COHESITY

Simple and Granular Data Protection for Microsoft SQL Server

Simple and Granular Data Protection for Microsoft SQL Server

Executive Summary

In enterprise environments, many mission critical workloads are dependent on SQL Server. As such, it is critically important to protect SQL servers against data loss, and against long term outages, both of which incur significant financial costs.

This paper explains why many backup applications only partially protect SQL Server against data loss. It also describes how Cohesity's technology provides comprehensive protection for SQL Server, while also allowing SQL Servers to be instantly recovered and brought back online in the event of a disaster.

The Requirements for Protecting SQL Server

As is the case for any other type of backup, organizations wishing to protect SQL Server must begin the process by establishing their data protection goals. One of the first steps in doing so is often to define the acceptable Recovery Point Objective (RPO) and Recovery Time Objective (RTO), and the required level of granularity.

The RPO generally refers to the frequency with which backups are created. For example, it was once common practice for organizations to create a nightly backup. In these situations, the RPO was approximately 24 hours, because one backup was taken per 24-hour period.

The problem with having such a long RPO is of course the potential for data loss. If a failure were to occur, then any data that has been created or modified since the completion of the most recent backup could potentially be lost. Hence, the longer the RPO, the greater the potential for data loss.

In recent years, many organizations have transitioned away from nightly backups to a continuous data protection solution. Although the actual mechanics vary by vendor, continuous data protection works by running very frequent backup operations, in which only the storage blocks that have been created or modified since the previous backup are written to the backup media. Although such backups are continuous, some vendor's solutions create recovery points that do not necessarily match up to the backup operation. For example, a backup application might create recovery points every few hours, even if data is backed up every five minutes. Unless such a backup application provides an option to create a synthetic recovery point on demand, the potential for data loss will be dictated by the frequency of recovery point creation.

Another metric that is commonly used when establishing backup criteria is RTO. RTO refers to how long it will take to restore a backup. The RTO is an important consideration, because there are very real costs associated with the outage of critical systems. The longer the RTO, the more money can potentially be lost while waiting for a restoration to be completed.

A server's RTO used to be measured in hours, or even days. Today however, instant recovery features (which will be discussed in detail later on) make it possible to recover from a failure almost instantly.

Although RPO and RTO are often the metrics that are used to describe backup requirements, granularity is an equally important concept. Granularity refers to the levels at which data can be restored from backup. For the sake of illustration, consider the limitations that existed in Windows Server Backup in Windows Server 2008. That version of Windows Server backup was capable of performing hypervisor (Hyper-V) level restorations, but lacked the granularity necessary to restore individual virtual machines.

When it comes to backup planning, it is necessary to determine the types of failures that could potentially occur, and then implement a backup solution that supports the level of granularity required for recovering from each of those failures. An administrator should not for example, be required to restore an entire virtual machine simply to recover a single SQL Server database. Similarly, a file level backup of a virtual machine will be of little use if the entire virtual machine has become corrupted at the hypervisor level.

In the case of a virtual datacenter in which SQL Server is running within one or more virtual machines, the following levels of granularity are required in order to achieve comprehensive protection:

- Host server recovery
- Virtual machine recovery
- File and folder recovery within virtual machines
- SQL Server database recovery

©2016 Cohesity, All Rights Reserved

Why VSS Backups Are Inadequate

Conventional wisdom has long held that backups of Microsoft server products should leverage the Volume Shadow Copy Services (VSS). When it comes to protecting Microsoft SQL Server however, VSS is incapable of providing comprehensive protection by itself. While it is undeniably important for a SQL Server backup solution to use VSS, other protective mechanisms must also be used.

The reasons why VSS backups only offer limited protection for SQL Server can be attributed to factors such as the way that backup applications usually interact with the server virtualization layer, and limitations to the VSS writer itself. In order to understand why these limitations come into play, it is necessary to examine the inner workings of VSS, Host level virtual machine backups, and SQL Server transaction logs.

SQL Server VSS Backups

Microsoft fully supports backing up SQL Server using VSS. In fact, ever since SQL Server 2005, Microsoft has included a VSS writer with SQL Server. It is worth noting however, that although this VSS writer is installed by default, some versions of SQL Server do not automatically start the SQL Server VSS Writer. It is possible to change this behavior by using the Service Control Manager (or PowerShell) to set the service's startup type to Automatic, as shown in Figure 1.

