CHAPTER 7

Backing Up and Recovering Virtual Machines

For virtual machines running in a production environment, backup and recovery is just as serious as for all the other servers on the network. When running servers inside VMs, you'll be faced with several new challenges, as well as advantages, when planning and implementing a backup strategy.

In this chapter, you'll explore the process of planning for and administering backup and recovery operations on VMs and VM hosts. Along the way, you'll see the different approaches you can take to secure VM data, which include the following:

- Traditional agent-based backups
- Non-agent-based backups
- Flat-file backups

Also, many organizations have embraced the idea of maintaining a warm standby VM server that can be brought online if a primary server fails. This approach, for many organizations, may mean that data is unavailable for a few minutes following the loss of a server. If you can't afford to cluster all your systems, or if some of your applications don't support clustering, then you may find this approach to be a perfect fit. Since the focus of this chapter is purely on VM backup and recovery, we'll walk you through the process of maintaining a standby VM server in Chapter 14.

Optimizing your backup strategy often means much more than simply installing software and letting it do its magic. Oftentimes, custom scripting is required to get the backup results you desire. Because of the importance of getting your virtual infrastructure to work around your needs (instead of the other way around), we'll also show you several scripting ideas to both enhance and automate the backup, recovery, and availability of your VMs. Let's start with the most common production backup method today—running agent-based backups.

Performing Traditional Agent-Based Backups

As VMs perform write operations, some data winds up in the host system's physical memory before it's passed to a virtual disk file on the hard disk. This architecture is common in both the VMware and Microsoft products, and as a result, neither vendor supports online backups of virtual disk files. Instead, both strongly recommend you install backup agent software on the VMs in order to back them up, as opposed to running a single backup agent on the host system.

Installing backup agents on your VMs is the only surefire way to guarantee the reliability of backing up their virtual disk files. Keep in mind, however, that there's more to a VM than just its disk files. Each VM has configuration files too, and these files should be backed up by a backup agent running on the VM's host system.

■**Note** All the major backup vendors, including EMC (Legato), Veritas, CommVault, and Computer Associates, have performed testing at some level with their backup agents and virtual machine software. You should find that the backup agent will behave as if it's running on a physical system, with no awareness of the VM software that's hosting the virtual machine.

If you don't have an enterprise-class backup software suite but still need to run VMs 24/7, then you could use the backup tool included with each OS to back up your running VMs. For example, you could use Windows Backup to back up Windows VMs and could use the `dump` command to back up Linux VMs. Each of these scenarios is described later in the "Performing Non-Agent-Based Backups" section, but first let's take a look at agent-based backups.

Running Backup Agents on VMs

Running backup agents on VMs is the preferred method for backing up virtual disks, according to both Microsoft and EMC. A *backup agent* is software that runs as part of a backup software application, allowing you to back up a system either over the LAN or through a storage network to backup media.

When backing up VMs, several considerations will directly affect backup and recovery performance. When planning your backup infrastructure to include VMs, you have three primary architectural choices:

- Backing up over the LAN
- Backing up to the host
- Backing up directly to storage (local or SAN-attached storage)

Let's first look at the traditional LAN backup configuration.

Backing Up over the LAN

Most organizations today employ LAN-based backups. With this technique, backup data is sent over a network from the system being backed up to a system acting as a media server. The media server provides a physical interface between the systems being backed up and the storage hardware, such as a tape library. Figure 7-1 shows this configuration.

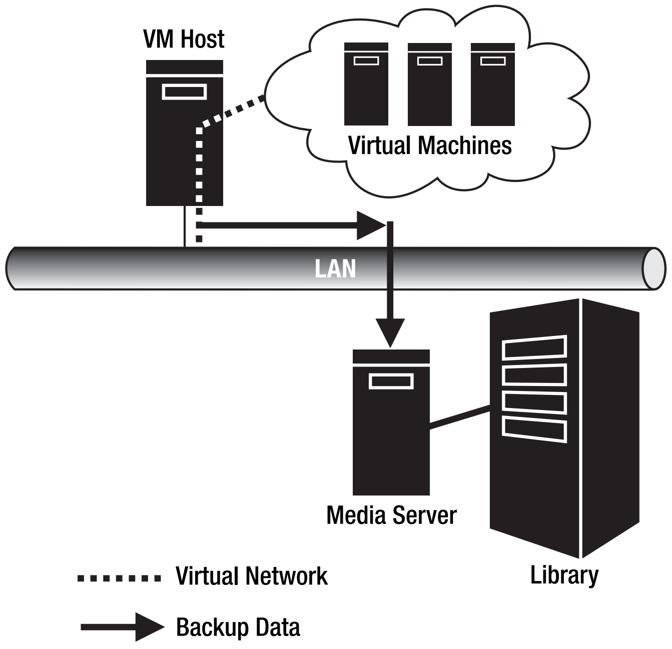


Figure 7-1. *Backing up a VM over the LAN*

With a LAN-based backup architecture, you could either network the VMs directly to the media server using bridged networking or have the VMs connect using a host-only network and route through the host to the media server. If the VMs are routing through a host system, remember that the media server will also need a static route present (or have the LAN router's route table updated) so that the media server can route back to the virtual machine. If routing isn't one of your strengths, then we recommend going with bridged networking to connect your VMs to the media server since this is the easiest scenario to configure.

Caution You can't use NAT to connect VMs on a private network to a media server. While the VMs could communicate to the media server through NAT, the media server wouldn't be able to directly communicate with a VM on its private IP address.

For those of you who want to have their VMs on a private host-only network, then let's consider a sample configuration. Assume the VM being backed up in Figure 7-1 has an IP address of 192.168.2.11/24. Now consider that the VM's host system has IP addresses of 192.168.2.1/24 (virtual interface) and 172.16.1.20/24 (production LAN interface). Finally, let's assume the media server has an IP address of 172.16.1.19/24. For the backups to work between the VM and media server, both systems will need to know how to route to each other. With the VM, this should be easy since you could just assign it the gateway address of 192.168.2.1. However, the media server may already have a default gateway on the production LAN with no knowledge of your private host-only network. If this is the case, you can assign a static route to the media server so that it knows how to reach the VM's host-only network. For the VM to reach the private network, its data will need to go through the production LAN interface of the VM's host system. So, at this point, you have all the information needed for creating a static route on the media server, which includes the following:

- **Destination network**: 192.168.2.0
- **Subnet mask**: 255.255.255.0
- **Gateway address**: 172.16.1.20

If you need a quick refresher on classless interdomain routing (CIDR) notation, remember that it's a means to represent a subnet mask as a single decimal value. So, the /24 that follows each IP address represents the number of consecutive binary 1s in the subnet mask. Since there are 8 bits in an IP address octet and since there are twenty-four 1s, you have three sets of eight consecutive 1s. If you convert eight consecutive 1s (1111111) from binary to decimal, you get 255.

■ **Note** If you're still craving more information on binary numbering, routing, and TCP/IP subnetting, point your Web browser to `http://www.learntosubnet.com`.

With this information now in hand, you can configure static routes on the media server. You do this using the `route add` command on either Windows or Linux systems. Here are sample commands to run if the media server is running on either a Windows box or a Linux box:

- **Windows**: `route add - p 192.168.2.0 mask 255.255.255.0 172.16.1.20`
- **Linux**: `route add –net 192.168.2.0 netmask 255.255.255.0 gw 172.16.1.20`

With the potential routing issues now behind you, you should have connectivity between your VM and the media server. If both systems allow ICMP requests, you should be able to run `ping` commands to test for the network connectivity between the two systems (for example, run `ping mediaserver` from the VM).

■ **Tip** If your backup software's architecture includes a central management server that initiates and manages all backup and recovery operations, ensure that network connectivity also exists between each VM and the management server.

At this point, you should be ready to install backup agents from your backup software vendor inside the VMs and to perform backups. With this architecture, it's likely that the LAN may become a bottleneck for backup data streams. For example, a 100Mb/sec LAN provides for 12.5MB/sec (100Mb ÷ 8) of bandwidth, which translates to 45GB per hour (12.5MB × 60 seconds × 60 minutes). Although this may seem like a pretty large number, it can be significantly reduced if multiple backups are running simultaneously and having to share the bandwidth of the media server's network interface. One alternative to solve a network bottleneck is to upgrade the network to 1Gb/sec or higher bandwidth. Another alternative is to configure the VM host system as a media server. We'll cover that option next.

Backing Up to the Host

When VMs are configured to back up directly to the host, no backup data has to traverse the production LAN. Figure 7-2 illustrates this configuration.

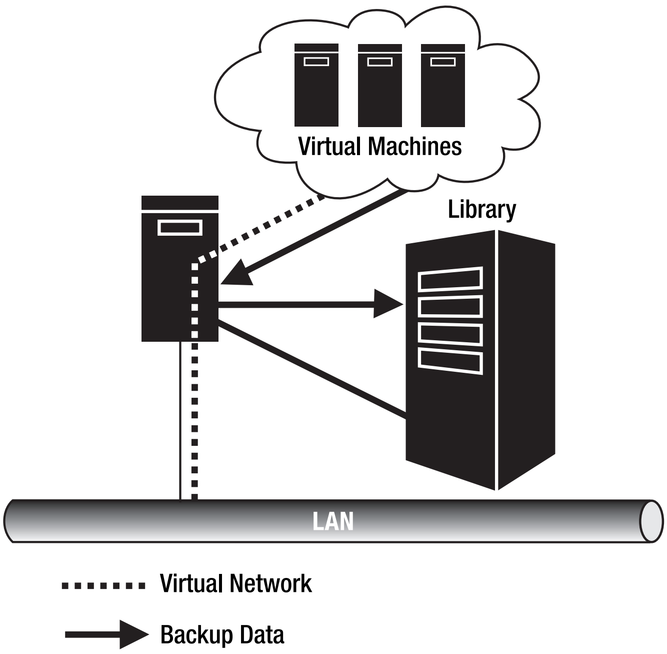


Figure 7-2. *Backing up a VM through the VM host*

In this configuration, the VM host is configured as a media server in your existing backup infrastructure and connects to physical backup devices such as a tape library either through a SCSI connection or through a storage network. Figure 7-2 shows a library SCSI attached to the VM host.

