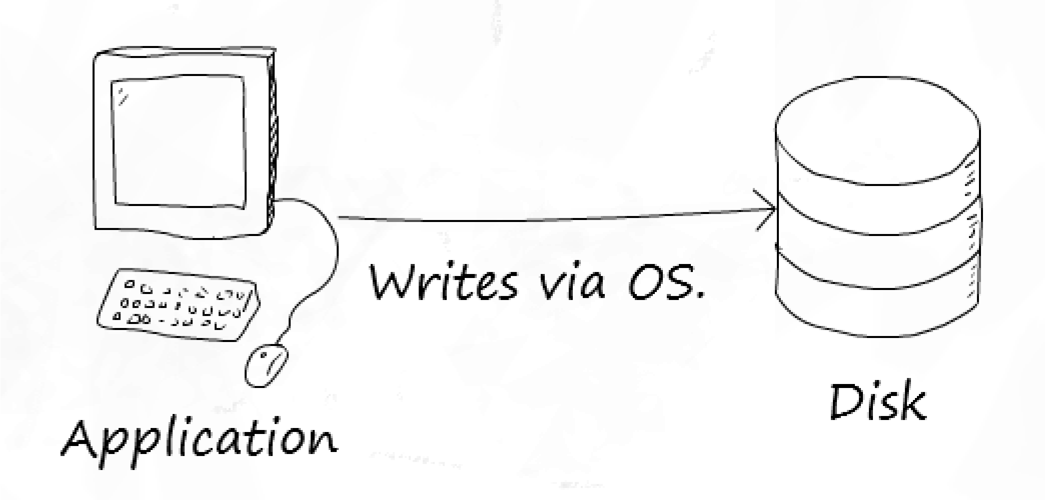
What is VSS, how does it work and why do we use it?

Backup Internals: What is VSS, how does it work and why do we use it?

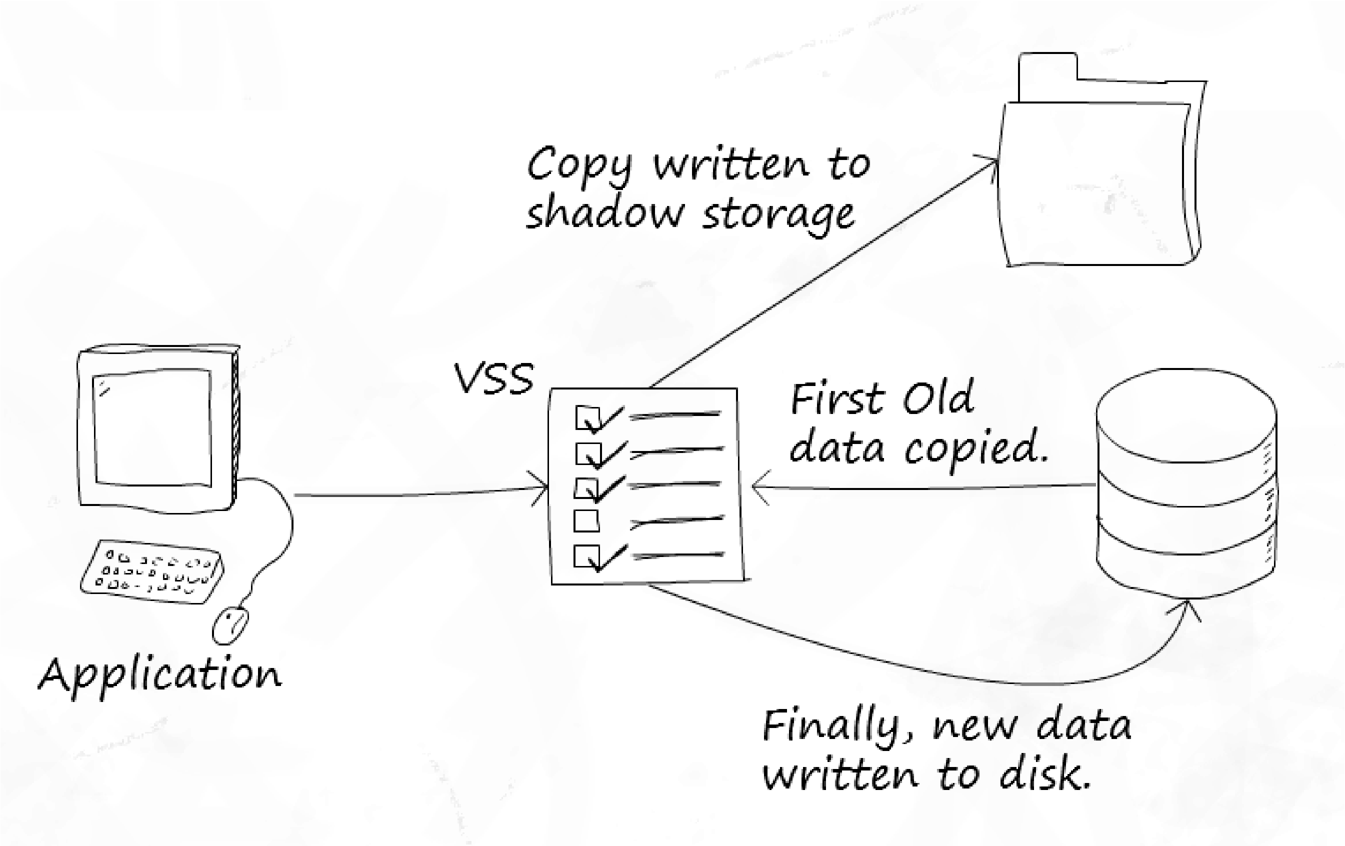
As happens sometimes, our more technically minded customers question a system that “just works”. We like these sort of customers, as they keep us on our toes. In this specific case, the question was is it safe to continue using my computer during imaging and the answer is yes, although the longer explanation is much more involved. After the ensuing discussion, one of our community members suggested we write this information up as a FAQ entry somewhere and internally we felt it was worth preserving the information from that thread, so this blog post is an extended, polished version of our notes there.

