Realtime publishers

The Shortcut Guide™ To

Architecting iSCSI Storage for Microsoft Hyper-V

sponsored by

Greg Shields

Copyright Statement

Chapter 4: The Role of Storage in Hyper-V Disaster Recovery

You've learned about the power of iSCSI in Microsoft virtualization. You've seen the various ways in which iSCSI storage is connected into Hyper-V. You've learned the best practices for architecting your connections along with the smart features that are necessary for 100% storage uptime. *You've now got the knowledge you need to be successful in architecting iSCSI storage for Hyper-V.*

With the information in this guide's first three chapters it becomes possible to create a highly-available virtual infrastructure atop Microsoft's virtualization platform. With it, you can create and manage virtual machines with the assurance that they'll survive the loss of a host, a connection, or any of the other outages that happen occasionally within a data center.

Yet this knowledge remains incomplete without a look at one final scenario: the complete disaster. That disaster might be something as substantial as a Category 5 hurricane or as innocuous as a power outage. But in every scenario, the end result is the same: *You lose the computing power of an entire data center.*

Important to recognize here is that the techniques and technologies that you use in preparing for a complete disaster are far, far different than those you implement for high availability. Disaster recovery elements are added to a virtual environment as an augmentation that protects against a particular type of outage.

Defining "Disaster"

Before getting into the actual click-by-click installation of Hyper-V disaster recovery, it is important first to understand what actually makes a disaster. Although the term "disaster" finds itself greatly overused in today's sensationalist media ("Disaster in the South: News at 11."), the actual concept of disaster in IT operations has a very specific meaning.

There are many technical definitions of "disasters" that exist, one of which your organization's process framework likely leverages to functionally define when a disaster has occurred. Rather than relying on any of the technical definitions, however, this chapter will simply consider a disaster for IT operations to be *an event that fully interrupts the operations of a data center.*

Using this definition, you can quickly identify what kinds of events can be considered a disaster:

- A naturally-occurring event, such as a tornado, flood, or hurricane, impacts your data center and causes damage; that damage causes the entire processing of that data center to cease
- A widespread incident, such as a water leakage or long-term power outage that interrupts the functionality of your data center for an extended period of time
- An extended loss of communications to a data center, often caused by external forces such as utility problems, construction, accidentally severed cabling, and so on

Although disasters are most commonly associated with the types of events that end up on the news, the actual occurrence of newsworthy disasters is in fact quite rare. In reality, the events making up the second group in the previous list are much more likely to occur. Both cause interruption to a data center's operations, but those in the first group occur with the kinds of large-scale damage that requires greater effort to fix.

It is important to define disasters in this way because those above are handled in much different ways than simple service outages. Consider the following set of incidents that are problematic and involve outage but are in no way disasters:

- A problem with a virtual host creates a "blue screen of death," immediately ceasing all processing on that server
- An administrator installs a piece of code that causes problems with a service, shutting down that service and preventing some action from occurring on the server
- An issue with power connections causes a server or an entire rack of servers to inadvertently and rapidly power down

The primary difference between these types of events and your more classic "disasters" relates to the actions that must occur to resolve the incident. In none of these three incidents has the operations of the data center been fully interrupted. Rather, in each, some problem has occurred that has caused a portion of the data center—a server, a service, or a rack—to experience a problem.

This differentiation is important because a business' decision to declare a disaster and move to "disaster operations" is a major one. And the technologies that are laid into place to act upon that declaration are substantially different (and more costly) than those used to create simple high availability. In the case of a service failure, you are likely to leverage your high-availability features such as Live Migration or automatic server restart. In a disaster, you will typically find yourself completely moving your processing to an alternative site. The failover and failback processes are big decisions with potentially big repercussions.

Defining "Recovery"

Chapter 3 started this guide's conversation on disaster recovery through its iterative discussion on the features that are important to Hyper-V storage. There, a graphic similar to Figure 4.1 was shown to explain how two different iSCSI storage devices could be connected across two different sites to create the framework for a disaster recovery environment.