This configuration offers fast backups when compared to pushing backup data over a production LAN, and it may also allow other LAN-based backups to run faster, with VM backups no longer running over the LAN. The only disadvantage to this approach is that you'll need to connect the VM host to a backup storage device, which may involve purchasing additional storage resources or reallocating existing resources.

With this type of backup architecture, you should first configure the host system as a media server in your existing backup network. We prefer to configure the media server first because the installation of your backup software may require a reboot. Also, you may need knowledge of the media server installation while installing the backup agent software on the VMs. Once you have the media server configured, you can then install the backup agent software on each virtual machine.

Backing Up Directly to Storage

Neither Virtual PC 2004 nor Virtual Server 2005 supports connecting VMs to storage devices such as tape drives or libraries either through a SCSI bus attached to the host or through a SAN. Hard disks are the only device located on a SAN that Virtual Server supports. All other storage devices can't be accessed by Virtual Server 2005 VMs. Basically, this means that if you're hosting VMs using either of the Microsoft virtualization applications, configuring your VMs to perform network-based backups—either internally to the VM host or over the LAN—is your best backup option.

VMware Workstation, GSX Server, and ESX Server all support connecting VMs directly to SCSI or Fibre Channel storage devices, such as tape drives and libraries. You can find more information on using the Generic SCSI Device hardware resource to create this connection in Chapter 6.

With support for other storage devices, you can configure a VM to be its own media server and thus back up directly to a local SCSI or Fibre Channel storage device. Figure 7-3 shows this configuration.

In backing up directly to a storage device such as a tape drive, you don't have any network bandwidth contention, whether on a logical host-only network or over a physical LAN. This approach offers the best possible performance but requires the additional storage resources to allocate to the VMs you want to back up. If your backup platform supports dynamic drive and library sharing in a SAN, then you may find backing up VMs directly to storage on a SAN to be the perfect solution.

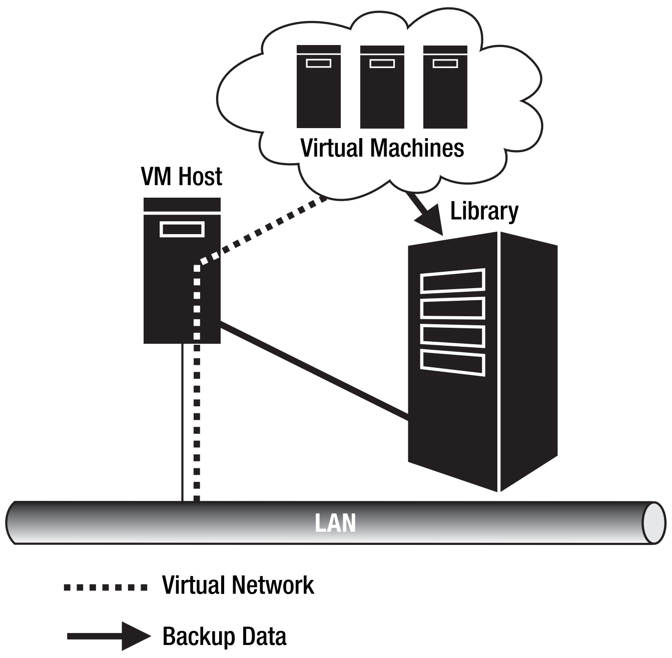


Figure 7-3. *Backing up a VM directly to a storage device*

If you're now sold on using your backup software to back up a VM directly to a storage device, here's the general procedure to make it all work on VMware GSX Server:

1. Add the storage device to the VM as a generic SCSI device.
2. Start the VM, and make sure it can see the storage device.
3. Install your backup application's media server software, and configure the storage device for use with your backup software.
4. Install your backup application's backup agent software on the VM.
5. Back up the virtual machine.

Although installing and running backup agents on VMs protects their virtual disk files, you still need to protect each VM's configuration files. You do this by backing up the host system.

Running Backup Agents on the Host

Running backups on the host system serves two general purposes:

- Secures each VM's configuration files
- Secures the host system's data, drivers, and configuration to prepare for a system failure or disaster

---

■**Caution** Running an open file agent such as St. Bernard Software's Open File Manager on the host system won't guarantee reliable backup of open virtual disk files and thus should be used with extreme caution. You should perform frequent test restores to verify the validity of a virtual disk file backed up via an open file backup agent.

■**Caution** Windows Server 2003's Volume Shadow Copy Service doesn't support online snapshots of virtual disk files, so this service shouldn't be used as a method of backing up VMware, Virtual PC, or Virtual Server virtual disk files.

In terms of setup, installing a backup agent on a VM host is no different from installing an agent on any other server on your network. With agents securing the data on the VM virtual disk files, remember that you don't even need to have your backup agent attempt to back them up. This means that if your backup agent supports filtering, you can filter out all virtual disk files from the backup, thus preventing the backup agent from even attempting to back up a virtual disk file.

If you don't plan to install backup agents on your VMs, another alternative is to power down the VMs prior to backing up the host. When a VM is powered down, its virtual disk files are no longer locked by the VM application and thus can be backed up safely. Later in the "Performing Flat-File Backups" section, you'll look at ways to automate powering down running VMs so that they can be backed up without needing a backup agent installed on them.

Now that you've seen how to protect VMs using enterprise backup applications, let's look at the tools operating systems have available for stand-alone local backups.

Performing Non-Agent-Based Backups

If you're running VMs in a small or home office, purchasing enterprise-class backup software may not be an option. Instead, you may find that the backup tools included with each VM's operating system are well-suited for supporting your backup needs. In this section, you'll examine tools for backing up the two most popular virtualized operating systems: Windows and Linux.

These tools support the three most popular backup types:

- **Full (normal)**: Backup of all files on a disk
- **Incremental**: Backup of all files that have changed since the time of the last backup
- **Differential**: Backup of all files that have changed since the time of the last full backup

Full backups are the most popular backup method for VMs, as they secure every file on a virtual hard disk. If you have a small tolerance for data loss, or if you don't have the media to support daily full backups, you may need to mix incremental or differential backups with the full backups. One popular approach is to run a full backup every Saturday and run incremental backups Monday to Friday. This means that if a virtual disk failed, you'd lose no more than one day's worth of data. If you need even better protection, you could run a weekly full backup on Saturday, differential backups daily (Monday to Friday), and incremental backups every four hours. This means that the most data ever lost would be four hours. The drawback to this approach, however, is that it would involve running backup jobs (the incrementals) during business hours, which can slow down server performance during the time of the backup. In this approach, you'd need to restore that last full backup, the last differential backup, and every incremental backup that occurred after the time of the last differential. Since a differential backup copies all files that have changed since the time of the last full backup, any earlier (prior to the time of the last differential) differential or incremental backups wouldn't be needed for recovery.

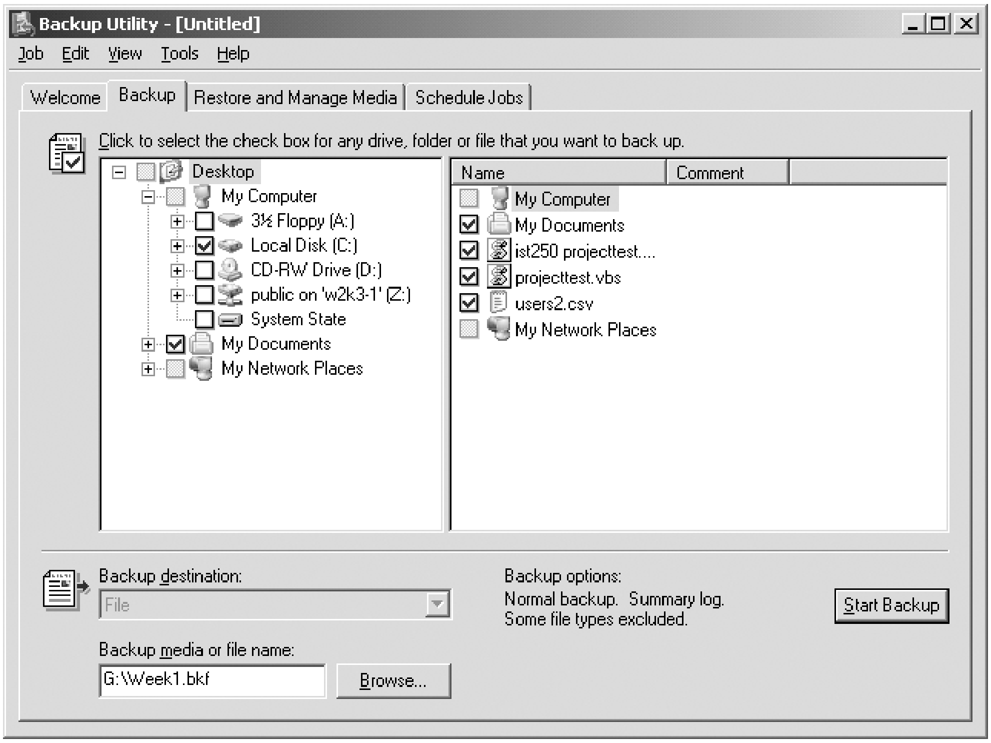
With backup strategy, there really is no single answer. You have to decide on your organization's tolerance for data loss and use that as a baseline to determine the frequency and types of backups to run on your VMs.

