Chapter 29

Server Virtualization with Hyper-V

If there is anything that the IT industry has shown us, it must be that technologies go in and out of fashion but seem to be resurrected when their time comes again. Server virtualization is one of these topics. Mainframes did it 30 years ago, and now virtualization is becoming big in the Windows world. With Windows Server 2008, Microsoft has introduced its own native server virtualization software, called Hyper-V. In Windows Server 2008 R2, Microsoft has upgraded Hyper-V with new possibilities and higher performance. The two versions are mostly compatible, and where relevant we will note the differences.

Virtualization is a huge subject. Typically, server virtualization is a game of big iron: many large multiprocessor servers, loads of memory, storage area networks, Fibre Channel networks, clustering technologies, management software, and so on. In one chapter of a general-purpose book on Windows Server 2008 R2, such as this one, it is impossible to cover all of that. In this chapter, you can expect an introduction on Hyper-V, sufficient to get you to first base and enabling you to build your own small environment using only native Windows software. We address topics such as what server virtualization is and what it's used for, how to install and use Hyper-V, the constituent components of Hyper-V and how they work together, how Hyper-V works with Windows Server Core, and how to manage Hyper-V installations, including Live Migration.

In this chapter, you will learn to:

- Determine whether a server can run Hyper-V
- Determine when it makes sense to virtualize a server
- Decide which technology to use to quickly move a virtual machine between hosts
- Advise on a backup strategy

What Is Server Virtualization?

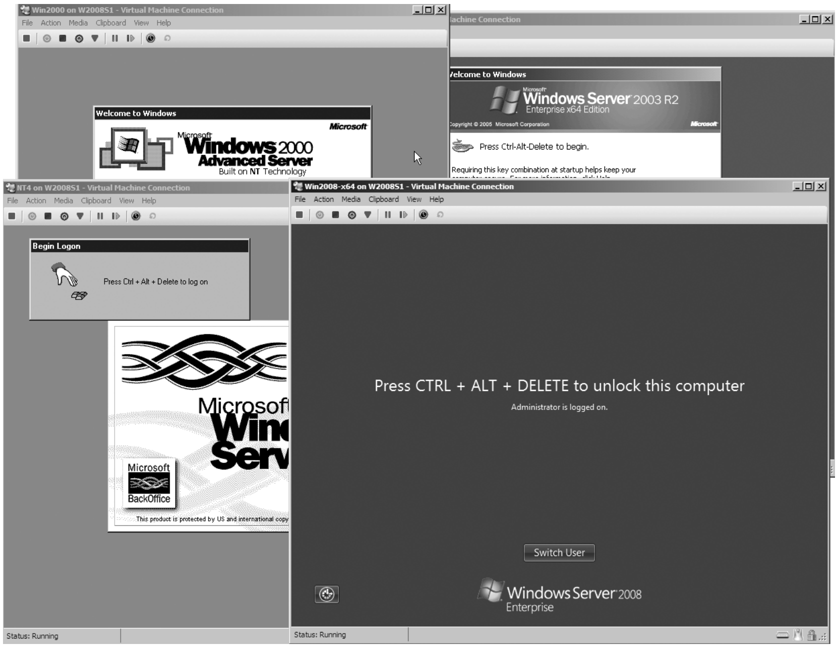
The term *virtualization* is used for a lot of different things nowadays. It is used in association with applications, storage, network, servers, screen presentation, and so on. In this chapter, *virtualization* means the ability to run a full operating system on a software platform in such a way that the OS thinks it is running on a "real" computer. This type of virtualization is called *hardware virtualization* or *server virtualization*. So, why would you want to have that?

Chances are that you are a system administrator, responsible for a number of servers in your organization. If you have been in the business a while, you will have noticed the trend that server power tends to grow faster than the resource hunger of applications.

Nowadays, you buy a low-end server with at least 4GB and probably 8GB or more. Moreover, you buy a 64-bit capable machine. Most such servers are just idling away with 5 percent CPU usage, have multiple gigabytes of free memory, and have I/O bandwidth to spare. Clearly, this is a waste of resources. This is where virtualization helps you out.

With virtualization, you can consolidate many servers on the same hardware. Not only will these servers make more effective use of the hardware, but because you have fewer physical servers, you will use less power and rack space. Even better, with the right software, you can move virtual servers between physical servers easily, giving you a flexible configuration.

To illustrate the principle, Figure 29.1 shows one physical server running Windows Server 2008 with Hyper-V virtualization software and a number of virtual machines (VMs). The machine running Hyper-V is known as the *host*.



Win2000 on W2008S1 - Virtual Machine Connection
File Action Media Clipboard View Help
Welcome to Windows
Microsoft
Windows 2000 Advanced Server
Built on NT Technology
achine Connection
Welcome to Windows
Windows Server 2003 R2
Enterprise x64 Edition
Copyright © 2005 Microsoft Corporation
Microsoft
Press Ctrl-Alt-Delete to begin.
Requiring this key combination at startup helps keep your
NT4 on W2008S1 - Virtual Machine Connection
File Action Media Clipboard View Help
Begin Logon
Press Ctrl + Alt + Delete to log on
Microsoft
Wind
Serv
Microsoft
BackOffice
This product is protected by US and international copy
Status: Running
Win2008-x64 on W2008S1 - Virtual Machine Connection
File Action Media Clipboard View Help
Press CTRL + ALT + DELETE to unlock this computer
Administrator is logged on.
Switch User
Windows Server 2008
Enterprise
Status: Running

FIGURE 29.1
Hyper-V on Windows Server 2008 R2 running multiple VMs

How does this work, generally speaking? Clearly, you cannot have two operating systems accessing the same hardware at the same time. One has to be in charge (that would be the host), and the other (the VM) will need to access that hardware through emulation or some other means. In principle, the same goes for the execution of CPU instructions and even memory access. Some virtualization systems can really emulate a different CPU than their own, but as you can imagine, VMs in such a system have nowhere near native performance. They need to translate each CPU instruction of the emulated system into ones that they can understand themselves. Such systems can still be useful, though, because they can solve problems that cannot be done in any other way. For instance, such systems might emulate an Intel PC on a Macintosh running a PowerPC CPU or the other way around. Another example is CCS64, which you can use to run a trusty old Commodore 64 on your Windows desktop—and quite a bit faster than the original if you want!

Modern dedicated virtualization systems like Hyper-V go out of their way to use system resources as efficiently as possible. They use real memory, and the CPU directly executes the code of the VM—with some exceptions that we will discuss later. The same argument holds for high-performance devices such as network, disk, or video interfaces. Emulation where existing hardware behavior needs to be simulated will cost you performance. Sometimes this is unavoidable, but

Hyper-V takes a different route. It uses its own driver architecture for each type of device to reduce such overhead as much as possible. This design tightly integrates with the computer architecture. From the VM you will see and use the same CPU that the host uses, so cross-CPU emulation is impossible. But that was not the point anyway. The point was to run a VM as fast as possible.

What Use Is Server Virtualization?

Now that you have some idea of what virtualization is about, let's discuss what to use it for. Some important applications are testing, consolidation of servers, and disaster recovery. These all benefit from the high degree of flexibility that virtualization offers.

The technology really got started as a test method. Administrators and consultants in need of hardware for a quick test were only too happy to use an existing machine with virtualization software to run a couple of VMs. Similarly, virtualization is great for giving technology demos. Because of the low-performance requirements for demos, you can run multiple VMs on a powerful laptop and show people how it actually works.

Testing still is a mainstay of virtualization. Larger organizations usually have multiple testing environments for various purposes. With virtualization, you can quickly add and remove VMs as required. Some organizations use a "network in a box": multiple VMs that taken together are a functionally representative copy of (part of) the production network. Whenever a new application comes along or a new infrastructure component needs to be integrated, a "network in a box" is deployed for the project to use.

A lot of administrators have a couple of VMs for their own private use so that they can quickly test and research changes before actually applying them in a production network. For instance, as an Active Directory specialist, you might run four or five DCs plus a Vista workstation in VMs on a normal 4GB desktop that is running Windows Server 2008 with Hyper-V. You might use this to research the fine points of Active Directory replication or the effects of a Group Policy change on the Vista client.

Testing is one thing, but the largest deployments of server virtualization are no doubt in datacenters where lots of servers are consolidated. A virtualized server offers the following advantages:

Conserves resources and saves costs One host running multiple VMs saves a lot of rack space, electrical power, and cooling capacity. It would not be unreasonable to consolidate 10 or 20 lowly utilized physical machines to one host machine.

Shares hardware A host offers the same "virtual" hardware to each VM. In other words, all VMs share common hardware. This makes them predictable and makes the maintenance of drivers easy. Deploying VMs is much easier than doing the same for physical machines, mainly because drivers are no longer a factor.

Increases flexibility The same feature of identical virtual hardware makes for a high degree of flexibility. You can move VMs between hosts for load distribution or maintenance.

Joins legacy operating systems A lot of organizations will be running a mix of operating systems—not only Windows 2008 but likely also older systems running Windows 2003 or even Windows 2000. Chances are that those older systems don't require much in the way of computing power on modern hardware. This makes them ideal candidates for consolidation. One host will have enough power for many of these *legacy* systems. Legacy consolidation will also benefit from the uniform virtual hardware.

In large environments with large storage area network (SAN) deployments, mirrored datacenters, and similar infrastructure, server virtualization is an asset for disaster recovery. Not only is server consolidation a benefit here, but all virtual machines have the same type of virtual hardware. You will have *no* driver or HAL issues when restarting a virtual machine on a new host, assuming of course that the new VM has the same configuration as the old one.

Each technology has its downside, and virtualization is no exception. Some may impact you more than others, but here are a few:

Increases complexity Virtualization adds a layer of complexity to the existing environment. You now need to know whether a given server is a VM or a physical server, or perhaps a host for VMs. Let's take the example of a SQL Server administrator. Before, he would be responsible for all the SQL Server: software, hardware, and configuration. If SQL Server is virtualized, he needs to depend on the administrators of the host server to keep his VM running. Any impact on the host will also impact the SQL Server VM.

Strains infrastructure A full-blown VM environment will need additional infrastructure: a SAN is mandatory in large environments, as is dedicated management software and a dedicated high-speed IP network.

Can cause large-scale failure If you are not specifically designing for service availability, a host is a single point of failure. If it goes down unexpectedly, for instance because the CPU overheats and shuts down, it will take all running VMs down with it.

Requires special maintenance If you have a library of offline VMs, you will need to do some form of maintenance on those as well, such as applying patches.

Creates unique security issues There are some nonobvious security considerations related to virtualization. For instance, in a SAN-based environment, you will have two additional groups of administrators who can access data in a VM: the administrators responsible for the host machines and the SAN administrators. A SQL Server administrator responsible for a VM may not be aware that those other administrators can in principle access his data at will, assuming the desire and knowledge to do so.

Requires a learning curve When you deploy a new technology, you need to learn it. While you are learning it, you will make mistakes. Some of those will impact your production environment. That's nothing new, but it's still a factor to consider.

Clearly, it's a balancing act. For most organizations (but not all), the advantages will outweigh the downsides. The main point to take away here is that server virtualization has arrived and is here to stay. If your organization has not deployed it yet, chances are that you will soon.

What Do You Need to Get Started with Hyper-V?

Not surprisingly, there are hardware and software requirements for running Hyper-V. In addition, there are some intricate licensing questions involved, which we'll discuss later in this chapter.

Hardware Requirements

The base requirements for running Hyper-V are quite simple. You need an x64-based CPU, hardware-assisted virtualization, and hardware Data Execution Prevention (DEP). Most, but not all, computers meeting these minimum requirements will run Hyper-V. A common problem is that although these features are offered by the system, they are not enabled in the BIOS. Make sure these features are turned on. If you need to change the DEP or virtualization settings, be aware that a cold boot is required: the computer must be turned completely off. A reset or software reboot is not sufficient. Note that server hardware from 2008 or earlier may need a BIOS upgrade as well.

Both Intel and AMD support DEP on the hardware level with modern processors, but they call it differently. Intel has the XD bit (eXecute Disable), and AMD has the NX bit (No eXecute). Look for that in the BIOS. The situation is a bit different for the virtualization extensions. Intel and AMD both created their own extensions around the same time in 2006. They roughly do the same thing but are incompatible. However, all major server virtualization products including Hyper-V support both the Intel and AMD extensions. One point to be aware of is that some BIOS systems for AMD CPUs refer to AMD-V as "SVM" or "secure virtualization." Also, because of incompatibilities, some early systems featuring AMD-V cannot actually run Hyper-V or can do so only with a specific BIOS version.

If you want to be really sure that the servers you will buy will run Hyper-V, you should check directly with your vendor. They are responsible for testing that Hyper-V actually runs on their hardware. Most large vendors also participate in the "Certified for Windows Server 2008" program, which requires them to test their hardware using Microsoft-standard procedures. After the server passes the test, the vendor can submit the configuration to Microsoft for inclusion on the public catalog. However, not all vendors submit all of their hardware. That's why you should ask them directly. The Microsoft catalog is at `http://windowsservercatalog.com`. You can specifically search for *Hyper-V compatible systems*.

Let's talk about the specifics now that you know generally what features to look for. There are two things to keep in mind when selecting hardware for virtualization: VMs like a lot of memory, and disk I/O bandwidth is critical. For a low-end testing system dedicated to virtualization, you should probably select something with at least 8GB of memory and a single or dual CPU motherboard with quad-core CPUs.

Get as many disk spindles as you can reasonably afford. Four independent medium-capacity disks will be faster than two large-capacity disks when using multiple VMs. If you have the money for it, invest in high-RPM disks. SATA disks are OK for testing and nondemanding applications. For best performance, SCSI or SAS are better, generally speaking. Avoid a RAID-5 configuration because it is slow on write operations. RAID 0 and RAID 1 combinations are fine for low-end systems. For high-end applications, consider RAID 10. Finally, think about the networking. The general recommendation is to have at least two NICs, one to manage the host and another for the VMs to access the network. If you expect high network throughput or iSCSI connections that require dedicated NICs, you will need even more.

Software Requirements

Now that you have your hardware sorted, we'll discuss the software side. The first thing to note is that a 64-bit version is required. Of course, this is the only option with Windows 2008 R2, but Windows 2008 also has 32-bit editions. Table 29.1 summarizes the options.

Table 29.1: Windows 2008 Editions and Hyper-V

Windows 2008 Edition	Has Hyper-V Role
Windows Server 2008 (R2) Standard edition x64	Yes
Windows Server 2008 (R2) Enterprise edition x64	Yes
Windows Server 2008 (R2) Datacenter edition x64	Yes
Microsoft Hyper-V Server 2008 (R2)	Yes
Windows Server 2008, any x86 edition	No
Windows Web Server 2008 (R2)	No
Windows Server 2008 (R2) for Itanium	No

Note the specific edition named Hyper-V Server. This is basically a 64-bit version of Server Core with Hyper-V enabled by default. We will discuss it briefly later in this chapter. Its main benefit over the regular Server Core editions is that it's available as a free download. The remaining editions differ in their failover clustering capabilities and licensing models. Briefly, Standard comes with one license for a Windows-based VM and has no clustering option. Enterprise has four VM licenses and allows clustering. Datacenter is licensed per physical CPU and allows unlimited VMs with clustering.

Windows Server 2008 Versions Without Hyper-V

As you can see from Table 29.1, not all editions of Windows 2008 R2 can run Hyper-V. Although that is clear enough, the situation was confused with its predecessor Windows 2008. It has Standard edition, Enterprise edition, and Datacenter editions with Hyper-V, and once again without Hyper-V at a minimal price difference. The reason for that was never really clear, but it is probably related to market regulation efforts. If you are buying Windows 2008, make sure you get the one with Hyper-V.

From the table it's clear that Hyper-V is a server component. Microsoft will not release a version of Hyper-V for a client OS such as Windows 7, Vista, or XP. Also, Windows 2008 is the first server version to support Hyper-V. It will not be implemented for earlier releases of the Windows operating system. If you want to run server virtualization on any of these other platforms, you need to look at other products in the market. Finally, if you're running Itanium servers (IA64), you should know that when Hyper-V was released, it did not run on IA64, and it probably never will.

The Hyper-V Feature Set

We'll now cover the functionality that Hyper-V offers in Windows Server 2008 R2. Since the first version in Windows Server 2008 SP1 (remember, there never was a SP0), there have been some quiet changes in certain limits, such as the number of supported processor cores or number of virtual machines. Those limits may well change again, so check the Microsoft site for the current feature set. For now, see Table 29.2.

Table 29.2: Windows 2008 R2 Hyper-V Feature List

Feature	Windows Hyper-V Server, Standard Edition	Enterprise Edition, Datacenter Edition
x86 VM	Yes	Yes
x64 VM	Yes	Yes
IA64 VM	No	No
Maximum number of VMs	384	384
Maximum host memory	32GB	1TB
Maximum host processors	64 cores	64 cores
Maximum VM memory	32GB	64GB
Hot-add VM memory	No	No
Maximum VM processors	4 per VM	4 per VM
Hot-add VM processor	No	No
IDE adapters	2	2
Maximum IDE devices	4	4
SCSI adapters	4	4
Maximum SCSI devices	255	255
Hot-add hard disk	Yes	Yes
Virtual networks	Unlimited	Unlimited
Maximum number of VM network cards	8	8
Hot-add virtual NIC	No	No
Failover clustering support	No*	Yes
Live Migration	No**	Yes

** Hyper-V Server R2 does support clustering and Live Migration.*
*** Live Migration is not included with Windows 2008. It's new with R2.*

There is not much difference between the editions until you really start to scale up in memory or require high availability using the various clustering options. Most of these limits are hard limits, in the sense that they cannot be exceeded. The exception is the maximum number of VMs per host. This is a soft limit, determined by the Microsoft testing labs. Perhaps the most stringent limit here is the maximum number of virtual processors, which translates to the number of physical CPU cores actually used. Currently, you cannot have a VM with more than four virtual CPUs, which means that your maximum CPU power is limited. On the other hand, if you have an application that really uses all your CPUs to the maximum, then it's probably not a great candidate for virtualization anyway. One notable missing feature is USB support, which would be great for lab situations. Microsoft's point of view is that Hyper-V is a server role where USB is not required, so there is little chance of ever having this feature added. Another feature set that is likely to change is the type of devices that support the ability for adding or removal from a running VM. Currently, this works only with virtual hard disks. It would certainly be nice to be able to do the same thing for memory, NICs, and CPU. Microsoft is planning this for future releases.

The final two items in the list are failover clustering support and Live Migration. Briefly, *failover clustering* is the ability to move a running application (such as a virtual machine!) from one server to another, either on-demand or after a detected hardware or failure. Shared storage between the servers is required, using Fibre Channel or iSCSI connections. Live Migration builds on failover clustering to move VMs between hosts with subsecond downtime. The point of this feature is that online users of the VM won't even know that the move occurred. With Live Migration, you can build a very flexible virtualization farm. We will walk you through how to set up Live Migration in the "Moving VMs: Quick Migration and Live Migration" section.

Installing the Host with a Virtual Machine

This book aims to be a practical resource for you. Although some theory is essential to fully understand Hyper-V, we can leave that for later. In this walk-through, you will learn how to install Hyper-V on Windows Server 2008, how to get a VM going, and how to connect to its console. Along the way, you will pick up practical details dealing with Hyper-V.

To get started, you need the following:

- You need Hyper-V capable hardware. Remember, VMs like memory and lots of disk I/O. One NIC is required; two would be nice.
- You need a version of Windows 2008 or 2008 R2 that supports Hyper-V. If you are using Windows 2008, make sure to install SP2 or newer first.
- An ISO file with your Windows OS of choice will come in handy but is not strictly required.
- You need IP addresses to use for the host and the VM.

Why You Should Avoid Windows Server 2008 SP1

The first version of Windows Server 2008 has the Hyper-V role but in a beta version. Don't use it! Although there have been interim updates to get the Hyper-V version up to 1.0, your best option is to simply install SP2 for Windows Server 2008, which includes many additional fixes. Of course, Windows Server 2008 R2 has no such problems since it includes the production version of Hyper-V 2.0.

Installing and Configuring Hyper-V

Since you have gotten this far in the book, you know how to install a server, join it to a domain, and so on. We will skip the details for the base installation. Table 29.3 shows the suggested configuration for a test setup. Feel free to vary the instructions as you see fit. This walk-through uses Windows 2008 R2, but there is hardly any difference if you use Windows 2008.

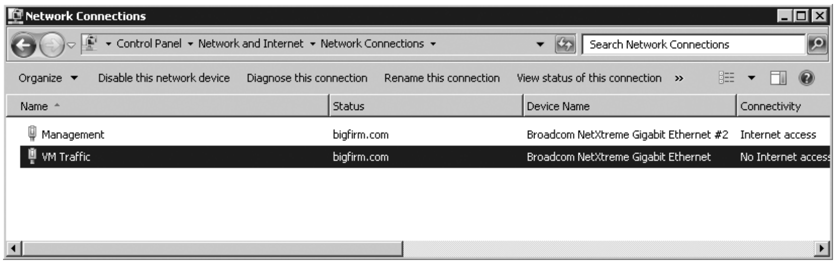
Table 29.3: Hyper-V Host System

Setting	Configuration
Internal memory	8GB; 4GB is the practical minimum.
Hard disks	2×200GB or more. One disk is acceptable for a test system, but expect low performance.
Partitions	Disk 1: System on C. Disk 2: Reserved for Hyper-V on E.
Network	2×1Gbit highly recommended. One NIC acceptable for test.
Operating system	Hyper-V enabled editions of Windows 2008 R2 or Windows 2008 SP2.
Installation type	Full GUI to follow the examples; Server Core is discussed later.
Hostname	bf5 to follow the examples; anything you like is fine.
IP configuration	Address: 192.168.1.54/24. Gateway: 192.168.1.1. DNS: 192.168.1.51.
Active Directory	Domain-joined recommended for production. Workgroup is workable for testing, although remote management is not possible because it requires Active Directory.

Just about the only thing you need to decide before you install the Hyper-V role is which NIC to use for managing the Hyper-V host. The idea is to have at least two NICs in the host, although you can do with one if you really must. Expect no performance miracles in that case. With two NICs available, dedicate one to managing the host and the second for VM network traffic. To make this obvious, one trick is the rename the network connections, as shown in Figure 29.2.

Let's start with the installation of the Hyper-V role:

1. Install the server using the parameters in Table 29.3 for the version, server name, IP configuration, and so on. Make sure to have dedicated disks or partitions for Hyper-V data.
2. Join the computer to the domain if you want to follow some of the later examples. The domain bigfirm.com is used throughout this book.


Network Connections
Control Panel ▸ Network and Internet ▸ Network Connections ▸
Search Network Connections
Organize ▾
Disable this network device
Diagnose this connection
Rename this connection
View status of this connection »

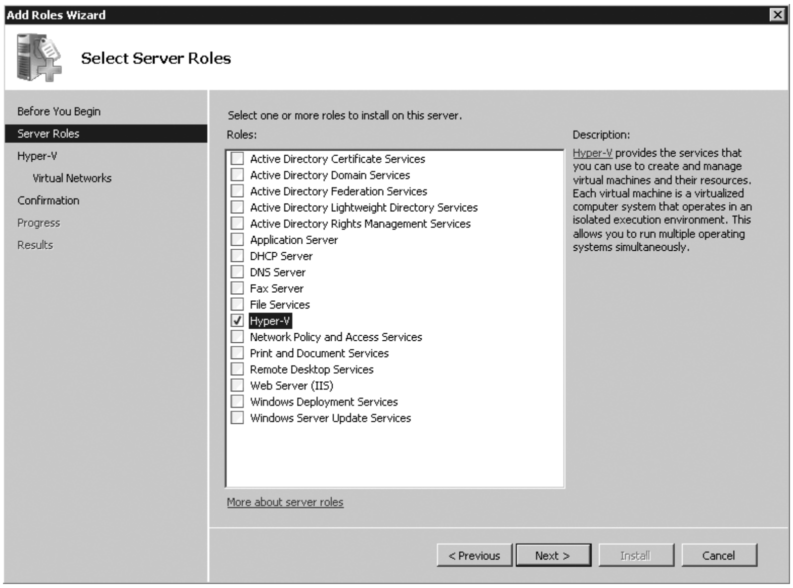
Name	Status	Device Name	Connectivity
Management	bigfirm.com	Broadcom NetXtreme Gigabit Ethernet #2	Internet access
VM Traffic	bigfirm.com	Broadcom NetXtreme Gigabit Ethernet	No Internet access

FIGURE 29.2 Renaming the network connections to reflect their role in the Hyper-V host

After you install the Hyper-V host server, you are ready to install the Hyper-V role. You might want to do this using a console session and avoid Remote Desktop. During the installation, the network connection will be broken once or twice because of upgraded network components.

3. Log on to the console on the host. At this time, do not use an RDP session, but use the physical console.

4. Open Server Manager, as shown in Figure 29.3. Click Add Roles, and select the Hyper-V role. As you can see, this is a clean server without any additional roles. This is the best practice for a Hyper-V host. Any other functionality should go to the VMs. In fact, the licensing structure for VMs stipulates that you do not use the host for anything but Hyper-V. We discuss the details later.



Select Server Roles

Select one or more roles to install on this server.

Roles:

□ Active Directory Certificate Services
□ Active Directory Domain Services
□ Active Directory Federation Services
□ Active Directory Lightweight Directory Services
□ Active Directory Rights Management Services
□ Application Server
□ DHCP Server
□ DNS Server
□ Fax Server
□ File Services
☑ Hyper-V
□ Network Policy and Access Services
□ Print and Document Services
□ Remote Desktop Services
□ Web Server (IIS)
□ Windows Deployment Services
□ Windows Server Update Services

Description:

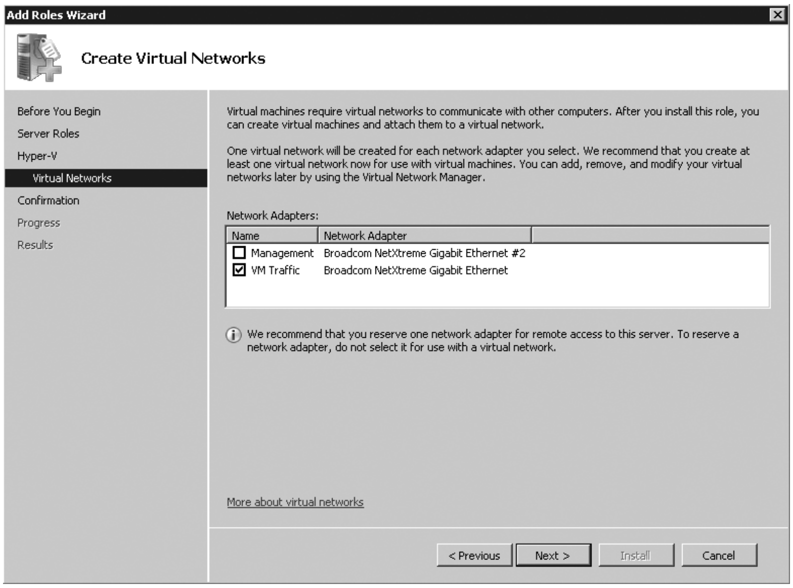
Hyper-V provides the services that you can use to create and manage virtual machines and their resources. Each virtual machine is a virtualized computer system that operates in an isolated execution environment. This allows you to run multiple operating systems simultaneously.