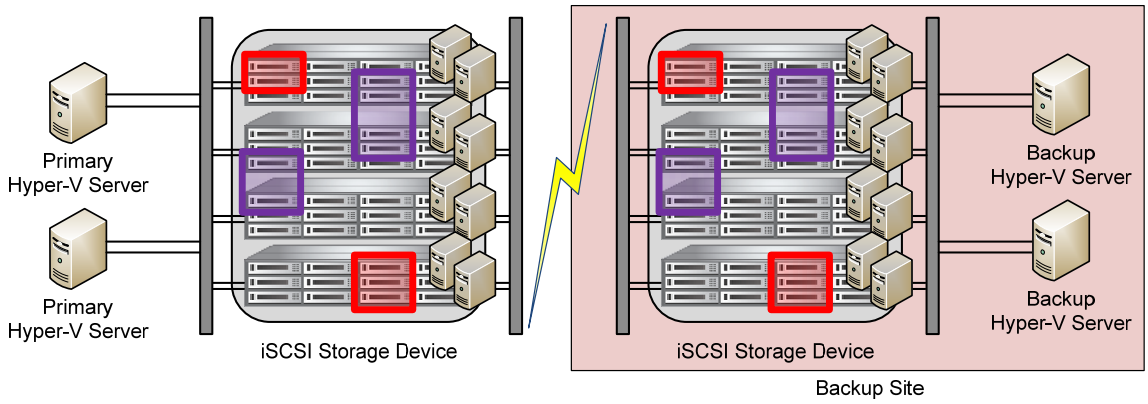

Figure 4.1: The setup of two different SANs in two different sites lays the framework for Hyper-V disaster recovery.

In Figure 4.1, you can see how two different iSCSI storage devices have been interconnected. The device on the left operates in the primary site and handles the storage needs for normal operations. On the left is another iSCSI storage device that contains enough space to hold a copy of the necessary data for disaster operations. Between these two storage devices is a network connection of high-enough bandwidth to ensure that the data in both sites remains synchronized.

This architecture is important because at its very core virtualization makes disaster recovery far more possible than ever before. Virtualization's encapsulation of servers into files on disk makes it both operationally feasible and affordable to replicate those servers to an alternative location.

At a very high level, disaster recovery for virtual environments is made up of three basic things:

- A storage mechanism
- A replication mechanism
- A target for receiving virtual machines and their data

The storage mechanism used by a Hyper-V virtual environment (or, really any virtual environment) is the location where each virtual machine's disk files are contained. Because the state of those virtual machines is fully encapsulated by those disk files, it becomes trivial to replicate them to an alternative location. Leveraging technology either within the storage device, at the host, or a combination of both, creating a fully-functional secondary site is at first blush as trivial as a file copy.

Now, obviously there are many factors that go into making this "file copy" actually functional in a production environment. There are different types of replication approaches that focus on performance or prevention of data loss. There are clustering mechanisms that actually enable the failover as well as failback once the primary site is returned to functionality. There are also protective technologies that ensure data is properly replicated to the alternative site such that it is crash-consistent and application-consistent. All of these technologies you will need to integrate when creating your own recovery solution.

The Importance of Replication, Synchronous and Asynchronous

Without delving into the finer details of how this architecture is constructed, a primary question that must first be answered relates to how that synchronization is implemented. Remember that above all else, an iSCSI storage device is at its core just a bunch of disks. Those disks have been augmented with useful management functions to make them easier to work with (such as RAID, storage virtualization, snapshots, and so on), but at its most basic, a storage device remains little more than disk space and a connection.

This realization highlights the importance of how these two storage devices must remain in synch with each other. Remember that the sole reason for this second storage device's existence is to create a second copy of production data comprised of both virtual machine disk files and the data those virtual machines work with. Thus, the mechanism by which data is replicated from primary to backup site (and, eventually, back) is important to how disaster recovery operations are initiated.

Two types of replication approaches are commonly used in this architecture to get data migrated between storage devices. Those two types are generically referred to as *synchronous* and *asynchronous* replication. Depending on your needs for data preservation as well as the resources you have available, you may select one or the other of these two options. Or, both.

Synchronous Replication

In synchronous replication, changes to data are made on one node at a time. Those changes can be the writing of raw data to disk by an application or the change to a virtual machine's disk file as a result of its operations. When data is written using synchronous replication, that change is first enacted on the primary node and then subsequently made on the secondary node. Important to recognize here is that the change is not considered complete *until the change has been made on both nodes*. By requiring that data is assuredly written on both nodes before the change is complete, the environment can also ensure that no data will be lost when an incident occurs.

Consider the following situation: A virtual machine running Microsoft Exchange is merrily doing its job, responding to Outlook clients and interacting with its Exchange data stores. That virtual machine's disk files and data stores are replicated using synchronous replication to a second storage device in another location. Every disk transaction that occurs with the virtual machine requires the data to be changed at both the primary and secondary site before the next transaction can occur.

Figure 4.2 shows a breakdown of the steps required for this synchronous replication to fully occur. In this situation, the Exchange server makes a change to its database. That change is first committed at the primary site. It is then replicated to the secondary site, where it is committed to the storage device in that location. An acknowledgement of commitment is finally sent back to the primary site, upon which both storage devices can then move on to the next change.