With a baseline for backup strategy under your belt, you'll now learn how to use Windows Backup to secure data.

■ **Caution** If you're using a local tool to back up your VMs or host system, use extreme caution when storing backup data in a single file on the host system. If the host system's hard disks are corrupt or become damaged, you'll lose not only your original data but your backup as well! A best practice when backing up to a file is to store the backup file on another server by specifying either a UNC path or a network file system (NFS) mount point as the location for the backup file.

Using Windows Backup

Windows Backup (formerly called NT Backup), shown in Figure 7-4, is a tool that has been included with the Windows operating systems as far back as Windows NT. You can run this tool as either a GUI application or a command-line utility; it supports backing up data to removable media devices such as tapes or to a single file on a hard disk. This flexibility gives you plenty of options when backing up VMs.



Click to select the check box for any drive, folder or file that you want to back up.

- Desktop
  - My Computer
    - ☐ 3½ Floppy (A:)
    - ☑ Local Disk (C:)
    - ☐ CD-RW Drive (D:)
    - ☐ public on 'w2k3-1' (Z:)
    - ☐ System State
  - ☑ My Documents
  - My Network Places

Name	Comment
My Computer	
☑ My Documents	
☑ ist250 projecttest....	
☑ projecttest.vbs	
☑ users2.csv	
My Network Places	

Backup destination: File

Backup options:
Normal backup. Summary log.
Some file types excluded.

Backup media or file name: G:\Week1.bkf

Figure 7-4. *Windows Backup*

■ **Note** The procedures outlined in this section are for a Windows Server 2003 virtual machine. The procedures for other Windows operating systems will be similar but may not be exactly the same.

To perform a backup with Windows Backup, follow these steps:

1. To open Windows Backup, click Start ➤ All Programs ➤ Accessories ➤ System Tools ➤ Backup.
2. If the tool opens in wizard mode, you'll need to answer a series of questions to start the backup. If the tool doesn't open in wizard mode, proceed to step 3.
3. Once the tool opens, click the Backup tab.
4. Next, select the disks (or specific data) to back up. You do this by checking the box that corresponds to the data to be backed up. You'll see that the GUI behaves similarly to Windows Explorer.

5. Now browse to or enter a destination for the backup data in the Backup Media or File Name field.

6. When you're satisfied with your selections, click the Start Backup button.

7. To begin the backup, verify that all your selections are accurate in the Backup Job Information dialog box, and then click the Start Backup button. If you want the backup job to run at a later time or on a recurring basis, you can click the Schedule button at this time to schedule when the job should run.

■ **Tip** To back up all the data on a virtual machine, in wizard mode you'd select the Back Up Everything on This Computer option. If the tool starts in standard mode, click the Backup tab and then select the My Computer check box. At a minimum, you should always back up all pertinent data drives, the boot drive, and the system drive, as well as the system state.

With the default options selected, Windows Backup will always perform a full backup. If you want to perform either an incremental or a differential backup, in step 7 you'd need to click the Advanced button in the Backup Job Information dialog box and then select the appropriate backup type in the Backup Type drop-down menu.

You can use the same tool to restore data by following these steps:

1. Open Windows Backup by clicking Start ➤ All Programs ➤ Accessories ➤ System Tools ➤ Backup.

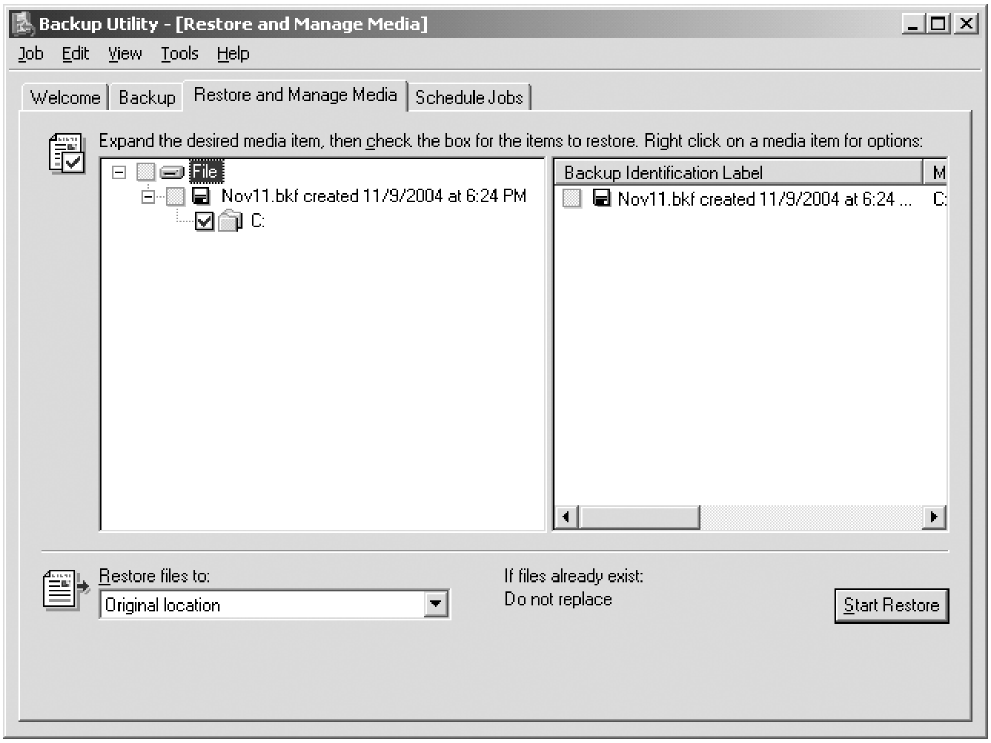
2. If the tool opens in wizard mode, you'll need to answer a series of questions to start the restore. If the tool doesn't open in wizard mode, proceed to step 3.

3. Once the tool opens, click the Restore and Manage Media tab.

4. Next, expand the appropriate media item (such as a file or tape), and select the data to restore (see Figure 7-5).

5. Now click the Start Restore button.

6. In the Confirm Restore dialog box, click OK. The restore job should run and complete.



Expand the desired media item, then check the box for the items to restore. Right click on a media item for options:

File
Nov11.bkf created 11/9/2004 at 6:24 PM
C:

Backup Identification Label	M
Nov11.bkf created 11/9/2004 at 6:24 ...	C:

Restore files to: Original location

If files already exist: Do not replace

Figure 7-5. *Selecting data to restore using Windows Backup*

At this point, you have the general procedures to use Windows Backup to back up and recover data stored on VMs. Keep in mind that you can still use two of the architectures mentioned in the previous section as methods for saving backup data. For example, you could store backup data in a file saved on a mapped network drive located on another physical server on the network. This will allow you to get your VM's backup data off the VM host. Since one of the primary purposes of backing up data is to protect against system failure, it's never a good idea to store a VM's backup data on its host system.

One other alternative with Windows Backup is to add a generic SCSI storage device to your VM (VMware only) and back up the VM directly to the storage device. In this scenario, you'll need to carefully manage backup media to ensure that the host system doesn't try to use the same tapes being used by a VM. Otherwise, one system may overwrite backup data generated by another system.

Now that you've seen the possibilities with Windows Backup, next you'll look at using `dump` and `tar` to back up Linux volumes. If you're interested in running Windows Backup from the command line, you can refer to the "Performing Flat-File Backups" section later in this chapter.

Backing Up Linux File Systems

The `dump` command has been around practically forever, dating back to early Unix operating systems (beginning with AT&T Unix version 6). This command allows you to perform full and incremental backups to a local storage device or volume or to a remote mount point. `tar` is another popular tool that can package files and folders for transferring over a network and for performing backups. In the following sections, we'll cover how to back up your Linux VMs using either dump or tar. Also, you'll see how to use the cron daemon to create an automated backup schedule for your Linux VMs. Let's get started by looking at dump.

Backing Up with dump

Listing 7-1 is a shell script that uses dump to perform a full backup of a VM named W2KFS1.

Listing 7-1. *Backing Up with dump*

```
#!/bin/sh
#
# Filename: LinuxVmDumpBackup.sh
# Purpose: Uses dump to back up a VM to Tape Drive 0. This script performs
# a full backup of a VM's hard disks.
#==================================================================
# ==== Assign variables ====
DriveToBackup="/dev/hda2"     # Name of drive to back up
BackupDestination="/dev/nrft0" # tape drive 0, tape not rewound after bu

# ==== Run the backup=====
/sbin/dump 0uf $BackupDestination $DriveToBackup
```

In the script, the DriveToBackup variable identifies the drives to be backed up, and the BackupDestination variable defines the target backup device. This script will perform full backups of a virtual machine. You can refine this script to perform incremental backups following a full backup by changing the dump level specified in the dump command. A level of 0 indicates a full backup. The values 1 to 9 in the command sequence indicate incremental levels. For example, the following are the script modifications necessary to run a full backup on Sunday and incremental backups from Monday to Friday:

- **Sunday full**: /sbin/dump 0uf $BackupDestination $DriveToBackup
- **Monday incremental**: /sbin/dump 1uf $BackupDestination $DriveToBackup
- **Tuesday incremental**: /sbin/dump 2uf $BackupDestination $DriveToBackup
- **Wednesday incremental**: /sbin/dump 3uf $BackupDestination $DriveToBackup
- **Thursday incremental**: /sbin/dump 4uf $BackupDestination $DriveToBackup
- **Friday incremental**: /sbin/dump 5uf $BackupDestination $DriveToBackup

With this approach, you'd have six versions of the script with the only difference in each version being a single line in the script. After the Friday backup completes, you'd want to insert a new tape in the drive to be used by the next backup cycle, starting on Sunday.