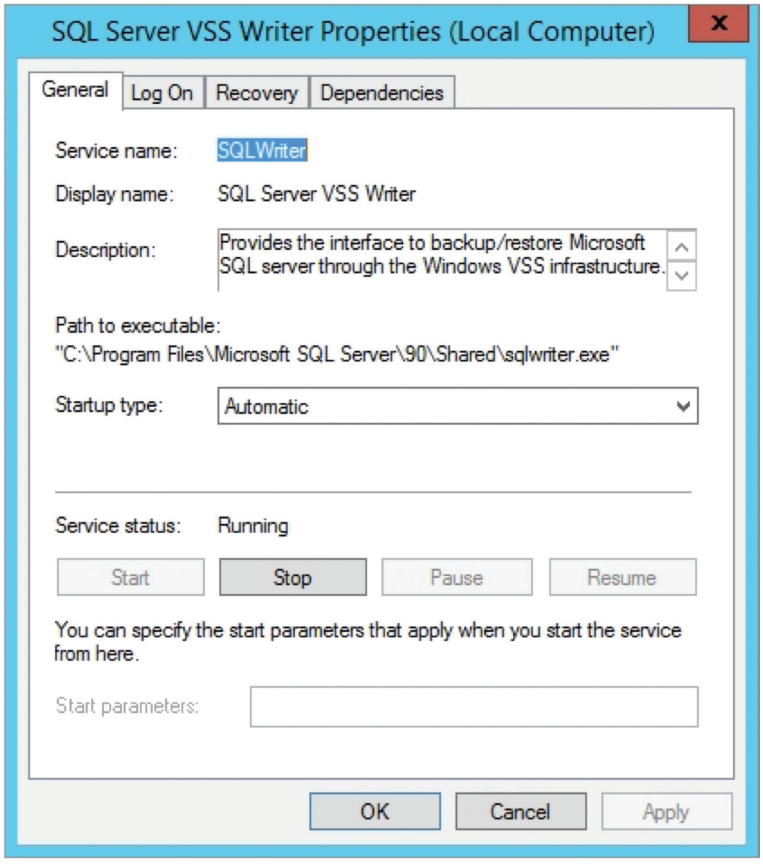

Figure 1: Data Protection for SQL Server

©2016 Cohesity, All Rights Reserved

Make sure that the SQLWriter service is running and that the Startup Type is set to Automatic.

Regardless of the backup application that is being used, the backup application itself functions as a VSS requestor. The requestor's job is to initiate the backup process. When backing up a SQL Server, the VSS requestor must begin by initializing the IVssBackupComponents interface. The requestor also instructs the IVssGatherWriterMetadata API to enumerate the server's metadata.

Next, the SQL Server VSS Writer connects to all running SQL Server instances, and collects the metadata for those instances. The VSS writer then sends that metadata back to the backup application, which then determines what needs to be backed up.

The next step in the process is the creation of a VSS snapshot. There are three things that occur during this part of the process. First, the SQL Server VSS Writer instructs the SQL Server to prepare itself to be backed up. This process puts the database into an application consistent state that is conducive to being backed up and restored. After that, the database is momentarily frozen. Doing so temporarily suspends write I/O to the database in an effort to prevent the database contents from being modified during the snapshot. Although suspending database I/O might seem counterproductive, this process has a very short duration and is not usually disruptive to normal database operations. Once the snapshot is created, and the database is thawed, which allows normal I/O to resume. The snapshot contents can then be safely written to the backup application.

The Server Virtualization Layer

Although the SQL Server VSS Writer is the primary mechanism used when backing up SQL Server, it is necessary to consider other architectural elements that may exist. Specifically, SQL Server is often installed within a virtual machine. Although it is possible to perform guest level virtual machine backups that mimic backing up a physical SQL server, most organizations opt to back up their virtualization infrastructure at the host level, because host level backups are easier to manage. Doing so however, adds a layer of complexity. Host level backups typically do not install backup agents into the individual virtual machines. This means that if a backup application is to back up a virtualized SQL server at the host level, then the entire backup process must be initiated, coordinated, and performed from outside of the virtual machine.

In these types of situations, the backup application still leverages VSS. However, the process usually leverages multiple VSS writers that must work in a coordinated manner to protect the virtual machines and the applications within them.

The specific details of how this process works vary by hypervisor. Some hypervisors, such as Microsoft's Hyper-V, include their own VSS writer. Just as the SQL Server VSS writer momentarily freezes storage I/O so that a snapshot can be taken of a SQL Server database, a hypervisor level VSS writer can temporarily suspend storage I/O so that snapshots can be taken of virtual machines, thereby allowing those virtual machines to be backed up.