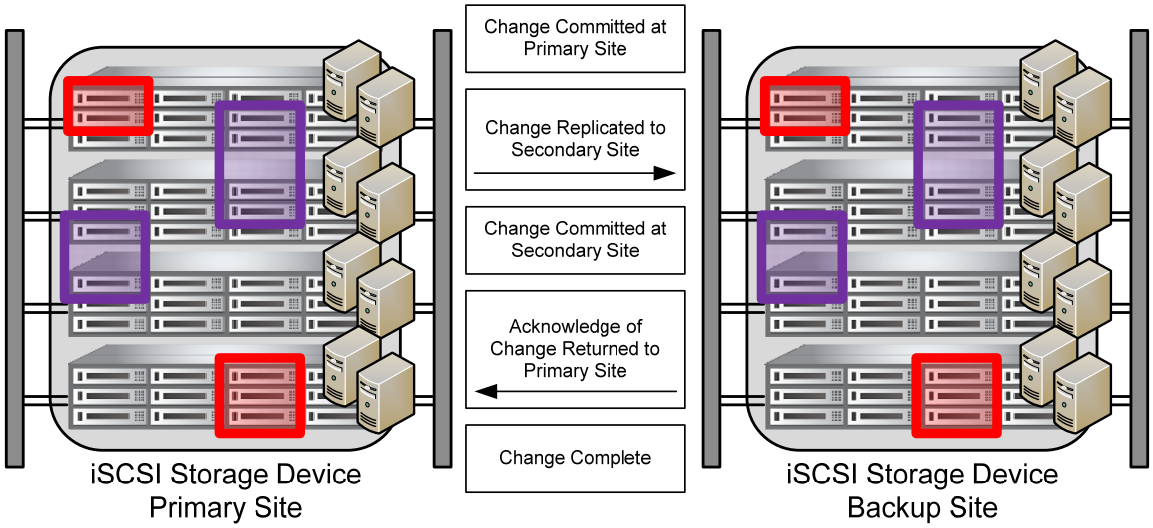

Figure 4.2: A breakdown of the steps required for synchronous replication.

This kind of replication very obviously ensures that every piece of data is assuredly written before the next data change can be enacted. At the same time, you can see how those extra layers of assurance can create a bottleneck for the secondary site. As each change occurs, that change must be acknowledged across both storage devices before the next change can occur.

Synchronous replication works exceptionally well when the connection between storage devices is of very high bandwidth. Gigabit connections combined with short distances between devices reduces the intrinsic latency in this architecture. As a result, environments that require zero amounts of data loss in the case of a disaster will need to leverage synchronous replication.

Asynchronous Replication

Asynchronous replication, in contrast, does not require data changes to occur in lock-step between sites. Using asynchronous replication between sites, changes that occur to the primary site are configured to *eventually be written to the backup site*.

Leveraging preconfigured parameters, changes that occur to the primary site are queued for replication to the backup site as appropriate. This queuing of disk changes between sites enables the primary site to continue operating without waiting for each change's commitment and acknowledgement at the backup site. The result is no loss of storage performance as a function of waiting for replication to complete.

Although asynchronous replication eliminates the performance penalties sometimes seen with synchronous replication, it does so by also eliminating the assurance of zero or nearly zero data loss. In Figure 4.3, you can see how changes at the primary site are queued up for eventual transfer to the backup site. Using this approach, changes can be submitted in batches as bandwidth allows; however, a disaster that occurs between change replication intervals will cause some loss of the queued data.

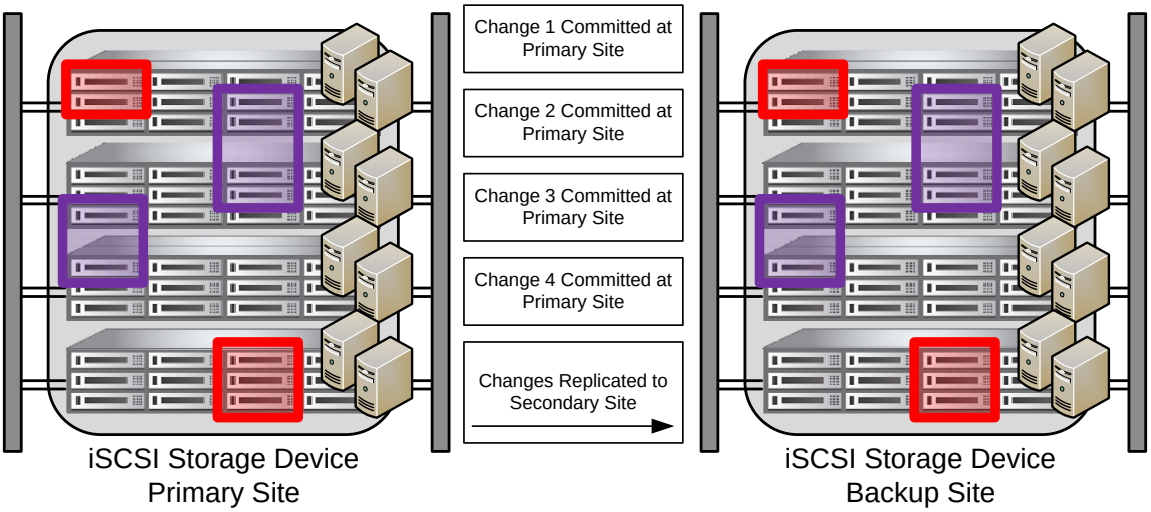

Figure 4.3: The steps involved with asynchronous replication.