To restore data backed up with dump, you need to use the restore command. To perform a complete restore of all data on a tape, use the following syntax:

```
restore rf <backup device>
```

Following the earlier example, run this command:

```
restore rf /dev/nrft0
```

Another popular method of using `restore` is to run the command in interactive mode. To run a restore in interactive mode, you use this syntax:

```
restore if <backup device>
```

In the previous backup example, you'd run `restore if /dev/nrft0` to start the interactive restore.

Using this approach, the command will show you an index of data backed up on a tape and allow you to select the files and folders you want to restore. When looking to restore just a few files or folders, this method is popular. Let's look at an example of an interactive-mode restore. Assume you need to restore a file named `contactinfo.doc` from the `/data/marketing` folder. Prior to running the restore operation, first navigate to the directory in which you want the files restored to; in this example, you'd run `cd /data/marketing`. Next, insert the tape that contains the data to be restored into the tape drive. Finally, run the following commands to recover the file:

```
#restore if /dev/nrft0
restore >ls
.:
data/          misc/
restore >cd data
restore >ls
./data:
marketing/          /sales          /service          /support
restore >cd marketing
restore >ls
./marketing:
brochure.doc          contactinfo.doc          suggestionlist.doc
restore >add contactinfo.doc
restore >ls
./marketing:
brochure.doc          *contactinfo.doc          suggestionlist.doc
restore >extract
```

Notice that after you add the `contactinfo.doc` file to the recovery list using the `add` command, you ran the `ls` command in the `marketing` folder, and the selected file (`contactinfo.doc`) was preceded with an asterisk (*). This indicates the file has been selected for recovery. Once you run the `extract` command, all files selected for recovery will be restored at that time.

While `dump` has been a popular backup tool, `tar` has significantly grown in popularity, primarily because of its reliability, ease of use, and flexibility. Next, we'll cover how to use `tar` to protect your VMs.

Backing Up with tar

`tar` is another popular tool for backing up data. With `tar`, you can back up a VM's contents into a single file. Listing 7-2 shows a script that uses `tar` to back up a VM's root directory and associated subdirectories to an NFS mount point.

Listing 7-2. *Backing Up with tar*

```
#!/bin/sh
#
# Filename: LinuxVmTarBackup.sh
# Purpose: Uses tar to back up a VM to an NFS mount point. This script performs
# a full backup of the root directory and filters out /tmp, /proc, and /mnt.
#=========================================================================
# ==== Assign variables ====
VMName="FS1"     # Name of VM to back up
Today=$(date +'%m-%d-%y')     #Loads current date as mm-dd-yyyy into variable
DirToBackup="/"     #Specifies to back up root.
NfsPath="mediasrv1:/data/Backup"     #Specifies NfsPath to back up to
#Name of backup file. Includes date stamp.
BackupDestination="/mnt/backup/"$VmName"-"$Today".tar"
#Name and location of backup log file
BackupLog="/mnt/backup/"$VmName"-Backup_Log_"$Today".log"
#Directories to exclude from backup
ExcludeList="--exclude /proc --exclude /mnt --exclude /tmp"
#==== Mount backup folder ====
mount –t nfs $NfsPath /mnt/backup

# ==== Run the backup=====
tar –cpvf $BackupDestination $DirToBackup $ExcludeList >&$BackupLog

#==== Unmount backup folder ====
umount /mnt/backup
```

In this script, you back up the root (/) folder while excluding the /tmp, /proc, and /mnt folders. The NFS mount point is mounted to /mnt/backup. This folder must be created before the script runs for the first time. The output backup file that's created will include information on the VM's name as well as the date that the backup was performed. This is useful if you intend to back up several VMs to the same location. In the sample LinuxVmTarBackup.sh script, if the backup was executed on December 20, 2004, the output file would be named FS1-12-20-2004.tar. The backup's associated log file would be named FS1-Backup_Log_12-20-2004.log.

To restore a tar backup file, you use the extract (x) option in place of the c option. For example, to extract a tar backup archive back to the root (/) directory, you run the following command:

```
tar xpvf /mnt/backup/FS1-12-20-2004.tar --overwrite
```

This command extracts the FS1-12-20-2004.tar archive to the root folder. The --overwrite option causes the tar command to overwrite like files during the restore operation.

Now that you've looked at both tar-based and dump-based backups and restores, we'll cover how to automate the backup process with cron.

Note Another excellent free Linux backup tool is the Advanced Maryland Backup Disk Archiver (AMANDA), which automates network-based backups of Linux hosts to a central media server. This is similar to the approach for backing up using backup agents. You can download a free copy of AMANDA and read more about this excellent product at `http://www.amanda.org`. In Chapter 14, we'll show you an example of using AMANDA to configure backups to run across several systems.

Automating Backups with cron

By using the cron daemon, you can automate the execution of scripted backups. This gives you the ability to set up your backup schedule and let it take care of itself. Jobs executed by cron are stored in the `/etc/crontab` file. By default, the file contains the following lines:

```
SHELL=/bin/bash
PATH=/sbin:/bin/:/usr/sbin:/usr/bin
MAILTO=root
HOME=/

# run-parts
01 * * * * root run-parts /etc/cron.hourly
02 4 * * * root run-parts /etc/cron.daily
22 4 * * 0 root run-parts /etc/cron.weekly
42 4 1 * * root run-parts /etc/cron.monthly
```

In this file, the purpose of the first four lines is to declare variables that establish cron's working environment. The `run-parts` section is where you can schedule tasks to run. By default, hourly, daily, weekly, and monthly tasks are already configured. Any scripts stored in the `/etc/cron.hourly` folder will run on the first minute of each hour, and scripts in the `/etc/cron.daily` folder run each day at 4:02 a.m., by default. Using these folders gives you an easy means to have several scripts execute simultaneously with a single line of code in the `crontab` file. To understand how the time and date settings are configured, let's examine the `crontab` file format illustrated in Figure 7-6.

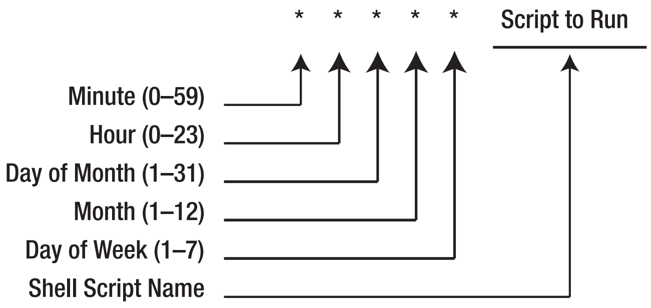


Figure 7-6. *`crontab` file text format*

The first five positions (separated by spaces) in a line of the `crontab` file specify the day and time to run the script, and the sixth position indicates the script to be executed. The * wildcard can represent all instances of a value. So, if you wanted a backup script job to run at 1 a.m. every day, you'd use the values `0 01 * * *`. With the day of the week parameter, the value `1` represents the first day of the work week, which is Monday. Sunday equals `7`.

Now let's consider the required line in the `crontab` file that's needed to execute the `LinuxVmTarBackup.sh` script. To perform this task every Friday night at 11 p.m., you'd add this line:

```
0 11 * * 5 /root/scripts/LinuxVmTarBackup.sh
```

Think you've got it now? Take a look at the last default line in the `crontab` file shown earlier. When do the scripts located in the `/etc/cron.monthly` folder run? If you want to test yourself, don't read the next sentence until you have an answer! Okay, if you came up with the first day of each month at 4:42 a.m., you're correct!

As you can see, cron can be invaluable in automating the backups of your VMs.

■ **Tip** cron is a powerful tool, and in this section we merely scratched the surface of its capabilities. For more information on cron, point your Web browser to `http://www.redhat.com/docs/manuals/enterprise` to download the Red Hat Enterprise Linux System Administration Guide.

Performing Flat-File Backups

If you don't plan to install backup agents on your VMs and don't plan to run local independent backups on each VM, then performing flat-file backups is your best alternative. The technique of flat-file backups originated as a way to back up databases without having to use a backup agent. When a flat-file backup is performed on a database, the database first shuts down (is taken to an offline state), and then its related files are backed up. Shutting down the database prior to backup guarantees that you'll be able to back up all the data contained within the database's related files. When a database is online, the database application locks the database's related files so that normal backups aren't possible. The only reliable way to back up an online database is by using a backup agent that's supported by the database application.

So, if you relate a VM's disk and configuration files to a database's data and log files, you have the same problem with backups as you would with databases. You can't just run a backup of the host system and expect to reliably back up running VMs. Not only does each VM application lock virtual machine files while VMs are running but also the VMs use physical memory as a buffer for writes to virtual disk files. This means that not all the data that a VM thinks is inside a virtual disk is actually in the virtual disk file on the host. So, backing up a virtual disk file while a VM is running could mean that you're backing up a corrupted disk, which will be useless when you attempt a restore.

Now that you've seen why you shouldn't try to back up a running VM directly from the host, let's look at the procedure for running a flat-file backup. To perform this backup, you must follow three general steps:

1. Shut down the VM.
2. Back up the VM's related files.
3. Start up the VM.

Normally, all VM files reside in a single folder, so backing up the offline VM should include simply making a copy of a VM's folder.

■Caution If a VM is configured to map one of its hard disks to a physical disk on the host system, you must be sure to include the disk in the VM's backup definition. For example, if a VM's configuration and a single virtual disk file are stored in the `E:\VMs\NT4Srv` folder but the VM also maps a disk to the host's F drive, you must back up the F drive on the host as well as the `E:\VMs\NT4Srv` folder when the backup job runs.