FIGURE 29.3 Adding the Hyper-V role using Server Manager

5. Click Next to start the Hyper-V role wizard, and read the introduction to Hyper-V if you like.

6. Click Next for the NIC selection screen, as shown in Figure 29.4. This requires you to select a NIC for a virtual network. You will find the details later in this chapter, but briefly, a virtual network is a Hyper-V software network switch. For each NIC that you select here, one such switch will be created. If you have two NICs or more, you should leave one blank. An unselected NIC will not have a switch associated with it and can be used to manage the host. If you have only one NIC, select it. If you don't select any NIC, your VMs have no easy way to communicate with the outside world, although you could correct this later. Because we remembered to give sensible names to the network connections, it's now easy to select the one we will use for VM network traffic.

FIGURE 29.4
Select one NIC to use for VM traffic. The unselected NIC is used to manage the Hyper-V host.



Virtual machines require virtual networks to communicate with other computers. After you install this role, you can create virtual machines and attach them to a virtual network.

One virtual network will be created for each network adapter you select. We recommend that you create at least one virtual network now for use with virtual machines. You can add, remove, and modify your virtual networks later by using the Virtual Network Manager.

Network Adapters:

Name	Network Adapter
□ Management	Broadcom NetXtreme Gigabit Ethernet #2
☑ VM Traffic	Broadcom NetXtreme Gigabit Ethernet

We recommend that you reserve one network adapter for remote access to this server. To reserve a network adapter, do not select it for use with a virtual network.

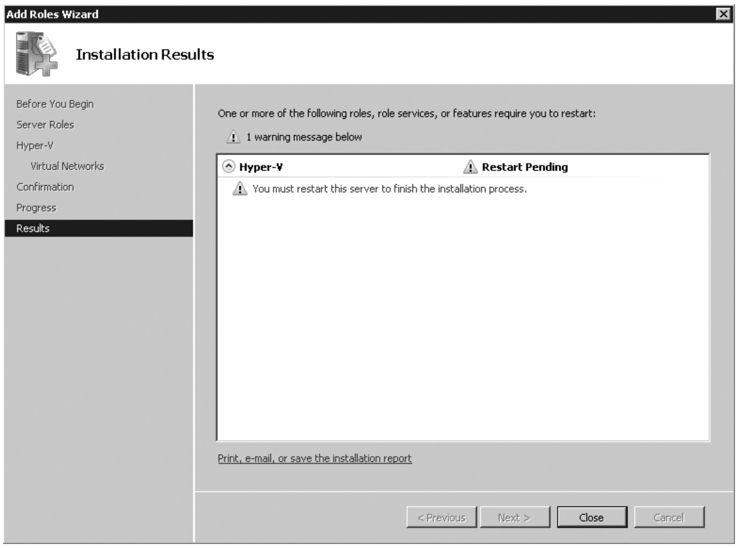
7. Click Next. A confirmation screen appears telling you that a reboot might be required. In fact, count on it.

8. Click Install. This should not take longer than a minute. After it's done, it tells you again that a reboot is needed, as shown in Figure 29.5.

9. Click Close. A dialog box offers to reboot the machine for you. Click Yes to accept its offer. The machine will reboot.

10. When it comes back, log on using an Administrator account. After the desktop initializes, the configuration process will resume. Give it some time to do its job and finalize the installation (Figure 29.6).

Let's take a look at what you've got now. Start the Hyper-V Manager from Administrative Tools. We have a console to administer the Hyper-V host setting with the usual layout, as shown in Figure 29.7. The left pane holds the Hyper-V host service you want to manage. Shown is the current server bf5, but you can have more than one server listed. Later, when we discuss a Server Core setup, you will add that server here.



One or more of the following roles, role services, or features require you to restart:

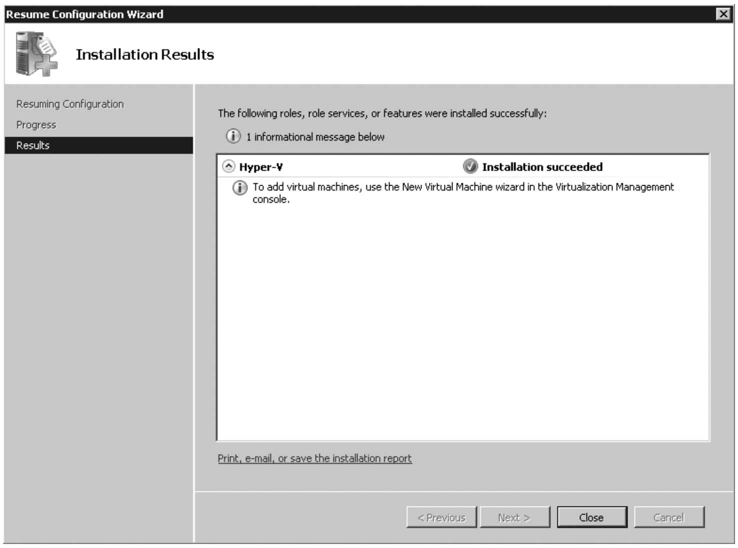
1 warning message below

Hyper-V Restart Pending

You must restart this server to finish the installation process.

Print, e-mail, or save the installation report

FIGURE 29.5 The final step of the wizard tells you that all is fine and that a reboot is required.



The following roles, role services, or features were installed successfully:

1 informational message below

Hyper-V Installation succeeded

To add virtual machines, use the New Virtual Machine wizard in the Virtualization Management console.

Print, e-mail, or save the installation report

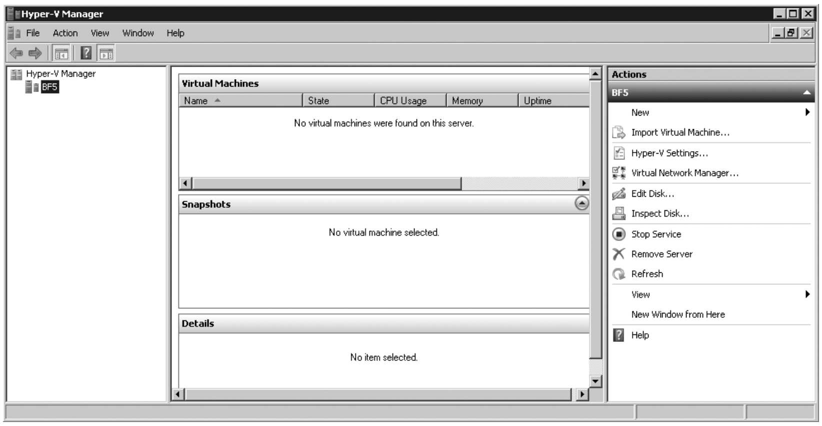
FIGURE 29.6 After the reboot and logging on, the Hyper-V installation wizard finalizes the installation. After this, Hyper-V is ready for use.

The middle pane has three regions:

Virtual Machines The list of VMs on this host with some relevant parameters such as its current state (running, off, saved, and so on).

Snapshots Briefly, these are point-in-time disk images, including memory and CPU states. You create a snapshot of a server if you want to be able to roll back to such an image. We will discuss this extensively later in the chapter.

Details This contains extra information on the currently selected VM—if any.



No virtual machines were found on this server.

No virtual machine selected.

No item selected.

FIGURE 29.7
The Hyper-V management console

The rightmost Actions pane is the most interesting for now. It allows you to manage various aspects of the Hyper-V setup. Some are pretty obvious, such as stopping or starting the Hyper-V Virtual Machine Management Service or removing the current server from the console. Others such as the Network Manager, Edit Disk, and Inspect Disk are not so obvious if you are new to the virtualization game. Before we go on with configuring Hyper-V, we'll take a step back and discuss virtual disks and virtual networks.

Virtual Disks: The Short Version

When you create a virtual machine, you assign virtual devices for network, video, and so on. Your VM also needs a virtual disk, of course. The question is, what disk? The physical disks are in use by the host! The solution is simple and obvious: because we are virtualizing and emulating already, use a file instead and present that to the VM as a disk. So, a virtual disk is nothing more than a very large file.

In fact, you can have multiple flavors. For test purposes, use a dynamically expanding disk. A virtual disk in this format will allocate the physical disk space it needs, and no more. For instance, you might allocate 127GB for a Windows 2008 Server VM, which only uses less than 10GB after a basic install. Every time the VM needs more disk space, the file is expanded. The expanding virtual disk makes best use of the available disk room on the host. But as you can imagine, there is a performance cost involved when an expansion is needed, and you run the risk of fragmentation. Also, a clear disadvantage is that you may run out of disk space on the host when dynamic virtual disk expands beyond available room. When that happens, all VMs are frozen by Hyper-V, and the event log turns red.

To get rid of overhead, you can also choose to use a fixed disk size. The full size of the virtual disk will be allocated when the disk is created. This is the recommended format for most production loads. The third flavor is a pass-through disk, where you assign a physical disk for dedicated use by a VM. This is possible when this disk is not used by the host. Pass-through disks are useful to connect a VM directly to a SAN or to an iSCSI target. Generally speaking, you will only use a pass-through disk in clustering scenarios and similar. If you don't need them, stick with normal virtual disks and let the host mount disks on the SAN or iSCSI target.

In fact, there is one more flavor: you can have a virtual floppy disk. Interestingly, you do not have the option to connect a VM to a physical floppy drive—not that you are likely to have one.

Virtual Networks: The Short Version

Early on in the server virtualization game, it was recognized that it would be very useful to have multiple types of network connections for VMs:

External This is full network access. Such a VM can communicate through the NIC of the host with the outside world. Any other network device will see the VM as if it were a normal computer. The only exception is the (physical) network switch connected to the host. This switch sees one host with two MAC addresses and two IP addresses. Be aware that some secure network environments may not allow this. The symptom would be that your VM can talk to your host but not to any other machine on the network even though you have specified external access.

Internal This is access between the host and VM only. This is suitable for most test installations.

Private VMs are connected to each other but cannot see the host. Effectively, they are completely isolated from the physical network. Again, this is suitable for testing. For instance, when you test the DHCP features of Windows 2008, you would not want to try this on the physical network. In some companies, that could get you fired! When in doubt, a private network is the safest option.

Later, this concept was expanded from virtual NICs to virtual switches with the same type of behavior. You will not be surprised to learn that Hyper-V implements them. Virtual switches are 100 percent software but are not visible outside of a host and its VMs since they are implemented in software. They deliver high-speed connections between VMs. On a virtual switch, you “plug in” virtualized NICs of VMs. A useful feature of virtual switches is that you can change their scopes. You might start a switch as internal and connect it to a physical NIC later.

Using virtual switches, you can generate quite complicated networks inside the host. Clearly, this is most useful in test situations; in production scenarios, all VMs should be able to talk to the physical network. Using three virtual switches, you could, for instance, build a classic DMZ setup: one switch for the outside interface connected to a physical NIC, one switch for the DMZ hosts, and one switch for the internal LAN.

Configuring the Hyper-V Host

With those asides out of the way, we can return to configuring the Hyper-V service in the right Actions pane of the Hyper-V management console. The New Wizard and the Import Virtual Machine Wizard deal with creating VMs and virtual disks, so we’ll leave that for the next section.

The Hyper-V Settings dialog box shown in Figure 29.8 allows you to set some parameters for Hyper-V. Not many, actually. Hyper-V was really designed to run out of the box with minimal tuning.

Each parameter shows its current setting. The two most important parameters are the default paths for virtual hard disks and for virtual machine configurations. In this example, they have been changed to point to a folder on E, a nonsystem partition and preferably a dedicated drive. In the Hyper-V philosophy, the files for virtual disks are separate from virtual machine settings. Make sure you set them to something sensible and different from each other. By default they end up somewhere in your local profile. Microsoft should probably have made these mandatory inputs during the Hyper-V setup.

The remaining settings are all related to the way you use Hyper-V and access the VMs:

Keyboard Specify how Windows special keys should behave in the VM console.

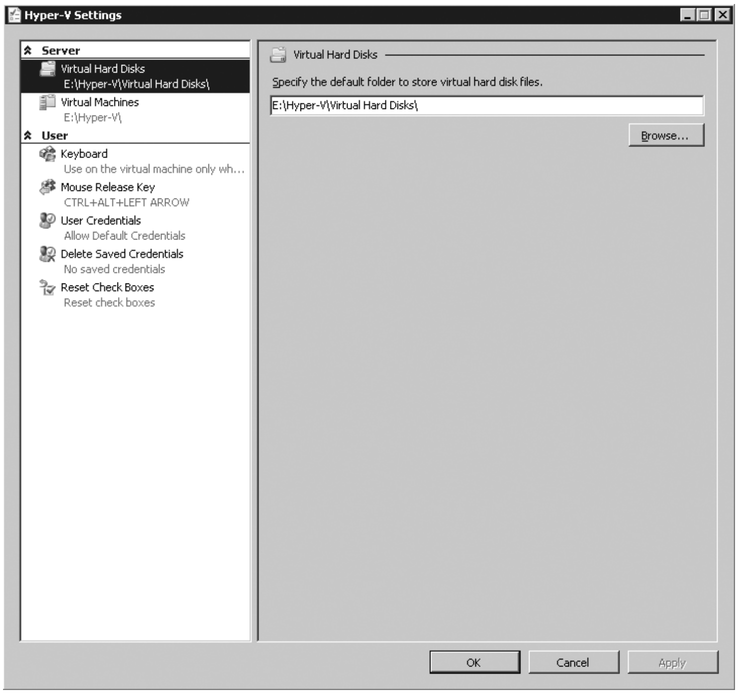
Mouse Release Key Specify what keyboard combination should be used to release input focus from a VM console.

User Credentials Specify what user ID should be used to access a VM console. By default, this is the currently logged on user. If you deselect this option, you will be required to supply a user ID and password when connecting to a VM console. These credentials can be cached, if you choose to do so.

Delete Saved Credentials If you have ever saved credentials, this dialog box has a button that will clear them.

Reset Check Boxes At various points in the Hyper-V console you can select boxes indicating that you never want to see a particular dialog box again. This option brings them all back.

Figure 29.8
The Hyper-V Settings dialog box. Set the default folders for virtual hard disks and VMs before you do anything else.



Hyper-V Settings

Server
Virtual Hard Disks
E:\Hyper-V\Virtual Hard Disks\
Virtual Machines
E:\Hyper-V\
User
Keyboard
Use on the virtual machine only wh...
Mouse Release Key
CTRL+ALT+LEFT ARROW
User Credentials
Allow Default Credentials
Delete Saved Credentials
No saved credentials
Reset Check Boxes
Reset check boxes

Virtual Hard Disks
Specify the default folder to store virtual hard disk files.
E:\Hyper-V\Virtual Hard Disks\

Continuing the overview of the Actions pane, we have three more wizards to discuss briefly. Virtual Network Manager is the central management point for virtual switches (or *networks*, as Hyper-V prefers to call them). Using this wizard, you view, create, and edit networks. Networks of type External can be assigned to physical NICs. In this example, we used the Hyper-V setup wizard to assign one NIC to the VMs, as shown in Figure 29.9. Be default, the corresponding virtual switch is set to External. Also, the host can access this switch, meaning that traffic between the VMs and their host can use this internal virtual switch.

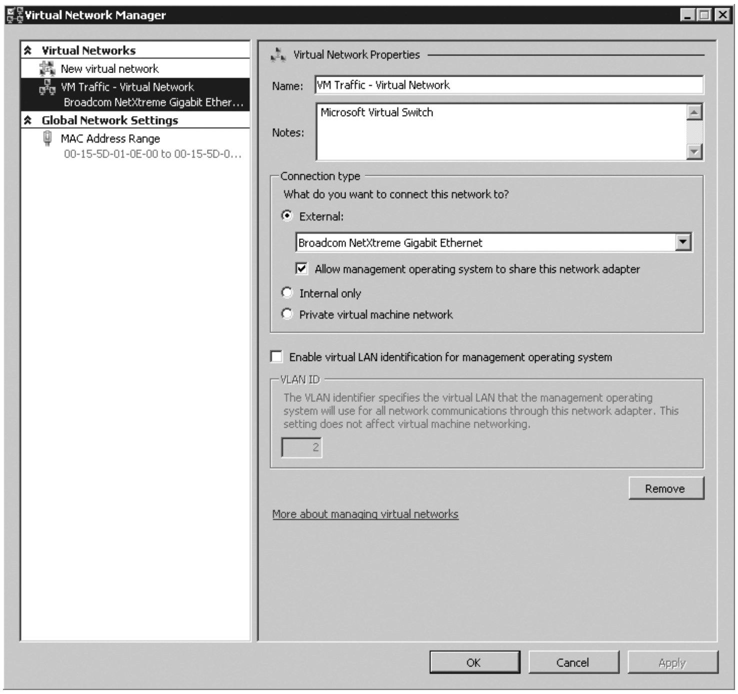


FIGURE 29.9
The virtual network manager after setup. The virtual switch called VM Traffic is currently the only one available for VMs.

Virtual Network Manager

Virtual Networks
- New virtual network
- VM Traffic - Virtual Network
  Broadcom NetXtreme Gigabit Ether…

Global Network Settings
- MAC Address Range
  00-15-5D-01-0E-00 to 00-15-5D-0…

Virtual Network Properties

Name: VM Traffic - Virtual Network

Notes: Microsoft Virtual Switch

Connection type

What do you want to connect this network to?

● External:

Broadcom NetXtreme Gigabit Ethernet

☑ Allow management operating system to share this network adapter

○ Internal only

○ Private virtual machine network

□ Enable virtual LAN identification for management operating system

VLAN ID

The VLAN identifier specifies the virtual LAN that the management operating system will use for all network communications through this network adapter. This setting does not affect virtual machine networking.

2

Surprisingly perhaps, the Edit Disk Wizard cannot be used to create virtual disk. That activity is reserved for the New Wizard. The Edit Disk Wizard does allow you to change a disk from dynamic to fixed, to expand its size, and to remove blank space from the virtual disk file. The Inspect Disk dialog box inspects a virtual disk for consistency.

Now that you have an understanding of how to install and configure the Hyper-V service, we'll turn to the heart of the matter: creating and managing virtual machines.

Configuring a Virtual Machine

At this point, you have the Hyper-V service up and running and are ready to start configuring a VM. Before you get started, check that you have all that you need:

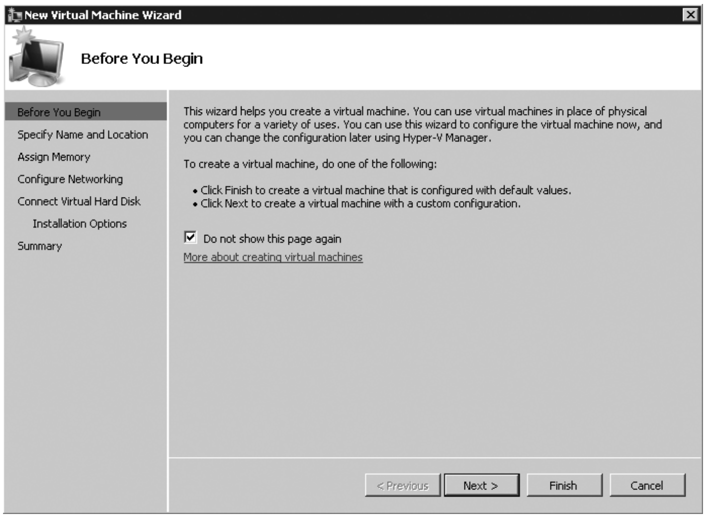
- A CD or DVD containing the operating system you want to install. A good alternative would be to use an ISO image of this CD/DVD, because an ISO image works much faster and is more convenient. In this example, we'll use an ISO file for Windows Server 2008 R2. The server bf5 has been set up with a share hosting the ISO files: `\\bf5\ISO`.
- A name for the new server.
- An idea on which network (External, Internal, or Private) you want to use the VM, and an IP address to go with it.
- How much memory to use. Make sure you are not too conservative with this. If the VM needs to start swapping its memory, it puts a heavy load on the disk I/O capacity of the

host—capacity that would be better used to accommodate more VMs. For a base install or Windows Server 2008 R2, use 1GB.

- The virtual disk type to use: dynamic for testing or fixed for production purposes.

Conceptually, it takes two steps to create a VM from scratch. First you configure the virtual hardware of the VM, and then you boot the VM and start installing the operating system. The New Wizard takes care of configuring the VM.

1. Open the Hyper-V management console, and select New ➢ New Virtual Machine. Figure 29.10 shows the first screen of the wizard, telling you briefly what a VM is for. Use the check box to skip this screen in the future. At this point, you could even use Finish to create a VM with the default settings, but that's not a good idea. Generally, no two VMs are the same.



This wizard helps you create a virtual machine. You can use virtual machines in place of physical computers for a variety of uses. You can use this wizard to configure the virtual machine now, and you can change the configuration later using Hyper-V Manager.

To create a virtual machine, do one of the following:

- Click Finish to create a virtual machine that is configured with default values.
- Click Next to create a virtual machine with a custom configuration.

☑ Do not show this page again

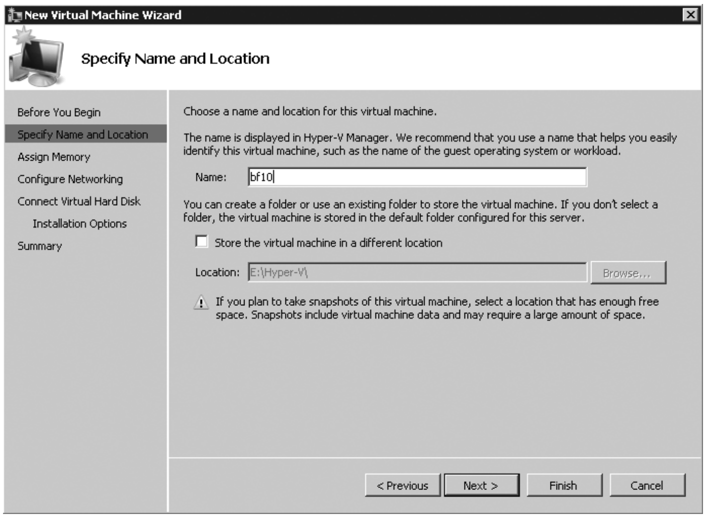
More about creating virtual machines

FIGURE 29.10 The start of the VM creation wizard. Select the check box to never see this first screen again.

2. Enter the name for the VM, as shown in Figure 29.11. Note that this is the friendly name used in the management console, not the actual hostname. Of course, it would make sense to make those names (almost) the same. To follow the example, set the name to bf10. By default, this name will also be used for its first virtual disk. To be clear, bf10 is not the hostname of the VM, although it makes good sense to keep the VM name and its hostname the same. If you want the configuration of the VM in a nondefault place, select the check box and the path you want. Remember, you should have set the default while configuring the host.

3. Click Next. The Assign Memory page asks you to specify the amount of memory for the VM. Specify 1024MB, as shown in Figure 29.12, as a minimum for a Windows Server 2008 R2 VM. Make sure to use enough memory. If the OS is too low on memory, it will start swapping it to disk and slowing down other VMs using the same disk. This is a general point to keep in mind: disk I/O is a precious resource on a virtualization host. Use it wisely.

FIGURE 29.11
Give the new VM its name.



Choose a name and location for this virtual machine.

The name is displayed in Hyper-V Manager. We recommend that you use a name that helps you easily identify this virtual machine, such as the name of the guest operating system or workload.

Name: bf10

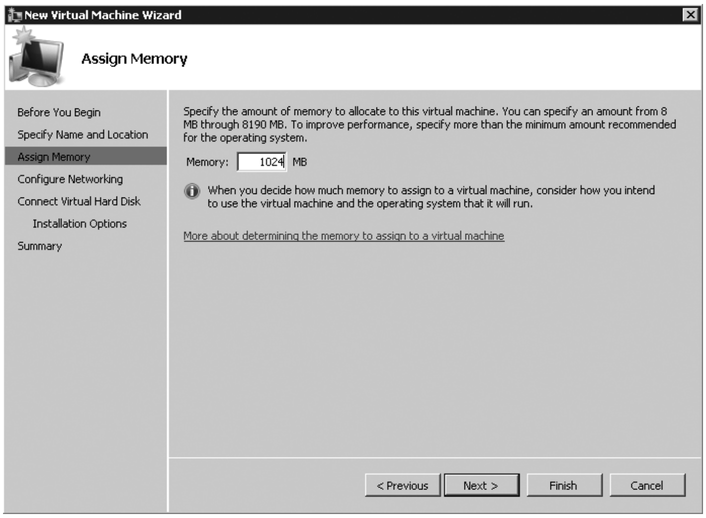
You can create a folder or use an existing folder to store the virtual machine. If you don't select a folder, the virtual machine is stored in the default folder configured for this server.

□ Store the virtual machine in a different location

Location: E:\Hyper-V\

If you plan to take snapshots of this virtual machine, select a location that has enough free space. Snapshots include virtual machine data and may require a large amount of space.

FIGURE 29.12
Specify enough memory for the OS you will install.



Specify the amount of memory to allocate to this virtual machine. You can specify an amount from 8 MB through 8190 MB. To improve performance, specify more than the minimum amount recommended for the operating system.

Memory: 1024 MB

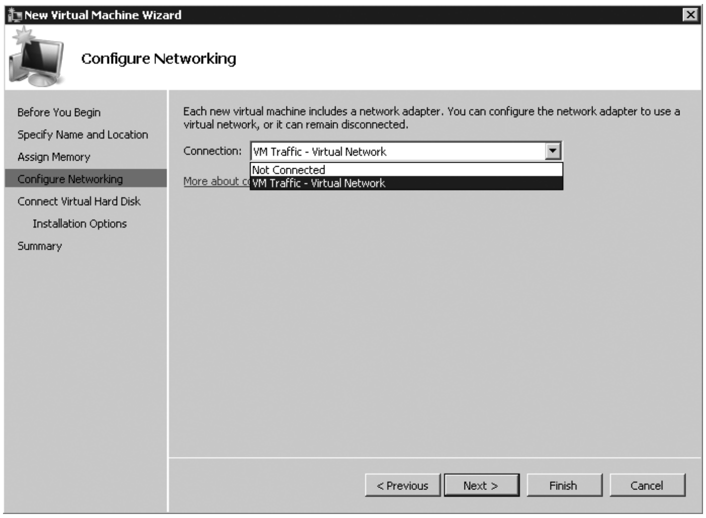
When you decide how much memory to assign to a virtual machine, consider how you intend to use the virtual machine and the operating system that it will run.

More about determining the memory to assign to a virtual machine

4. Click Next. The Configure Networking page shown in Figure 29.13 is used to specify the virtual switch to use. The default is Not Connected, which is a safe but not very useful option. Select the VM Traffic virtual switch to connect to. Remember, this switch is external and allows the VM to talk to the outside world directly.

5. Click Next for the Connect Virtual Hard Disk page shown in Figure 29.14. Here you will create a new virtual disk or assign an existing one. The defaults are good for a test setup: a new virtual disk in the default location, 127GB in size. Although the dialog box does not show it, this disk will be dynamic. If you want a fixed size disk, you must build one before you create the VM or attach it afterward. The wizard proposes a default name for the virtual disk. Accept it if you like it.