Although using a VSS backup in this way will allow point in time image backups to be created for virtual machines, the hypervisor level VSS writer is unable to look inside of the virtual machines (at least not without assistance) in order to determine which applications are running on the virtual machine. As such, a hypervisor level VSS writer (acting alone) would not be able to tell the difference between a virtual machine that is running SQL Server and a virtual machine that is running Exchange Server. This means that if a backup application had to depend solely on a hypervisor level VSS writer, it would be able to create an image backup of a SQL Server, but that image would only be crash consistent, not application consistent. Furthermore, such a backup would not support granular recovery within SQL Server.

In order to protect applications that run within virtual machines, the hypervisor must have a way to interact with virtual machines at the guest operating system level. Hypervisor vendors enable this functionality through a driver package that can be installed onto the guest operating system. VMware for example, enables this functionality through the VMware Tools, which you can see in Figure 2.

Microsoft provides similarfunctionality through the Hyper-V Integration Services. Although Cohesity does not support Hyper-V, Microsoft has done an especially good job of illustrating the relationship between the Integration services and the backup process, as shown in Figure 3.

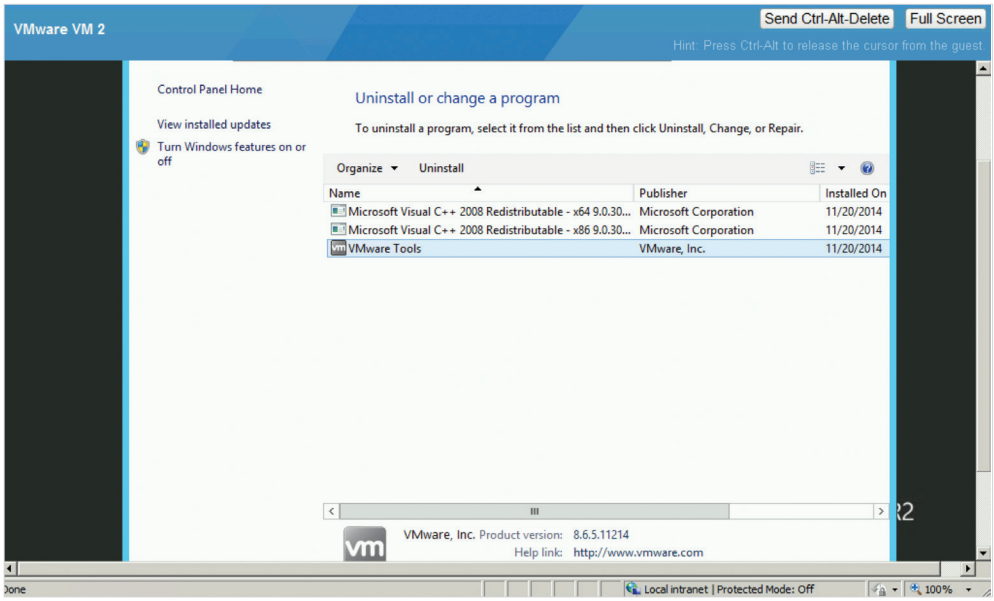

Figure 2: Data Protection for SQL Server

Installing the VMware Tools onto a virtual machine allows the VM to be better protected.