Reflect can take an image of a live system. How does this work?



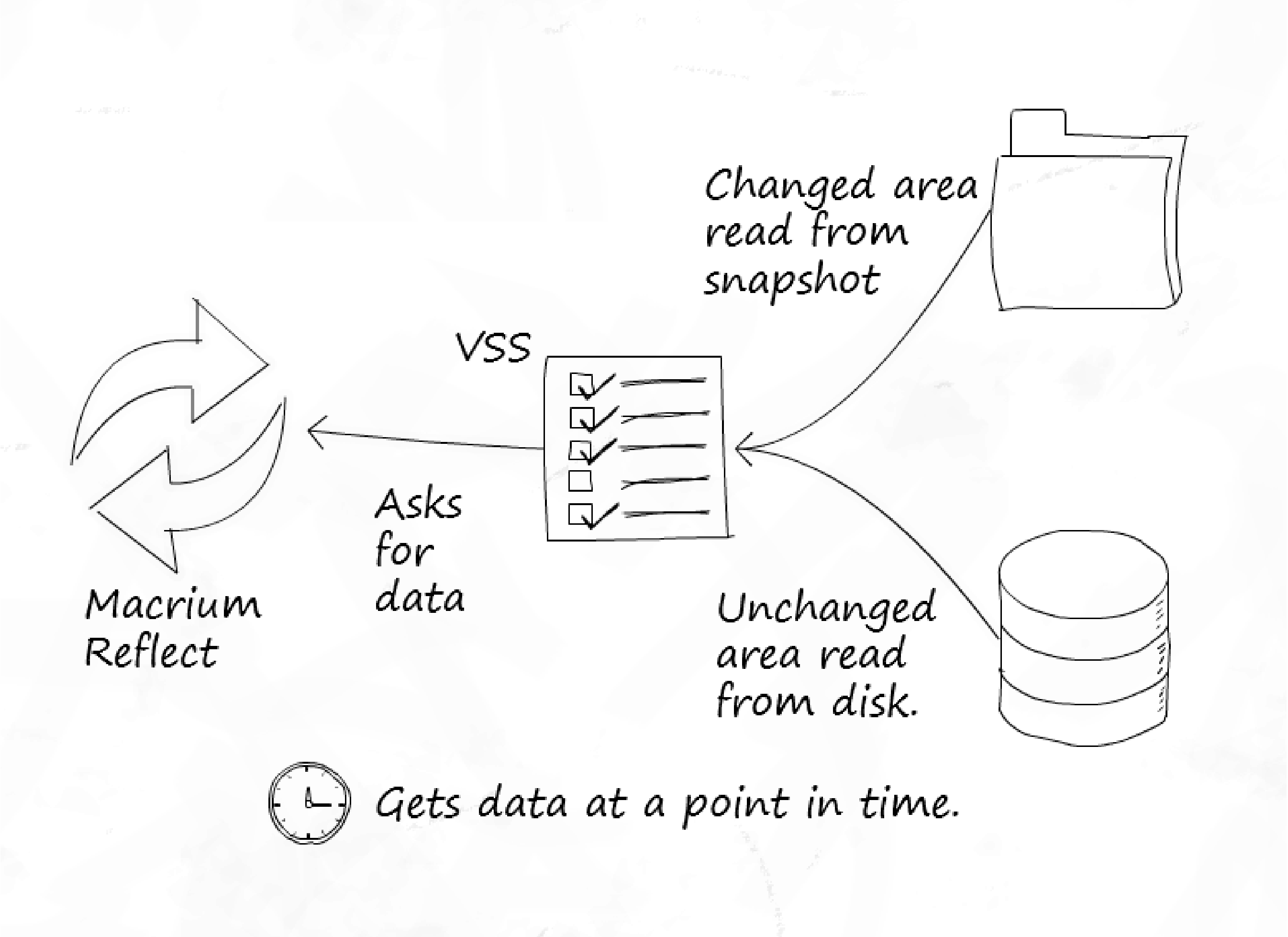
A simple disk write

Internally Macrium Reflect relies on a Microsoft Windows component called VSS, which stands for Volume Shadow Storage. Microsoft’s VSS operates by taking what is called a copy on write snapshot of your system. This allocates a small temporary storage space. Then, every time you write to a part of your disk, the information on the disk is first copied to the snapshot before allowing the write to take place.