FIGURE 29.13
Connect the VM to the outside world using an externally connected virtual switch.



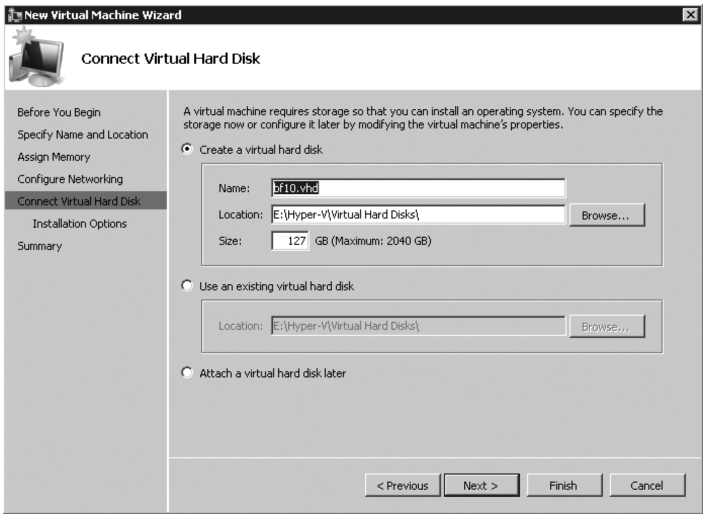
New Virtual Machine Wizard

Configure Networking

Each new virtual machine includes a network adapter. You can configure the network adapter to use a virtual network, or it can remain disconnected.

Connection: VM Traffic - Virtual Network

FIGURE 29.14
The new virtual disk is dynamic, which means that its real-world size is roughly as large as the data it contains, much less than 127GB usually.



New Virtual Machine Wizard

Connect Virtual Hard Disk

A virtual machine requires storage so that you can install an operating system. You can specify the storage now or configure it later by modifying the virtual machine's properties.

● Create a virtual hard disk

Name: bf10.vhd

Location: E:\Hyper-V\Virtual Hard Disks\

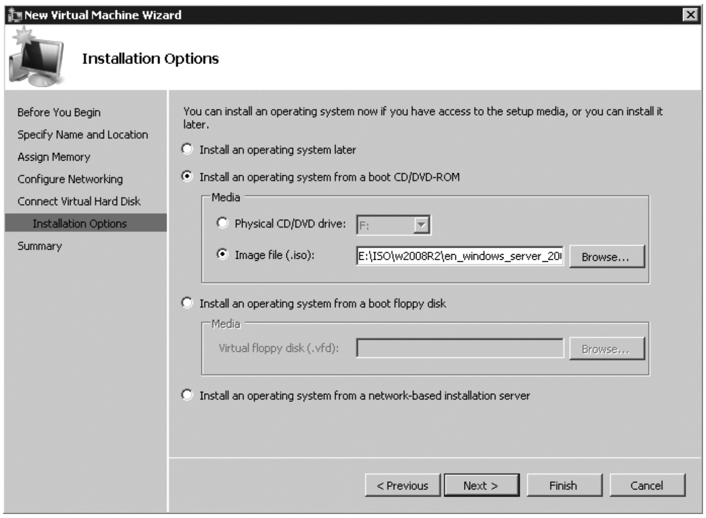
Size: 127 GB (Maximum: 2040 GB)

○ Use an existing virtual hard disk

Location: E:\Hyper-V\Virtual Hard Disks\

○ Attach a virtual hard disk later

6. Click Next for the final options. This page basically allows you to set the initial installation media: a physical or virtual (ISO) CD/DVD, a virtual floppy disk, or something from the network. In this example, you install using an ISO file. Use the radio button "Install an operating system from a boot CD/DVD-ROM" to get started. Here you see one of the great conveniences of the VM world: you can use ISO files directly, as shown in Figure 29.15. Browse to the spot where you have stored the ISO for Windows 2008 R2. Of course, if you only have the physical DVD, you can use that as well.



New Virtual Machine Wizard

Installation Options

Before You Begin
Specify Name and Location
Assign Memory
Configure Networking
Connect Virtual Hard Disk
Installation Options
Summary

You can install an operating system now if you have access to the setup media, or you can install it later.

○ Install an operating system later

● Install an operating system from a boot CD/DVD-ROM

Media

○ Physical CD/DVD drive: F:

● Image file (.iso): E:\ISO\w2008R2\en_windows_server_20 Browse...

○ Install an operating system from a boot floppy disk

Media

Virtual floppy disk (.vfd): Browse...

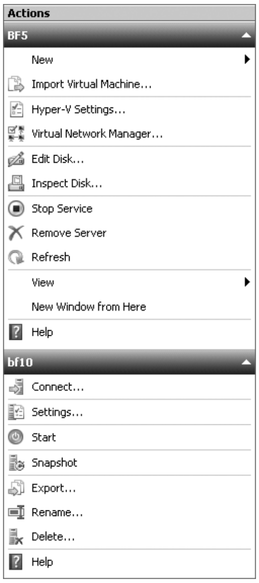
○ Install an operating system from a network-based installation server

< Previous | Next > | Finish | Cancel

FIGURE 29.15
Mount an ISO file in the virtual DVD player of the VM. When starting the VM, it will boot from it.

7. Click Next for the summary screen. It's slightly more than just a summary. Click Finish to let the wizard start working. After a couple of seconds, you will have a new VM.

In the Hyper-V Manager, you will now have your first VM. Select this VM, and the Actions pane is extended to display specific options for the VM (named bf10), as shown in Figure 29.16.



Actions

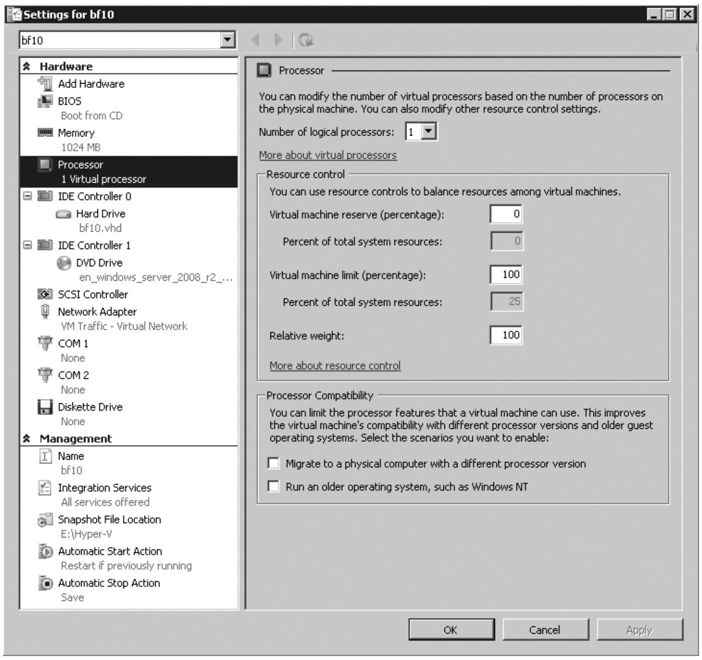
BF5
New
Import Virtual Machine...
Hyper-V Settings...
Virtual Network Manager...
Edit Disk...
Inspect Disk...
Stop Service
Remove Server
Refresh
View
New Window from Here
Help

bf10
Connect...
Settings...
Start
Snapshot
Export...
Rename...
Delete...
Help

FIGURE 29.16
Select a VM to see the actions that apply to it. This menu varies, depending on the power state of the VM among other things.

Click the Settings menu to take a look at the VM configuration, as shown in Figure 29.17. This is one dialog box you will see a lot in the future. There is no point in covering all items here exhaustively, but we'll talk about the most important ones. The left pane has two sections, Hardware and Management.

FIGURE 29.17
Managing all VM parameters from the Settings dialog box



Settings for bf10

bf10

Hardware
- Add Hardware
- BIOS
  Boot from CD
- Memory
  1024 MB
- Processor
  1 Virtual processor
- IDE Controller 0
  - Hard Drive
    bf10.vhd
- IDE Controller 1
  - DVD Drive
    en_windows_server_2008_r2_...
- SCSI Controller
- Network Adapter
  VM Traffic - Virtual Network
- COM 1
  None
- COM 2
  None
- Diskette Drive
  None

Management
- Name
  bf10
- Integration Services
  All services offered
- Snapshot File Location
  E:\Hyper-V
- Automatic Start Action
  Restart if previously running
- Automatic Stop Action
  Save

Processor

You can modify the number of virtual processors based on the number of processors on the physical machine. You can also modify other resource control settings.

Number of logical processors: 1

More about virtual processors

Resource control

You can use resource controls to balance resources among virtual machines.

Virtual machine reserve (percentage): 0

Percent of total system resources: 0

Virtual machine limit (percentage): 100

Percent of total system resources: 25

Relative weight: 100

More about resource control

Processor Compatibility

You can limit the processor features that a virtual machine can use. This improves the virtual machine's compatibility with different processor versions and older guest operating systems. Select the scenarios you want to enable:

□ Migrate to a physical computer with a different processor version

□ Run an older operating system, such as Windows NT

The Hardware section manages all virtual hardware. The topmost entry is for the Add Hardware Wizard, although the choices are limited. You can add a SCSI controller that you can use in turn to add virtual SCSI drives. You can have up to four SCSI controllers, with 255 devices each; that should be enough. One thing to note here: you cannot boot from a virtual SCSI disk. It must be an IDE disk. However, because of the way Hyper-V implements the virtual IDE and SCSI adapters, there is no performance penalty for using IDE.

The second entry is BIOS. Contrary to some other virtualization software, there is no direct access to the BIOS screen from the VM console. If you boot the VM from the console, there is no function key to get you to a BIOS menu. Here you see again the idea that the hypervisor should run mostly out of the box with little tuning. The BIOS entry in the VM settings allows you to select the order of boot devices: disk, CD/DVD, floppy, or network. And for some reason, you can also tell the VM to turn Num Lock on or off on boot.

The Memory entry specifies the amount of memory available to the VM, as you have seen before. The Processor entry is more interesting. You can specify the number of logical processors configured for this particular VM. On boot, the VM will see that number of processors. Of course, they are not by default reserved to that particular VM. The hypervisor is the sole owner of the processors and decides which VM gets how much time. The host itself is under the control of the hypervisor in this aspect. To say this another way, you can have many more virtual CPUs in total than the number of real CPU cores.

Do Not Give a Windows 2003 VM More Than Two CPUs

Although Hyper-V allows you to assign up to four logical CPUs to a VM, this is not always a good idea. It turns out that Windows 2003 may react badly when presented with three or four virtual CPUs on Hyper-V. Anyway, the Microsoft product team for Hyper-V supports Windows 2003 with one or two CPUs only.

Using the "Resource control" settings, you specify what percentage of the selected processors should be reserved for this VM. The grayed-out boxes tell you how much this is in terms of the full host capacity. This is a lower limit that is always available. The upper limit is set by the virtual machine limit. The VM will never use more than this particular percentage. That feature is very useful to limit misbehaving applications that use CPU resources for no good reason.

At the bottom are two check boxes to tweak processor functionality:

Migrate to a physical computer with a different processor version This is used in a situation where you have clustered machines with slightly different physical CPUs. Before, such a configuration would block features such as Live Migration and Quick Migration where you move online VMs between hosts. Select the box to improve your chances. Tests show that compatibility between CPUs of the same family is quite good, but beyond that it doesn't really help. You will certainly not be able to do online moves between AMD and Intel systems. The feature works by disabling certain CPU instructions whose functionality varies between processors, such as SSE3, SSE4, 3dNow!, and others. The price you pay is potential lower performance, depending on your workload.

Run an older operation system, such as Windows NT This is required to run OSs that were built before the year 2000, roughly speaking. It alters the behavior of some processor instructions, notably CPUID. In the Windows 2008 version of Hyper-V, this check box was obscurely called "Limit processor functionality." Not something you would think of using in case of trouble.

The next items on the list are the two IDE controllers. You cannot delete them. The first IDE controller is used for the virtual hard disk that you configured. As usual, you can have two IDE devices per controller. The second IDE controller has the CD/DVD drive. Using this setting you can change the configuration while the VM is running. Why is that useful? Well, you can mount other ISO files or connect to a physical CD/DVD drive if you want. There also is a default SCSI controller if you're running Windows 2008 R2, but it has no devices connected.

Next are the network adapter, the COM ports, and the floppy drive. Most notable about the network adapter is that you can change its binding while the VM is running. This is equivalent to pulling the cable from a physical NIC and connecting it to another switch. One use for it is to install the operating system, patch it on Windows Update, and then connect it to an internal switch for testing. The COM ports are not quite what they seem. The do *not* connect to any physical COM port your server may have. Instead, they can connect only to a named pipe, which in turn must have a process behind it to talk to. Similarly for the floppy drive, it does not connect to a physical drive, but only to a file image. A virtual floppy is created using the New Wizard; then select Floppy Disk.

Continuing the discussion of the left pane, we come to the Management section. Normally you don't need to change any settings here, since the defaults are sensible. You can change the name of the VM here, configure the Integration Services components, specify the snapshot file location (discussed later when we fully discuss virtual disks), and decide what should happen when the host starts up or shuts down. Of these settings, the Integration Services setting deserves extra discussion.

You can only configure the Integration Services setting from the host. You have five options to choose from. All of them are enabled by default, and again, the defaults should be good for most situations:

Operating System Shutdown This allows the host to signal to the VM that it should start the regular shutdown procedure right now, as opposed to just turning off. This is a neat trick, because this allows you to shut down the host, which in turn will cleanly shut down all the VMs for you. Clearly, this works only when this option is enabled and for VMs that actually have Integration Services installed.

Time Synchronization As you probably know, this option is critical in an Active Directory domain, because Kerberos authentication relies on accurate timekeeping. If the host is a member of the same Active Directory forest as its VMs, you can make the host responsible for timekeeping of the VMs; this is the default. If the time of the host is not managed in any way, you must deselect this setting to allow the VM to manage its own time source.

Data Exchange With this option, it's possible for the host and VM to exchange data through selected registry keys.

Heartbeat This is used by the host to keep track of the state of the VM. When heartbeats stop coming, the host signals that the VM is in trouble.

Backup (volume snapshot) This important setting enables Volume Shadow Copy Services (VSS) integration for the VM. In other words, a backup application on the host can signal the VM that it is going to be part of a backup and should make sure all its applications are ready for it.

Real World Scenario

Why Accurate Timekeeping Is Important

Accurate timekeeping is not just important for Kerberos. Many applications depend on good time administration to work reliably and may act strangely if time is reset into the past. If you are lucky, the application just completely stops working when its internal administration is corrupted. Jumps into the future are usually fine. After all, that is not really different from turning a server off and turning it on a year from now. The trouble starts when you set the time back to the present after a jump into the future.

To illustrate what might go wrong, consider a real-world example of a VM running an Active Directory domain controller (DC). It is running on a host that is not joined to a domain. Instead, it takes its time from a network component, a centrally located switch. During maintenance, the switch gets a new firmware update and accidentally sets the time a year ahead. The host picks this up and also sets its time a year ahead. Then the virtual DC does the same. At that point, it stops replicating with its peers because Kerberos authentication is broken.

Another problem is that the internal Active Directory administration for deleted objects goes badly wrong if the time jumps ahead too far. Some objects will be permanently deleted; others will not. The result is a badly corrupted forest. The list goes on and on, if you start thinking about it.

Bottom line: in a production environment, make really sure to have a correct time synchronization configuration, especially for VMs.

We are not done discussing the Actions pane in Figure 29.16 yet. With a VM selected, you have these additional options:

Connect Used to start the virtual console.

Settings To configure the VM, as discussed.

Start To boot the VM. Depending on the state of the VM, you will have more options here such as Shutdown and others.

Snapshot To create a point-in-time image of the VM.

Export To save the entire VM, including configuration and virtual hardware.

Rename To give the VM another name.

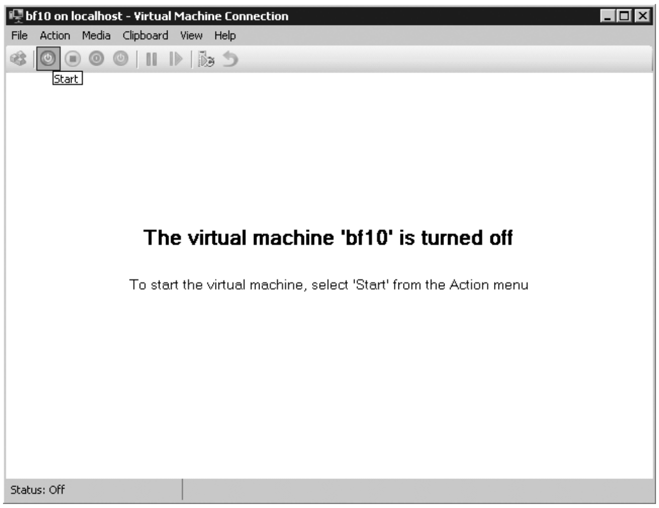
Delete To remove the VM configuration, but not its virtual hard disks.

Actually, the exact items in the Actions pane vary with the power state of the VM. The previous assumes that the VM is still turned off. When a VM is running, you have additional items to turn the VM off, reset, shut down, and so on.

Installing a Virtual Machine

With the VM configured, the hard part is done. The next step is to install the operating system in the VM. When you are used to dealing with physical servers, the details of installing in a VM are a bit different.

The first question is how to connect to the console of the VM. After all, you want to see what is going on. To do that, open Hyper-V Manager, select the VM (bf10) you want to connect to, and click Connect in the lower-right corner. A virtual console opens, as shown in Figure 29.18. Another way to do this is to double-click the thumbnail at the bottom.



The virtual machine 'bf10' is turned off

To start the virtual machine, select 'Start' from the Action menu

FIGURE 29.18 Connecting to a VM shows you its console

The virtual console has a menu bar and a button bar. Most of the functions have duplicates in the Hyper-V management console that you are already familiar with. The black screen telling you that the VM named bf10 is turned off is your virtual screen. When you boot the VM, this will show you the familiar messages of Windows progressing from boot to full GUI. Before you go ahead and boot the VM, there is something you need to be aware of. It's clear how the screen works, but what about the keyboard and mouse?

The console can "capture" the keyboard and mouse. You do so by clicking the virtual screen. When captured, all input from keyboard and mouse are sent to the VM. Initially, you cannot release control from the VM by just moving the mouse. You press the key sequence Ctrl+Alt+left arrow for release. In a fully running VM with Integration Services installed, the experience is much better: you can move the mouse out of the virtual screen onto the desktop, and when that happens, the host has control of the keyboard and mouse again. There is one special case: the Ctrl+Alt+Del sequence is special. Even when the VM has control, the host will process it. To send the Ctrl+Alt+Del sequence to a VM, you can either press Ctrl+Alt+End or use the console menu Action (Ctrl+Alt+Delete).

The Virtual Console Under the Hood

The Hyper-V virtual console uses the Remote Desktop Protocol (RDP) to talk to the VM, the same protocol that is used for Remote Desktop Services. The difference is that it does not use the default RDP port but instead uses TCP port 2179. When you start the virtual console from the Hyper-V Manager, it starts a client application called `vmconnect.exe`, located in `%programfiles%\hyper-v`. This application is similar to the Remote Desktop Client but additionally allows you to select the VM you want to connect to. The Virtual Machine Management Service is the listening service. When you connect to it using `vmconnect.exe`, it tells the client which VMs are available and makes sure the RDP traffic goes to the correct VM. In other words, this service acts as an RDP multiplexer.

Using the RDP protocol and client code means that VMConnect shares a number of keyboard shortcuts with RDP and introduces some of its own. Table 29.4 lists the most relevant ones.

Table 29.4: Hyper-V Virtual Console Keyboard Shortcuts

Hyper-V Key	Windows Key	Explanation
Ctrl+Alt+End	Ctrl+Alt+Del	The well-known three-finger salute to display the logon screen or security dialog box
Alt+Page Up	Alt+Tab	Switches to next program
Alt+Page Down	Shift+Alt+Tab	Switches to previous program
Ctrl+Alt+left arrow		Releases keyboard and mouse focus from the VM
Ctrl+Alt+Pause		Toggles full-screen mode

If you open the Media menu on the virtual console and select DVD drive, you will see that the ISO file for Windows Server 2008 R2 is mounted—assuming you are following the example to the letter. With the DVD ready for booting, you are ready to go. You can find Start under the Action menu, but the quickest way is to just push the green power-on button. The console will display a few messages and quickly show the familiar boot screen of Windows Server 2008 R2. Click the virtual console to let it capture the keyboard and mouse. Don't forget: to release the mouse, you need to press Ctrl+Alt+left arrow.

Install the operating system in the usual way using a full installation—not Server Core. We will get to a Server Core installation later in this chapter.

Real World Scenario

Installing Older or Unsupported Operating Systems

Hyper-V was developed to run only current and common operating systems. The programmers at Microsoft do not test with older operating systems such as Windows NT 4, Windows 98, OS/2 Warp, Linux distributions, and so on. This does not mean that such operating systems will not work as a VM on Hyper-V, but it does mean that they will not have Integration Services. They will have to rely on the legacy (emulated) devices, which will be much slower than enlightened VMs.

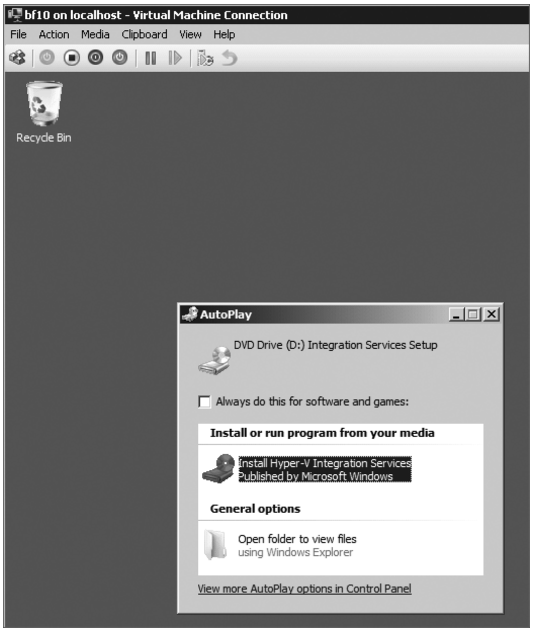
We tested some old operating systems, just to see what would happen. Windows NT 4 is known to run on Hyper-V, while Windows 98 does not boot. DOS 6.22 runs fine, even with network access. We took one old OS from a different vendor: OS/2 Warp from IBM. It did not install.

After the installation and final reboot, you end up with a fully running Windows Server 2008 R2 VM. If you have DHCP on the LAN connected to its virtual switch, it will be fully network enabled as well.

However, the work is not quite done. Unless your VM runs the same OS and service pack as the host system, its Integration Services will not match that of the Hyper-V host. In fact, any OS older than Windows 2008 will not have Integration Services at all. In both cases, you need to install the latest version. This is easily done using the built-in ISO file with the Integration Services software. The details may vary a bit, depending on the OS version and/or previously installed versions of Integration Services.

1. Log on to the VM using an Administrator account.
2. Select Console ➢ Action ➢ Insert Integration Services Setup Disk.
3. After a short while, the Autoplay dialog box should offer to run Setup, as shown in Figure 29.19. If this does not happen, perhaps because Autoplay is disabled, you can run Setup directly from the virtual DVD drive in the VM.
4. Run Setup. After this, you know the drill. One or more reboots may be required.

If you look through the VM using the virtual console, there is little that shows that it's a VM. From its own point of view, it's just another server. One place where it shows its virtual nature is in the Device Manager. This is actually a good place to check that all is well with your VM. Table 29.5 lists the names for the virtual drivers of Hyper-V. If you *don't* see these, something went wrong. You might be missing a reboot, installed the wrong version of the Integration Services, or something like that.


bf10 on localhost - Virtual Machine Connection
File Action Media Clipboard View Help
Recycle Bin
AutoPlay

DVD Drive (D:) Integration Services Setup

□ Always do this for software and games:

Install or run program from your media

Install Hyper-V Integration Services
Published by Microsoft Windows

General options

Open folder to view files
using Windows Explorer

View more AutoPlay options in Control Panel

FIGURE 29.19 Autoplay dialog box for setting up Integration Services

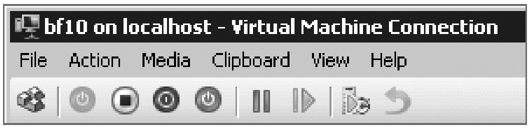
TABLE 29.5: Hyper-V Devices in the VM

TYPE OF DEVICE	NAME
DVD/CD-ROM	Msft Virtual CD/ROM ATA Device
IDE/SATA disk	Virtual HD ATA Device
SCSI controller	Storvsc miniport
SCSI disk	Msft Virtual Disk SCSI Disk Device
Display adapter	Microsoft Virtual Machine Bus Video Device
Mouse	HID-compliant mouse
Network adapter (enlightened)	Microsoft Virtual Machine Bus Network Adapter
Network adapter (legacy)	Intel 21140 Based PCI Fast Ethernet Adapter or DEC PCI Fast Ethernet DECchip 21140
System devices	Many, such as the Integration Component drivers, the Virtual Machine Bus, and various filter drivers

A final word on the power states of a VM: you can boot, shut down, hibernate, suspend, and reset a physical machine. But a VM has an additional power state: it can be in a "saved state." This feature is very nice to have in a test environment. It is similar to a hibernated state but initiated from the host. This means that it always works, no matter what OS is installed in the VM. The "save state" action can be found in the usual place in the Hyper-V management console or the VM virtual console. Start the VM again, and the saved stated will be reloaded to resume exactly where you left off.

For some tests, it's useful to freeze the VM in place. This is different from a saved state because no data is saved to disk. A freeze (or *pause*, as Hyper-V calls it) just stops the VM in its tracks, instantaneously. This is again a feature that is hard to find on a physical machine. Figure 29.20 shows the power state buttons on the Hyper-V console.

FIGURE 29.20
Virtual machine power state buttons on the console



From left to right, you can do the following:

- Send a Ctrl+Alt+Del sequence to the VM.
- Start the VM after a shutdown or saved state.
- Turn off the VM directly. Windows will ask you whether you are sure you want this. It is the equivalent of pulling the plug.
- Shut down the VM. This works only with properly installed Integration Services. The Hyper-V host will work with the VM to initiate a normal shutdown sequence. Quite useful! There's no need to log on to the console anymore.
- Save the current state of the VM. For test, this is probably the one we use most.
- Pause (or freeze) the VM. Hit the button again to resume.
- Reset the VM, similar to a hardware reset on a physical host.
- Snapshot the VM: save all current state and configuration, and bookmark it. The VM continues running, but you can revert to the saved snapshot later. This is very useful, but it's potentially dangerous, as you will see later
- Revert to a saved snapshot.