If you're running VMs in production, it's likely you'll want the VM to be offline for as little time as possible. With this in mind, the easiest way to automate a VM shutdown, backup, and startup is by using a script. In the following sections, we'll cover how to script flat-file backups of the following VM configurations:

- VMware Workstation on Windows
- VMware Workstation on Linux
- VMware GSX Server on Windows
- VMware GSX Server on Linux
- Microsoft Virtual PC 2004
- Microsoft Virtual Server 2005

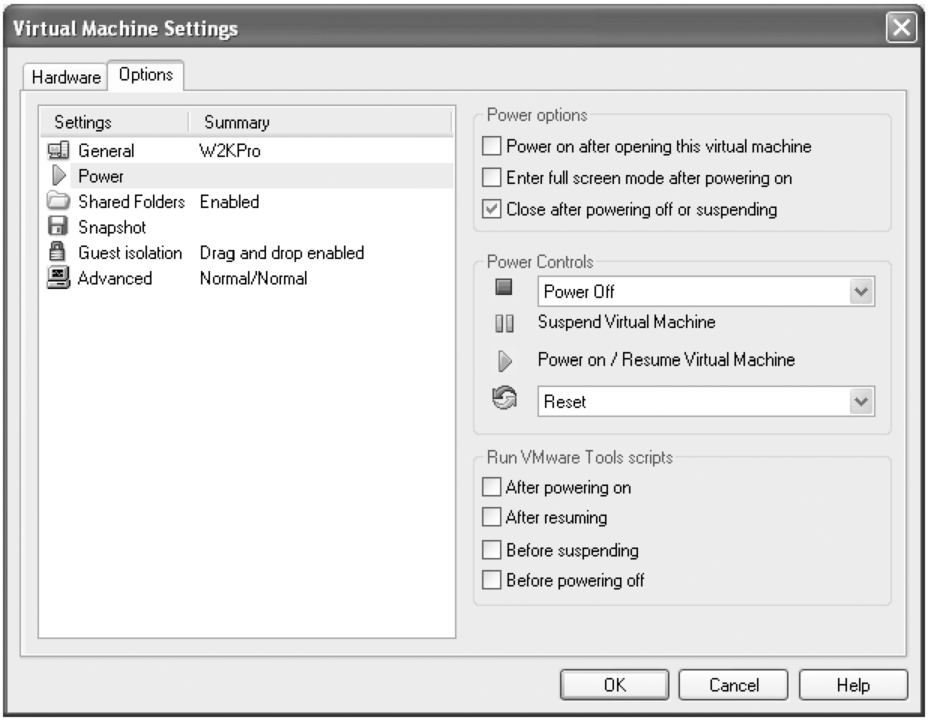
Each section details scripts for automating the backup process, so you can just turn to the section that's right for you, depending on the virtual machine applications in your environment. We'll start by looking at scripting flat-file backup jobs for VMware Workstation.

■Tip If you want to schedule Windows batch scripts to run on a recurring basis, you can do so using the Scheduled Tasks system tool. This tool allows you to select a schedule pattern in which to run a job and also gives you the ability to supply user credentials under which to execute the job. This prevents you from having to code the user credentials inside the script.

Running VMware Workstation Flat-File Backups

Workstation VMs lock their configuration file while a VM is open, so to fully back up a VMware VM, you must not only power down the VM but also close the specific VM you plan to back up. Lucky for you, VMware Workstation provides an easy method for automating this. To configure a VM to automatically close once it's powered down, perform the following steps:

1. In the Workstation UI, click the VM you plan to back up. Then click the VM menu, and select Settings.
2. In the Virtual Machine Settings dialog box, click the Options tab.
3. Under the Options tab, click the Power setting, and then check the Close After Powering Off or Suspending box (see Figure 7-7).
4. Click OK to close the Virtual Machine Settings dialog box.


Virtual Machine Settings

Settings	Summary
General	W2KPro
Power	
Shared Folders	Enabled
Snapshot	
Guest isolation	Drag and drop enabled
Advanced	Normal/Normal

Power options
- [ ] Power on after opening this virtual machine
- [ ] Enter full screen mode after powering on
- [x] Close after powering off or suspending

Power Controls
- Power Off
- Suspend Virtual Machine
- Power on / Resume Virtual Machine
- Reset

Run VMware Tools scripts
- [ ] After powering on
- [ ] After resuming
- [ ] Before suspending
- [ ] Before powering off

Figure 7-7. *Selecting for the VM to automatically close when powered down*

In the next sections, we'll begin by looking at backing up Workstation VMs running on Windows operating systems and finish with a look at backing up Workstation running on Linux.

Running VMware Workstation on Windows Flat-File Backups

One of the easiest methods to script backup jobs on Windows operating systems is using tried-and-true batch scripting. We've selected this method for the examples in this book because of its ease of use and overall adoption in the field. Since many of the administrators we encounter run both Windows and Linux operating systems inside VMs, we'll provide examples of scripting flat-file backups for both operating systems. In each example, backups are executed using the `ntbackup.exe` command-line utility. Both of the batch files covered use a core set of variables to establish a source and destination for the backup. For your own use, you need to change the variables to suit your specific environment. Here's a quick overview of each of the variables that will be used:

- **VMName**: Host name of VM to be backed up
- **VMFolder**: Location of VM's files on the host system
- **VMXPath**: Full path to VM's configuration file
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **BackupFolder**: Backup data destination folder (create this folder prior to running backup)
- **BackupFile**: Filename of backup job (stamped with current date)
- **JobName**: Friendly name for job (stored in backup file)
- **Description**: Description of job (stored in backup file)

With the variables now under your belt, let's look at backing up a Windows VM running on a Windows host.

Backing Up a Windows VM

Listing 7-3 shows a batch file that can be used to automatically shut down a Windows virtual machine, back up its files, and then start it up.

Listing 7-3. *Backing Up a Windows VM on a Windows Host*

```
:: Filename: WinVMWSWinBackup.bat
::
:: Purpose: Automates the backup of a virtual machine by shutting down the VM,
:: using ntbackup.exe to back up the VM's files, and then automatically starting
:: back up the VM. This script backs up a VM named W2KFS, which is located in
:: the E:\VMs\W2KAS folder. You would need to alter the VM name and paths for
:: this script to run in your environment.
::==================================================================================
@echo off
:: Set Variables
set VMName=W2KFS
set VMFolder= E:\VMs\W2KAS
set VMXPath= E:\VMs\W2KAS\W2KFS.vmx
set Today=%Date:~4,2%-%Date:~7,2%-%Date:~10,4%
set BackupFolder=G:\Backup\%VMName%
set BackupFile=%BackupFolder%\%Today%.bkf
set JobName=%Today% Full Backup
set Description=Starting %Today%

:: Stop VM Using shutdown.exe (requires UNC path to VM)
:: On W2K and earlier hosts, shutdown.exe requires installation of the resource kit
shutdown.exe /m \\%VMName% /s /t 0

:: Wait for VM to shut down (uses ping -n <seconds> to the loopback address
:: to create a delay)180 seconds = 3 minute delay.
ping -n 180 127.0.0.1 > nul

:: Back up VM
IF NOT EXIST %BackupFolder% md %BackupFolder%
ntbackup backup "%VMFolder%" /J "%JobName%" /D "%Description%" /F "%BackupFile%"

:: Start VM (path to VMware.exe file must be specified)
"D:\Program Files\VMware\VMware Workstation\VMware.exe" -x "%VMXPath%"
```

In this example, a VM is backed up to a local drive on the system. If you wanted to back up the VM to a UNC path, you'd need to change the BackupFolder variable. For example, to direct backup data to the \\Max\Backups share, you edit the BackupFolder variable line so that it reads set BackupFolder=\\Max\Backups\%VMName%.

Notice that the ping command is used in the script for the purpose of creating a delay. The –n parameter for the ping command specifies the number of echo requests to send. Each echo request will take approximately one second, so using this command gives you a simple means to insert a delay into the script. An alternative to using ping is to use the Sleep.exe resource kit tool.

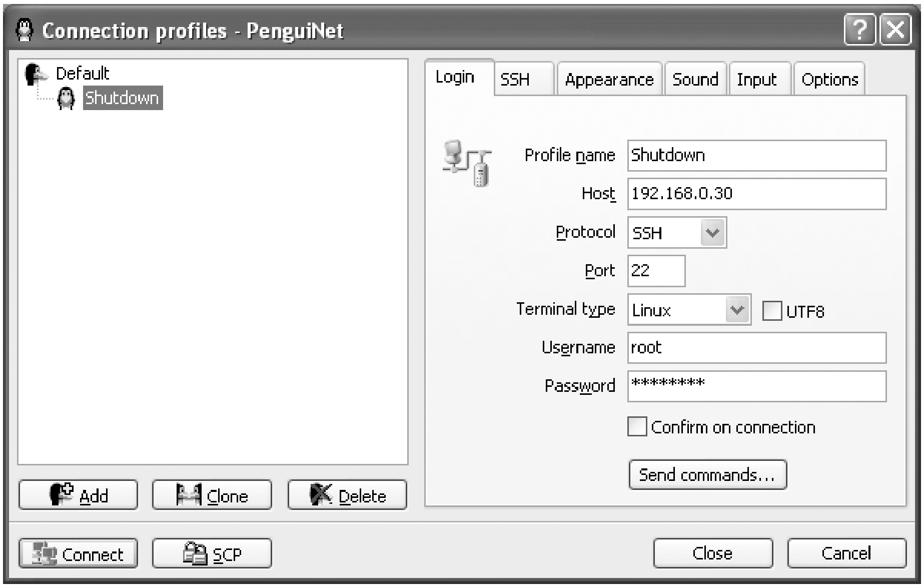
After the VM shuts down and the batch script waits the allotted time, the ntbackup.exe command executes to back up the VM. Control isn't returned to the batch file until ntbackup.exe completes the backup job, so there's no need to insert a delay into the batch file following the ntbackup.exe command. The last line of the batch file uses the VMware.exe command to start the VM. You'll see almost identical logic in the next script, which will perform a flat-file backup of a Linux VM.