Copy written to shadow storage
VSS
First Old data copied.
Application
Finally, new data written to disk.

VSS in use

This technique makes VSS quite efficient. A snapshot only contains as much data as has changed since the snapshot started – you do not need an entire copy of your disk. Also, writes are only affected for used space – free space does not need to copy anything, as there is no original to preserve.


Changed area read from snapshot
VSS
Asks for data
Macrium Reflect
Unchanged area read from disk.
Gets data at a point in time.

Reading data

However there is another side to this advantage. The more writes made after the snapshot springs into existence, the more VSS must store. So if you perform an excessive amount of disk activity during imaging, the snapshot storage space may become large. VSS imposes a cap on this space, and should a snapshot attempt to exceed it, VSS will stop its copy on write behaviour and delete the temporary storage.

Once the snapshot is finished with, the space can be freed up again – so the temporary storage space is only needed during use.

What determines where this storage space is?

There is no magic to this – VSS chooses sensible defaults based on your available drives. This configuration can of course be altered and Microsoft provide a tool for this called VSSAdmin.

Is the data taken at a point in time really consistent?

This is a difficult question to which there is no straightforward answer. As you may have realized already, if a copy is in progress when the snapshot starts, only the completed part of the copy will be included in the snapshot. The rest of it will count as new writes and the old, inconsistent blocks will be copied to the snapshot!

To understand how VSS gets around this we need to understand a little more about VSS architecture so I'll introduce three terms:

- **Provider**: in VSS land, a provider provides the VSS service and is responsible for all the co-ordination elements. Essentially this is the "core" of VSS. Microsoft include a VSS Provider with Windows, and Reflect uses this. (Technically speaking, there are two components, the co-ordinator and the provider, however, for simplicity we have combined these for the purposes of explaining VSS.)
- **Requester**: a requester is, as you might expect from the name, an application that requests a snapshot to be made. This request goes to the provider.
- **Writers**: these components provide a mechanism for applications to be alerted to the creation of a snapshot, so they can prepare their data for snapshotting. VSS providers can notify VSS writers that a snapshot has been created, and the VSS writer can then perform the appropriate action. Once the snapshot has been taken, the provider again notifies the writers, so they may let the applications resume.

VSS Writers are an important part of VSS for certain environments. Imagine you are running a heavily loaded database server in production, or a virtual machine cluster. The database server will have open and be manipulating its database files, processing transactions etc. while the VMs are continually performing disk operations for the virtualised OS. Taking a point-in-time snapshot might be fine but, due to the amount of IO, it is likely to catch one or more of these services off guard in an inconsistent state.

The role of writers is to inform these applications a snapshot is about to be created. The application can then perform any tidy up operations necessary to ensure what is on disk is consistent. The snapshot starts and the applications are then notified they can continue. Now the on disk state should be fine.

When do I need to check my applications are VSS aware? I.e. when do I need VSS writers?

You need VSS writers for applications that perform large amounts of IO and depend heavily on the state of the files they are writing if you wish to back them up. Virtual machine disk images are a perfect candidate for this – the running image could easily become inconsistent.

Most applications do not write to disk in quantities of data large enough to be a problem. Many applications from Microsoft are also VSS aware, which helps greatly.

What about current and previous snapshots?

VSS is smart enough to keep track of its temporary storage location, and will exclude anything in it, including the current temporary storage area and any persistent snapshots that have been previously created. So when the data is read by your backup software, you only get what you need.

When is VSS in use and when is it not with regards to taking a backup?

Now a question you may not have thought of – when is VSS in use? Well, VSS solves the problem of imaging a live system neatly, so whenever you image your system with Reflect you are probably using VSS.

There is one exception to this – we have a fall-back mechanism for live versions of 32-bit Windows client operating systems up to Windows 7 (so XP, Vista and 7) called pssnap.sys. Reflect can use this on these systems if VSS is unavailable. pssnap provides the same copy-on-write technique as VSS, but with fewer extra features (pssnap.sys is not a VSS provider and is independent from VSS). Why have both systems? Well, VSS is a core component of Windows and available on every edition of Windows greater than XP SP2. We know it will be there as Microsoft are committed to this feature and it makes sense to use it as it is highly reliable. Before VSS we had no such mechanism so we built our own. It is equally as reliable but we felt the feature additions of VSS and the support for writers for enterprise scenarios really meant VSS made sense.

However in the rescue environment we do not use, or need to use, either pssnap.sys or VSS. Why? Because the system is not in use so the data should be perfectly consistent at all times. Of course we may well end up including persistent snapshots if they exist because without VSS they are not excluded.

You keep saying VSS is a Windows feature, yet only Reflect uses it…

Actually, that's not true. Windows itself uses VSS for system restore points! These system restore points are the persistent snapshots that might exist in your shadow storage.

That's it. If you'd like to read about VSS directly from Microsoft, you can read their article on TechNet.