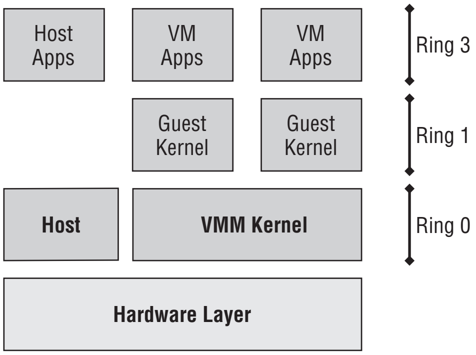
Understanding Hyper-V Architecture

By this time, you should have a feeling for the basic functionality of Hyper-V. You have learned what server virtualization is used for and how to set up a basic server running Hyper-V with some VMs. However, to understand what's going on and to be able to troubleshoot problems, you need a deeper understanding of how Hyper-V was designed. In this section, you will learn more about the software architecture of Hyper-V.

CPU capabilities play an essential role in the implementation of server virtualization. The Intel/AMD model for Pentium-class processors has four privilege levels, known as *rings*. Ring 0 has the highest privilege. The Windows kernel and device drivers use this level. Processes in ring 0 are able to access any hardware in the system. Ring 1 and ring 2 are not normally used in current versions of Windows. Ring 3 is the lowest level. It runs normal "user" programs. In practice, this (should) mean any code that does not require kernel privileges. The trick here is that the CPU forbids any code running in a higher ring to write data or code belonging to a lower ring. In other words, it's a hardware security feature.

Let's take a look at the various ways that server virtualization is implemented in general. The oldest methods have a hybrid architecture that is similar to the diagram shown in Figure 29.21. The label Hybrid means that it is a merger of a normal kernel and a kernel that is aware of virtualization. The (now free) Microsoft Virtual Server product line is an example of such a hybrid architecture.

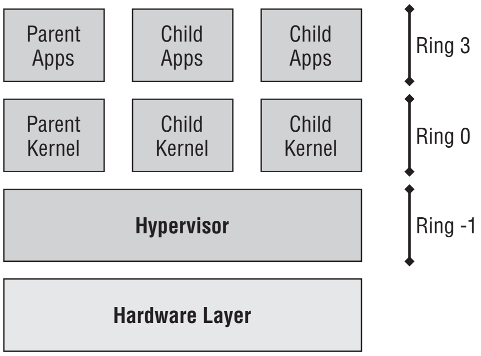
Figure 29.21
Overview of a hybrid virtualization platform, where the host and the virtualization layer are on the same level



The host kernel is split in two pieces. The normal kernel runs side by side with code that takes care of all virtualization aspects related to hardware: the Virtual Machine Manager (VMM) kernel. The VMM kernel takes care of all interactions of the VMs with the host. The VMM runs in ring 0 and has full access to all hardware. The VMs have their own kernel. This kernel should believe that it's running in ring 0, but as you can see from the diagram, it is really running in ring 1. The VMM kernel is taking care of this translation. The principle is known as *ring compression*. To make this work, both the host and the VM kernels need adjustments, and the VMM needs to translate certain VM requests to the host.

Contrast this with a hypervisor architecture. A *hypervisor* is a software layer between the hardware and the operating systems running on the host. This is known as the *bare-metal* approach: virtualization at the lowest possible level. The main purpose of the hypervisor is to create isolated execution environments (partitions) for all operating systems. In line with that function, it is responsible for arbitrating access to the hardware. Figure 29.22 illustrates the point.

Figure 29.22
With a hypervisor, the host operates on the same level as the VMs: atop a hypervisor layer.



Let's take a look at the ring structure. You will note the addition of ring -1. This is not a mandatory feature of a hypervisor, but most modern hypervisors use it. This ring is the main feature of the CPU virtualization additions discussed earlier: Intel VT and AMD-V. It's a new access level of even higher priority than ring 0. It allows all kernels to really run on ring 0 without the tweaking that is required in a hybrid model. This makes for a cleaner architecture, implying fewer bugs in code and ideally better performance.

The diagram uses Microsoft's terminology for the technology. It illustrates that all VMs are created equal but that one is more equal than the others: the *parent* partition that is responsible for the management and high-level arbitration of all VMs. It is the default owner of all hardware resources and controls the startup and shutdown of the *child* partitions.

The Hyper-V hypervisor is *microkernelized*. As the term suggests, this means it was written to be as lean and mean as possible. It contains no drivers, no GUI code, but just enough intelligence to do its main job: manage memory and regulate access to the hardware. Other (non-Microsoft) hypervisors may take a different approach. In the monolithic approach, the hypervisor contains drivers and takes more responsibility for inter-VM communication. One advantage of a monolithic hypervisor is that it theoretically can deliver a higher maximum performance because of its tighter integration with drivers to access the hardware. On the other hand, if this hypervisor has no drivers for your specific hardware, you are out of luck. For mature products, this should not be a real problem, though. In the microkernel hypervisor, the drivers actually reside in the parent partition, which in the case of Hyper-V must be a Windows 2008 (R2) server. The parent partition has the drivers, so if your hardware has drivers for Windows 2008, it can work with Hyper-V.

Let's step away from the generalities and take a look at the specifics of Hyper-V. Microsoft designed it with the following goals in mind:

- The hypervisor should be as lean and mean as possible.
- It should be manageable using open APIs.
- Reliability and performance should be maximized.
- It should be a built-in feature of the Windows Server.

What they came up with is an implementation roughly along the lines of Figure 29.23. You will probably need to look at this a couple of times before it starts to make sense. We know we did the first time.

The bottom of the diagram shows the by-now-familiar layers of the hardware and the hypervisor. On top of the hypervisor run four VMs in this particular example. From left to right we have the parent partition and three types of child partitions. Let's start with the parent partition.

The Hyper-V Parent Partition

The parent really has two parts. The lower block runs in ring 0, or kernel mode. It contains four blocks, three of which are new in this discussion: the VMbus, the independent hardware vendor (IHV) drivers, and the virtualization service provider (VSP). The upper block corresponds to ring 3 code, or user mode. Shown here are only the components that are relevant to Hyper-V. The parent partition must run Windows Server 2008 or newer in order to support the hypervisor. Depending on your needs, this can be a full GUI version or the Server Core edition.

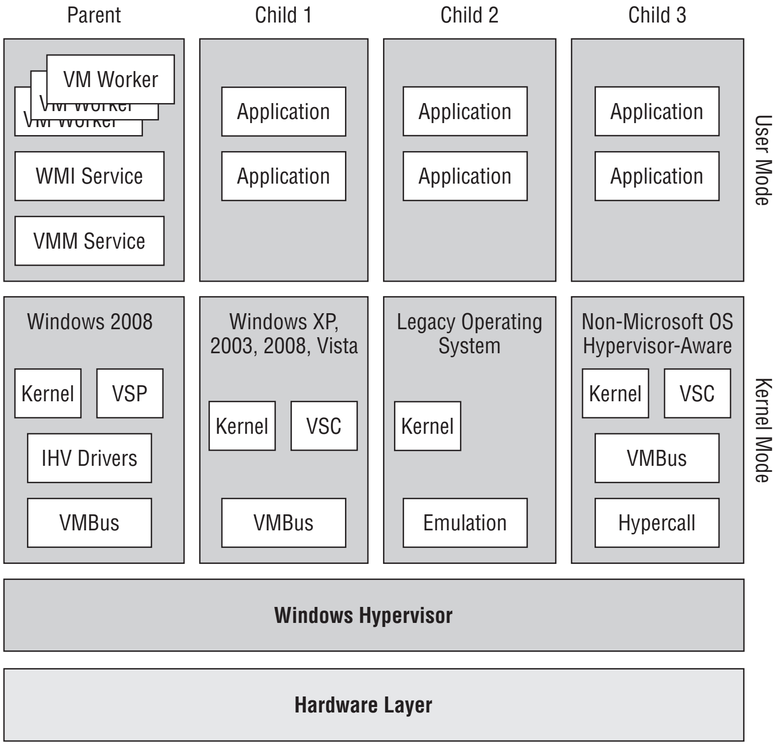


Figure 29.23 Hyper-V architecture: the hypervisor, the virtual machines, and their relations

Installation and Boot Process of the Hypervisor

You may wonder about this chicken and egg problem: the hypervisor needs a Windows 2008 parent to manage it, but the parent needs the hypervisor before it can do anything. What happens is this. You first install the Windows Server 2008 edition with Hyper-V, such as Standard, Enterprise, or Datacenter. At this point, there is no hypervisor. When you enable the Hyper-V role, Windows will install all required software components such as the VMbus and the VSP, but not the hypervisor.

Instead, Windows installs the device driver `Hvboot.sys`, which will load the actual hypervisor on next boot. This can either be the `%Systemroot%\System32\Hvax64.exe` hypervisor for AMD processors or `%Systemroot%\System32\Hvix64.exe` for Intel. These files are both less than 1MB on the initial version of Windows 2008, illustrating their microkernel nature. Once loaded, the hypervisor uses the virtualization extensions of the CPU to insert itself as a ring -1 process, taking over control of the hardware. It proceeds to load the Windows 2008 kernel of the parent partition, prepared with the VMbus and VSP.

Coming back to the ring 0 part of the parent partition, you see the VMbus component at the lowest level. As the name suggests, the VMbus is used for data communication. It is 100 percent virtual, meaning that it has no hardware components. It is a point-to-point connection between partitions. It does not communicate with the hypervisor directly and uses shared memory and Inter Process Communication (IPC) mechanisms to share data. The point to note here is that data is shared and not copied for obvious performance reasons. The VMbus is an important differentiator compared to the hybrid model.

The IHV drivers live on top of the VMbus, but only in the parent partition. This illustrates that you install no external drivers in the child partitions. The IHV drivers are the usual drivers that are supplied with every version of Windows: disk drives, SCSI bus, RAID, network, video, and so on. A characteristic for this virtualization model is that you depend on exactly the same drivers as you would for nonvirtualized Windows servers. The good news is that you have a lot to choose from. The bad news is that those drivers are as good as the vendor chooses to make them, since they are not written or verified by Microsoft. Make sure you use drivers that are digitally signed and supported by your vendor.

Key to the performance of Hyper-V is the combination of the VSP in the parent and the virtualization service client (VSC) in the child partition. The parent VSP component is responsible for translating data between the VMbus and the IHV drivers. The VSP is actually a combination of multiple modules for each hardware type: storage, networking, video, input devices, and so on. This makes sense, because each of these has very different requirements regarding the transaction speed and the amount of data to be processed. Table 29.6 shows an overview of the most important hypervisor files.

Table 29.6: Relevant Hypervisor Files

Function	Path to File
Hypervisor boot driver	`%systemroot%\system32\drivers\hvboot.sys`
Hypervisor (for AMD)	`%systemroot%\system32\hvax64.exe`
Hypervisor (for Intel)	`%systemroot%\system32\hvix64.exe`
Virtual Machine Management service	`%systemroot%\system32\vmms.exe`
VM worker process	`%systemroot%\system32\vmwp.exe`

These VS**P** modules are paired with corresponding VS**C** modules. For example, the storage module of the VSC will communicate with the storage module in the VSP. Where the VSC communicates only with the VSP in the parent partition, the VSP communicates with all VSC components. It's possible for non-Microsoft developers to design and build their own modules since the relevant APIs are open to all.

We have covered the kernel mode part of the parent partition and progress to the user mode components. The Virtual Machine Management service (VMM service) is responsible for managing all child partitions. Whenever a child partition is started, the VMM service starts a new virtual machine worker process (VMW process) for this child. This process doesn't really do any work, except for monitoring, starting up, shutting down, and so on.

What Process "Runs" the VM?

Contrary to similar worker processes in the hybrid model, the VMW process does not really "run" the child partition. It just controls it. You cannot tell from the CPU usage of the VMW process if the child is doing anything, with the exception of data transfer by legacy (nonenlightened) devices.

In fact, all child partitions may be running up to 100 percent CPU without anything showing up in the parent partition directly. To find out what the child is doing, you either need to connect to the child directly using normal server management tools, use Performance Monitor counters from the parent, or access the WMI provider in the parent partition.

The final component in the parent partition is the WMI provider. Years ago Microsoft has decided that WMI would be the preferred way to manage and monitor system resources, and Hyper-V is no exception. The WMI provider is publicly documented on MSDN and provides the following functionality:

- Status reporting on mouse and keyboard
- Access to the VM BIOS
- Managing and reading the configuration of networking, serial devices, storage, guest partitions, and so on
- Reading current CPU properties
- Managing power state

From this feature set you can see that the WMI provider has all the functionality required to manage the Hyper-V environment. In fact, most management systems for Hyper-V are expected to use WMI. Unfortunately, the WMI interface is a little too involved to use directly for most daily management activities, although we will discuss some examples later in this chapter. Only people with extensive WMI programming experience will be able to work productively with this.

The Two Hyper-V APIs

Hyper-V actually has two built-in APIs. Besides WMI, it has the Hypervisor interface. The Hypervisor API is a low-level interface capable of configuring the hypervisor, managing partition states, handling interpartition communication (VMbus), handling the scheduler, and so on. The Hypervisor interface is really meant for those writing their own VSP/VSC modules and other system-level coding. As an administrator, you will certainly never touch the Hypervisor interface API directly.

To complete the parent overview, you should have an idea of the services running in the Hyper-V parent partition. There are three of them:

- Hyper-V Virtual Machine Management. This is for managing virtual machines from the parent partition.

- Hyper-V Image Management Service. This is dedicated to managing disk snapshots.
- Hyper-V Networking Management Service.

All three services are set to start on boot.

Hyper-V Child Partitions

Now you have most of the picture. Just the child partition architecture is left to discuss, and we have actually already covered some elements of that. The first child partition in Figure 29.23 runs an operating system that is aware of the hypervisor. Microsoft calls such an OS *enlightened*, while others may call it *paravirtualized*. As such, it has a VMbus interface and a Virtualization Service Client. The question is, how does the child partition become enlightened? In the parent, you installed the Hyper-V role, but clearly you need to do something else in a child partition to supply the required software components. The Hyper-V software for VMs is Integration Services. Its version must match the Hyper-V server version.

Enlightenment Comes in Two Flavors

Enlightenment exists in two variants, a general and a more specific one. Generally speaking, an operating system that works with the VMbus interface is enlightened. That includes older systems such as Windows XP and Windows 2003 that need the Integration Services to be enlightened. The specific variant applies to the kernel: if the kernel natively knows about the hypervisor, the operating system is said to have an *enlightened kernel*. This would include Windows 2008 and Vista SP1 and newer versions. The advantage of an enlightened kernel is that it can further optimize access to the VMbus. For instance, Windows 2008 can bypass some internal software layers for network and disk access when it realizes the data is intended for a virtual device.

Only Windows 2008 and newer have a version of Integration Services built in, although you will probably need to upgrade that version to match the current Hyper-V version. For other operating systems, you will need to install them explicitly. One of the ways to do this is to insert an ISO file with Integration Services into the VM. This ISO file comes with Hyper-V.

Integration Services is actually a set of modules installed in `%systemroot%\Virtualization\<version>` in the child partition. There is one `<version>` folder for each update of Integration Services. Taking a deeper look at its functionality, you can see that it is really split into two parts. On one hand, there are drivers for storage, video, and network. Another driver directly integrates with the kernel to optimize its performance running as a VM. On the other hand, there is a multipurpose service (`vmicsvc.exe`) executable that is started four times with different arguments to provide the following functionality:

Shutdown service On request of the parent partition, this service will shut down the VM in an orderly manner.

Heartbeat service The parent needs a way to tell whether the child is still alive. If the parent can no longer hear the heartbeats, it is safe to assume the child has a severe problem.

Time Synchronization service A child partition has no direct access to the hardware clock. This service offers an alternative. Don't use this feature in an Active Directory domain environment, because that has its own time synchronization infrastructure.

Data Exchange service This service is the only way for user (ring 3) processes in the parent and child services to communicate outside of the regular devices. It works through certain registry keys.

Shadow Copy Requestor service This service will work with a backup service on the parent to provide consistent backups for both the parent and the child partitions, using VSS technology.

It's worth noting that Integration Services cannot be configured from the child partition. The only (proper) way to do it is through the parent partition, either using the Hyper-V management console or using the WMI interface.

Continuing the discussion of child partitions, let's look at the second child partition in the diagram. It's labeled as a "legacy operating system." What this really means is that the operating system is not enlightened. This may be because Integration Services are not installed or are not available. For instance, you can have Windows 2003 installed both as a legacy system and as enlightened. For older systems such as Windows 2000 or Windows NT 4, Microsoft will not write Integration Services, although in principle a third party could do so. Lack of Integration Services does not mean that the VM will not install or run, but it does imply that it will never reach a high level of performance. On the other hand, many legacy operating systems should perform just fine on modern hardware, even if virtualized. The diagram shows that a legacy system has no VMbus or VSC provider.

The third and final type of child partition runs a non-Microsoft operating system using a custom hypervisor adapter, such as the Hypercall adapter of SUSE Linux. These are kernel-level drivers that provide access to the VMbus infrastructure. On top of that you see the usual VMbus component, which in turn is used by enlightened drivers from the VSC provider, again written specifically for that operating system. You may expect near-native performance from such an enlightened system. At the time Hyper-V came on the market, the only such operating system was SUSE Linux with a Xen-enabled kernel.

Security Design in Hyper-V

Now that you have a good overview of the Hyper-V architecture, we should say a few words about security. After all, you can now run multiple computers on the same hardware, separated only by a thin layer of hardware and software. If one of the partitions gets compromised, what could it do to the others? It could attack the hypervisor and, if it succeeds in compromising it somehow, attack the other partitions. In other words, the hypervisor is the key to the whole thing. Hyper-V was designed with such security issues in mind. We'll cover what was done to make an attack as hard as possible, but keep in mind that no sane person could ever guarantee 100 percent that an attack from one VM to the other through the hypervisor is impossible. It is software, and any piece of software has bugs.

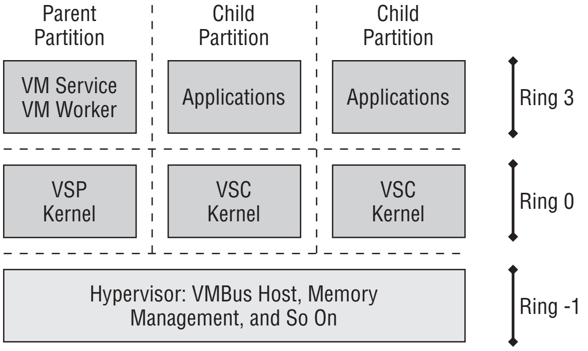
As discussed, the Hyper-V hypervisor was designed to have a small footprint. That by itself is good for security: the less code and complexity, the better. Also, the Hypercall APIs that are needed to talk to the hypervisor are open to public scrutiny. Therefore, Microsoft had to make the assumption that all possible Hypercall functions are accessible to attackers and took measures to protect against invalid arguments, bad usage, and so on.

A basic premise of the Hyper-V security model is that the parent must be trusted, both by the hypervisor and by the child partitions. After all, all system resources are controlled by the parent. For instance, if an attacker controls the parent partition, he or she can stop and start VMs at will, inspect their disks, and so on. Given that premise, the Hyper-V architecture has to defend against two avenues of attack against the parent: external and internal.

An external attack against the parent is nothing more than a normal hacking attack: guess a password, try to compromise a running service, intercept management traffic, and so on. To minimize the attack surface, we recommend you use a Windows 2008 Server Core or the dedicated Hyper-V Server edition for the parent and use only the strict minimum of services required. Don't fall into the temptation of using it for a file server, domain controller, or anything else. If it's not related to Hyper-V functionality, it needs to go into a VM.

An internal attack would have to come from a compromised child partition, using Hyper-V resources such as the VMbus, VSC, or Hypercall API to try to attack services in the parent partition. Also, an attack from one VM to another VM on the same host would constitute an internal attack. To illustrate the principle, take a look at Figure 29.24.

FIGURE 29.24
The dashed lines are security boundaries in a Hyper-V machine.



Each dashed line represents a security boundary, either as a ring transition in the CPU or the software separation between partitions. To defend against an internal attack, the Hyper-V architecture has implemented the following measures:

- The hypervisor uses unique resource pools for each partition. In other words, no VM-specific data is shared with other VMs. Such data cannot be used for a direct attack. Along the same line, no memory is shared between VMs. Each VM has its own memory space.
- There is no direct communication from one child to another. All communication either goes through the parent or goes through a really external medium such as the network.
- Each child has its own VMbus instance to the parent. So, even if one VMbus instance is compromised, this has no direct influence on other child partitions.
- Each child has its own controlling process in the parent, a VM worker process, as noted earlier.
- DMA attacks between VMs are not possible because a child partition can never own a physical device.

There's more. Architecture is one thing; actual coding is quite another. The programmers of the hypervisor have taken steps to make an attack as difficult as possible and to try to detect an attack if it happens. Some of these include the following:

- The hypervisor uses Address Space Layout Randomization (ASLR) whenever possible. This is a technique where different pieces of code are always loaded at different addresses.

This makes buffer overrun attacks very hard, since those depend on a predictable starting point for executable code.

- Hardware Data Execution Prevention is mandatory, meaning that only in-memory code that is marked as executable can actually be run. This protects against stack overflow attacks. In addition, memory pages containing code are marked as read-only. Any attempt to modify such data should generate a blue screen.
- Stack Guard cookies are known pieces of data on the stack that the hypervisor uses with certain internal functions. If these cookies are modified, the system will know there has been a stack overflow or an illegal but successful modification of the stack. A blue screen will probably be the next event.
- Last but not least, the hypervisor file is digitally signed. Any attempt to modify it offline will stop it from loading.

All of these countermeasures against internal attacks are no guarantee that an attack cannot succeed. But they do indicate that Microsoft has taken security very seriously.

Using Virtual Disks

In your day-to-day working with Hyper-V, the virtual disks of your VMs play an important role. To use them effectively, you need a good understanding of the various types of disks and their performance characteristics. We will also discuss VM snapshots and disk maintenance.

Virtual Disks and Their Controllers

A *virtual disk* is a file with a specific format. Its normal extension is `.vhd`, and it's common to all Microsoft virtualization products. Even more important, its structure is published and freely usable by anyone. The specification is low level from a disk point of view, meaning that it does not assume any type of filesystem in the VM. The VM may run Windows with NTFS, FAT32, or FAT or run Linux with EX2, ReiserFS, or whatever will be developed in the future. The open specification of VHD files means that it can be used for virtualization products from other companies and that anyone can write tools to handle them.

Windows accesses disks through controllers, and a VM running on Hyper-V is no different. The two usual types are present: IDE and SCSI. In a physical machine, SCSI systems generally perform better on a server. Not only are SCSI disks faster than IDE disks (very generally speaking!), but the SCSI interface was designed to handle multiple I/O requests at the same time. Interestingly, from a Hyper-V VM perspective, there is no performance difference. In an enlightened VM, both IDE and SCSI data transfers are quickly translated to VMbus requests that in turn depend on the storage system of the host. In fact, the `.vhd` file has no knowledge of its controller. This means that the same `.vhd` file can be used with an IDE and SCSI controller.

Still, there is one important difference between virtual SCSI and IDE controllers: you cannot boot a VM from a SCSI controller. The reason behind this is the magic that Hyper-V needs to perform to make the enlightened controllers work. If you have read the section "Understanding Hyper-V Architecture," you will know that the VMbus drivers are not the first ones to load in the boot process. If the VMbus is required for booting, we would have a deadlock situation. Hyper-V solves this by making the IDE controller automorphing, meaning that it behaves as a legacy IDE controller when this is necessary. As soon as an enlightened driver starts talking to it, the controller will use the VMbus instead of using emulation. The virtual SCSI controller has no such automorphing capability, explaining why it cannot be used to boot a VM. In practice, it should not be a problem because the virtual IDE performance is equal to that of virtual SCSI.

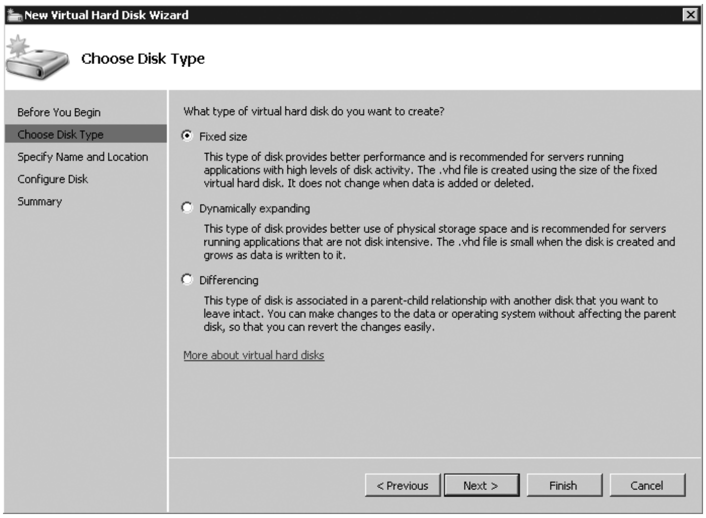
VHD Disks as First-Class Citizens

Previously, virtual disks based on VHD were just files that needed to be run by a virtualization engine. Both Windows Server 2008 R2 and its little brother Windows 7 offer native support for VHD files. Not only can you mount them as you can with physical disks, but you can also boot from them.

Virtual Disk Types and When to Use Them

Hyper-V has three types of virtual disk systems to offer to a VM, as you can see when you start the wizard for a new virtual hard disk (shown in Figure 29.25). All these virtual disks are implemented as normal files on an NTFS volume.

Figure 29.25
The New Virtual Hard Disk Wizard



New Virtual Hard Disk Wizard

Choose Disk Type

What type of virtual hard disk do you want to create?

● Fixed size

This type of disk provides better performance and is recommended for servers running applications with high levels of disk activity. The .vhd file is created using the size of the fixed virtual hard disk. It does not change when data is added or deleted.

○ Dynamically expanding

This type of disk provides better use of physical storage space and is recommended for servers running applications that are not disk intensive. The .vhd file is small when the disk is created and grows as data is written to it.

○ Differencing

This type of disk is associated in a parent-child relationship with another disk that you want to leave intact. You can make changes to the data or operating system without affecting the parent disk, so that you can revert the changes easily.

More about virtual hard disks

Dynamically expanding disk You create such a disk with a virtual size. For instance, you create the disk with a size of 127GB. The VM using this disk will indeed see 127GB. However, on the host, the file will allocate only as much space as it really needs. This is very useful for test situations, but it's not as fast as a fixed-size disk. Another disadvantage is that the host may run out of disk space unexpectedly because of growing virtual disks. You will definitely want to avoid this in production situations. Finally, each time the host needs to allocate new disk space to expand the disk, you will take a (brief) performance hit. Although a dynamically expanding disk is the default selected by the wizard, it is really only suitable for testing.

Fixed size disk When you create such a disk, a file is created with the full size that you specify. During creation, the file is filled with binary zeros. This may take quite some time, depending on the size you specified. A fixed size disk should be your default for production systems.

Differencing disk This is an interesting feature where you have a parent-child relation between two virtual disks. The parent disk is a static, read-only reference. The differencing disk (child) stores all changes with respect to the parent. The point of this feature is to save disk space on the host and be able to create a new VM really quickly. However, if the VM using the differencing disk is writing a lot of new data, the advantage of saving disk space will disappear quickly. Clearly, this feature is suitable only for test purposes.

A VM can use any type of virtual disk. However, it is also possible to use a physical disk on the host. This is known as a *pass-through disk*. To dedicate a physical disk to a VM, it must appear as a physical disk to the Windows host machine. This may be a local disk, a disk on a SAN, an iSCSI disk, and so on. It cannot be a partition. It really must be a full disk, as visible in Disk Manager. Also, to be able to use the disk in a VM, it must be offline in the host to stop the host from using it. You can set a disk offline or online in Disk Manager or using the command-line tool `diskpart`.