Although the idea of "eventual replication" might seem scary in terms of data integrity, it is in fact an excellent solution for many types of disaster recovery scenarios. To give you an idea, turn back a few pages and take another look at the types of incidents that this chapter considers to be disasters. In either of these classes of events, the level of impact to the production data center facility is enormous. At the same time, those same types of disasters are likely to cause an impact to the people who work for the business as well.

For example, a natural disaster that impacts a data center is also likely to impact the brick-and-mortar offices of the business. This impact may impede the ability of employees to get the job of the business done. As a result, a slight loss in data may be insignificant when compared with the amount of business data that is saved, that will be used in the immediate term, and that can be reconstructed from other means.

Which Should You Choose?

To summarize the discussed concepts, remember always that synchronous replication has the following characteristics:

- Assures no loss of data
- Requires a high-bandwidth and low-latency connection
- Write and acknowledgement latencies impact performance
- Requires shorter distances between storage devices

In contrast, asynchronous replication solutions have the following characteristics:

- Potential for loss of data during a failure
- Leverages smaller-bandwidth connections, more tolerant of latency
- No performance impact to source server processing
- Potential to stretch across longer distances

Your decision about which type of replication to implement will be determined primarily by your Recovery Point Objective (RPO), and secondarily by the amount of distance you intend to put between your primary and secondary sites.

Recovery Point Objective

RPO is a measurement of your business' tolerance for acceptable data loss for a particular service, and is formally defined as "the point in time to which you must recover data as defined by your organization." Business services that are exceptionally intolerant of data loss are typified by production databases, critical email stores, or line of business applications. These services and applications cannot handle any loss of data for reasons based on business requirements, compliance regulations, or customer satisfaction. For these services, even the most destructive of disasters must be mitigated against because the loss of even a small amount of data will significantly impact business operations.

You'll notice here that this definition does not talk about the RPO *of your business* but rather the RPO *of particular business services*. This is an important differentiation as well as one that requires special highlighting. Remember that every business has services that it considers to be Tier I or "business critical". Those same businesses have other services that it considers to be Tier II or "moderately important" as well as others that are Tier III or "low importance."

This differentiation is critically important because although virtualization indeed makes disaster recovery operationally feasible for today's business, disaster recovery still represents an added cost. Your business might see the need for getting its production database back online within seconds, but it likely won't need the same attention for its low-importance WSUS servers or test labs.

Distance Between Sites

Remember too that synchronous replication solutions require good bandwidth between sites. At the same time, they are relatively intolerant of latency between those connections. Thus, the physical distance between sites becomes another factor for determining which solution you will choose.

Of the different types of disasters, natural disasters tend to have the greatest impact on this decision. For example, to protect against a natural disaster like a Category 5 hurricane, you likely want your backup site to sit in a geographic location that is greater than the expected diameter of said hurricane. At the same time, Category 5 hurricanes are relatively rare events, while other events like extended power outages are much more likely.

It is for these reasons that combinations of synchronous, asynchronous, and even non-replication for your servers can be an acceptable solution. Some of your servers need to stay up no matter what, while others can wait for the disaster to end and normal operations to return. Others can be protected against low-impact disasters through short-distance synchronous replication, while a tertiary site located far away protects against the worst of natural cataclysms. In all of these, cost and benefit will be your guide.

Note

An additional and yet no less important determinant here relates to your support servers. When considering which virtual servers to enable for disaster recovery, remember to also make available those that provide support services. You don't want to experience a disaster, fully failover, and find yourself without domain controllers to run the domain or Remote Desktop Servers to connect users to applications.

Ensuring Data Consistency

No discussion on replication is complete without a look at the perils of data consistency. Bluntly put, if you expect to simply file-copy your virtual machines from one storage device to another, you'll quickly find that the resulting copies aren't likely to power on all that well. Nor will their applications and databases be immediately available for use when a disaster strikes.

Data Consistency: An Exchange Analogy

The best way to explain this problem is through a story. Have you or someone in your organization ever accidentally pulled the power cable on your Exchange Server? Or have you ever seen that Exchange Server crash, powering down without a proper shut down sequence? What happens when either of these two situations happens?