Backing Up a Linux VM

The toughest aspect of scripting a Linux VM flat-file backup on a Windows host is having to gracefully shut down the VM prior to backup. To do this, you must send a remote shutdown command to the Linux VM. To do this securely, we decided to use one of the many available secure shell (ssh) tools available for Windows operating systems.

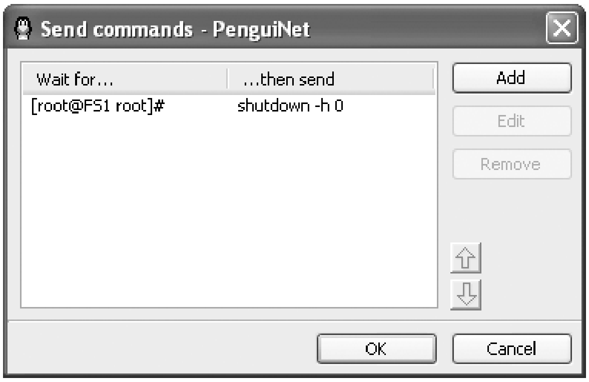
We selected the PenguiNet shareware software to perform the remote ssh connection and shutdown command. You can download this tool from http://www.download.com. With PenguiNet, you can create a profile that includes all the ssh connection settings and commands to run during the ssh connection. This allows you to fully script a remote shutdown in a batch file without having to include any confidential information in the batch file. For example, the username, password, system name to connect to, and commands to run are all embedded in the profile.

In PenguiNet, you can create connection profiles using the connection profiles toolbar in the PenguiNet GUI. Figure 7-8 shows the connection profile used in our `WinVMWSLinuxBackup.bat` file. When you click the Send Commands button in the Connection Profiles dialog box, you can enter commands to automatically run once the session is established. Figure 7-9 shows these settings.



Profile name: Shutdown
Host: 192.168.0.30
Protocol: SSH
Port: 22
Terminal type: Linux
UTF8: [ ]
Username: root
Password: *********
Confirm on connection: [ ]

Figure 7-8. *PenguiNet shutdown profile settings*



Wait for...	...then send
[root@FS1 root]#	shutdown -h 0

Figure 7-9. *PenguiNet shutdown profile Send Commands settings*

Once you've created the profile, you must turn the profile into a shortcut so the batch file can use it. You can create a shortcut to the profile by simply dragging the profile from the PenguiNet GUI and dropping it into any folder in Windows Explorer. Once this is done, you can use the command referenced by the profile shortcut in the batch file. Listing 7-4 shows a batch file that remotely shuts down a Linux VM, backs it up, and then restarts the VM.

Listing 7-4. *Backing Up a Linux VM on a Windows Host*

```
:: Filename: WinVMWSLinuxBackup.bat
::
:: Purpose: Automates the backup of a Linux virtual machine by shutting down
:: the VM, using ntbackup.exe to back up the VM's files, and then automatically
:: starting back up the VM. This script backs up a VM named LinuxFS1, which is
:: located in the E:\VMs\LinuxFS1 folder. You would need to alter the VM name
:: and paths for this script to run in your environment.
::==================================================================================
@echo off
:: Set Variables
set VMName=LinuxFS1
set VMFolder=E:\VMs\LinuxFS1
set VMXPath=E:\VMs\LinuxFS1\LinuxFS1.vmx
set Today=%Date:~4,2%-%Date:~7,2%-%Date:~10,4%
set BackupFolder=G:\Backup\%VMName%
set BackupFile=%BackupFolder%\%Today%.bkf
set JobName=%Today% Full Backup
set Description=Starting %Today%

:: Stop VM Using PenguiNet.exe to issue shutdown command
"D:\Program Files\PenguiNet\PenguiNet.exe" profnum://4278190084

::Wait for VM to shut down (uses ping -n <seconds> to the loopback address
:: to create a delay) 180 seconds = 3 minute delay.
ping -n 180 127.0.0.1 > nul

:: Back up VM
IF NOT EXIST %BackupFolder% md %BackupFolder%
ntbackup backup "%VMFolder%" /J "%JobName%" /D "%Description%" /F "%BackupFile%"

:: Start VM (path to VMware.exe file must be specified)
"D:\Program Files\VMware\VMware Workstation\VMware.exe" -x "%VMXPath%"
```

As you can see, this batch script is almost identical to the Windows VM backup script. The primary difference with this script is the methodology used to remotely shut down the Linux VM. Since the Windows `shutdown.exe` command couldn't be directly run against the Linux VM, we launched PenguiNet to run a remote shutdown command via an ssh connection. Next, we'll cover backing up VMs running on a Linux host system.

Running VMware Workstation on Linux Flat-File Backups

In this section, we show how to use shell scripts to automate the backups of VMs running on a Linux host running VMware Workstation. As with the previous section, this script will shut down the VM, back it up, and then restart the VM.

Here's a quick overview of each of the variables that will be used:

- **VMName**: Host name of VM to be backed up
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **DirToBackup**: Location of virtual machine's files on the host system
- **VMXPath**: Full path to VM's configuration file
- **NfsPath**: Backup data destination folder (create this folder prior to running backup)
- **BackupDestination**: Path and filename of backup job (stamped with current date)
- **BackupLog**: Name and location of `tar` backup log file

Listing 7-5 shows the shell script that will shut down a Windows VM, back it up to an NFS mount point using `tar`, and then restart the VM. The `shutdown` command is issued via a remote ssh connection from the host to the VM. Since Windows doesn't natively support ssh, you'll need to install it on the Windows VM. Since ssh is natively supported by Linux, this script can also be used to back up a Linux VM running on a Linux host.

■**Tip** You can download the free OpenSSH ssh server application for Windows at `http://sshwindows.sourceforge.net`. OpenSSH is also packaged for Windows Services for Unix and can be downloaded at `http://www.interopsystems.com`.

Listing 7-5. *Backing Up a Windows VM on a Linux Host*

```
#!/bin/sh
#
# Filename: LinuxVmWSBackup.sh
# Purpose: Uses ssh and shutdown.exe to remotely shut down a Windows VM. Then uses
# tar to back up the VM. When the backup completes, the VM is restarted.
#=======================================================================
# ==== Assign variables ====
VMName="W2KFS1"     # Name of VM to back up
Today=$(date +'%m-%d-%y')     #Loads current date as mm-dd-yyyy into variable
DirToBackup="/data/vms/W2KFS1"     #Specifies to back up W2KFS1 folder
VMXPath="/data/vms/W2KFS1/W2KFS1.vmx"
NfsPath="mediasrv1:/data/Backup"     #Specifies NfsPath to back up to
#Name of backup file. Includes date stamp.
```

```
BackupDestination="/mnt/backup/"$VmName"-"$Today".tar"
#Name and location of backup log file
BackupLog="/mnt/backup/"$VmName"-Backup_Log_"$Today".log"
#==== Shutdown VM ====
ssh $VMName shutdown -h 0

#==== Mount backup folder ====
mount -t nfs $NfsPath /mnt/backup

# ==== Run the backup=====
tar -cpvf $BackupDestination $DirToBackup >&$BackupLog

#==== Unmount backup folder ====
umount /mnt/backup

#==== Restart VM ====
/usr/bin/vmware.exe -x $VMXPath
```

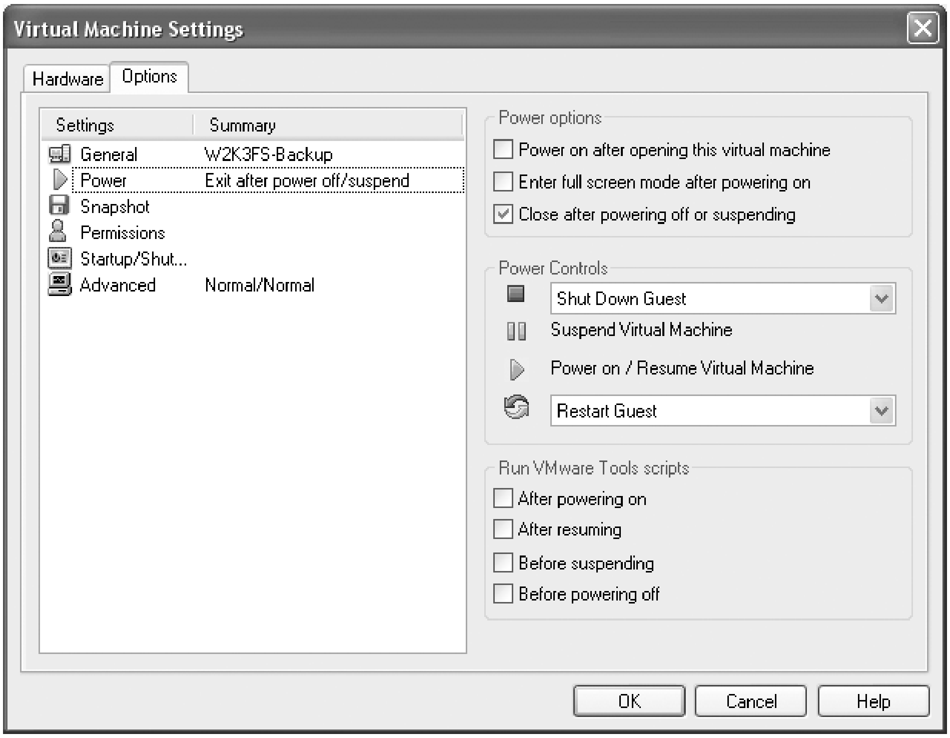
You may have noticed that this script is similar to the one used in the "Backing Up with `tar`" section of this chapter. The primary differences with the script are the lines added to shut down and start the VM. To stop the VM, an ssh connection is established to issue the `shutdown` command. To restart the VM, the `vmware.exe` command-line tool is used. The `-x` parameter starts the VM specified in the `$VMXPath` variable.