A pass-through disk is the fastest possible configuration, in theory. In practice, the performance difference with a fixed disk will be very small. In some cases, a pass-through disk has another advantage: it may be larger than 2TB, which is the limit for a virtual disk. You can decide for yourself if it is wise to use such a large volume, considering the time it would take to back up or restore such a volume or to run the command-line tool `chkdsk` on it to verify the filesystem integrity.

A pass-through disk has some disadvantages. You cannot copy it to another machine as you would with a virtual disk, although you can convert a physical disk to a virtual disk when creating a virtual disk. A pass-through disk cannot be part of a differencing set, and you cannot snapshot a VM that has such a disk.

In short, your default type of drive for production purposes should be a fixed-size virtual disk. Use pass-through disks for the scenarios that require them, such as some clustering configurations. Use dynamically expanding and differencing disks only for testing.

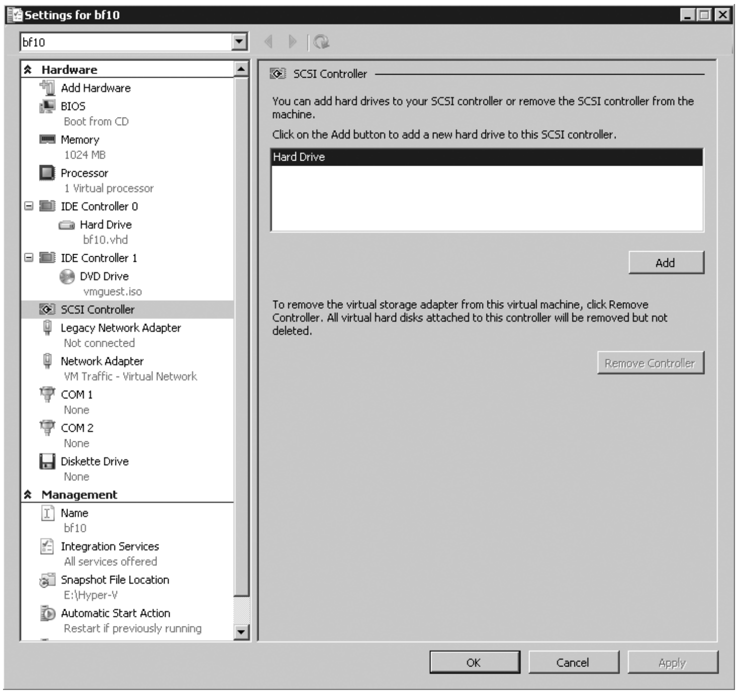
Adding a Disk to an Existing VM

When you create a VM using the New Virtual Machine Wizard, you have the option to add the boot disk. To add more disks, you can either prepare a disk using the New Disk Wizard directly from the Hyper-V console or, more conveniently, start this wizard from the VM settings. In this example, we use the hot-add feature of Hyper-V V2; this won't work on Windows 2008 Hyper-V. We will add a disk to the SCSI controller. Remember, the `.vhd` virtual disk format is interchangeable for IDE and SCSI because it is the same for both. We could have added it to an IDE controller except for the little fact that IDE does not support hot-add!

1. Select the VM, and open its Settings menu. Select the SCSI controller on the left, and then select Hard Drive on the right, as shown in Figure 29.26. The Add button will activate.

2. The next step is to select an unused SCSI channel. Channel 0 is reserved for the controller, so picking channel 1 for the first disk makes sense. You now have the option to select an existing virtual hard disk using the Browse button or to create a new one using New. This enters the familiar New Disk Wizard. In the screenshot of Figure 29.27, we have created a new 127GB fixed-size disk (the default!) and used a name that relates it to its VM.

FIGURE 29.26
Adding a new virtual disk to the SCSI controller



Settings for bf10

bf10

Hardware
Add Hardware
BIOS
Boot from CD
Memory
1024 MB
Processor
1 Virtual processor
IDE Controller 0
Hard Drive
bf10.vhd
IDE Controller 1
DVD Drive
vmguest.iso
SCSI Controller
Legacy Network Adapter
Not connected
Network Adapter
VM Traffic - Virtual Network
COM 1
None
COM 2
None
Diskette Drive
None
Management
Name
bf10
Integration Services
All services offered
Snapshot File Location
E:\Hyper-V
Automatic Start Action
Restart if previously running

SCSI Controller

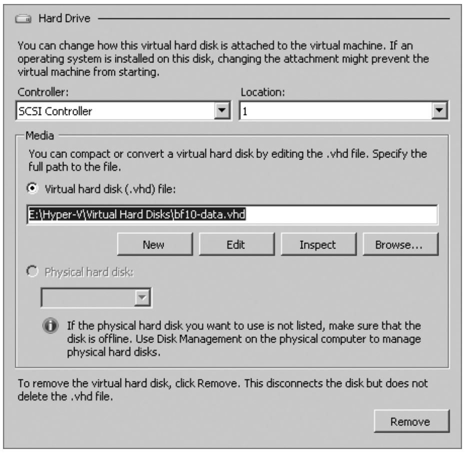
You can add hard drives to your SCSI controller or remove the SCSI controller from the machine.

Click on the Add button to add a new hard drive to this SCSI controller.

Hard Drive

To remove the virtual storage adapter from this virtual machine, click Remove Controller. All virtual hard disks attached to this controller will be removed but not deleted.

FIGURE 29.27
Select an unused SCSI channel, and either add an existing disk or create a new one.



Hard Drive

You can change how this virtual hard disk is attached to the virtual machine. If an operating system is installed on this disk, changing the attachment might prevent the virtual machine from starting.

Controller: SCSI Controller
Location: 1

Media

You can compact or convert a virtual hard disk by editing the .vhd file. Specify the full path to the file.

● Virtual hard disk (.vhd) file:

E:\Hyper-V\Virtual Hard Disks\bf10-data.vhd

○ Physical hard disk:

If the physical hard disk you want to use is not listed, make sure that the disk is offline. Use Disk Management on the physical computer to manage physical hard disks.

To remove the virtual hard disk, click Remove. This disconnects the disk but does not delete the .vhd file.

3. Click OK to finalize your changes. Now, let's see if the hot-add worked. Log on the VM with an Administrator account, and start Server Manager. Open Storage and Disk Management. You will see the newly added disk, ready to be initialized and formatted, as shown in Figure 29.28. Interestingly, you can even remove an online disk on operating systems that support it, such as Windows Server 2008 R2.

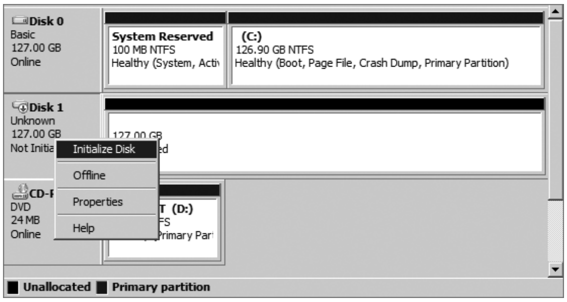


Figure 29.28 The newly added disk is visible in the VM and just needs to be initialized and formatted.

Using Differencing Disks with a Base Image

Let's take a closer look at the use for differencing disks. We often have the need to quickly try something in a lab. For instance, say we want to examine the effects of raising the Active Directory domain and forest levels. How would we quickly build a one-shot lab to try this? The trick is to use a base installation of Windows Server 2008 and make differencing disks from that VM. From each differencing disk, you can create a new VM. This is similar to working with an image, so you must take care to apply `sysprep` to the new VMs. These are the basic steps that you would take in preparation:

1. Prepare a VM with a dynamically expanding virtual hard disk. Take note of the full path of the corresponding .vhd file.
2. Install Windows Server 2008 R2 as usual, including Integration Services. Apply service packs and patches as required. Do not join it to a domain, make it a domain controller, or install any service or application.
3. Install tools that you might need in each VM, such as resource kits, the Sysinternals tooling, and so on.
4. Run sysprep: `%systemroot%\system32\sysprep /generalize /oobe /shutdown`.
5. Find to the .vhd file, and mark it as read-only.

You would have the same preparation even if you planned to copy the virtual disk afterward to deploy a new VM. In this case, for each new VM you want to create, you need to create a differencing disk to use it for a new VM. To create the differencing disk, follow these steps:

1. Start the New Hard Disk Wizard from the Hyper-V console.
2. Click Next to skip the welcome screen.
3. Select Differencing Disk.
4. Click Next, and enter a name for the virtual disk such as `vm2-diff.vhd`. Specify the full path to the disk as well, or accept the default.
5. Click Next, and browse to the parent virtual disk that you have marked as read-only when you prepared it.
6. Click Next and Finish to create the differencing disk.

Finally, to create a new VM using the differencing disk, follow these steps:

1. Start the New Virtual Machine Wizard from the Hyper-V console.
2. Step through the wizard (you know the drill), and when you get to the step where you specify the virtual disk, you set the check box for an existing virtual disk. Browse to the virtual disk you created, `vm2-diff.vhd` in this example.
3. Boot the VM, and the `sysprep` procedure will do its work to create a new instance of Windows.

Those are all the steps—it's quicker than copying the full virtual disk of the parent and uses less disk space.

There are two things you can do to speed things up a bit. The first and rather obvious suggestion is to create the differencing disks on a different physical disk for better performance. The second is to use NTFS compression on the differencing disks. Experience teaches us that differencing disks compress very well. A CPU decompressing such a file will outperform an uncompressed file in terms of MB per second, as long as the CPU is not too busy. Again, this is for testing purposes only, as you know by now.

Finally, a note of warning. If you change the parent disk at any time, you will completely invalidate all its children. Setting the parent disk to read-only prevents such accidents.

Real World Scenario

VMs Mysteriously Running Slow

Bigfirm has deployed a number of Hyper-V servers as a test environment. Each server has 16GB of internal memory and 2TB in usable disk space. The Hyper-V hosts are run by the IS department. Other departments use and manage the VMs.

One day, complaints start coming in that certain VMs are slow to respond, although there is no obvious load on the VMs. The IS department takes the complaints and investigates. None of the VMs shows any sign of trouble. The CPU usage is normal, all applications are running, the OS is not swapping, and there is no memory pressure. In short, there is no obvious cause. One pattern emerges: most "slow" VMs share the same host.

The IS department starts running performance counters on the host, including the CPU and Logical Disk objects. After an hour of measuring, it is noted that some counters are consistently high: the counter "disk seconds/read" averages in at 30 milliseconds, and the counter "disk seconds/write" is mostly over 50 milliseconds. The "average disk queue length" counter of the disk hosting the VMs often peaks at more than 20. Compared to other hosts, these are really high values. Clearly, the host has trouble delivering the required disk I/O.

An inspection of the host shows that the VM disk is a RAID 5 configuration, which is good at reads but slow at writes. The combined disk I/O of all VMs has pushed the host capacity over the limit. The problem is solved by rearranging the disks in a RAID 10 configuration and adding more disks to increase the maximum I/O throughput.

This case illustrates the added complexity of managing a virtualized server environment. The combined VMs may put the host in trouble.

Disk Maintenance

The most essential parts of a VM are its disks. Not only do they contain the data, but they determine in large part the performance of the VM. In this section, you will learn how to maintain virtual disks.

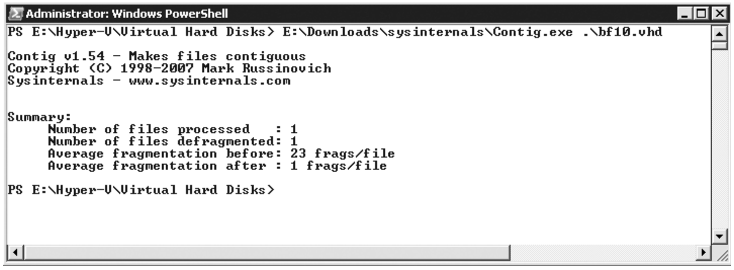
We'll discuss disk fragmentation first. You are familiar with NTFS fragmentation, where a file may get chopped into many pieces for several reasons. For large files there may not be sufficient continuous space on disk to create an unfragmented file. More commonly, a file may get updated many times and write to a different section of the disk each time. If you think about it, you see that VMs using virtual disks may suffer doubly from fragmentation! Not only does it have fragmentation in its own file systems, but its virtual disk may also fragment. There are some measures you can take to reduce the effects of fragmentation.

First, it is good to realize that a very large file having a few fragments does not impact performance significantly. If you have a 127GB fixed size virtual disk in a 100 pieces, each piece is still larger than 1GB on the average. Reading one such chunk from disk will take multiple seconds, while skipping to another section on disk takes no more than a few milliseconds. You won't notice the few additional milliseconds loading that data. So, fragmentation on this level is nothing to worry about.

Second, dynamic and differencing disks are different. They increase their size on demand and may get fragmented quickly. The most reliable way to get a dynamic disk back in shape is to defragment it. But how do you defragment a single file? The built-in tools can defragment only a complete volume. One way to do it is to use the command-line tool `contig.exe`, which you can download from `www.microsoft.com/sysinternals`. Assuming you have downloaded and unzipped `contig.exe`, you could do it like this (as illustrated in Figure 29.29):

1. Copy `contig.exe` to a convenient spot, such as `E:\downloads\sysinternals`.
2. Stop the VM whose disk you want to defragment, either by shutting down or saving its state.
3. Open a command prompt or PowerShell window, and change the working directory to `E:\Hyper-V\Virtual Hard Disks`.
4. To analyze the virtual disk named `bf10.vhd`, type **`E:\downloads\sysinternals\contig.exe -a bf10.vhd`**.
5. If the file is significantly fragmented, fix it by typing **`E:\downloads\sysinternals\contig.exe bf10.vhd`**.

FIGURE 29.29 Defragmenting files using the Sysinternals tool `contig.exe`



```
PS E:\Hyper-V\Virtual Hard Disks> E:\Downloads\sysinternals\Contig.exe .\bf10.vhd

Contig v1.54 - Makes files contiguous
Copyright (C) 1998-2007 Mark Russinovich
Sysinternals - www.sysinternals.com


Summary:
     Number of files processed   : 1
     Number of files defragmented: 1
     Average fragmentation before: 23 frags/file
     Average fragmentation after : 1 frags/file

PS E:\Hyper-V\Virtual Hard Disks>
```

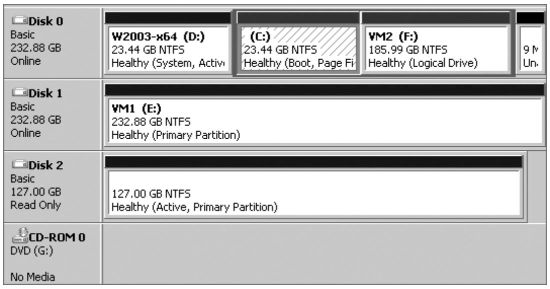
There is another fragmentation-like effect that only affects dynamic and differencing disks. If you create a lot of data and then remove it, you will leave a lot of empty space on the disk. In other words, the virtual disk is larger than it needs to be. Hyper-V can remove this empty space with an operation called *compaction*. To do that, it mounts the virtual disk in the parent partition. If the file system on the virtual disk is NTFS, the parent can inspect its data structures to figure out where the empty space is. It uses this information to reduce the size of the file representing the virtual disk. If you have any other file system, such as FAT32 or EXT2, you must use some means to fill the empty space with zero bytes first. The compaction procedure recognizes such zero bytes and can adjust the VHD file to remove them. Tools to fill empty space with zeros are not included with Hyper-V, so compaction is effectively limited to NTFS file systems. If you do try to compact a FAT32 disk, it will start the process but will probably not do much because it will not find many zero byte sequences to compact.

To compact a virtual disk, the corresponding VM cannot be running. You must shut it down or save its state. Once that's done, it goes like this:

1. From the Actions pane in the Hyper-V management console, start the Edit Disk Wizard.
2. Locate the virtual disk you want to compact, such as the familiar `bf10.vhd`.
3. In the Choose Action dialog box, select the default action Compact.
4. Click Finish. The actual process may take a while.

Interestingly, you can open Disk Manager and see that the virtual disk is mounted while the compaction process works. In Figure 29.30, disk 2 is the virtual disk being compacted. Note that it has no drive letter assigned to it and is mounted read-only. Clearly, it shows the virtual size and not the real size of the disk. Note that there is a potential issue with mounting the virtual disk like this. If the host is running something like an antivirus or indexing program, it might find this disk and start analyzing it as well.

FIGURE 29.30
Disk Manager while the compaction operation is running. The VM disk of 127GB is actually mounted in the host!



Disk 0
Basic
232.88 GB
Online

W2003-x64 (D:)
23.44 GB NTFS
Healthy (System, Activ

(C:)
23.44 GB NTFS
Healthy (Boot, Page Fi

VM2 (F:)
185.99 GB NTFS
Healthy (Logical Drive)

9 M
Un.

Disk 1
Basic
232.88 GB
Online

VM1 (E:)
232.88 GB NTFS
Healthy (Primary Partition)

Disk 2
Basic
127.00 GB
Read Only

127.00 GB NTFS
Healthy (Active, Primary Partition)

CD-ROM 0
DVD (G:)

No Media

You might find yourself in a situation where you want to change the type of disk you are working with. For instance, you may decide that a dynamic disk was not a great choice and that you would prefer to have a fixed disk instead. Or, you have a differencing disk that you would like to disentangle from its parent. Hyper-V offers a number of disk conversion options, as shown in Table 29.7.

Table 29.7: Disk Operations

Type of Disk	Possible Operations
Dynamically expanding disk	Create a new fixed size disk, compact, expand.
Fixed-size disk	Create a new dynamic disk, expand.
Differencing disk	Merge with parent disk, compact, or reconnect to a parent disk that was moved to a different path.

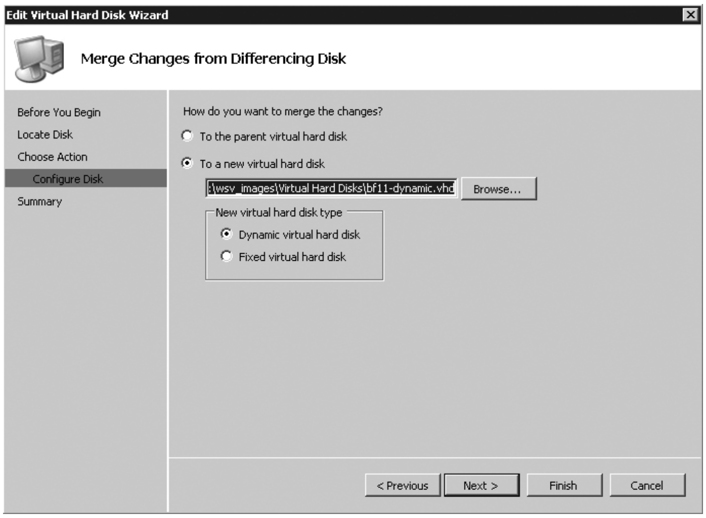
These options are all accessed from the Edit Disk Wizard. If you do a lot of testing using differencing disks, you will at some point want to create stand-alone disks, because a much-used differencing disk will be larger than its parent—exactly the point you wanted to avoid. We'll show an example. Say you have a differencing disk called `bf11-differencing.vhd`, and you want to create a new independent virtual disk. Again, for such a procedure, the VM must be turned off.

1. From the Actions pane on the Hyper-V management console, start the Edit Disk Wizard.
2. Locate the disk you want to merge, which is `bf11-differencing.vhd` in this example. The wizard will determine that it's a differencing disk and present you with the relevant options only.
3. From the Choose Action dialog box, select the Merge option.
4. A subdialog box appears giving you two basic options. Do you want to merge the differencing disk into its parent, or do you want a new dynamic of fixed differencing disk? Be really careful when merging a disk into its parent because any other child disks will be invalidated. For testing, you will want to create a new dynamic disk. Choose a suitable name, such as `bf11-dynamic.vhd`, as shown in Figure 29.31.
5. Click Next and Finish to start the actual procedure.
6. To use the new disk, edit the virtual machine properties, and replace the current differencing disk.

From the discussion you may have seen that virtual disks come in various types. The `.vhd` extension does not tell the whole story. If you look at the Actions pane of the Hyper-V management console, you see one additional menu related to disks: Inspect Disk. This is a simple dialog box that does nothing more than to open a virtual disk and tell what type it is with some additional information. It has one useful feature for differencing disks: if the parent disk is missing, it will tell you. It will give you the option to reconnect it to its parent, which you do by browsing to it.

Time Travel with Snapshots

As you have seen, a lot of features of Hyper-V help you with testing. Dynamic disks save you valuable disk space, and differencing disks allow you to quickly deploy new VMs. But wouldn't it be nice if you could prepare a VM for testing and save it easily so that you can revert to it when things go wrong? This is what snapshots offer you—and more besides.



Merge Changes from Differencing Disk

How do you want to merge the changes?

○ To the parent virtual hard disk

● To a new virtual hard disk

:\wsv_images\Virtual Hard Disks\bf11-dynamic.vhd

New virtual hard disk type

● Dynamic virtual hard disk

○ Fixed virtual hard disk

FIGURE 29.31 Merging a difference disk and its parent to a new (dynamic) virtual hard disk

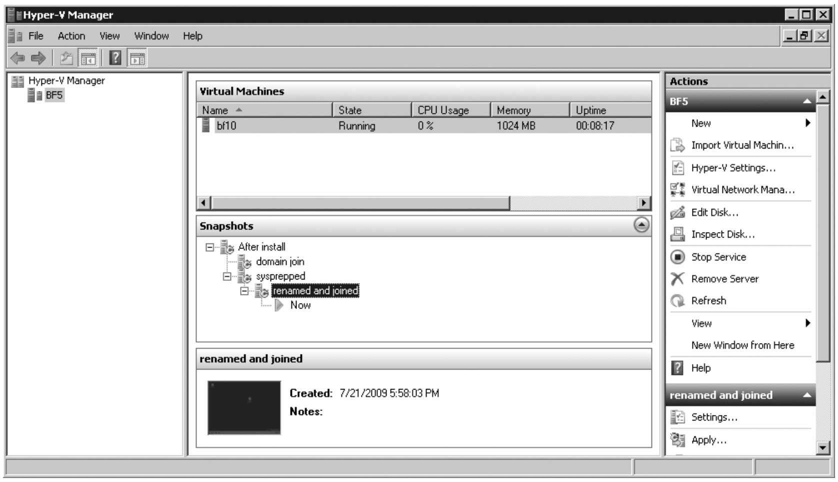
A *snapshot* is best understood as a point-in-time copy of a VM. The copy includes virtual disks, memory, processor state, and configuration of the VM. The VM may be turned off, may be running, or may be in a saved state—it doesn't matter. Of course, not all the VM is really copied. Hyper-V is efficient about it. When you snapshot a VM, Hyper-V does the following:

1. The VM is paused.
2. For each virtual disk connected to the VM, a new differencing disk is created.
3. The virtual disks are decoupled from the VM and are replaced by the corresponding differencing disks.
4. The files containing the configuration of the VM are copied.
5. The VM is resumed. Usually, at this point, you have less than a second of downtime.
6. While the VM is running, its memory is written to disk. If the VM writes to memory, the write action is intercepted, and the original memory is quickly written to disk. Then the write action is allowed to succeed. In this way, all the memory at the time of the snapshot is preserved, while allowing the VM to keep running.
7. When the memory dump is completed, the snapshot is done.

If you look at this carefully, you can conclude some interesting things. First, it allows a true undo. Not only are the disk contents preserved, but the memory and even the configuration are too. You could change memory and add disks and networks and still be able to revert to the original situation. Second, there is no reason why you could not repeat this process. You can have a chain or even multiple chains of snapshots. A limitation of this process is that it works only with virtual disks. If your VM uses pass-through disks, you cannot snapshot it. Finally, taking a snapshot is quick. Anyone using the VM would not notice much downtime, perhaps a second or so.

Taking a snapshot can be done in various ways, similar to changing power states. To do it from the Hyper-V management console, select the VM, and take the snapshot from the Actions pane. Figure 29.32 illustrates what this looks like.

FIGURE 29.32 A VM with multiple snapshot trees. The base snapshot is "After install" and has two subtrees called *domain join* and *sysprepped*.



Name	State	CPU Usage	Memory	Uptime
bf10	Running	0 %	1024 MB	00:08:17

```
After install
    domain join
    sysprepped
        renamed and joined
            Now
```

Created: 7/21/2009 5:58:03 PM
Notes:

This example already has a number of snapshots. The current snapshot is highlighted. Its icon is different from the others. The default name is the current time and date, but renaming it immediately makes more sense.

It's not hard to move around between snapshots. However, if you want to preserve the current state, you need to take a new snapshot first. If you don't, the latest preserved state is that of its latest snapshot. So, to go to a different snapshot while the current one is running, you do the following:

1. Select the snapshot you want to move to. You can select any one you like, even the ones in the middle of a chain.
2. Use the Apply action from the snapshot menu.
3. You get a dialog box with three choices: Take Snapshot and Apply (the default), Apply, and Cancel.
4. Make your choice, and after a couple of seconds, the selected snapshot starts running.

In the same way, you can also delete any snapshot you want. From the management console, this seems simple, but under the hood there is more going on. The key to snapshots is the use of differencing disks. So, what happens if you remove a snapshot in the middle of a chain? In that case, the differencing disks of the removed snapshot are preserved, keeping the disk chain intact. Just the configuration files are removed. When VM is shut down, the host will merge the orphaned differencing disks into their parents. The Hyper-V management console will show a status of "Merging… (34%)." There are more scenarios to consider here, but the key point is that Hyper-V is aware of them and will do the right thing.

One particular example is when you remove a VM that has snapshots. The dialog box will tell you that it will not remove the associated virtual disks. That's only partially true. When you do delete the VM, it will take longer than you might expect. A dialog box stays on-screen telling you that it is "Destroying… (12 percent)" and counting. It is not removing the disks; it is merging the active snapshot with its parents, leaving you with just one `.vhd` file. That's probably the right thing to do, because it leaves you with just one `.vhd` representing all the data of the VM you just deleted.

As a testing feature, snapshots are just great. You can mess up your environment all you want, and you have your get-out-of-jail-free ticket ready to bail you out. Always? No, not always. There are some situations where applying an earlier snapshot can get you into serious trouble. For a stand-alone computer without relations to others, snapshots are perfectly safe. For any situation where computers share some configuration, you need to be careful. The most well-known example of this is Active Directory.

In fact, there is a number of ways where snapshots can get you into trouble with Active Directory. The most obvious example is where you have a snapshot of a member server taken 40 days ago. As you know, a server that is a member of a domain changes its password every 30 days. Since 40 days ago, that server has changed its password in Active Directory at least once and maybe twice already. In its local database, it stores two passwords, the current and the previous. If it has changed its password two times, the snapshotted server no longer has a matching password in Active Directory. In other words, it is no longer a member of the domain and needs to be rejoined. This problem is the same as with regular backups. However, there is a more subtle and dangerous problem that affects only domain controllers.