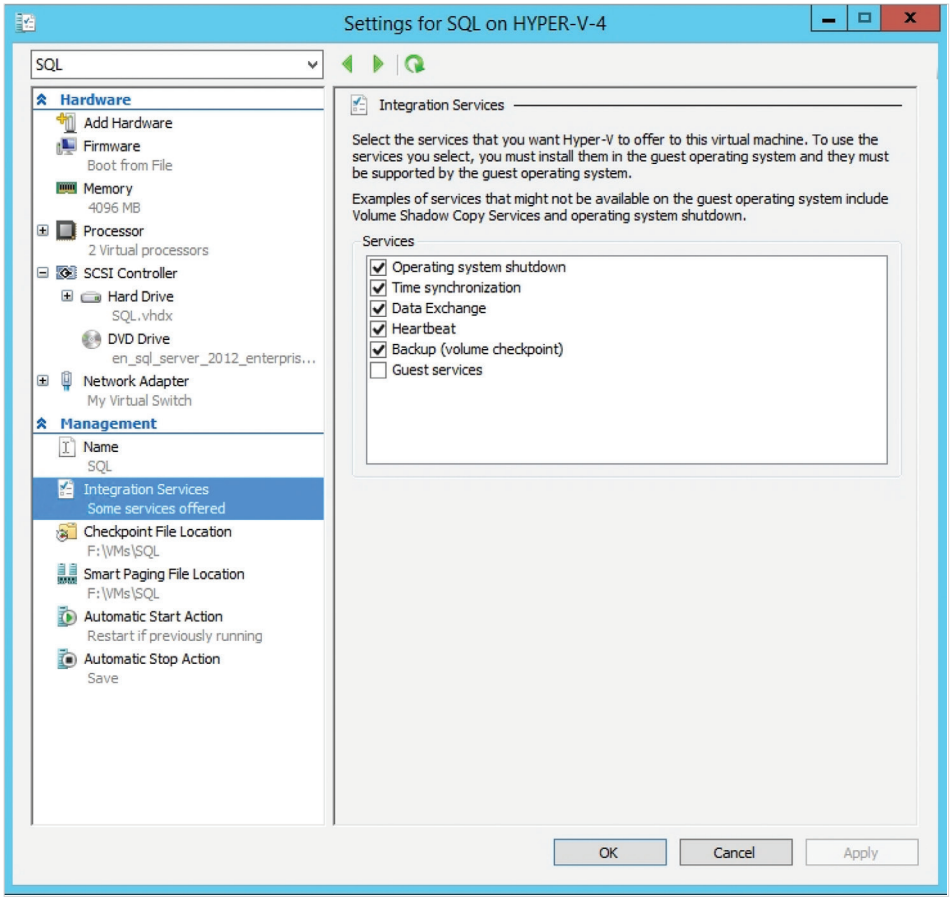

Figure 3: Data Protection for SQL Server

Microsoft's Hyper-V Integration Services are the Hyper-V equivalent to the VMware Tools, and include a backup component, which can be enabled or disabled separately from the other integration services.

Protecting SQL Server

As previously noted, host level backups of virtualized SQL servers will generate point in time image backups of the virtual machine and its contents. In order to ensure database integrity however, the backup application must also be SQL Server aware and support VSS-based SQL Server backups. The backup application must be designed in such a way that allows the hypervisor VSS writer and the SQL Server VSS writer to work together to create an application aware backup of virtualized SQL servers. Most of the backup solutions that are designed for use in virtualized environments do support VSS backups of the hypervisor, SQL Server, and other application servers.

SQL Server Transaction Logs

Although the SQL Server VSS writer does allow backup applications to create application consistent backups of SQL Server, the SQL Server VSS writer does not provide comprehensive protection by itself. The reason for this is because the SQL Server VSS Writer protects the SQL Server database, but not the corresponding transaction logs. In fact, SQL Server differentiates between database backups and log file backups, and even makes note of the last time that databases were backed up, and the last time that the log files were backed up. You can see an example of this in Figure 4.

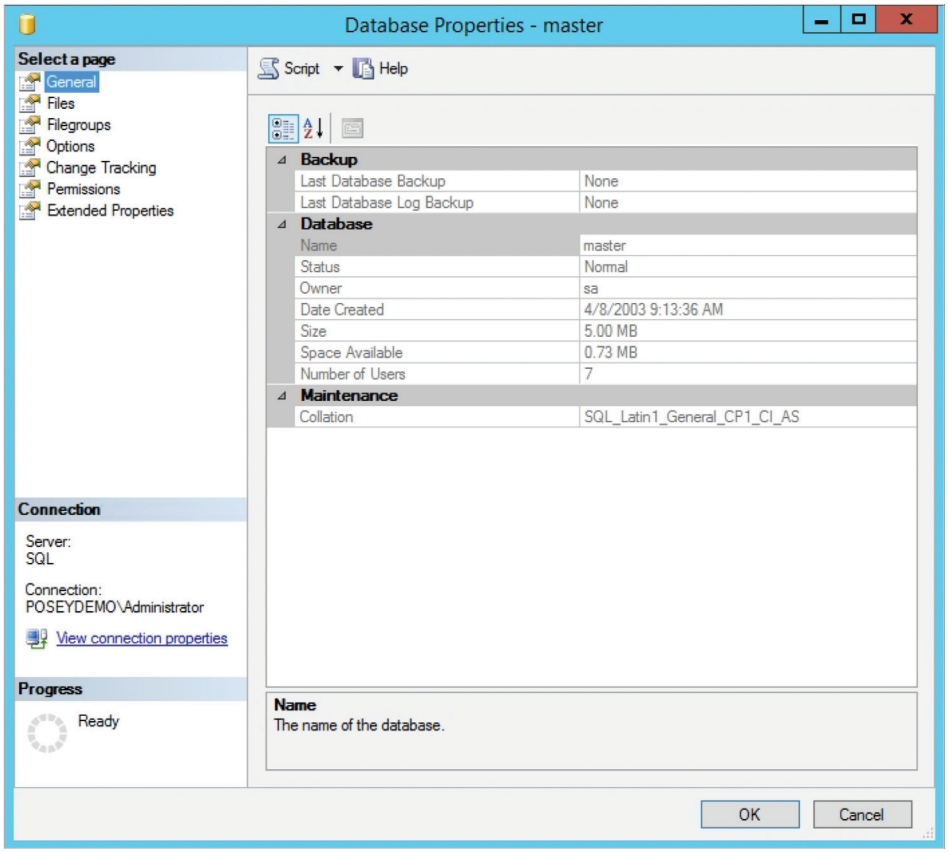

Figure 4: Data Protection for SQL Server

SQL Server keeps track of the last time that its databases were backed up, and the last time that the transaction logs were backed up.