Now that we've covered VMware workstation backups, we'll cover VMware GSX Server.

Running VMware GSX Server Flat-File Backups

VMware GSX Server products, being built for production, offer much more to aid in backing up your VMs. To begin with, GSX Server allows you to configure VMs to gracefully shut down when they're stopped. All that's required is to install the latest version of VMware Tools in the VM and to configure the VM to shut down the guest when powering down. To configure the VM power-off settings, you need to perform these steps prior to running any scripted backups:

1. In the VMware Virtual Machine Console UI, click the VM you plan to back up. Then click the VM menu, and select Settings.
2. In the Virtual Machine Settings dialog box, click the Options tab.
3. Under the Options tab, click the Shutdown drop-down menu, and select the Shut Down Guest power-off option. Then check the Close After Powering Off or Suspending box. Figure 7-10 shows both of these settings.
4. Click OK to close the Virtual Machine Settings dialog box.



Virtual Machine Settings

Hardware | Options

Settings	Summary
General	W2K3FS-Backup
Power	Exit after power off/suspend
Snapshot	
Permissions	
Startup/Shut...	
Advanced	Normal/Normal

Power options
- ☐ Power on after opening this virtual machine
- ☐ Enter full screen mode after powering on
- ☑ Close after powering off or suspending

Power Controls
- Shut Down Guest
- Suspend Virtual Machine
- Power on / Resume Virtual Machine
- Restart Guest

Run VMware Tools scripts
- ☐ After powering on
- ☐ After resuming
- ☐ Before suspending
- ☐ Before powering off

Figure 7-10. *Selecting GSX VM power-down options*

Notice that the previous steps also had you configure the VM to close when powered down. This is required to unlock all VM files so that they can be backed up. Once the VM options are set, all that's left is to put together a script to automate its backup. In the next two sections, you'll see how to script backups on both Windows and Linux hosts.

Running VMware GSX on Windows Flat-File Backups

Since GSX Server can gracefully shut down either a Windows VM or a Linux VM with a single command, there's no difference between scripting a backup for either. Prior to looking at the backup script, let's first examine the purpose of each variable that's used:

- **VMName**: Host name of VM to be backed up
- **VMFolder**: Location of virtual machine's files on the host system
- **VMXPath**: Full path to VM's configuration file
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **BackupFolder**: Backup data destination folder (create this folder prior to running backup)
- **BackupFile**: Filename of backup job (stamped with current date)
- **JobName**: Friendly name for job (stored in backup file)
- **Description**: Description of job (stored in backup file)

The script uses the VMware-cmd.exe tool, which is located by default in the Program Files\VMware\VMware VmPerl Scripting API folder on the host system. The syntax of VMware-cmd used in this section's script is as follows:

```
VMware-cmd <path to vmx file> <start | stop>
```

As you can see, the syntax is relatively simple. Following the command, you need to specify the path to the VM's configuration file and then use the start or stop parameter. Now that you've examined the unique aspects of this script, refer to the script itself (see Listing 7-6).

Listing 7-6. *Backing Up a VM on a Windows GSX Server Host*

```
:: Filename: WinVMGSXBackup.bat
::
:: Purpose: Automates the backup of a virtual machine by shutting down the VM,
:: using ntbackup.exe to back up the VM's files, and then automatically starting
:: back up the VM. This script backs up a VM named W2KFS, which is located
:: in the E:\VMs\W2KAS folder. You would need to alter the VM name and paths
:: for this script to run in your environment.
::==================================================================================
@echo off
:: Set Variables
set VMName=W2KFS
set VMFolder= E:\VMs\W2KAS
set VMXPath= E:\VMs\W2KAS\W2KFS.vmx
set Today=%Date:~4,2%-%Date:~7,2%-%Date:~10,4%
set BackupFolder=G:\Backup\%VMName%
set BackupFile=%BackupFolder%\%Today%.bkf
set JobName=%Today% Full Backup
set Description=Starting %Today%

:: Stop VM Using VMware-cmd
"D:\Program Files\VMware\VMware VmPerl Scripting API\VMware-cmd" %VMXPath% stop

::Wait for VM to shut down (uses ping -n <seconds> to the loopback address
:: to create a delay) 180 seconds = 3 minute delay.
ping -n 180 127.0.0.1 > nul

:: Back up VM
IF NOT EXIST %BackupFolder% md %BackupFolder%
ntbackup backup "%VMFolder%" /J "%JobName%" /D "%Description%" /F "%BackupFile%"

:: Start VM Using VMware-cmd
"D:\Program Files\VMware\VMware VmPerl Scripting API\VMware-cmd" %VMXPath% start
```

Compared to VMware Workstation, scripted flat-file backups are much simpler with GSX Server. The same holds true for GSX Server running on a Linux host, which you'll look at next.

Running VMware GSX on Linux Flat-File Backups

Scripting GSX Server flat-file backups on Linux hosts is almost identical in theory to Windows hosts. In fact, both use the `vmware-cmd.exe` tool to facilitate the backup. The only major difference is in the tool used to back up the data itself (`tar` with Linux and `ntbackup` with Windows). Before getting to the script itself, we'll briefly define its variables:

- **VMName**: Host name of VM to be backed up
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **DirToBackup**: Location of virtual machine's files on the host system
- **VMXPath**: Full path to VM's configuration file
- **NfsPath**: Backup data destination folder (create this folder prior to running backup)
- **BackupDestination**: Path and filename of backup job (stamped with current date)
- **BackupLog**: Name and location of `tar` backup log file

This section uses the `vmware-cmd` tool to stop the VM prior to backing up and restarting the VM once the backup completes. Its syntax is identical to the Windows version of the tool. So, for example, to stop a VM named Mail1, you'd run `vmware-cmd /data/VMs/W2KMail1/mail1.vmx stop`. Aside from this command, you should see that the format and syntax of this script, shown in Listing 7-7, resembles that of the `tar` backup script presented earlier in the chapter in Listing 7-5.

Listing 7-7. *Backing Up a VM on a Linux GSX Server Host*

```
#!/bin/sh
#
# Filename: LinuxVmGSXBackup.sh
# Purpose: Uses vmware-cmd to remotely shut down a Windows VM. Then uses tar
# to back up the VM. When the backup completes, the VM is restarted.
#==================================================================
# ==== Assign variables ====
VMName="W2KFS1"     # Name of VM to back up
Today=$(date +'%m-%d-%y')     #Loads current date as mm-dd-yyyy into variable
DirToBackup="/data/vms/W2KFS1"     #Specifies to back up W2KFS1 folder
VMXPath="/data/vms/W2KFS1/W2KFS1.vmx"
NfsPath="mediasrv1:/data/Backup"     #Specifies NfsPath to back up to
#Name of backup file. Includes date stamp.
BackupDestination="/mnt/backup/"$VmName"-"$Today".tar"
#Name and location of backup log file
BackupLog="/mnt/backup/"$VmName"-Backup_Log_”$Today".log”
#==== Shutdown VM ====
vmware-cmd $VMXPath stop

#==== Mount backup folder ====
mount –t nfs $NfsPath /mnt/backup

# ==== Run the backup=====
tar –cpvf $BackupDestination $DirToBackup >&$BackupLog
```

```
#==== Unmount backup folder ====
umount /mnt/backup

#==== Startup VM ====
vmware-cmd $VMXPath start
```

Now that you've tackled flat-file backups of the most popular VMware products, next you'll explore scripted flat-file backups for the Microsoft VM applications.

Running Virtual PC 2004 Flat-File Backups

Although Virtual PC supports the command-line startup of VMs, it doesn't support command-line VM shutdowns. Because of this, performing flat-file backups of Virtual PC VMs requires a little creativity. Since Virtual PC officially supports only Microsoft OS VMs, in this section we'll cover a batch script that will shut down a Windows VM, perform a backup, and then restart the VM.

In the batch script we put together to back up Virtual PC VMs, we use `shutdown.exe` to remotely shut down the VM and then use `taskkill.exe` to close the Virtual PC application. This is needed because Virtual PC will hold a lock on its VM's files while the application is still running (regardless of the state of its VMs). After shutting down the VM and closing Virtual PC, the script then uses the `ntbackup.exe` command-line tool to back up the VM. When the backup completes, the `Virtual PC.exe` command-line tool restarts the VM. A few variables are used in the script:

- **VMName**: Host name of VM to be backed up
- **VMFolder**: Location of VM configuration and virtual disk files
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **BackupFolder**: Target location for backup
- **BackupFile**: Name of backup file (includes path loaded into `BackupFolder` variable)
- **JobName**: Friendly name of backup job, referenced by `NTBackup.exe`
- **Description**: Describes backup, referenced by `NTBackup.exe`

Now let's look at the actual script (see Listing 7-8).