For domain controllers (DCs) to replicate, they need some administration that tells them what information they already have received from their partners and what they are still missing. This administration depends on the update sequence numbers (USNs) that each DC generates with each modification of its database. Any change has a unique USN associated with it. The point is, when you restore a snapshot of a DC, you also restore the USN configuration of that time. The problem is that the other DCs do not know this has happened! When they look at the DC that had a snapshot restored, they see that they already have processed the current USN of that DC. So, the replication partners of that DC see no need to replicate any data. Effectively, that DC will not replicate until its USN is finally higher than its partners have administered. All data that the affected DC generates in that period will never leave the DC.

This complicated but very serious problem is known as *USN rollback* and is documented in Knowledge Base article 875495. Luckily, if the DC detects that this is going on, it will immediately stop all Active Directory activity and log the problem in the Directory Services event log. The point of this illustration is to make you aware that using snapshots can lead to unexpected problems, even in systems you think you know well. The best thing to avoid such problems is to snapshot computers as a group after shutting them down and to restore them as a group. You may wonder why such problems do not occur when you do a regular backup of a DC and restore it. The reason is that the Windows restore operations are aware of the Active Directory replication administration and reset it. When a restored domain controller comes online, it signals to its partners that it should be treated as if it has never replicated before.

Snapshots and Disk Imaging Are Very Similar

If you look at how snapshots work, you'll realize that it's technically similar to disk imaging. The difference is, most modern disk-imaging software (not all!) is aware of this problem and takes measures to prevent it. In most cases, this requires the use of software agents.

SANs that are capable of making disk snapshots face a similar problem. Make sure that you install the required agent software on all computers whose disks may be snapshotted.

To finish up this section on Hyper-V snapshots, the two key summary points are probably these:

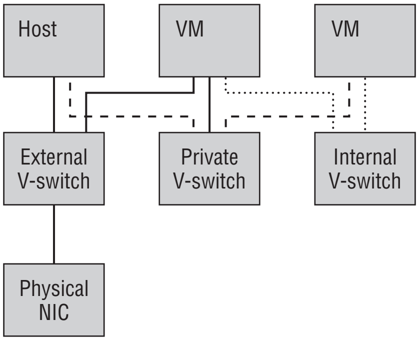
- Snapshots are a great testing feature that can make your life a lot easier.
- Snapshots should not be used in production networks.

Using Virtual Networks

Once you have set up the virtual networks on your Hyper-V hosts, you won't have to think about them much. In fact, once you have set the defaults during the setup of the Hyper-V role, chances are that you won't touch them again. Still, they offer some interesting possibilities and have some limitations that you should be aware of. In this section, we'll discuss everything you need to know about virtual networks for your daily job.

In addition to virtual network interface cards (NICs), Hyper-V supports virtual switches. You really should think of them like that, because they do have some of the characteristics of physical switches. The Hyper-V architecture is such that a virtual NIC must be connected to a virtual switch. You cannot connect a virtual NIC directly to a physical one. Clearly, the Hyper-V virtual switches play an essential role. In fact, there are three types of virtual switches: external, internal, and private. You might take a look at the diagram in Figure 29.33 while reading the following explanation. The diagram shows the Hyper-V parent partition (the host) and two virtual machines. Connections to the virtual switch of type external are solid lines, to internal are dashed, and to private are dotted.

FIGURE 29.33 Three types of virtual switches: external, internal, and private



An *external* switch is one that is connected to a physical NIC on the host. This allows the VM to communicate directly with the outside world. Any other computers on the network will see two different machines, since the VM and its host each has different MAC and IP addresses. From the physical switch connecting the host, the situation looks different: one (physical) NIC with multiple MAC addresses and IP addresses for each VM. Although most switches allow this configuration by default, on some corporate networks this would not be allowed by policy. Note that there is a one-to-one relation between external switches and physical NICs. External switches cannot share NICs, and you cannot have more than one switch per NIC. If you require load balancing or redundancy on the NIC level, you will need software and drivers from the NIC manufacturer.

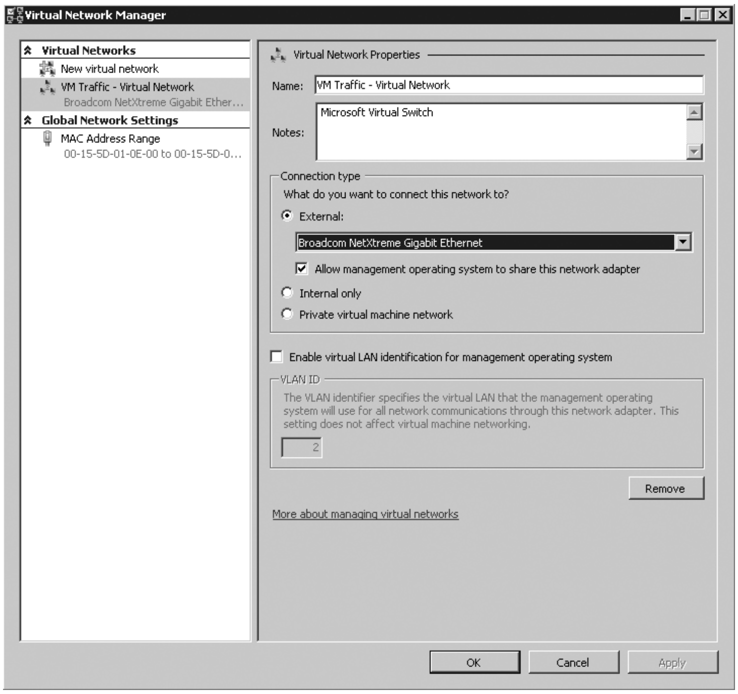
The *internal* switch is interesting. It allows communication between all VMs with the host but not directly to the outside world. For each internal virtual switch that you create, Hyper-V creates an additional Local Area Connection directly connected to the new switch. There is no DHCP on that particular NIC (unless you put it there), so everything on that switch will get an APIPA address. So yes, there is network connectivity, but to use it, you need to configure the IP settings yourself. Another way of generating an internal switch would be to configure a loopback network adapter on the host and create an external switch connected to it. In fact, this is how Virtual Server and Virtual PC work. The existence of the internal switch is to make your life easier in this aspect.

The *private* switch is now easy to understand. There is no host NIC or Local Area Connection associated with it. Only VMs can connect to it, which is a great feature for testing. There is no way a fully internally connected VM can reach the host or the external network directly.

Understanding Virtual Switches

There are two parts to configuring virtual networking: virtual NICs and virtual switches. The switches are managed using the Virtual Network Manager, which you open from the Actions pane in the Hyper-V management console, as shown in Figure 29.34. In the screenshot, the virtual switch VM Traffic – Virtual Network is selected. This switch was created when we installed the Hyper-V role and told the wizard which NICs to use for VM traffic. Look back to Figure 29.4 to see this again.

FIGURE 29.34 The Virtual Network Manager dialog box. Open this from the Actions pane of the Hyper-V management console.



Virtual Network Manager

Virtual Networks
- New virtual network
- VM Traffic - Virtual Network
  Broadcom NetXtreme Gigabit Ether…

Global Network Settings
- MAC Address Range
  00-15-5D-01-0E-00 to 00-15-5D-0…

Virtual Network Properties

Name: VM Traffic - Virtual Network

Notes: Microsoft Virtual Switch

Connection type

What do you want to connect this network to?

● External:

Broadcom NetXtreme Gigabit Ethernet

☑ Allow management operating system to share this network adapter

○ Internal only

○ Private virtual machine network

☐ Enable virtual LAN identification for management operating system

VLAN ID

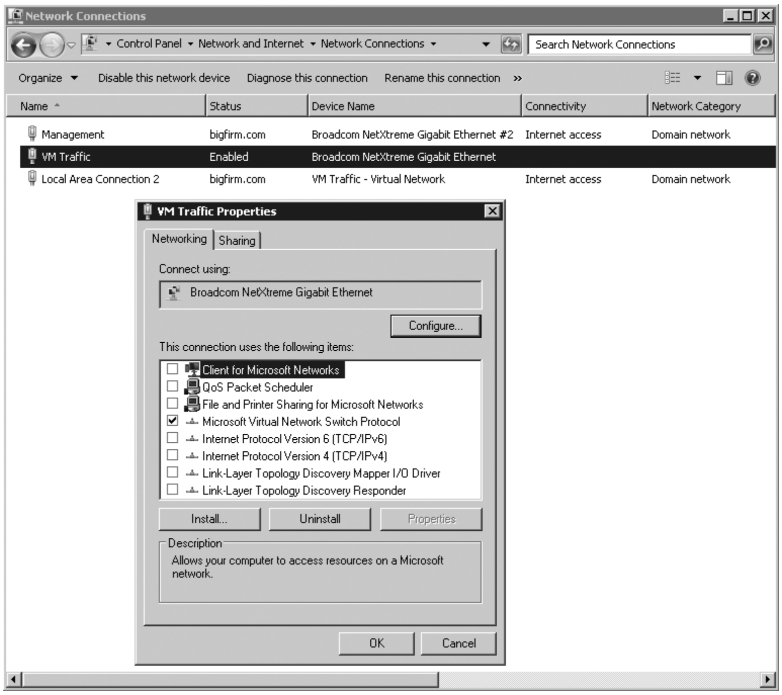
The VLAN identifier specifies the virtual LAN that the management operating system will use for all network communications through this network adapter. This setting does not affect virtual machine networking.

2

A virtual switch has a number of properties: a name, some notes if you like, a network scope (external, internal, or private), and an optional VLAN tag. This switch is external and connected to physical NIC Broadcom NetXtreme Gigabit Ethernet. The LAN connection in the host for this NIC is called VM Traffic; unfortunately, that does not show here. Note the important check box "Allow management operating system to share this network adapter." With this check box, a new virtual NIC is created in the host, allowing network traffic from the host to the internal VM world.

The result of this configuration can be slightly confusing if you look at the networking configuration of the host. Figure 29.35 shows host bf5, which has the Hyper-V role enabled.

Figure 29.35 Hyper-V host with one NIC dedicated to VM traffic. The virtual NIC Local Area Network 2 shows up because of the enabled check box "Allow management operating system to share this network adapter."



Network Connections
Control Panel ▸ Network and Internet ▸ Network Connections
Search Network Connections
Organize
Disable this network device
Diagnose this connection
Rename this connection

Name	Status	Device Name	Connectivity	Network Category
Management	bigfirm.com	Broadcom NetXtreme Gigabit Ethernet #2	Internet access	Domain network
VM Traffic	Enabled	Broadcom NetXtreme Gigabit Ethernet		
Local Area Connection 2	bigfirm.com	VM Traffic - Virtual Network	Internet access	Domain network

VM Traffic Properties

Networking | Sharing

Connect using:

Broadcom NetXtreme Gigabit Ethernet

Configure...

This connection uses the following items:

☐ Client for Microsoft Networks
☐ QoS Packet Scheduler
☐ File and Printer Sharing for Microsoft Networks
☑ Microsoft Virtual Network Switch Protocol
☐ Internet Protocol Version 6 (TCP/IPv6)
☐ Internet Protocol Version 4 (TCP/IPv4)
☐ Link-Layer Topology Discovery Mapper I/O Driver
☐ Link-Layer Topology Discovery Responder

Install... Uninstall Properties

Description

Allows your computer to access resources on a Microsoft network.

OK Cancel

We have three NICs here, one more than we started with when we built the machine. Let's look at them all:

Management This NIC is physical and dedicated to the host. Remote management of VMs goes through this NIC. This is a best-practice configuration: have at least two physical NICs, leave one for dedicated host and management traffic, and use the other one(s) for VM traffic. The NIC is connected to the device Broadcom NetXtreme Gigabit Ethernet.

VM Traffic This is also a physical NIC but with a much different configuration. Look at the binding properties for this NIC. All protocols are deselected, except for Microsoft Virtual Network Switch Protocol. So, it has no IP stack, cannot act as a server, and is effectively invisible. Give the command `ipconfig /all`, and the NIC does not show up. The NIC is connected to Broadcom NetXtreme Gigabit Ethernet, illustrating that this is a link to the outside world for the VMs.

Local Area Network 2 This is a virtual NIC. Its configuration is the same as the Management NIC, except that it connects to the virtual switch VM Traffic – Virtual Network. All protocols are bound, except the Virtual Network Switch Protocol. This NIC is created when you select the check box "Allow management operating system to share this network adapter." And yes, it will go away when you deselect it. Using this NIC, the host is connected to the outside network through the virtual switch.

Figure 29.36 sums it all up. The physical NICs talk to the outside world and are connected to the parent partition directly or to a virtual switch. The host (parent partition) can use both virtual switches and hardware NICs. VMs can only use virtual switches.

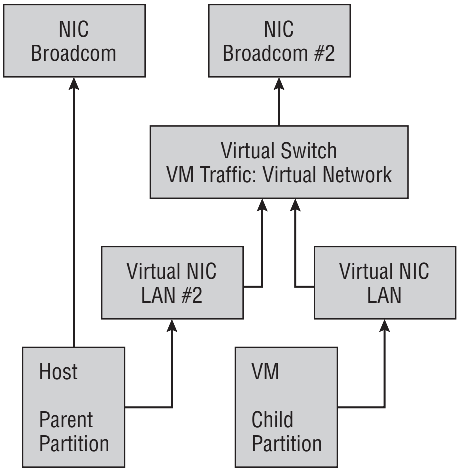


Figure 29.36
The virtual network puzzle: physical NICs, virtual NICs, and virtual switches

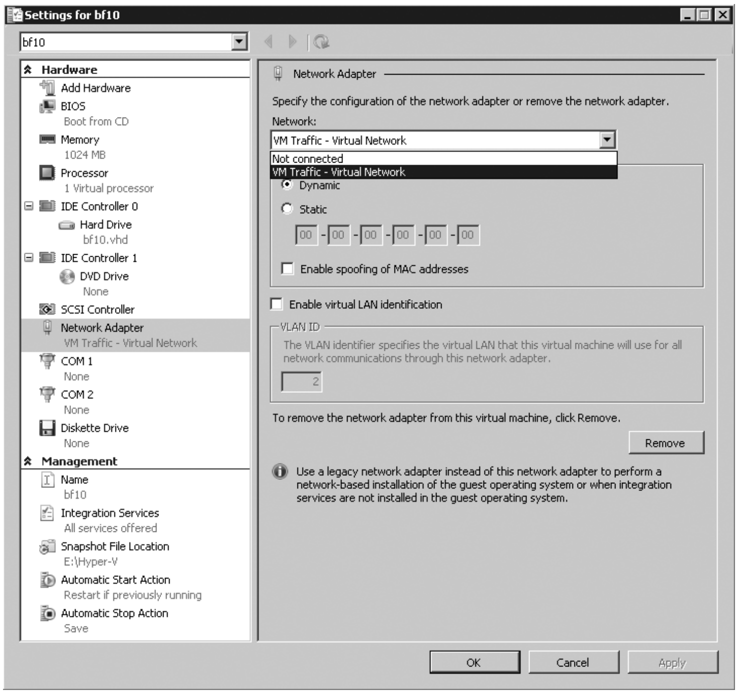
These virtual switches are a powerful concept. Using them, you can create an internal network as complex as you like. Great for testing and exploring new technology!

Connecting VMs to Virtual Switches

Virtual network connections are managed from the VM settings dialog box. This VM has only one NIC. For each NIC, you can select any virtual switch you like, as shown in Figure 29.37. The interesting setting is where you choose which network (virtual switch, really) to use for the selected NIC. You can change this while the VM is running. From the VM point of view, this is like pulling the cable from one switch and plugging it into another.

The option to use a static MAC address should not be necessary. Hyper-V manages its own pool of MAC addresses. Unless you encounter a MAC conflict, you should probably leave this setting at Dynamic.

Then there is the option "Enable spoofing of MAC addresses." That sounds like a bad thing, doesn't it? Well, it was the only behavior available in Hyper-V 1.0. One compromised VM could start flooding the switch or impersonate another VM. Hyper-V 2.0 is by default more secure. The virtual switch learns which MAC corresponds to which VM and does not allow it to change its MAC address anymore. Usually this is not a problem, except for those situations where you really do need multiple MAC addresses. Can't think of one? Consider network load balancing or failover clustering.



bf10

Hardware

- Add Hardware
- BIOS: Boot from CD
- Memory: 1024 MB
- Processor: 1 Virtual processor
- IDE Controller 0
  - Hard Drive: bf10.vhd
- IDE Controller 1
  - DVD Drive: None
- SCSI Controller
- Network Adapter: VM Traffic - Virtual Network
- COM 1: None
- COM 2: None
- Diskette Drive: None

Management

- Name: bf10
- Integration Services: All services offered
- Snapshot File Location: E:\Hyper-V
- Automatic Start Action: Restart if previously running
- Automatic Stop Action: Save

Network Adapter

Specify the configuration of the network adapter or remove the network adapter.

Network: VM Traffic - Virtual Network

- Not connected
- VM Traffic - Virtual Network

● Dynamic

○ Static

00-00-00-00-00-00

□ Enable spoofing of MAC addresses

□ Enable virtual LAN identification

VLAN ID

The VLAN identifier specifies the virtual LAN that this virtual machine will use for all network communications through this network adapter.

2

To remove the network adapter from this virtual machine, click Remove.

Use a legacy network adapter instead of this network adapter to perform a network-based installation of the guest operating system or when integration services are not installed in the guest operating system.

Figure 29.37 Configuring VM network settings. The important choice is which virtual switch to use: none (the "Not connected" option) or VM Traffic – Virtual Network.

When you configured the switch, you saw the option to configure a VLAN tag. A similar option is present in the dialog box for a virtual NIC. A VLAN is a virtual LAN on the network level. The term has been in general use since the late-90s and is unrelated to Hyper-V virtualization.

Briefly, a virtual LAN (VLAN) can be regarded as a broadcast domain. It operates on the OSI network layer 2. The exact protocol definition is known as 802.1Q. Each network packet belonging to a VLAN has an identifier. This is just a number between 0 and 4095, with both 0 and 4095 reserved for other uses. Let's assume a VLAN with an identifier of 10. A NIC configured with the VLAN ID of 10 will pick up network packets with the same ID and will ignore all other IDs. The point of VLANs is that switches and routers enabled for 802.1Q can present VLANs to different switch ports in the network. In other words, where a normal IP subnet is limited to a set of ports on a physical switch, a subnet defined in a VLAN can be present on any switch port—if so configured, of course.

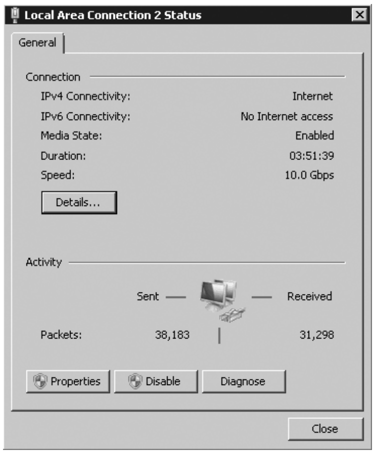
Getting back to the VLAN functionality in Hyper-V: both virtual switches and virtual NICs can detect and use VLAN IDs. Both can accept and reject network packets based on VLAN ID, which means that the VM does not have to do it itself. The use of VLAN enables Hyper-V to participate in more advanced network designs. One limitation in the current implementation is that a virtual switch can have just one VLAN ID, although that should not matter too much in practice. The default setting is to accept all VLAN IDs.

Network Tracing on Hyper-V Switches

What all these virtual switches have in common is that they do not behave like network hubs, but like true switches. This includes the capability to transport packets from port to port without offering them to all connected NICs. Usually this means nothing to you, but there is one situation where you will notice it. A VM will see only its own MAC address on the virtual switch port. This means that there is no way to analyze the network with a NIC in promiscuous mode. The port will simply see no traffic that is not directly targeted at its VM. To see all traffic, you would need to be able to assign one port as a monitor, duplicating all traffic in the switch. This feature is not present in versions 1.0 and 2.0 of Hyper-V, although it is being considered for later versions.

If you look at the status of a NIC in Figure 29.38, you may note one unusual feature. It has a link speed of 10Gbit per second! That's clearly not realistic in most cases. There is a subtle point here. What speed do you assign to a software NIC connected to a software switch? It really doesn't matter. Hyper-V will move the data as fast as it can. The Hyper-V development team arbitrarily chose to label the link speed as 10Gbit per second, and perhaps you may even reach that rate under some circumstances.

Figure 29.38
A virtual NIC is connected to its switch using a link speed of 10Gbit per second.



Local Area Connection 2 Status

General

Connection

IPv4 Connectivity: Internet
IPv6 Connectivity: No Internet access
Media State: Enabled
Duration: 03:51:39
Speed: 10.0 Gbps

Activity

Sent — Received
Packets: 38,183 31,298

NIC Teaming: Be Careful

Using NIC teaming, you can logically combine two or more physical NICs and present them to the OS as a single NIC. The idea is to supply redundancy or load balancing. Many NIC vendors supply teaming drivers for this purpose. The problem is, most teaming solutions (anno 2009) don't work in combination with Hyper-V, with the exception of specifically designed drivers.

So, before you deploy teaming on your Hyper-V hosts, make really sure that your teaming solution supports Hyper-V.

Managing Virtual Machines

In the previous sections of this chapter, you learned the main points of Hyper-V: how to install it, its software architecture, and the ins and outs of virtual disk and networks. There is a lot more to say about server virtualization that would deserve a book on its own. In this remaining sections, we will discuss some of these aspects, some just for awareness, and others more in depth.

Licensing Hyper-V Hosts and Their VMs

As with all commercial software, you need a license to run Hyper-V legally. Generally, licensing aspects are an uninteresting bit of overhead when you buy and deploy software. But with virtualization, there is something important going on. Since 2006, Microsoft has had a specific licensing scheme for server virtualization. Before we get into that, you should realize that such licensing may change over time. To get the current state of affairs, please visit www.microsoft.com/licensing. To have the exact details explained to you, you should call Microsoft. It has people trained in the science of Microsoft licensing; rumor has it that such training takes years.

Anyway, the point is this. If you buy a Windows server license enabled for virtualization, you obtain the right to run one or more VMs with a Windows Server operating system on it using that same license. In other words, you do not have to buy an additional license for the VM—under certain conditions. This right is valid for Windows 2003 R2 licenses and later versions of Windows Server. Table 29.8 lists the main points.

Table 29.8: Virtual Machine Licensing

Server Version	Extra License Rights
Hyper-V Server	None
Standard edition	One VM
Enterprise edition	Four VMs
Datacenter edition	Unlimited VMs, licensed per *physical* processor

Here is how it works. A license for a higher edition always gives you permission to run a lower edition. Buy Enterprise edition, and you are allowed to run Standard edition as well. Assume you want to run 10 new VMs with Standard edition. The host server has two physical processors. Again, note that licensing depends on these physical processors (motherboard sockets) of the host, not on the number of processor cores or on any hyperthreading features. Your options for these 10 VMs are as follows:

- Buy 10 licenses for Standard edition.
- Buy two licenses for Enterprise edition, giving you the right for 2 × 4 = 8 VMs and two additional licenses for Standard edition. This gives you exactly what you need, but no more.

- Buy three licenses for Enterprise edition for 3 × 4 = 12 licenses. Yes, two too many, but that is room to expand.
- Buy two licenses for Datacenter Edition, one for each processor on the host. In this case, you have unlimited expansion room with your licenses.

The cost effectiveness depends on your future growth plans and on the prices for each edition. As you can see, you may have some unexpected options to license your VMs. Note that the license you buy does not require you to install that particular operating system edition as a host or VM. A typical scenario might be to buy a high-end four-processor quad-core server with 128GB of internal memory, where you will run 50 or maybe even 100 lightly used servers, most of them probably running Standard edition. You will certainly buy four Datacenter licenses for the host, but you do not have to install Datacenter just to run Hyper-V. A Standard or Enterprise edition with Hyper-V is good enough.

What's really interesting about this licensing mechanism is that it does not talk about a specific virtualization technology. The license does not specify what edition of Windows to install on the host, as noted, but it in fact allows you to install anything you like as a virtualization host. This includes Microsoft's own products such as Virtual Server R2 but also any server virtualization products by the competition.

What operating system are you licensed for, exactly? There are not only editions to think about but also versions. The host will be Windows 2008 or newer if you run Hyper-V, but what about the operating system in the VMs? Unfortunately, the answer is that it depends on your licensing scheme. Some have downgrade options allowing you to install Windows 2003; others do not. Contact Microsoft or talk to your reseller to find out.

You may be planning to virtualize existing physical machines in your environment. Clearly, you have already paid for those licenses. There is no need to purchase extra licenses from this special virtualization licensing scheme.

One point about the spirit of the license. If you buy one Enterprise edition, you are allowed to run four VMs, but including the host with Hyper-V, you are really running five instances of Windows. The stipulation is that the host should not do anything except be a virtualization platform. You should not make the host a domain controller, run DHCP, and so on. If you plan on doing so, it costs one license, which would leave you with only three licenses for VMs.

Moving VMs Around: Export and Import

One important benefit of server virtualization technology is that the VM no longer strongly depends on the physical server. With a physical server, you would at the very least need to move physical disk around, which works only if the hardware is similar enough. With dissimilar hardware, you would probably have to reinstall the whole machine. Because all Hyper-V hosts offer almost identical hardware to their VMs (with the possible exception of the CPU) and most VMs will use virtual disks, VMs are much easier to move around. With additional tools, it becomes easier still, but moving a VM with Hyper-V is not hard. All you need for this section is a regular high-speed network because of the amount of data you are moving. A SAN is not required.

A VM has three parameters: configuration, current state, and data. When moving a VM to another Hyper-V host, you need to make sure that all of these come along. There is no easy way to do this by hand, though in a pinch you could just take the virtual disks and re-create the VM from there. The Hyper-V console has the Export Wizard, which appears only when the VM is not running. This wizard collects all relevant data, including snapshots, and copies it to a target folder. On the Hyper-V host where you want to run the VM, you run the Import Wizard, and that is all there is to it—well, almost.

The first point to be aware of is that an export operation simply copies all the VM: configuration, state, and data. The import operation is different. You have two choices to make:

- Do you keep the old virtual machine ID (default) or generate a new one?
- Do you use the export files directly to run the VM or copy them first? If you do not copy the files, you can import only once.

Virtual Machine IDs

As most experienced Windows administrators know, many objects in the Windows world have multiple names. A well-known example is an Active Directory object. Its "true" name is a *globally unique identifier* (GUID), but we prefer to use human-readable names such as *display name* or *common name*. It's the same with VMs. Their true name is a GUID, which cannot change. Then there is the display name, which can be anything you like. Other objects in the VM world also have GUIDs, such as virtual switches and NICs.

An interesting property of VMs is that their IDs need to be unique, but the display names may be identical. This typically happens when you import a VM on the same host that you used for export. As long as you use a different ID on import, this will work fine.

A second, more subtle point is that of the virtual network configuration. How does the VM know which virtual switches to use on the new host? Internally, each virtual switch is known by its GUID, allowing you to give the switch any name you like without interfering with its relations to virtual NICs. But if you create a virtual switch on another host, it will have a different GUID. Hyper-V has one little trick to get around this. On import, it looks for virtual switches with exactly the same display name (including character case) as the ones included in the export files. If it finds them, the virtual NICs are connected. If not, the imported VM will have disconnected NICs.

