creating hard links to each shadow, it becomes possible to access the shadows contents with more useful tools, allowing VSS on the VM to do all the heavy lifting of rebuilding each file on the live file system whilst overlaying the 16KB blocks present in the shadow file.

Using this established technique, it is possible to ignore the architecture and get straight to the artefacts that are of interest. A “for” loop is used to automate the process of linking each shadow.

```
for /l %i in (start,1,stop) do mklink /d c:\rp%i
  \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy%i\
```

The loop shown above will “do” the “mklink” command, substituting the %i variable with the value in *start*, incrementing by 1 each time and finish when it reaches the value held in *stop*. It will make the hard link and call it c:\rp%i. This is better explained with an example.

If the first shadow is 2 and the last is 3, then the command would be as shown in figure 5, below.

This loop starts at 2, increments by 1 and stops at 3 so that two shadows will be mounted, one each at c:\rp2 and c:\rp3.

```
c:\>for /l %i in (2,1,3) do mklink /d c:\rp%i
  \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy%i\

c:\>mklink /d c:\rp2 \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy2\
symbolic link created for c:\rp2 <====> \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy2\

c:\>mklink /d c:\rp3 \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy3\
symbolic link created for c:\rp3 <====> \\?\GLOBALROOT\Device\HardDiskVolumeShadowCopy3\
```

Figure 5. The output from the mklink loop command for 2 shadow files

With these shadow files now mounted, it is possible to examine them with forensic tools more familiar to the analyst – for example a logical acquisition could be made of the mount points using FTK Imager. So far, nothing new has been presented. All of the techniques shown have been discovered and documented by others such as *Dave Titheridge* of North Wales Police and *Richard Drinkwater* from the Sausage Factory Blog. Where existing techniques have caused issues involves the time and disk space taken to image each of the

shadow files that are now mounted. If a suspect’s disk is 100GB in size and there are 20 shadow files present, there is potentially 2TB of data to be extracted, imaged and analysed. The vast majority of which has little or no relevance to the investigation.

This is where the use of the Windows *RoboCopy* command is particularly helpful and where this whitepaper breaks new ground. The command to use is shown below:

```
robocopy c:\rp2\Users\james z:\Shadows\rp2\Users\james *.jpg /S /COPY:DAT /XJ /w:0 /r:0
```

This command, will take its root source as “c:\rp2\Users\James” and copy out the specified files to the target folder at “z:\Shadows\rp2\Users\james”. It will target all jpg files and will recurse subfolders. The copy:dat parameter will ensure all metadata associated with each file is preserved (timestamps), the /xj ignores any directory junction points and the w:0 and r:0 parameters define that in the event of an error, the wait time is zero seconds and to retry zero times.

one of them, the others can be ignored. This results in a lot less data to analyse later. The structure of the command means that if all the files for all of the users are required, then the path can be truncated to end at “users” and *.*be used for the filename parameter. Multiple file types can be defined, delimited by a space. What this really means to the investigation is that all of the irrelevant data present in the shadow file can be bypassed. On the other hand, if all that is required are the prefetch files or the Registry hives, then these can be focussed on, accurately and precisely.

What this means in real terms is that when the command is complete, the folder at z: will contain a copy of all .jpg files present in the user’s profile that are present in the shadow copy. The power of this command comes in its proportionality or focus. If there are three user accounts on a target system, but the investigation is only focussed on

Implementing this command in a loop to process all of the shadow files available is a simple process shown in figure 6, as is getting all of this extracted data back in to EnCase in a logical evidence file so that it can be analysed. This is covered in the next section.

```
for /l %i in (2,1,3) do robocopy c:\rp%i\Users\%user% z:\Shadows\rp%i\Users\%user% *.jpg  
*.bmp *.png /S /COPY:DAT /XJ /w:0 /r:0
```

Figure 6. The loop version of the robocopy extract command

Despite the overwhelming urge to do so, it is important that the extracted data present in the target folder, z in this example, is not navigated to or browsed. This may alter the timestamps and modify the internal metadata for some file types, for example; Microsoft Office documents. These files can be copied into a new EnCase case file as single files and then a logical evidence file created. It is important to create the logical evidence file, not just from a sound procedural point, but also because errors sometimes occur using this process when analysing the single files within EnCase prior to creating the logical image.