Transaction logs are important, because they contain an intact record of the database transactions. With a couple of exceptions, every SQL Server transaction is written to the SQL Server transaction log store. The log store is collection of individual files called Virtual Log Files. SQL Server sequentially writes transactions to the virtual log files. Virtual log files have a fixed capacity, and when a log file is filled to capacity, a new log file is created.

It is worth noting that the log files behave differently depending on the recovery model that a database is configured to use. There are three different recovery models – Full, Bulk Logged, and Simple.

As its name implies, the full recovery model fully logs every database operation. As such, the log files will continue to accumulate as the database is used. Log files are eventually truncated (which reclaims storage space), but only after a database backup and a log file backup have been completed. The advantage to using full recovery mode is that it allows the SQL Server database to be recovered to a specific point in time, and it also supports database mirroring.

The Bulk Logged recovery model is similar to the full logged model, except that there are some types of transactions (such as rebuilding an index) that are minimally logged. Like the full logged model however, the logs are not truncated until a database backup and a log backup have been made. The bulk logged model is also similar to the full logged model in that it supports restoring a database to a specific point in time. The advantage that the bulk logged recovery model has over the full logged model is that it has the potential to reduce the volume of logging data that must be stored.

Like the bulk logged recovery model, the simple recovery model also logs all transactions. However, the simple recovery model does not allow for log file backups. The advantage to this is that log files can be truncated more quickly (when a checkpoint occurs), but the recovery options are much more limited due to the limited amount of logging data that is retained.

The takeaway from all of this is that unless a SQL Server is configured to use the simple recovery model, then a log file backup is also necessary, and not every backup solution is designed to back up the transaction logs. Although the simple recovery model is commonly used to circumvent the need for log file backups, the simple recovery model carries with it a risk of data loss.

Protection Through Deep Integration

Throughout the history of IT, backups have commonly existed as a collection of individual components that have been made to work together. A simple backup solution might for example include backup software that has been provided by a particular vendor, and a tape drive that was manufactured by a different vendor. Although this approach to data protection usually works, it is far from being optimal.

One of the major trends that has dominated corporate IT over the last few years is that of hyper convergence. The term Hyper Convergence commonly refers to hyper converged servers, but the concept of hyper convergence can also be extended to data protection in an effort to create an integrated backup solution.

The Advantages of Using an Integrated Solution for SQL Server Protection

The advantages to using an integrated backup solution are very similar to the advantages associated with using hyper converged infrastructure for server virtualization. The solution leverages performance matched hardware that has been especially chosen for its ability to effectively handle data protection requirements. Additionally, the backup application itself is specifically designed for use on the underlying hardware platform, and because the hardware and software are provided by a single vendor, there is a single point of contact for support.

The deep integration between hardware and software means that vendors who offer integrated backup solutions are able to design backup software that specifically leverages the underlying hardware capabilities. An especially good example of such a vendor is Cohesity (www.cohesity.com). Cohesity provides an integrated solution that is designed to achieve Web scale performance and scalability through its proprietary SnapTree technology.

As its name might imply, SnapTree is a snapshot mechanism that is used within the backup process. While the convergence of backups and snapshots is nothing new, Cohesity's approach is different from that of most other vendors.

Most of the snapshot solutions that are available today are based around the use of differencing disks, and are sometimes referred to as Copy on Write snapshots. When this type of snapshot is created, the software creates a differencing disk. All write operations are redirected to this differencing disk, thereby leaving the original media (typically a virtual hard disk) in an unmodified state.

There are a number of different problems associated with the use of differencing disk-based snapshots, but one of the primary issues is that of performance. The ongoing snapshot creation process can result in large chains of differencing disks being created. These differencing disk chains can negatively impact read performance, since the software may need to query multiple differencing disks before locating the data that needs to be read.

Cohesity's SnapTree technology does not use differencing disks. Instead, SnapTree is a very powerful and highly scalable, global file system. SnapTree uses a technique called Distributed Redirect on Write to achieve its required functionality. The underlying concept is both simple and effective. Rather than using differencing disks to capture write operations, write operations are directed to new blocks. As such, both current and previously used storage blocks reside within the same file system. One of the major advantages to this technique is that it does not degrade read or write performance, because the number of traverses that are required in order to retrieve data from any snapshot is fixed.