In either situation, the Exchange database does not return back to operations immediately with the powering back on of the server. Instead it refuses to start Exchange's services, reporting that its database was shut down *uncleanly*. The only solution when this occurs is a long and painful process of running multiple integration checks on the database to return it back to functionality. Depending on the size of the database, those integrity checks can require multiple hours to complete. During their entire process, your company must operate with a non-fully-functional mail system. It is for this reason that businesses that use Microsoft Exchange add high-availability features such as battery backup, redundant power supplies, and even database replication to alternative systems.

Now, you might be asking yourself, "How does this story relate to data consistency in replicated virtual environments?" The answer is, Without the right technology in place, a dirty Exchange database can occur from a poorly-replicated virtual machine in the exact same way that it does with a power fault. In either case, you must implement the right technologies if you're to prevent that unclean shutdown.

The problem here has to do with the ways in which virtual machine data is replicated from primary site to backup site. Remember that a running virtual machine is also a virtual machine that is actively using its disk files. Thus, any traditional file copy that occurs from a primary site to a backup site will find that the file has changed during the course of the copy. Even ignoring the obvious locked-file problems that occur with such open files, it becomes easy to see how running virtual machine disk files cannot be replicated without some extra technology in place.

Further complicating this problem are the applications that are running within that virtual machine itself. Consider Exchange once again as an example, although the issue exists within any installed transactional database. With a Microsoft Exchange data store, its .EDB file on disk behaves very much like a virtual machine's disk file. In essence, although it may be possible to copy that .EDB file from one location to another, you can only be guaranteed a successful copy if the Exchange server is not actively using the file. If it is, changes are likely to occur during the course of the transfer that result in a corrupted database.

It is for both of these reasons that extra technology is required at one or more levels of the infrastructure to manage the transfer between primary and secondary sites. This technology commonly uses one of many different snapshotting technologies to watch for and transfer changes to virtual machines and their data as they occur.

Data integration technologies often require the installation of extension software to either the Hyper-V cluster or the individual virtual machines. This software commonly integrates with the onboard Volume Shadow Copy service along with its application-specific providers to create and work with dynamic snapshots of virtual machines and their installed applications. The result is much the same as what is seen with traditional application backup agents that integrate with applications like Exchange, SQL, and others, to successfully gather backups from running application instances. The difference here is that instead of gathering backups for transfer to tape, *these solutions are gathering changes for replication to a backup site.*

Other solutions exist purely at the level of the storage device. These solutions leverage on-device technology for ensuring that data is replicated consistently and in the proper order. It should be obvious that leveraging storage device-centric solutions can be of lesser complexity: Using these solutions, installing agents to each virtual host or machine may not be required. Also, fewer "moving parts" are exposed to the administrator, allowing administrators to enable replication on a per-device or per-volume basis with the assurance that it will operate successfully with minimal further interaction. Depending on your environment, one or both of these solutions may be necessary for accomplishing your needs for replication.

Note

When considering a secondary storage device for disaster recovery purposes, you must take into account the extra technologies required to ensure data consistency. In essence, if your backup site cannot automatically fail over without extra effort, you don't have a complete disaster recovery solution.

Architecting Disaster Recovery for Hyper-V

All of this introductory discussion brings this conversation to the main topic of how to actually enable disaster recovery in Hyper-V. You'll find that the earlier discussion on storage devices and replication is fundamentally important for this architecture. Why? Because creating disaster recovery for Hyper-V involves stretching your Hyper-V cluster to two, three, or even many sites and implementing the necessary replication. The first half of accomplishing this is very similar to the cluster creation first introduced in Chapter 2.

Note

As in Chapter 2, this guide will not detail the exact click-by-click steps necessary to build such a cluster. That information is better left for the step-by-step guide that is available on Microsoft's Web site at http://technet.microsoft.com/en-us/library/cc732488(WS.10).aspx.

Microsoft's terminology for a Hyper-V cluster that supports disaster recovery is a *multi-site cluster*, although the terms *stretch cluster* and *geocluster* have all been used to describe the same architecture. By definition, a Microsoft multi-site cluster is a traditional Windows Failover Cluster that has been extended so that different nodes in the same cluster reside in separate physical locations.

Figure 4.4 shows a network diagram of the same cluster that was first introduced in Figure 4.1. In Figure 4.4, you can see how the high-availability elements that were added into the single-site cluster have been mirrored within the backup site.