Let's walk through an example where you export and import a VM. Any VM will do, but in this case we will use bf10. Ideally you should have a second Hyper-V host for import, but the example works with one host.

Assume you have a second host name, bf6. Create a shared folder with a descriptive name like Imported Virtual Machines. The required share permissions are a bit tricky. The export process runs as SYSTEM, not as you. So, the share should have permissions for the host computer account to write data. Use either the computer account (bf5) or the Everyone security principal to give write access. Then, start the Export process on the source host, bf5:

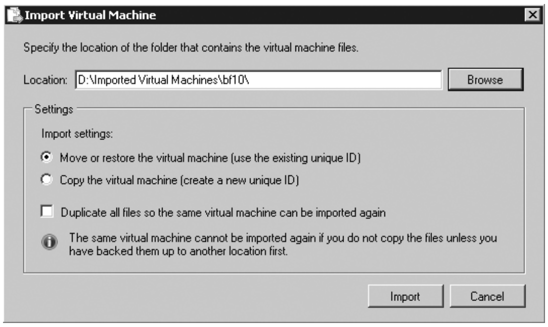
1. On the Hyper-V management console, select the VM you will export, which is bf10 in this example.
2. If the VM is still running, shut it down. A saved state may work but requires compatibility on the CPU level between hosts. For the attentive reader, yes, enabling the ability to migrate to a computer with a different processor version will definitely help to migrate saved states. You would set this on the virtual CPU properties.

3. Select Export, and enter a destination path for your export. It could be a local drive, USB drive, or a remote network location. This time, we will export directly to another Hyper-V host: `\\bf6\Imported Virtual Machines`. If the path does not exist, the wizard will create it. Whatever path you specify here, the wizard will create a subdirectory with the name of the VM.

After the export procedure is finished, you will import the VM. You can to this from the Hyper-V management console on bf5 by adding bf6 as an additional managed machine, but for the example you will log on to bf6:

1. Log on to bf6.
2. If you want the virtual NICs to connect automatically on import, make sure there are virtual switches with exactly the same names. If you have followed the examples, you need to create Management and VM Traffic.
3. Start the Hyper-V management console.
4. From the Actions pane in the Hyper-V management console, select Import Virtual Machine, as shown in Figure 29.39. Specify the location of the VM using the full path: `D:\Imported Virtual Machines\bf10`. The default settings in the dialog box deserve some attention. Keeping the same unique ID (see the "Import/Exports in Hyper-V 1.0" sidebar) makes sense. But if you plan to use this export again, you should select "Duplicate all files so the same virtual machine can be imported again."

FIGURE 29.39
Importing an exported VM



Specify the location of the folder that contains the virtual machine files.

Location: D:\Imported Virtual Machines\bf10\

Settings

Import settings:

● Move or restore the virtual machine (use the existing unique ID)

○ Copy the virtual machine (create a new unique ID)

□ Duplicate all files so the same virtual machine can be imported again

The same virtual machine cannot be imported again if you do not copy the files unless you have backed them up to another location first.

5. Click Import to start the process. It will take just a few seconds and come back with a status update. If there was a problem, you need to search the event log. A common issue is that no matching virtual switch was found, leaving the NICs unconnected.
6. Select the newly imported VM. You will see that the snapshots came along. Check the VM properties, and note the paths to the virtual disks. Verify the connection state of the NICs.

Now that you have imported this VM, you might be tempted to turn it on. But remember, you have now effectively cloned the old VM. You'd have two identical machines on the network if you were to turn them both on! Good idea? You be the judge.

Import/Exports in Hyper-V 1.0

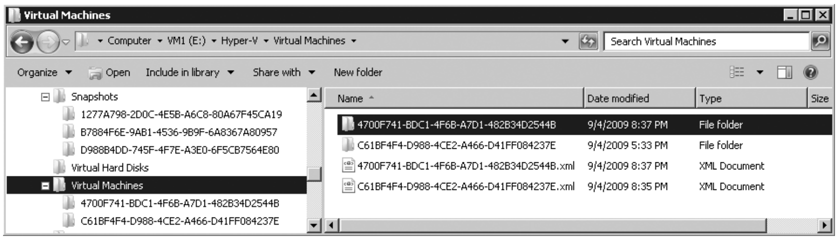
The first version of Hyper-V in Windows Server 2008 has different behavior on export and import operations. In fact, to call it broken would not be far from the truth.

In Hyper-V 1.0, if you imported a VM, it would always use the import location directly. The GUI offers no option to copy the import files first. So, unless you thought to copy them yourselves, you could import the exported VM only once. More than a few people have been caught out with this, because this behavior is totally nonobvious—unless you know exactly how it works.

A second point is that Hyper-V 1.0 does not deal elegantly with snapshots. On export you would get all the intermediate files as well. In 2.0, the whole snapshot tree is merged on export, leaving you with one `.vhd` file.

Just when you think you have the whole concept nailed, there comes one more point to consider. While reading about snapshots, you learned about the risks of rolling back images or restoring image-like backups. Exported VMs are no different. If you export a VM and start it again, the exported VM becomes an image-like backup. If you import the VM on another host, two bad things might happen: you have two almost identical computers running (same name, same IP address, same roles, and so on), and at the same time you did the trick of going back in time for the VM. As noted in the snapshot discussion, this exposes you to issues such as USN rollback and related fun. The safe thing to do is never to use a VM again after you export it. The export mechanism is used to store a VM offline. It is not a replacement for backup.

You may be interested to know how Hyper-V actually keeps track of VM configuration. The actual VM configuration is the easy part. When you create a VM, you tell it where to put the top-level folder containing the VM configuration. Figure 29.40 shows the configuration on Hyper-V host bf5 at `E:\Hyper-V`.



Snapshots
1277A798-2D0C-4E5B-A6C8-80A67F45CA19
B7884F6E-9AB1-4536-9B9F-6A8367A80957
D988B4DD-745F-4F7E-A3E0-6F5CB7564E80
Virtual Hard Disks
Virtual Machines
4700F741-BDC1-4F6B-A7D1-482B34D2544B
C61BF4F4-D988-4CE2-A466-D41FF084237E

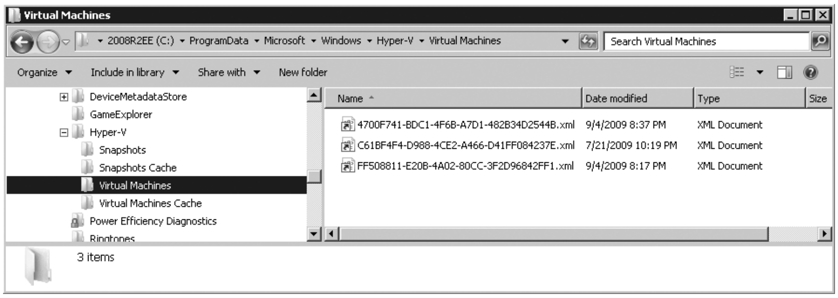
Name	Date modified	Type	Size
4700F741-BDC1-4F6B-A7D1-482B34D2544B	9/4/2009 8:37 PM	File folder	
C61BF4F4-D988-4CE2-A466-D41FF084237E	9/4/2009 5:33 PM	File folder	
4700F741-BDC1-4F6B-A7D1-482B34D2544B.xml	9/4/2009 8:37 PM	XML Document	
C61BF4F4-D988-4CE2-A466-D41FF084237E.xml	9/4/2009 8:35 PM	XML Document	

Figure 29.40 The file structure in the default Hyper-V folder

The screenshot shows three folders: Snapshots, Virtual Hard Disks, and Virtual Machines. There are two virtual machines registered here. Each has one folder named after its GUID, which is the real name of the virtual machine. For instance, the folder name C61BF4F4-D988… corresponds to bf10. This folder contains the current memory state if you save its state. Each VM also has an XML document, again named after the GUID of the VM. This document is fully readable, although you probably should not edit it by hand. If you look though it, you will find the entire configuration of the VM. As an illustration, the XML file has all virtual switches the VM is connected to listed by the GUID of the virtual switch, not by the friendly names.

The Snapshot folder has GUID folders, containing state information of each snapshot: the current configuration of the processes and virtual hardware in a `.vsv` file, and all of virtual memory in a `.bin` file. The difference files of the snapshots are saved as `.avhd` files in the Virtual Hard Disks folder.

As you know, the VM configuration may be all over the place. When creating a VM or virtual disk, you can put it basically any place you like. So, how does Hyper-V know where to find everything? It uses *symbolic links*, which are very small files that refer to other files in a way that is transparent to applications. In this case, all relevant symbolic links are in the central configuration location for Hyper-V: `%systemdrive%\ProgramData\Microsoft\Windows\Hyper-V`. The folder ProgramData is hidden; it's visible only if you tell Explorer to not hide hidden and system files. Starting with Windows 2008 R2, you can actually view symbolic links as such from Explorer. In Figure 29.41, you can see from the small curved arrows in the icons that these are symbolic links, not ordinary files. Also, the file size of 0 bytes gives it away. In earlier versions of Windows, you would need to revert to the command prompt to tell the difference between symbolic links and normal files.



Name	Date modified	Type	Size
4700F741-BDC1-4F6B-A7D1-482B34D2544B.xml	9/4/2009 8:37 PM	XML Document	
C61BF4F4-D988-4CE2-A466-D41FF084237E.xml	7/21/2009 10:19 PM	XML Document	
FF508811-E20B-4A02-80CC-3F2D96842FF1.xml	9/4/2009 8:17 PM	XML Document	

FIGURE 29.41 The starting point for Hyper-V configuration files. All files here are symbolic links, pointing to the physical location of the XML files that define virtual machines.

This Hyper-V folder has four subdirectories: Snapshots, Snapshots Cache, Virtual Machines, and Virtual Machines Cache. The screenshot shows the Virtual Machine folder containing three symbolic links to XML files. This is one more than Figure 29.41 actually shows. The third VM definition is not in `E:\Hyper-V\*`, but it turns out to be on `C:\ClusterStorage\*`, which is used for Live Migration. The Cache folders are also used to support Live Migration. Read more about Live Migration in section "Moving VMs: Quick Migration and Live Migration."

Knowing how this reference mechanism works can be quite useful for repairing the Hyper-V configuration after some disaster. Keep it in mind!

Backing Up and Restoring Virtual Machines

From a systems management point of view, you should treat VMs as if they were physical machines, with some exceptions. When it comes to backup and restore, there are some special issues to consider. If you look at a Hyper-V host machine running a number of VMs, you may wonder whether you can back up all of that in one go. In the ideal world, you could do the following:

- Back up all VMs, and do so incrementally.
- Make all VMs aware of the backup so that they can do the right thing on restore. This is particularly important for Active Directory and Exchange.
- Restore VMs individually.

In principle you could do it this way. The reality is different, however. We'll review briefly how a modern Windows backup works.

Since the days of Windows 2003, we have had Volume Shadowcopy Services (VSS). This component does two things for the backup process. First, it can freeze a disk volume, take a snapshot, and unfreeze the disk again. The snapshot stays on the volume. It works using a differencing algorithm and takes little space. Second, the VSS service interacts with applications during the backup to allow them to clean up and prepare, such as flush buffers, stabilize data structures, and so on. You can see how this benefits complex applications. VSS allows them to be backed up quickly and with data structures intact. Such applications would be Active Directory, SQL Server, Exchange Server, and of course…Hyper-V.

So, to make VSS work in a virtual environment, you need a VSS provider for Hyper-V and a component in the VM that integrates with this provider. How else would the applications in the VM be aware that there is a backup going on? In other words, it's a two-stage rocket: the VSS process in the host needs to trigger the VSS process in the VM during backup, using the Hyper-V VSS provider as the communication channel. In the VM itself, it's the Integration Services that takes care of the integration. This is an important point to note: without working Integration Services, you cannot have a real backup from the host. Yes, using VSS you can back up the `.vhd` files that make up the VM, but the VM will not be aware of the backup. It would be nothing better than a live image backup with all its limitations.

Let's recap. You have two options to back up operating systems and applications in a VM:

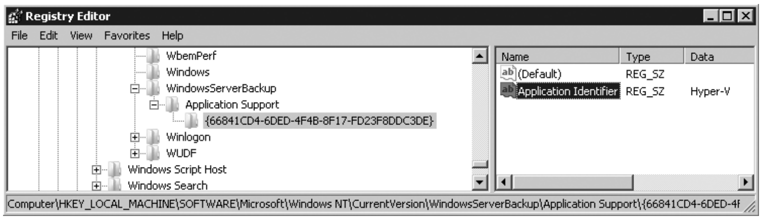
- Run a backup from the host, using a backup tool that is VSS aware. The Integration Services in the VM are required.
- Treat the VM as a physical machine, and run backups software from the virtualized OS.

There is a lot going for the second option. It will fit right it with your existing backup system, it won't break anything, and you know how to do it. This way, the host can be a black box that can be replaced at will. Be aware that a backup initiated from a VM generates a lot of disk I/O. If you have 10 VMs on one host all doing their backup at the same time, you generate 10 times the load on the host. It takes pretty good hardware to keep up with that!

On the other hand, backing up all those VMs in one shot is attractive as well, so let's take a look at that. The built-in tool for the job is Windows Server Backup. There is one problem you need to take care of before you start. By default, Hyper-V is not registered with VSS. If you made a backup now, the VSS service in the VMs would not be used. The following steps show you how to install WSB and to register the Hyper-V VSS provider. For more details, please read Knowledge Base article 958662.

1. Start Server Manager, select Features, then select Add Features. Find Windows Server Backup Features, select all subfeatures, and install it.
2. Start the registry editor, `regedit.exe`. Browse to `HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion`.
3. Create a new key called WindowsServerBackup.
4. Below this key, create a new key called Application Support.
5. Below this key, create a new key called {66841CD4-6DED-4F4B-8F17-FD23F8DDC3DE}. Yes, we're serious. This is the GUID corresponding to the Hyper-V VSS writer. Don't forget the curly brackets.

6. Below this key, create a new `REG_SZ` value called `Application Identifier`.
7. Set this value to `Hyper-V`, as shown in Figure 29.42.



Name	Type	Data
(Default)	REG_SZ	
Application Identifier	REG_SZ	Hyper-V

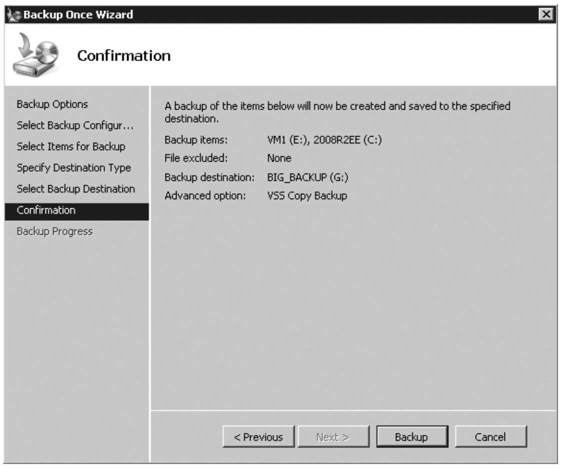
Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\WindowsServerBackup\Application Support\{66841CD4-6DED-4F

FIGURE 29.42 Required registry setting to register the Hyper-V VSS writer with Windows Server Backup

If you are going to do this more than once, you will probably want to export this key into a `.reg` file. There is no need to reboot. Windows Server Backup looks for VSS registrations every time it starts.

To get all VMs and their configuration, you must back up all volumes where you have the VMs, as well as the system volume that stores the Hyper-V configuration. In most cases, the best thing is to back up all volumes except the one that will hold the backup files. For this example, you'll to do a one-time backup to a local USB drive:

1. From the Administrative Tools, start Windows Server Backup. In the Actions pane, select "Backup once."
2. On the first screen, just click Next for Different Options.
3. Select Custom, and click Next.
4. Add Items, and select the system partition (C, most likely) and all partitions containing VMs.
5. Click Next to select the destination drive. This can be a local drive or a remote share. Figure 29.43 shows the summary screen.



A backup of the items below will now be created and saved to the specified destination.

Backup items: VM1 (E:), 2008R2EE (C:)
File excluded: None
Backup destination: BIG_BACKUP (G:)
Advanced option: VSS Copy Backup

FIGURE 29.43 Windows Server Backup, about to start the Hyper-V backup

6. Click the Backup button to start the process.

Although this is clear enough, the restore process is not nearly as nice. WSB offers no way to restore individual VMs. Either you can restore all the Hyper-V configuration and data or you need to hand-pick the VHD files you want recovered. Still, the full restore is good enough in a disaster recovery situation where the whole host is lost and you need to recover the VM setup. If that is your main purpose, you are all set!

Hyper-V Backup in the Real World

The limitations of Windows Server Backup make it unsuitable for larger companies. Its worst problem is not being able to restore individual VMs. Not only does this force you to jump through hoops for VM recovery, but it means you need to know on which host the VM was running at the time. So, what to do?

You probably have third-party backup software already. Many vendors have promised support for Hyper-V host-based backup, so with a bit of luck, you have a solution already in-house. The only supported solution from Microsoft is to use Data Protection Manager 2007 SP1, or any newer edition.

It may have occurred to you to use the Export feature as a backup. Still, an exported VM is not equivalent to a normal backup because there is no VSS integration. When you import a VM, it does not know it has been "restored." The downsides of using exports are that you do them one by one and that importing them requires you to delete the current VM first (if it still exists). If you know what you are doing, an export/import will do in a pinch. Otherwise, it's not a backup solution.

Server Core and the Hyper-V Server

If there is one scenario where the Server Core edition really shines, it's virtualization. What would you like in a virtualization host? Minimal memory footprint, as few services as possible, locked down by default, small attack surface, reduced patch frequency—Server Core has it. There is just one little disadvantage: you must manage it remotely because there is no MMC on Server Core and no out-of-the-box command-line interface to Hyper-V. In larger environments, this is no problem, since remote management should be the default option anyway.
In this section, we show you how to install Server Core with Hyper-V and how to manage it remotely. This example is suitable for both Windows 2008 and Windows 2008 R2. In reality, the R2 version of Server Core is easier to manage because it includes a menu-oriented command named `sconfig` to manage many configuration settings. Similarly, the Hyper-V Server has the command `hvconfig` to do the same thing.

The system requirements for Hyper-V on Server Core are basically the same as for a full installation of Windows Server 2008, with one strong recommendation: join the host to a domain. Remote management is just so much easier when the server is a member of an Active Directory forest. To recap, Table 29.9 is the suggested minimal configuration.

Table 29.9: Hyper-V on Server Core

Setting	Configuration
Internal memory	8GB; 4GB is the practical minimum
Hard disks	2×200GB or more
Partitions	Disk 1: System on C Disk 2: Reserved for Hyper-V on E
Network	2×1Gbit highly recommended.
Operating system	Windows 2008 SP2 or later with Hyper-V, 64-bit edition
Installation type	Server Core
Hostname	bf6 to follow the examples
IP configuration	Address: 192.168.1.55/24 Gateway: 192.168.1.1 DNS: 192.168.1.51
Active Directory	Domain-joined strongly recommended

Next, you will install Server Core, configure the network and firewall, name the server, join it to a domain, and install the Hyper-V role. Since Server Core is covered extensively elsewhere in this book, we will not show you the full installation. Please refer to Chapter 3, "The New Server: Introduction to Server Core," to remind yourself how it was done and to familiarize yourself with the command-line tools you might need.

In this example, the Server Core installation will work closely with the first Hyper-V host you installed (bf5). The IP configuration is compatible. Make sure they are connected to the same network using a hub or switch. Now perform the following steps:

1. Install Windows Server 2008 with Hyper-V, using the Server Core installation option. Install to the point where you log on for the first time using Administrator with a blank password, and set it to something meaningful. The next step is to set the IP configuration.
2. Examine the LAN interfaces. Take note of the IDX (index) number of the interface that you need to use. Enter the following at the command prompt:

   ```
   netsh interface ipv4 show interfaces
   ```

3. Set the IP number, mask, and interface using the IDX number from the previous step:

   ```
   netsh interface ipv4 set address name="<IDX>" source=static↩
   address=192.168.1.55 mask=255.255.255.0 gateway=192.168.1.1
   ```

4. Add a DNS server using the IP address of the DC bf1.bigfirm.com:

   ```
   netsh interface ipv4 add dnsserver name="<IDX>" address=192.168.1.51 index=1
   ```

With the IP configuration done, next up is giving the server a proper name and joining it to the domain bigfirm.com:

1. Rename the server to bf6:

   ```
   netdom renamecomputer localhost /NewName:bf6
   ```

2. As the `netdom` command told you, a reboot is required:

   ```
   shutdown /r /t 0
   ```

3. Let the machine restart, and log on again using the password you set in step 1.
4. To double-check, verify that you can resolve the domain. The following command should succeed and give you the IP address of at least one domain controller of the domain bigfirm.com:

   ```
   nslookup bigfirm.com
   ```

5. Join the server to the domain. You use the Administrator of the domain and need its password. The asterisk causes `netdom` to prompt you for the password:

   ```
   netdom join localhost /domain:bigfirm.com /userD:Administrator /passwordD:*
   ```

6. Again, a reboot is required:

   ```
   shutdown /r /t 0
   ```

To use the server in a domain with Hyper-V, you need to make it more manageable than its current locked-down state. Enable Remote Desktop, allow remote management through MMCs on other computers, and allow remote computers to access file shares:

1. Log on using a Domain Admin account such as BIGFIRM\Administrator.
2. Enable Remote Desktop:

   ```
   cscript \windows\system32\scregedit.wsf /ar 0
   ```

3. Optionally, you might want to allow older systems than Windows 2008 to use RDP:

   ```
   cscript \windows\system32\scregedit.wsf /cs 0
   ```

4. Enable the use of remote management tools:

   ```
   netsh advfirewall set currentprofile settings remotemanagement enable
   ```

5. Allow remote computers to access shared folders on this server:

   ```
   netsh advfirewall firewall set rule group="File and Printer Sharing" new
   enable=Yes
   ```

The final step is to install Hyper-V:

1. Log on to bf6 using Domain Admin credentials.
2. Install the Hyper-V role:

   ```
   ocsetup Microsoft-Hyper-V
   ```

3. Click Yes in the dialog box to restart. The server may reboot again automatically.

At this point, bf6 is a fully prepared Hyper-V host. As noted, it's not easy to manage Hyper-V from the command line without additional tools. WMI scripts are your only option using native instrumentation. Normally, you will use another server with a Hyper-V management console to control your Server Core hosts. As an illustration, follow these steps:

1. Log on to bf5, and start the Hyper-V management console.
2. Right-click Hyper-V Manager, and add bf6 as an additional server.

Use Group Policy to Configure Server Core

Take advantage of the power of Group Policy to configure Server Core (or Hyper-V Server) as much as possible. After the computer has joined the domain, you can use a firewall policy, use startup scripts, and so on.

Moving VMs: Quick Migration and Live Migration

It's impossible to have a full discussion of server virtualization without mentioning Quick Migration and Live Migration. These are related but different technologies to move virtual machines between hosts using shared storage and shared high-speed networks. Live Migration is the newer technology. Its key feature is that the unavailability of the VM during the actual move is very short. Subsecond times are not unusual. This time is short enough to keep TCP/IP sessions alive. In other words, anyone using that VM will at most notice a subsecond delay, but everything keeps working.

The ability to move VMs quickly between hosts opens up some interesting scenarios. Let's say you have at least three of four Hyper-V hosts deployed, and you are running all your VMs on shared storage with Live Migration enabled.

Easier host management Any time you need to do something on a host that requires downtime, you can move all VMs away without downtime for those VMs. Think hardware maintenance like adding memory, patching, and so on.

On-demand resource allocation You may have VMs that are running "hot" and are using a lot of CPU or disk I/O resources. With Live Migration, you can move this VM to a lightly loaded host or, alternatively, get other VMs out of the way instead.

Enabling green IT With Live Migration, you have the potential to shut down lowly used hosts, although in the real world you would need additional tooling to pull this off. The idea is that you move VMs away from lightly used hosts, putting them all on a few hosts that are well utilized—say, up to 60 percent average CPU time or whatever you prefer. The hosts without VMs can be shut down until they are needed again, with corresponding power savings.

All the major server virtualization vendors offer similar features but may use different names. However, shared storage implies SANs, iSCSI, and other storage technologies that we can't fully discuss here. Still, we'll cover the basic principles and show you a brief walk-through assuming that you are able to set up the basic storage infrastructure yourself. If you have never worked with remote storage before, we will have some hints to help you along.

Live Migration relies on failover clustering, known as *server clustering* in Windows 2003 and 2000. *Failover clustering* is the ability to transfer running applications between hosts, including all their data and current state. This transfer (also known as *failover event*) can be initiated by the user, but usually a number of conditions are set that should trigger the failover. Dependencies of the application are good examples of such a condition: disks, networks, certain services that should be running, and so on. If any of these goes missing, failover clustering will trigger a failover. An application needs to know that it's running on a cluster and take action when requested. Well-known examples are SQL Server, Exchange Server, File and Print Services, and so on. You can see where the shared storage comes in: a data disk for an application running on one node needs to go to another node if the application fails over.

Each server taking part in a clustering setup is a node. Since Windows 2008, the maximum number of nodes has been pushed to 16, an increase from 8 in Windows 2003. The main point of failover clustering is to provide high availability: if one host goes down, either planned or unexpectedly, the clustering services ensure that the application is restarted on another node.

An important limitation of Quick Migration in Hyper-V version 1 is that each VM requires its own logical unit (LUN). A LUN is a unit of storage on a SAN that is presented as a single disk to Windows. This leads quickly to drive letter exhaustion and other administrative complications. One of the major changes in Windows 2008 R2 is the introduction of a cluster shared volume (CSV). This is a way to store multiple files on a single LUN, simultaneously shared among two or more hosts. There is a locking mechanism that determines which node actually owns which file. CSV was written with Live Migration in mind, although it also works with Quick Migration. Its main function is to make life easier by simplifying the setup and configuration of a Hyper-V cluster and its VMs.

With failover clustering applied to Hyper-V virtual machines, you actually have two different failover technologies. First, Quick Migration works by saving the state of the VM, transferring control of its disk to another node, and restarting it on that node. This process is predictable and reliable but not so fast that it passes unnoticed. Time is lost during saving and restoring the VM state, which may take up to perhaps a minute for VMs with a lot of memory. Still, this is not a problem during a maintenance window where you need to work on the cluster node(s).

The second technology is Live Migration, called that because it is fast enough to move a VM to another node without loss of service. It works roughly like this; the key difference with Quick Migration is the way the VM memory is transferred:

1. The migration is initiated. The VM configuration is transferred to the destination host, which builds a skeleton VM.
2. The memory store of the VM is locked, and a difference file is started. All memory changes are written to this file.
3. The memory store of the VM is transferred to the destination node using the shared network. Clearly, this network should be as fast as possible. The destination node starts loading this memory into the skeleton VM.

4. The first difference file is locked, and a second one is started. The first difference file is transferred to the destination node. This process repeats until the difference files become small.
5. Up to now, the VM keeps running, but at this point, the VM is frozen, and the final difference file is transferred as well.
6. Control of the VHD files of the VM is transferred to the destination node. This happens quickly.
7. The VM configuration is removed from the original node and registered on the destination node.
8. The VM starts running on the destination.