Another advantage to SnapTree's design is that SnapTree allows for an unlimited number of snapshots to be stored. Differencing disk based snapshots tend to have operational limits and fixed limits. Some vendors for example, hard code a limit to the number of snapshots that can exist at any one time. Often times however, performance begins to significantly degrade long before this limit is reached. Some vendors periodically consolidate snapshots by using the snapshots to generate full data copies. However, this consolidation process consumes additional storage I/O and can further degrade performance until the consolidation operation completes.

SnapTree goes beyond simply providing better performance than differencing disk based solutions. Cohesity has engineered SnapTree to provide Web scale performance. Cohesity achieves this by distributing snapshot data across all of the nodes in the cluster. Hence, the larger the cluster, the more disks are involved in storing snapshot data. As such, snapshot performance is able to keep pace with cluster growth.

Cohesity achieves restoration granularity and snapshot application awareness through the use of resource specific protection jobs, as shown in Figure 5. By creating a SQL Server specific protection job for example, Cohesity ensures that the SQL Server VSS Writer is used during the snapshot creation process, and that SQL Server transaction logs are also protected.

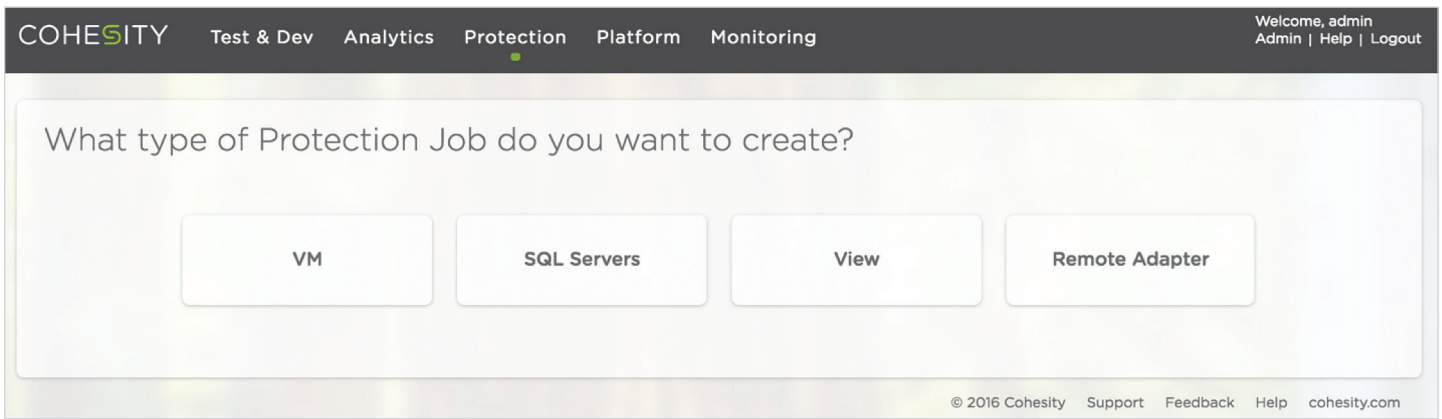

Figure 5: Data Protection for SQL Server

Cohesity allows for the creation of SQL Server protection jobs.

SQL Server protection jobs are tied to simple policies that define the frequency with which SQL Server resources are protected. Figure 6 for instance, shows a policy instructing the software to create snapshots on an hourly basis. Because these snapshots are being specifically used to protect SQL servers, the snapshots are application aware. This same policy also instructs the backup software to capture the SQL Server transaction logs every fifteen minutes.