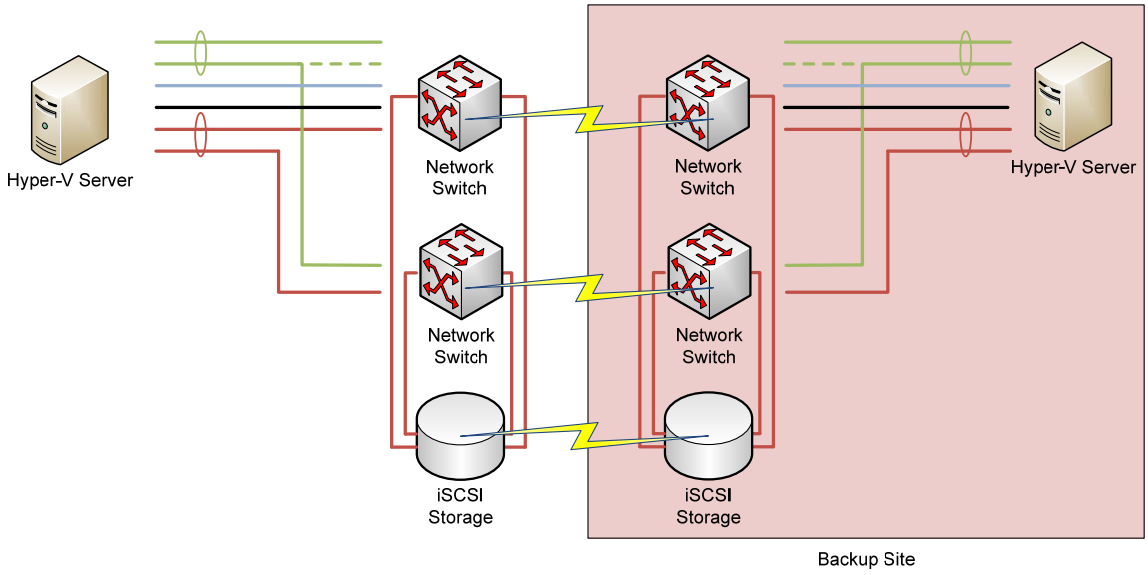

Figure 4.4: A network diagram of a multi-site cluster that includes high-availability elements.

Full Redundancy Isn't Always Necessary at the Backup Site

This mirroring of high-availability elements is present for completeness; however, it is not uncommon for backup site servers to leverage fewer redundancy features than are present in the production site.

The reason for this reduction in redundancy lies within the reason for being for the cluster itself: Backup sites are most commonly used for disaster operations only—often only a small percentage of total operations—so the cost for full redundancy often outweighs its benefit. As you factor in the amount of time you expect to operate with virtual machines at the backup site, your individual architecture may also reveal that fewer features are necessary.

Important to recognize in this figure is the additional iSCSI storage location that exists within the backup site. Multi-site Hyper-V clusters leverage the use of local and replicated storage within each site. Although each Windows Failover Cluster generally requires this storage to be local to the site, its services provide no built-in mechanisms for accomplishing the replication. You must turn to a third-party provider—commonly either through your storage vendor or an application provider—to provide replication services between storage devices.

Note

Although Microsoft has a replication solution in its Distributed File System Replication (DFS-R) solution, this solution is neither appropriate nor supported for use as a cluster replication mechanism. DFS-R only performs replication as a file is closed, an action that does not often happen with running virtual machines. Thus, it cannot operate as a cluster replication solution.

Choosing the Right Quorum

In Windows Server 2008, Microsoft eliminated the earlier restriction that cluster nodes all reside on the same subnet. This restriction complicated the installation of multi-site clusters because the process of extending subnets across sites was complex or even impossible in many companies. Today, the click-by-click process of creating a cluster across sites requires little more than installing the Windows Failover Clustering service onto each node and configuring the node appropriately.

Although clicking the buttons might be a trivial task, it is designing the architecture of that cluster where the greatest complexity is seen. One of the first decisions that must be made has to do with *how the cluster determines whether it is still a cluster*. This determination is made through a process of obtaining quorum.

Obtaining quorum in a Hyper-V cluster is not unlike how your local Kiwanis or Rotary club obtains quorum in their weekly meetings. If you've ever been a part of a club where decisions were voted on, you're familiar with this process. Consider the analogy: Decisions that are important to a Kiwanis club should probably be voted on by a large enough number of club members. In the bylaws of that club, a process (usually based on the rules of Parliamentary Procedure) is documented that explains how many members must be present for an important item to be voted on. That number is commonly 50% of the total members plus one. Without this number of members present, the club itself cannot vote on important matters, *because it does not see itself as a fully-functioning club*.

The same holds true in Hyper-V clusters. Remember first that a cluster is by definition always prepared for the loss of one or more hosts. Thus, it must always be on the lookout for conditions where there are not enough surviving members for it to remain a cluster. This count of surviving members is referred to as the cluster's *quorum*. And just like different Kiwanis clubs can use different mechanisms to identify how they measure quorum, there are different ways for your Hyper-V cluster to identify whether it has quorum. In Windows Server 2008, four are identified.