Before shutting down the Windows 7 virtual machine, it is important to back out the steps carried out to ensure the VMware image is clean for the next set of shadows to be examined. The following loop command will remove the hard links created and ready the VM for a clean shutdown.

```
for /l %i in (2,1,3) do rd c:\rp%i
```

Once the VM is shutdown and the logical evidence file created and loaded back into EnCase, be sure to remove the physical drive added to the VM and then to cancel the PDE emulated logical disk.

The complete files present in the new logical evidence file can now be examined for keywords, scripts run across them and far more detailed Internet history analysis backed up with the contents of the users cache can be carried out. There will be some degree of duplication present in the event that multiple shadow copy files have been processed. It is important to remember that physical sector number is of no help in determining these duplicates, only filename, hash value and path are really of any great assistance. An EnScript is available by request for this purpose.

4. Considerations

There are a few things to consider when dealing with data recovered from shadow files – this is especially true in child protection cases where the quantity of indecent material present may be a distorted view of reality. Take for example, a thumbnail cache. A thumbnail file contains

multiple pictures – if 1 picture changes, the whole 16KB block containing that picture will get copied into the shadow file. If that block contains 10 pictures, there are now the 10 original pictures in the shadow file, plus the 9 original pictures in the live block, plus the 1 new picture, giving a total of 20 pictures resulting from a change to only 1. It is clear that the data being examined must be taken in context and not simply at face value.

Another point to consider is that VSS will not copy out file slack for a file. If a file is 17KB long, VSS will reserve 32KB for it. Data found in the VSS using keyword searches may be in the remaining 15KB of file slack and this will not be copied out. Put another way, VSS only deals with logical file lengths, not physical ones.

5. Analogy for court

When considering how to present what is a very complex architecture to a non-technical audience, the following analogy seems appropriate:

The VSS system handles files much like a paper based mail order catalogue:

- The first edition is received – this is the equivalent of the original file.
- Price changes occur and entire new pages are received, despite there possibly being changes to only 1 item on the page.
- The old pages are removed from the catalogue and replaced with the new ones. The old pages are saved. This is where the Shadow copy is made.
- Each batch of page updates comprises a single Shadow copy file and any grouping of them can be rolled back, mimicking a single file or whole folder restore.

There are now 2 ways to look at the catalogue:

1. With new pages inserted – this is the live, up-to-date view, or;
2. As originally sent with the new pages replaced by the original pages again.

This analogy works very well and despite various attempts at rebuttal, it stands as an effective way of explaining shadow copy technology to a jury.

6. Summary and Conclusions

Volume shadow copies are a very new technology and forensic analysis of them is evolving almost weekly. Much of the inner workings are currently unknown and the implications for users and analyst alike have yet to be determined. At the time of writing, shadow copy based evidence remains untested in court and it will be interesting to see how the various parties present their arguments for and against.

In determining a reliable way to pragmatically recover usable and useful data from shadow copies without generating enormous amounts of data, several of the early steps presented here are based on the previous work by others. The *RoboCopy* aspect of the solution is our addition to this gradual easing forward of the boundaries which currently surround the forensic analysis of shadow files. No doubt work will continue and a new utility will soon be developed making all of this redundant!

To finish, there are some important points that are worth re-iterating, just to ensure the concepts are understood.

VSS covers all files on the disk, not just the system areas as Windows XP did. Volume shadow copies are switched on by default and are beyond most user's ability to influence or control. This includes programs like Evidence Eliminator and the like. Microsoft have done a good job in protecting the Shadow files from this type of tampering so the data should be well preserved for the analyst. Shadows are rich in forensic artefacts which allow us to provide accurate provenance to files that previously would only be present in unallocated clusters, probably in non-sequential data runs.

The problems associated with carving files in non-consecutive data runs is elegantly handled by the VSS infrastructure – providing a robust and effective way to access both user and system data.

Already analysts at our laboratory are finding evidence which is key to investigations and are able to accurately provenance their source.

Shadow copies are not difficult to process once the procedure has been completed a few times.

7. References

David Titheridge: *North Wales Police*, Cranfield MSc dissertation into VSS Operation.

Richard Drinkwater: *The Sausage Factory Blog*, the forensic one stop shop for all things VSS.

Samantha Bennett: *Oceanic Systems*, For an effective analogy.

George M. Garner Jr: *Forensic Acquisition Utilities*

Troy Larsen & Harlan Carvey: *Presentation to the Digital Crimes Consortium 2009*.

Rob Lee: *SANS Computer Forensic Blog*