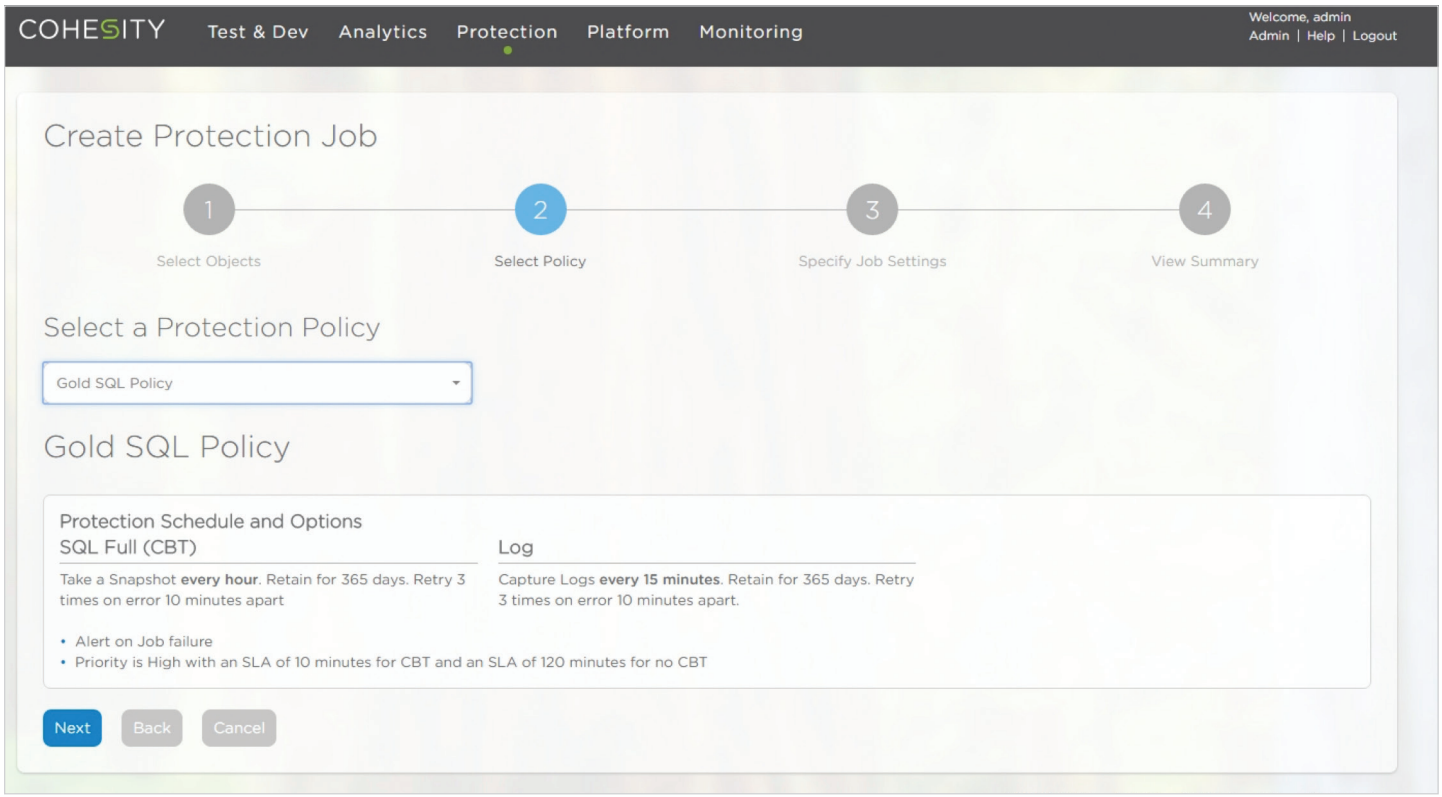

Figure 6: Data Protection for SQL Server

The Cohesity software creates application aware snapshots.

Instant Recovery

One of Cohesity's key capabilities is the instant recovery of SQL Server. SQL Servers commonly support mission critical workloads, and there are tangible costs associated with workload outages. Traditional SQL Server restorations can take hours or even days to complete. As such, any technology that reduces the RTO for SQL Server is effectively also reducing the cost of an outage.

Cohesity uses integrated data protection, meaning that the backup software is running on dedicated, enterprise grade hardware. The integrated storage hardware contains the snapshots that have been created by the backup process. These snapshots can be mounted as a virtual machine and run directly on the Cohesity DatPlatform. Doing so means that it is possible to bring a failed SQL Server back online almost immediately, while also protecting the backup contents against unwanted modifications stemming from using the backup VM.

One of the reasons why the instant recovery process allows virtual machines to be brought online so quickly is that unlike some of the competing backup solutions, every Cohesity backup is fully hydrated. All of the required storage blocks already exist within the SnapTree file system, so there is no need to consolidate deltas or to merge differencing disks. Storage blocks can be used as-is. Cohesity's approach to instant recovery is not only efficient, but also allows an unlimited number of snapshots to be retained. Competing backup solutions that are based on differencing disks commonly require older snapshots to be periodically deleted, archived, or merged. Cohesity however, has no such requirement and snapshots can be retained indefinitely.

With the SQL Server now online, and available for production use, a traditional restoration is performed in the background. Once the restoration completes, users are redirected from the Cohesity DataPlatform to the newly restored production virtual machine.

Cohesity uses similar technology to support its VM Cloning feature. Like instant recovery, VM cloning mounts and runs virtual machines directly on the Cohesity DataPlatform. While instant recovery is designed to reduce RTO however, VM cloning allows administrators to safely use VM backups as a sandboxed dev / test environment. The use of snapshots means that these environments can be created without altering the backup contents in the process. Because the dev / test environment perfectly mimics the production environment, VM clones can be used to accurately test configuration changes, software upgrades, and more. You can see some of the clone options for SQL Servers in Figure 7.