Node and Disk Majority

In the Node and Disk Majority model, each node gets a quorum vote, as does each disk. Here, a single-site four-node cluster would have five votes: one for each of the nodes plus one for its shared storage. Although useful for single-site clusters that have an even number of nodes, Node and Disk Majority is not a recommended quorum model for multi-site clusters. This is the case because the replicated shared storage introduces a number of challenges with multi-site clusters. The process of replication can cause problems with SCSI commands across multiple nodes. Also, storage must be replicated in real-time synchronous mode across all sites for the disks to retain the proper awareness.

Disk Only Majority

In the Disk Only Majority model, only the individual storage devices have votes in the quorum determination. This model was used extensively in Windows Server 2003, and although it is still available in Windows Server 2008, it is not a recommended configuration for either single-site or multi-site clusters today.

Node Majority

In the Node Majority model, only the individual cluster nodes have votes in the quorum determination. It is strongly suggested that environments that use this model do so with a node count that is equal to three or greater in single-site clusters, and only with an odd number of nodes in multi-site clusters. Clusters that leverage this model should also be configured such that the primary site contains a greater number of nodes than the secondary site. Further, the Node Majority model is not recommended when a multi-site cluster is spread across more than two sites.

The reason for these recommendations has to do with how votes can be counted by the cluster in various failure conditions. Consider a two-site cluster that has five nodes, three in the primary site and two in the secondary site. In this configuration, the cluster will remain active even with the loss of any two of the nodes. Even if the two nodes in the secondary site are lost, the three nodes in the primary site will remain active because three out of five votes can be counted.

Node and File Share Majority

The Node and File Share Majority adds a separate *file share witness* to the Node Majority Model. Here, a file share on a server separate from the cluster is given one additional vote in the quorum determination. It is recommended that the file share be located in a site that is not one of the sites occupied by any of the cluster nodes. If no additional site exists, it is possible to locate the witness file share within the primary site; however, its location there does not provide the level of protection gained through the use of a completely separate site.

This introduction of the file share witness to the cluster quorum determination provides a very specific assist to multi-site clusters in helping to arbitrate the quorum determination when entire sites are down. Because an entire-site loss also results in the loss of network connectivity to all hosts on that site, the cluster can experience a situation known as “split brain” where multiple sites each believe that they have enough votes to remain an active cluster. This is an undesirable situation because each isolated and independent site will continue operating under the assumption that the other nodes are down, creating problems when those nodes are again available. Introducing the file share witness to the quorum determination ensures that an entire site loss cannot create a split brain condition, no matter how many nodes are present in the cluster.

Further, the Node and File Share Majority also makes possible the extension of clusters to more than two sites. A single file share in an isolated site can function as the witness for multiple clusters. Figure 4.5 shows a network diagram for how a witness disk can be used to ensure complete resiliency across a multi-site cluster even with the loss of any single site.