How to Choose Between Live Migration and Quick Migration

Why would you ever want to use Quick Migration when you can have Live Migration? Well, one reason is that you might be running Hyper-V on Windows 2008, which works only with Quick Migration.

Another reason is that Live Migration could sometimes fail. If you look at how it works, you can see its theoretical weak spot: it needs to transfer memory faster to the destination host than the VM is changing it. If you have an application that writes a lot of data to memory very quickly and keeps doing this, it could make Live Migration impossible. Clearly, a fast network is required to make Live Migration reliable. Your legacy 10Mb Ethernet is not going to cut it.

But this does show the advantage of Quick Migration: it is deterministic because it saves and restores the VM state between migrations. So, if Live Migration ever fails, try it with Quick Migration.

After all this explaining, let's get on with the walk-through. Take a look at Figure 29.44 for the basic setup. To make this work, you need to get hold of some servers and shared storage:

- Two similar servers. The CPU and motherboard need to be as closely matched as possible. The Failover Clustering Wizard will tell you if you have a problem. The servers need two NICs.
- Installation media for a cluster-enabled version of Windows 2008 R2, meaning Enterprise edition or Datacenter.
- An exported VM ready for import will be very convenient. Alternatively, you will build a VM from scratch.
- Shared network, at least 1Gbit.
- Shared storage, either iSCSI or Fibre Channel. Standard two-node shared SCSI no longer works for clustering since Windows 2008. The simplest setup uses iSCSI, and this is what we will use as an example. There are many ways to get hold of an iSCSI storage device. Lots of NAS devices support it nowadays, there is Windows Storage Server, and if you have a TechNet or MSDN subscription, you can even download a software iSCSI target to install on Windows Server.

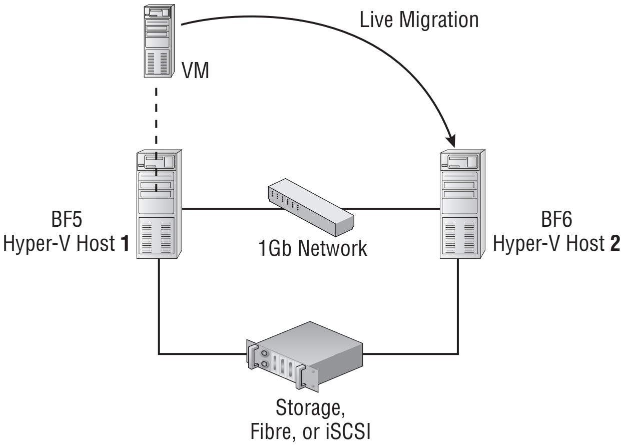


Figure 29.44
Moving VMs between hosts using Live Migration requires a high-speed network and shared storage.

In preparation, configure the following. Yes, we are assuming that you know how to set up and configure your iSCSI storage here or that you can have somebody do it for you. Apologies if this turns out to be a problem!

1. Install two Hyper-V servers, named bf5 and bf6. A setup as described before for server bf5 will do fine. Use one NIC for management traffic and another for the VMs. The key point is to *make sure to use the same name for the virtual networks on both nodes.* Also, deselect the check box on the virtual network called "Allow management operating system to share this network adapter," because you don't want the cluster to become confused about which NIC to use.
2. Use static IP configuration for bf5 and bf6. In our example, we use 172.16.1.5/24 and 172.16.1.6/24. Find an unused IP address to be used as cluster IP address, such as 172.16.1.9.
3. Join both machines to an Active Directory domain, such as bigfirm.com, as used in this book.
4. Set up a LUN on your iSCSI device for the shared cluster information (a witness disk). This one needs to have a size of at least 1GB.
5. Set up another LUN for the VM that we will use for Live Migration; 50GB should be a comfortable size.
6. Both LUNS should be configured to be used by the two Hyper-V machines simultaneously.

The next step is to create the witness disk and to assign it a drive letter. W seems like a good choice, though any unused drive letter will do. All machines in the cluster must use the same drive letter for the witness disk, though. We will deal with the shared VM disk later. Note that the order in which you do this does matter. Another order may "confuse" the Create Cluster Wizard when it looks for disks and networks to use.

1. Start out with bf5, and log on using a Domain Admin account.
2. Using the iSCSI Initiator console, add the LUN intended as the witness disk. Make sure to add it to the list of favorite targets, ensuring the drive stays mounted after a reboot.

3. Open Server Manager, and select Storage and Disk Management. Mark the witness disk as online, and initialize it. Create a new simple volume, and assign it drive letter W. Then, format it using NTFS, and assign it a label like Witness.
4. On the other machine (bf6), mount the witness disk using the iSCSI initiator. The same procedure applies. Online the disk, but do not initialize or format it; we did that already. Change the drive letter to W.

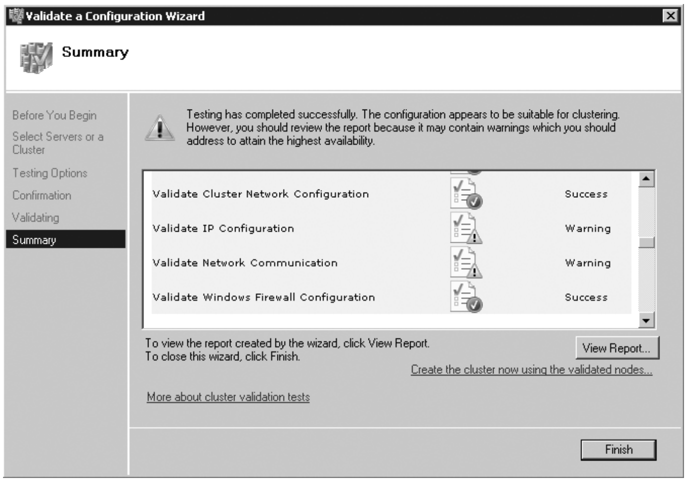
Now that you have storage set up, you can install the failover clustering features. No, it's not a role, although something as heavyweight as failover clustering should deserve to be one.

1. On bf5, open Server Manager, select Features, and add the Failover Clustering role. A reboot is not required.
2. Repeat for bf6.

With the Failover Clustering role installed on both nodes, you can configure the failover cluster itself. Step 1 is to validate the configuration; step 2 is to create the cluster. These are the most critical steps. If any of the following fails, you should go back and troubleshoot.

1. On bf5, open Failover Cluster Manager from the Administrative Tools. In the right pane, select Validate a Configuration.
2. The Validate a Configuration Wizard starts. Close the opening screen, and add bf5 and bf6 to the list of servers to be tested on the next screen.
3. Next up is the choice for which tests to run. Just run them all.
4. For some reason, you need to confirm this step in the next stage.
5. Start the actual validation. This could take several minutes.
6. The wizard will present you with the results that look something like Figure 29.45. Examine any warnings, and fix all errors before proceeding. The warnings shown in the figure are because we have only one NIC for cluster communication, which is a single point of failure. No problem for a test, of course.

FIGURE 29.45
The results of the cluster validation should look like this. If all is well, you are in a good position to go ahead and configure Live Migration for these nodes.


Validate a Configuration Wizard
Summary
Before You Begin
Select Servers or a Cluster
Testing Options
Confirmation
Validating
Summary
View Report...
Create the cluster now using the validated nodes...
More about cluster validation tests
Finish

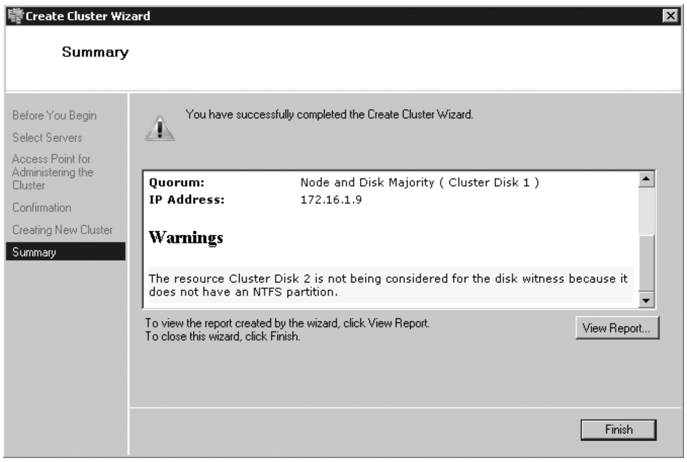
Testing has completed successfully. The configuration appears to be suitable for clustering. However, you should review the report because it may contain warnings which you should address to attain the highest availability.

Validate Cluster Network Configuration	Success
Validate IP Configuration	Warning
Validate Network Communication	Warning
Validate Windows Firewall Configuration	Success

To view the report created by the wizard, click View Report.
To close this wizard, click Finish.

7. Go back to Failover Cluster Manager, and select Create a Cluster.
8. The Create Cluster Wizard starts. Click away the opening screen, and add bf5 and bf6 as nodes to be added to the cluster.
9. The next screen is for the access point of the cluster. Enter a network name for the cluster, such hv1. The name must be unique in the Active Directory forest. Additionally, specify an IP address for this cluster name such as 172.16.1.9 in our setup.
10. Again, you need to confirm that you're sure you really want this.
11. Create the cluster. The wizard will look for the witness disk itself and configure it to be used accordingly.
12. If all is well, you end up with Figure 29.46. Note the warning that says that the second disk that it found is not considered as a witness. This is exactly what we need, because this disk will be used to hold the VMs.

FIGURE 29.46
Finalizing the creation of the Hyper-V cluster



You have successfully completed the Create Cluster Wizard.

Quorum: Node and Disk Majority (Cluster Disk 1)
IP Address: 172.16.1.9

Warnings

The resource Cluster Disk 2 is not being considered for the disk witness because it does not have an NTFS partition.

To view the report created by the wizard, click View Report.
To close this wizard, click Finish.

So, we have a cluster, but there is no Hyper-V in sight yet. That is changing now. The next step is to enable CSV and to add the prepared VM disk.

1. On bf5, open Failover Cluster Manager. Select the cluster configuration (hv1.bigfirm.com), right-click, and Enable Shared Cluster Volumes. You will need to approve a notice.
2. From Server Manager, open the Disk Manager. Mark the disk intended for VMs as online, and initialize it. Create a new simple volume, and assign it drive letter V. Then, format it using NTFS, and assign it a label like VMDISK1.
3. Back to the Failover Cluster Manager, right-click Storage, and select Add a Disk. From the dialog box, pick the disk you just created (should be only one).
4. On Failover Cluster Manager, right-click Cluster Shared Volumes, select Add Storage, and add the new disk.

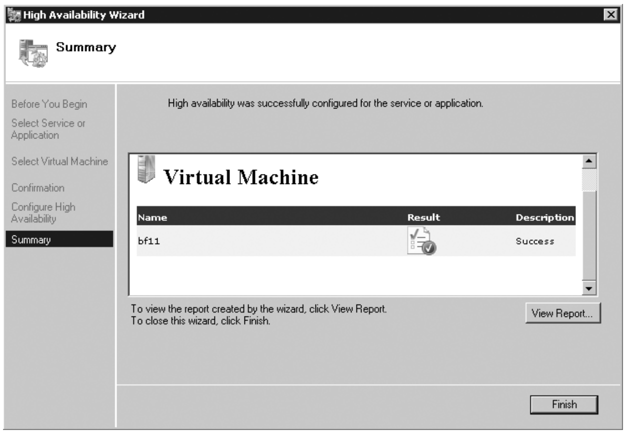
To check that all is well, you should look for a newly created folder named `C:\ClusterStorage\Volume1` on both nodes. At this point, you have a shared volume that you can use for virtual machines. You are almost there now. The whole system is ready for use. The next step is to create a virtual machine that you can use for testing. Create or import a new VM named bf11 (or whatever you prefer) on Hyper-V host bf5, and make sure that you do the following:

1. Place the virtual hard disks on the shared cluster storage at `C:\ClusterStorage\Volume1`.
2. Start the VM, and install the Integration Services, if required.
3. For networking, pick the network that you have created for use by VMs. Remember, this network should have the same name on both machines.
4. Give it an IP address that is reachable from the hosts, such as 172.16.1.50. Test this before you continue.
5. After you finalize the configuration, shut down the VM. Note: saving state is not sufficient, a shutdown is needed.

You have a VM, but the cluster does not know about it yet. To bring it under control of the cluster, you need to add the VM to its configuration.

1. On bf5, open Failover Cluster Manager. Select the cluster configuration (hv1.bigfirm.com), right-click Services and Applications, and select Configure a Service or Application. This starts the High Availability Wizard.
2. Click Next on the welcome screen.
3. On the second screen, you'll see a list with applications suitable for high availability. Pick Virtual Machine, and proceed.
4. On the final screen, choose the VM you want to make highly available. Pick the machine you just created, probably bf11.
5. Once again, confirm that you want to do this.
6. After the wizard finishes, the confirmation screen should show all green like Figure 29.47.

FIGURE 29.47
You just made virtual machine BF11 highly available.



High availability was successfully configured for the service or application.

Virtual Machine

Name	Result	Description
bf11	✓	Success

To view the report created by the wizard, click View Report.
To close this wizard, click Finish.

7. Finally, verify that the VM is ready to be used with Live Migration. In Failover Cluster Manager, open Services and Applications again, and select your VM. In the Summary pane, select the properties of virtual machine bf11. It has a tab called "Network for live migration." Make sure that at least one cluster network is selected. Note: you may have to refresh the GUI or to start the VM first to actually see this tab. We have seen some inconsistencies here.

The configuration is done: the VM is ready for Live Migration and Quick Migration as well. To set up the test, a nice trick is to connect a remote desktop to the VM and play with Explorer or do a `ping /t 172.16.1.50` from a remote machine to keep checking availability during the Live Migration. To initiate Live Migration, follow these steps:

1. Open Failover Cluster Manager, expand Services and Applications, and select bf11 (the VM that is enabled for Live Migration).

2. Right-click the VM, and select "Live Migrate virtual machine to another node." A list will expand showing all available nodes; pick one, and keep watching the ping window. After about 10 seconds, the VM will have moved to the other node while you have missed at most one ping—if all went well, of course.

3. Similarly, you can test quick migration and see that this is not as quick. Most of our tests kept the migration time under 10 seconds, which is short enough to keep a remote desktop session alive. During the migration, this desktop is frozen because the VM is being saved and restored.

That about sums it up. Configuring Live Migration is not a one-step process, but once you get the hang of it, it is easy enough. Having seen how it works, there are some final points you need to know.

First, there is backup and restore. In the section "Backing Up and Restoring Virtual Machines," we discussed how to back up a VM using Windows Server Backup and its VSS integration. However, in Hyper-V versions 1 (Windows 2008) and 2 (Windows 2008 R2), there is no VSS integration for clustered disks. So, you are back to making backups from inside the virtualized OS or need to use an external backup application that is cluster aware.

Second, you have some choices about which OS to use. As noted, both Enterprise edition and Datacenter editions are enabled for clustering and Live Migration. It does not matter if you install the full operating system or the Server Core edition. However, you have an additional choice: the Windows 2008 R2 Hyper-V Server is a valid cluster node. This is not a bad choice as a virtualization platform. It has a minimal footprint in terms of memory, disk space, and features. It can even be run from an USB device. The downside is that you only have the command line and the built-in menu system `hvconfig`. Still, for a large virtualization cluster, this should not be a problem.

Malware Protection and Patching

Most people are convinced by now that having some type of malware protection on servers is a good idea. The question is more, to what extent do you need protection? Do you need continuous scanning, or can you get away with daily or weekly scans? As long as you are in position to counter new threats, you are probably OK.

There is a possible complication with virtualization that may have occurred to you. Not only do you have the host to consider, but disk I/O is a precious resource on a virtualization platform. If you have a host with 10 VMs, all of them performing continuous malware scanning, the I/O system of the host gets 10 times more load because of scanning than a single server would have. In practice, this is not often a limiting factor. Modern antimalware software, certainly the server-class kind, knows how to be careful with I/O bandwidth. So, for VMs, the advice is clear: install malware scanners in all of them. Treat a VM the same way as a physical server.

But what about the Hyper-V host? If it runs no services except Hyper-V, what is the need for malware scanning? Especially on Server Core, there is little to attack for malware. Although that's true, that's not quite the point. Any operating system connected to a network is potentially vulnerable to attacks. Even a flaw in the TCP/IP stack could lead to a compromise of the system. So if you want full protection, you need to scan the host as well. Consider that by far the most disk activity will be performed by the VMs and you see that the performance impact of a malware scanner on the host will be low. Clearly, you will need to exclude the virtual disks and the Hyper-V processes from scanning. To be precise, exclude the default VM Configuration folder, the VHD folder, the Snapshot folder, `VMMS.exe`, and `VMWP.exe`.

Although malware scanning is a direct form of protection, keeping up-to-date with patches is indirect but just as important. There's no need to tell you that you should patch the VMs and the host. Patching the host may require a reboot, taking all the VMs offline. That requires planning and may cause problems if you cannot shut down VMs reliably. That's why Integration Services are so important. One of its functions is to initiate the save-state procedure in the VM when the host asks for it. The default action configured on each VM is that is saves state when the host shuts down. Clearly, the fewer patches the host needs, the less server interruption you will have. That is one of the reasons that Server Core makes a good virtualization platform—it needs fewer patches than a full server.

Scripting Hyper-V

One of the focus points of this book is to show you how to manage your network from the command line. The bad news is that there is no direct command-line interface to Hyper-V. There are no built-in tools to query the configuration of the host, start or stop VMs, and so on. The good news is that there is an extensive WMI interface to Hyper-V. This same interface is used by the Hyper-V management console. Theoretically speaking, you should be able to write a script to do anything the Hyper-V console can do. In practice, this is not easy. The WMI model of Hyper-V is quite complex, to put it mildly.

However, there are some simple and useful illustrations that show the basics of working with WMI to manage Hyper-V. All of the examples use VBScript, which is guaranteed to be present on any Windows Server, including Server Core. The scripts shown are stripped of all nonessentials so as to not distract from the main points. Production-quality scripts should include error checking, use explicitly declared variables, and so on.

Unfortunately, if you are not familiar with scripting or WMI, these scripts will make little sense to you. However, you will still be able to run a script using the following steps:

1. Find a folder to run these scripts, such as `C:\scripts`.
2. Open Notepad, type the script literally, and save it in `C:\scripts` with a descriptive name such as `list-vm.vbs`. Note that the `.vbs` extension is required.

3. Open a command prompt, and change the directory to `C:\scripts`.
4. Type **`cscript list-vm.vbs`**.

A typical example of working with WMI is to ask a Hyper-V host which VMs it has. The following script asks the local host (.) for its VMs. The WMI root namespace for Hyper-V is `\root\virtualization`, and the object class you need is `Msvm_ComputerSystem`. From this class, you can query the registered virtual machines.

```
strHost = "."
Set objWMIService = GetObject("winmgmts:\\" & strHost & "\root\virtualization")
Set vmCollection = objWMIService.ExecQuery("SELECT * FROM Msvm_ComputerSystem",,48)

For Each vm In vmCollection
    Wscript.Echo vm.Elementname & ", " & vm.Description & ", " & vm.EnabledState
Next
```

The output of this script is useful but shows some of the complications that are typical of the WMI interface to Hyper-V:

```
C:\tools>cscript list-vms.vbs
Microsoft (R) Windows Script Host Version 5.7
Copyright (C) Microsoft Corporation. All rights reserved.

bf5, Microsoft Hosting Computer System, 2
bf11, Microsoft Virtual Machine, 3
bf10, Microsoft Virtual Machine, 32769
```

In addition to the VMs (bf10, bf11), it also shows the parent partition (bf5). This is not so strange considering that the parent partition is just a special type of VM, but the result is probably not what you expected. The Description field can be used to distinguish between them. The `EnabledState` field is a number indicating the current power state. Not very intuitive, is it? Table 29.10 shows what the numbers mean. Interpreting the table, bf5 is running (of course, it is the host!), bf11 is shut down, and bf10 is powered off with a saved state.

TABLE 29.10: WMI Power State Codes

POWER STATE	MEANING
0	Unknown
2	Running
3	Powered off
4	Shutting down
10	Resetting
32768	Paused

Table 29.10: WMI Power State Codes *(continued)*

Power State	Meaning
32769	Saved (state)
32770	Starting up
32771	Snapshot in progress
32772	Migrating
32773	Saving State
32774	Stopping
32775	Deleted
32776	Pausing
32777	Resuming
<anything else>	Unknown/undocumented

Using this knowledge, you can take it one step further. These codes are used in other places, for instance to change the current power state of a VM. In the previous script, you see the object called `vm`, representing a virtual machine. This object has a method named `RequestStateChange()`, which accepts the desired power state as an argument. The method has a variety of return codes. The most important one is 4096, indicating that the start procedure was initiated successfully. The method is asynchronous, meaning that it directly returns control and does not wait until it knows that the action has fully succeeded. It just tells you it has started the action. The final complication here is that you do not want to start the host itself. The easiest way to check which VM is the host is to check the `Description` field of the VM object. The following script tries to start all VMs on the Hyper-V host it's run on.

```
strHost = "."
Set objWMIService = GetObject("winmgmts:\\" & strHost & "\root\virtualization")
Set vmCollection = objWMIService.ExecQuery("SELECT * FROM Msvm_ComputerSystem",,48)

For each vm In vmCollection

if vm.EnabledState <> 2 and vm.Description <> "Microsoft Hosting Computer System"
Then
        StatusCode= vm.RequestStateChange(2)

        if StatusCode = 4096 Then
            Wscript.Echo "Start signal sent to " & vm.ElementName
        Else

Wscript.Echo "An error occurred while starting " & vm.ElementName & ": " &
StatusCode
        End If
    End If
Next
```

The script uses code 2 from the table to check whether it is not running and start the VM. Change it to 32769, and the VM will save its state instead. Of course, you would check first that current state is 2 (running). These are the simplest kinds of examples of what you can do with the Hyper-V WMI interface.

Another class type worth knowing about is `Msvm_ImageManagementService`, which represents virtual hard disks. The following example does something very interesting: it mounts a `.vhd` file as a disk in the parent partition. To do that, it uses the first object returned by `Msvm_ImageManagementService`. This object always belongs to the parent. Using the `Mount()` method, the VHD file is made visible to Windows, although it will be offline at first if you use Windows Server 2008 Enterprise edition or Datacenter edition. The following script mounts the virtual disk belonging to bf10, specified by the variable `strVHD`:

```
strVHD = "E:\wsv_images\Virtual Hard Disks\bf10.vhd"
strHost = "."

Set objWMIService = GetObject("winmgmts:\\" & strHost & "\root\virtualization")
Set objVHDService = objWMIService.ExecQuery("SELECT * FROM Msvm_
ImageManagementService").ItemIndex(0)
objVHDService.Mount(strVHD)
```

To examine how this works, try the following:

1. Turn off or save-state the VM bf10. You cannot mount a `.vhd` file if it is in use by a running VM.
2. Open a command prompt, and start the Disk Manager by typing **`diskmgmt.msc`**.
3. Start Notepad, type the script, and save it as `c:\scripts\mount-vhd.vbs` on the Hyper-V host bf10.
4. From the command prompt, type **`cscript c:\scripts\mount-vhd.vbs`**.
5. Check the Disk Manager, and notice a new disk appears. The disk may be offline, as indicated by a red arrow pointing downward. On Windows 2008 Standard edition, it will be online.
6. Right-click the new disk, and select Online.
7. A drive letter is automatically assigned. At this point, you can open the drive in Explorer and change anything you like. From Windows' point of view, the `.vhd` file is now a local drive.

To reverse this, taking the drive offline is not enough. You need to unmount it. One way to do it is to use almost exactly the same script, but to replace the Mount() method by Unmount(). If you want to automate the online/offline procedure of the disk, investigate the command diskpart.

To finish up this section, we will cover another command-line trick, just to show you that there is more in life than WMI. Likely, you will have more than one Hyper-V host in your Active Directory forest. Is there an easy way to find them? Not surprisingly, there is such a way, and it's related to the computer account of the Hyper-V host.

As you know, each domain-joined computer has its own account in Active Directory. Any object in Active Directory is allowed to have child objects, although not all types are allowed. One of the child objects any computer account allowed to have is a Connection object. You may be familiar with Print Queue objects, allowing you to find printers just by querying the global catalog. Services such as Hyper-V can implement their own specific type of service connection point (SCP). This is a child object of any domain-joined Hyper-V host. Its name is always Microsoft Hyper-V.

Armed with this knowledge, you can find all Hyper-V hosts in your forest. If you want to script this, you will need the ADSI interface, not WMI. Both scripting technologies are covered on http://technet.microsoft.com, but there is another and simpler way to do it. Because the information is stored in Active Directory, you can use any LDAP query tool to read it. On a computer with the RSAT tooling or a Directory Services role installed, you will have the command-line tool dsquery. You will search for objects named Microsoft Hyper-V, of type serviceConnectionPoint. You ask specifically for the attribute serviceBindingInformation. From a command prompt, run the following:

```
dsquery * forestroot -filter↵
"&(cn=Microsoft Hyper-V)(objectCategory=serviceConnectionPoint)" -attr
serviceBindingInformation
```

The output will be similar to the following. It's a bit cryptic, but it's perfectly usable. The query returned two Hyper-V hosts in the forest, bf5 and bf6. serviceBindingInformation is a multivalued attribute, and its first member is always the name of the host. The second member is interesting as well; it tells you which TCP port to use for a remote connection using VMConnect.

```
serviceBindingInformation

bf5.bigfirm.com;
RDP listener port=2179;
msxml://C:\ProgramData\<…>\InitialStore.xml;

bf6.bigfirm.com;
RDP listener port=2179;
msxml://C:\ProgramData\<…>\InitialStore.xml;
```

The Bottom Line

Determine whether a server can run Hyper-V You are buying new servers whose main role will be to run Hyper-V. However, you are concerned that the new servers may not be capable of running Hyper-V because they do not meet the minimum requirements.

> **Master It** What are the CPU requirements to run Hyper-V? What brands may be considered, and are there other factors to be considered?

Determine when it makes sense to virtualize a server Your company is standardizing on Hyper-V virtual machines for all production servers. The strategy is to virtualize all servers, unless there is a good reason not to do so. One of the critical business applications (CalcIT) is a multithreaded modeling application that is very CPU intensive and may take days to run even on a big 16-core server.

> **Master It** Decide whether CalcIT is a good candidate for virtualization. If it is, explain why. If it is not, you need good arguments to convince your management to deviate from the standard.

Decide which technology to use to quickly move a virtual machine between hosts You are designing a Hyper-V testing lab. One of the requirements is that the hardware is used effectively for a varying collection of virtual machines. The idea is to maximize the use of available hardware and do so with the least overhead the next time the VM collection changes. Also, the process must be as reliable as possible.

> **Master It** What technology would you use, and how would your choice meet the requirements?

Advise on a backup strategy As an IT consultant you are hired by a company to have a look at their Hyper-V configuration. They are especially proud of their backup system, describing it as simple and effective. Using a script, they save the state of all VMs one by one, and during the suspended period, they copy the VHD file to a backup location.

> **Master It** Give the customer your opinion of their backup methods. If it is fine, compliment them on their choice, and leave a (self-)satisfied customer. If there is a problem, explain it to them, and propose an alternative. What do you recommend?