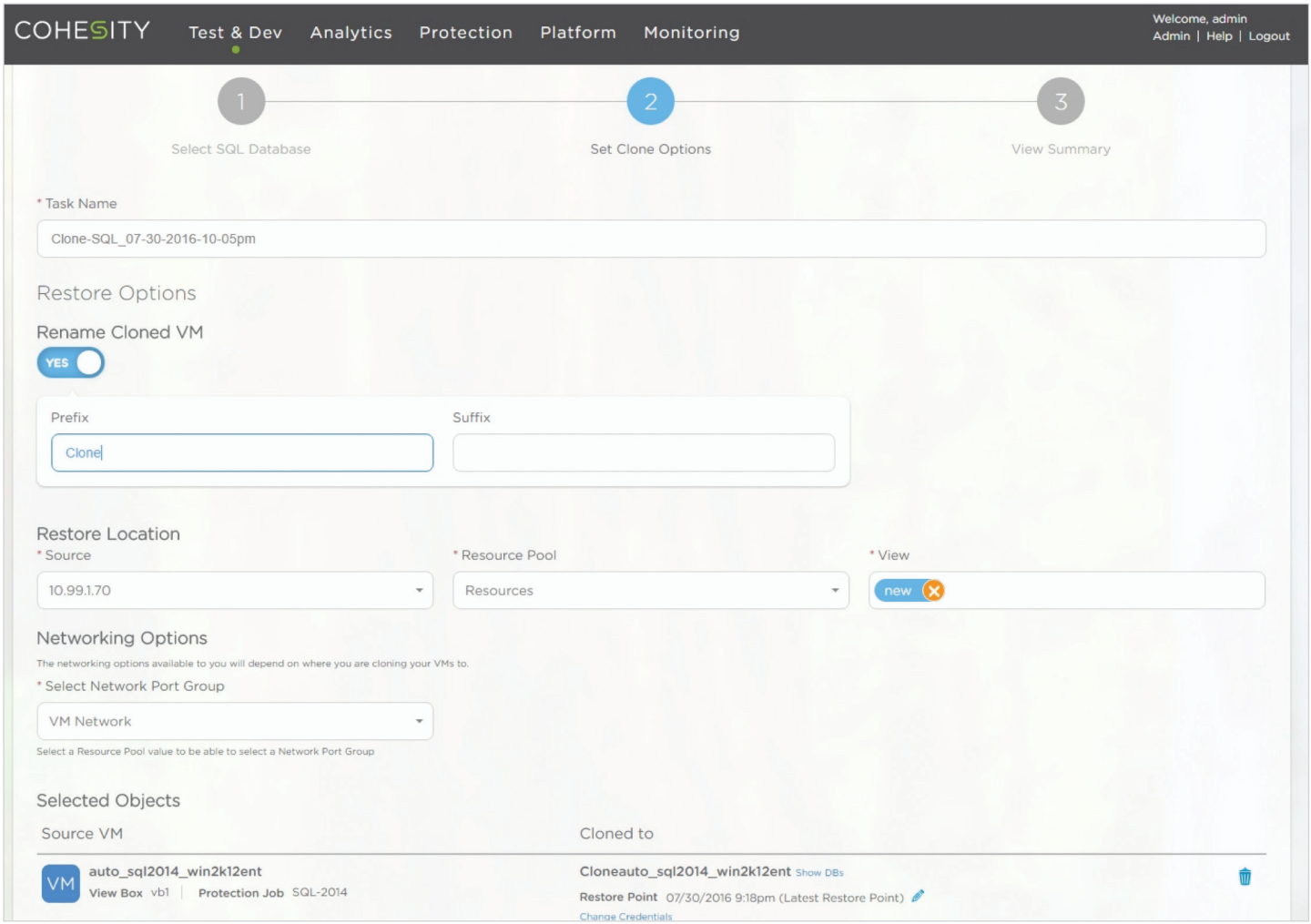

Figure 7: Data Protection for SQL Server

Cohesity allows SQL Servers to be cloned for use in dev / test environments.

Conclusion

Although most backup applications are able to create VSS backups of SQL Server, the only way to completely protect against data loss is to also backup the SQL Server transaction logs. Of course establishing comprehensive protection for SQL Server is only part of what is required. Administrators must also work to minimize the RTO associated with SQL Server restorations so that mission critical workloads can be brought back online as quickly as possible.

©2016 Cohesity. All Rights Reserved.