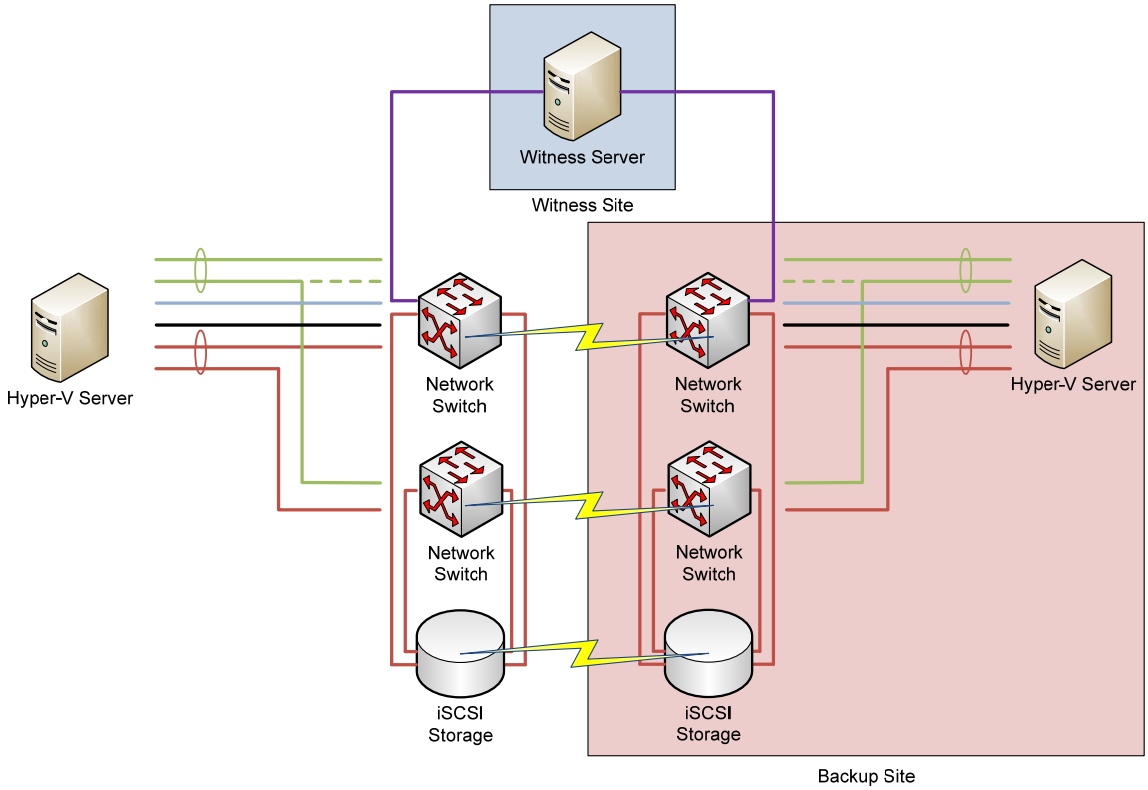

Figure 4.5: Introducing a Witness Server further protects a multi-site cluster from a site failure.

Obtaining Quorum

If you are considering a multi-site cluster for disaster recovery, you will need to select one of the two recommended quorum options (Node Majority or Node and File Share Majority). That decision will most likely be based on the availability of an isolated site for the witness disk but can be based on other factors as well.

The actual process of obtaining quorum is an activity that happens entirely under the covers within the Windows Failover Cluster service. To give you some idea of the technical details of this process, on its Web site at http://technet.microsoft.com/en-us/library/cc730649(WS.10).aspx Microsoft identifies the high-level phases that are used by cluster nodes to obtain quorum. Those phases have been reproduced here:

- As a given node comes up, it determines whether there are other cluster members that can be communicated with (this process may be in progress on multiple nodes simultaneously).
- Once communication is established with other members, the members compare their membership "views" of the cluster until they agree on one view (based on timestamps and other information).

- A determination is made as to whether this collection of members "has quorum" or, in other words, has enough members that a "split" scenario cannot exist. A "split" scenario would mean that another set of nodes that are in this cluster was running on a part of the network not accessible to these nodes.
- If there are not enough votes to achieve quorum, the voters wait for more members to appear. If there are enough votes present, the Cluster service begins to bring cluster resources and applications into service.
- With quorum attained, the cluster becomes fully functional.

Ensuring Network Connectivity and Resolution

The final step in architecting your Hyper-V cluster relates to the assurance that proper networking and name resolution are both present at any of the potential sites to which a virtual machine may fail over. This process is made significantly easier through the introduction of multi-subnet support for Windows Failover Clusters. That support eliminates the complex (and sometimes impossible) networking configurations that are required to stretch a subnet across sites.

This is very obviously a powerful new feature. However, at the same time, the use of multiple subnets in a failover cluster means that virtual machines must be configured in such a way that they retain network connectivity as they move between sites. For example, the per-virtual machine addressing for each virtual machine must be configured such that its IP address, subnet mask, gateway, and DNS servers all remain acceptable as it moves between any of the possible sites. Alternatively, DHCP and dynamic DNS can be used to automatically re-address virtual machines when a failover event occurs.

Any of these events will involve some level of downtime for clients that attempt to connect to virtual machines as they move between sites. The primary delay in connection has to do with re-convergence of proper DNS settings both on the servers as well as clients after a failover event. It may be necessary to reconfigure DNS settings to reduce their Time To Live (TTL) setting for DNS entries, or flush local caches on clients after DNS entries have been updated to reconnect clients with moved servers.

Disaster Recovery Is Finally Possible with Hyper-V Virtualization

Although this chapter's discussion on disaster recovery might at first blush appear to be a complex solution, consider the alternatives of yesteryear. In the days before virtualization, disaster recovery options were limited to creating mirrored physical machines in alternative sites, replicating their data through best-effort means, and manually updating backup servers in lock-step with their primary brethren.

Today's solutions for Hyper-V disaster recovery are still not installed through any Next, Next, Finish process. These architectures remain solutions rather than any simple product installation. However, with a smart architecture and planning in place, their actual implementation and ongoing management can be entirely feasible by today's IT professionals. Doing so atop iSCSI-based storage solutions further enhances the ease of implementation and management due to iSCSI's network-based roots.

Your next step is to actually implement what you've learned in this guide. With the knowledge you've discovered in its short count of pages, you're now ready to augment Hyper-V's excessively simple installation with high-powered high-availability and disaster recovery. No matter whether you need a few servers to host a few virtual machines or a multi-site infrastructure for complete resiliency, the iSCSI tools are available to manifest your needed production environment.

Download Additional eBooks from Realtime Nexus!

Realtime Nexus—The Digital Library provides world-class expert resources that IT professionals depend on to learn about the newest technologies. If you found this eBook to be informative, we encourage you to download more of our industry-leading technology eBooks and video guides at Realtime Nexus. Please visit http://nexus.realtimepublishers.com.