Listing 7-8. *Backing Up a VM on a Virtual PC Host*

```
:: Filename: VPCWinBackup.bat
::
:: Purpose: Automates the backup of a Virtual PC 2004 virtual machine by shutting
:: down the VM, closing the Virtual PC application, using ntbackup.exe to back up
:: the VM's files, and then automatically starting back up the VM. This script
:: backs up a VM named W2KWeb, which is located in the E:\VMs\W2KWeb folder.
:: You would need to alter the VM name and paths for this script to run in your
:: environment.
::==================================================================================
@echo off
:: Set Variables
set VMName=W2KWeb
set VMFolder= E:\VMs\W2KWeb
set Today=%Date:~4,2%-%Date:~7,2%-%Date:~10,4%
set BackupFolder=\\Dempsey\Backups\%VMName%
set BackupFile=%BackupFolder%\%Today%.bkf
set JobName=%Today% Full Backup
set Description=Starting %Today%

:: Stop VM Using shutdown.exe (requires UNC path to VM)
:: On W2K and earlier hosts Shutdown.exe requires installation of resource kit
shutdown.exe /m \\%VMName% /s /t 0

::Wait for VM to shut down (uses ping -n <seconds> to the loopback address
:: to create a delay) 180 seconds = 3 minute delay.
ping -n 180 127.0.0.1 > nul

:: Close Virtual PC Application and any child processes
taskkill /F /IM "Virtual PC.exe" /T

:: Back up VM
IF NOT EXIST %BackupFolder% md %BackupFolder%
ntbackup backup "%VMFolder%" /J "%JobName%" /D "%Description%" /F "%BackupFile%"

:: Start VM
"Virtual PC.exe" -singlepc -pc %VMName% -launch
```

Remember that tools such as `shutdown` and `taskkill` require administrative rights to execute, so this batch script should run under an administrator account. You can easily accomplish this by configuring the script to run periodically using the Scheduled Tasks system tool and configuring the option to run the script under a specific user account.

Running Virtual Server 2005 Flat-File Backups

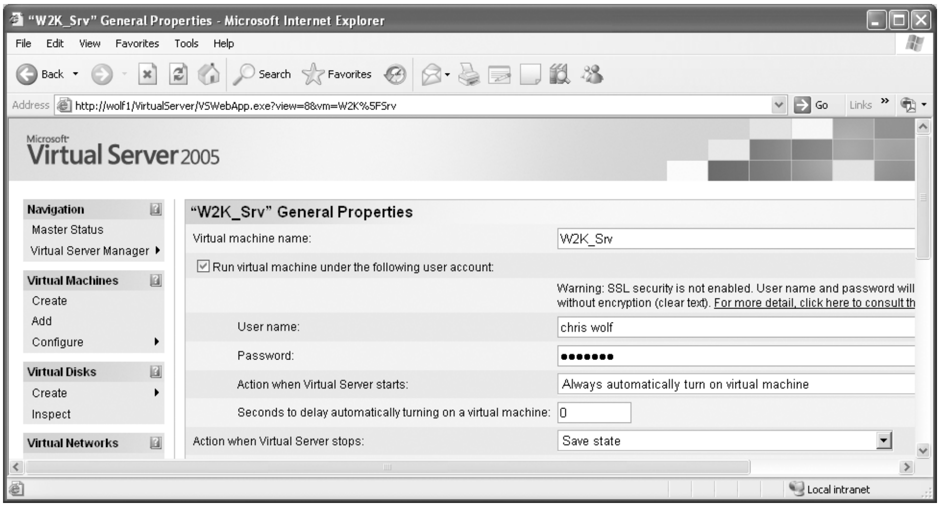
Flat-file backups with Virtual Server 2005 are slightly more challenging compared to the VMware GSX product line. This is because Virtual Server 2005 doesn't gracefully shut down Linux VMs when powered down. However, it supports a graceful shutdown of Windows VMs. Even without graceful shutdown of Linux VMs, Virtual Server supports automatically saving the state of any running VM at the time the Virtual Server service stops. Furthermore, this product supports automatically resuming a saved VM when the service restarts.

If the bulb is starting to flicker a little bit, it's for a good reason. The built-in automatic save and restart capabilities of Virtual Server make it easy to use a batch script to back up all VMs on a single host system. If your preference is to back up each VM independently, then the methodology for backing up the VM will change slightly. Virtual Server 2005 doesn't have native support for command shell administration; instead, it supports using COM for scripted administration. What this essentially means is that backing up a single VM will require a script written in a scripting language such as Visual Basic Script (VBScript) or Perl. To cover both the "single VM" and "all VMs" backup scenarios, the next two sections will show you scripts that will do each task. To back up all VMs, we'll show you how to use a simple batch script, and to back up a single VM, we'll show you some VBScript code that will get the job done.

Backing Up All Virtual Server VMs

The key to backing up all VMs on a Virtual Server host using a single script is to prepare the VMs to automatically save their state if the Virtual Server service stops. Doing so will allow you to automatically suspend all running VMs when Virtual Server stops, which will place each VM's related files in a state so that it can be successfully copied. To prepare the VMs for backup, you need to set their default behavior for when Virtual Server starts or stops. To do this, follow these steps:

1. On the Virtual Server host system, click Start ➤ All Programs ➤ Microsoft Virtual Server ➤ Virtual Server Administration Website. If any VMs are running, shut them down before proceeding.
2. When the Web site opens, mouse over the Configure item under the Virtual Machines menu, and select the first virtual machine listed.
3. You should now see the status of the VM you selected. Scroll down the Web browser window until you see the Configuration section of the page. In this section, click the General Properties link.
4. Now check the Run the Virtual Machine Under the Following User Account box. Enter a username and password of an account with administrative rights under which to start the virtual machine.
5. Now in the Action When Virtual Server Starts menu, select Always Automatically Turn on Virtual Machine. In the Action When Virtual Server Stops menu, select Save State. Figure 7-11 shows the settings described in steps 4–5.
6. Scroll to the bottom of the General Properties page, and click the OK button.
7. Repeat steps 2–6 for each additional VM on the server.


"W2K_Srv" General Properties - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://wolf1/VirtualServer/VSWebApp.exe?view=8&vm=W2K%5FSrv
Go Links
Microsoft Virtual Server 2005
Navigation
Master Status
Virtual Server Manager
Virtual Machines
Create
Add
Configure
Virtual Disks
Create
Inspect
Virtual Networks
Local intranet

"W2K_Srv" General Properties

Virtual machine name: W2K_Srv

☑ Run virtual machine under the following user account:

Warning: SSL security is not enabled. User name and password will without encryption (clear text). For more detail, click here to consult th

User name: chris wolf

Password: •••••••

Action when Virtual Server starts: Always automatically turn on virtual machine

Seconds to delay automatically turning on a virtual machine: 0

Action when Virtual Server stops: Save state

Figure 7-11. *Setting Virtual Server VM start and stop actions*

Once you've set the start and stop actions for each VM, you're ready to run a script to back them up. The script used in this section to back up all VMs on a Virtual Server host uses the following variables:

- **VMRootFolder**: Location of parent drive or folder containing all VMs
- **Today**: Current date formatted as mm-dd-yyyy (used to date stamp backup files)
- **BackupFolder**: Backup data destination folder (create this folder prior to running backup), which can be a local or UNC path
- **BackupFile**: Filename of backup job (stamped with current date)
- **JobName**: Friendly name for job (stored in backup file)
- **Description**: Description of job (stored in backup file)

With an understanding of the variables used, refer to the script (see Listing 7-9).

Listing 7-9. *Backing Up All VMs on a Virtual Server Host*

```
:: Filename: VS2005AllVMBackup.bat
::
:: Purpose: Automates the backup of all VMs on a VS 2005 host. VMs
:: must be configured to Save State when the VS service stops and to automatically
:: turn on when the VS service starts. After the service stops, ntbackup.exe is
:: used to back up the root folder that contains all of the hosts VMs.
:: When the backup completes, the Virtual Server service is restarted, which in
:: turn resumes all saved VMs.
::=================================================================================
@echo off
:: Set Variables
set VMRootFolder= E:\VMs
set Today=%Date:~4,2%-%Date:~7,2%-%Date:~10,4%
set BackupFolder=\\Zeus\Backups\VMs
set BackupFile=%BackupFolder%\%Today%.bkf
set JobName=%Today% Full Backup
set Description=Starting %Today%

:: Stop Virtual Server Service - Saves state of all VMs
net stop "Virtual Server"

:: Back up VMs
ntbackup backup "%VMRootFolder%" /J "%JobName%" /D "%Description%" /F "%BackupFile%"

:: Start Virtual Server Service - Resumes all saved VMs
net start "Virtual Server"
```

As you can see, this is one of the shortest scripts in the entire chapter, yet it's one of the most powerful. By taking advantage of the built-in save state and VM startup features of Virtual Server, you're able to back up all VMs on the server with just a few commands. If you're just interested in backing up one VM, you'll see how to do that next.

Backing Up a Single Virtual Server VM

To back up a single VM running on Virtual Server 2005, you must first save the VM's state. This will quiesce the VM while the backup runs. Once the backup completes, the VM can be restarted. Suspending the VM prior to backup will place the VM's configuration and virtual disk files in a fixed state so that they then can be reliably copied.

Unfortunately, Virtual Server 2005 doesn't allow you to directly administer VMs from the command line, so for this specific task you'll use VBScript. This script uses the following variables, which you'll need to modify for it to successfully back up one of your VMs:

- **strVMName**: Identifies the name assigned to virtual machine to back up (visible in Virtual Server's Web UI)
- **strVMFolder**: Provides location of VM's files
- **strToday**: Defines current date formatted as mm-dd-yyyy
- **strBackupFolder**: Identifies location where backup data is stored
- **strBackupFile**: Stores name and full path to backup (`.bkf`) file
- **strJobName**: Defines friendly name for job (stored in backup file)
- **strDescription**: Holds description of job (stored in backup file)
- **strNTBackupCommand**: Stores complete backup command syntax that's passed to the `ObjShell` command shell object
- **objVirtualServer**: Provides connection to Virtual Server application
- **objVirtualMachine**: Provides connection to specific virtual machine
- **objShell**: Runs `ntbackup.exe` shell command from within script
- **errBackup**: Holds return code for `ntbackup.exe` command

With the variables out of the way, let's look at the script (see Listing 7-10).

Note The methods used in this script are supported only on Windows Server 2003 or higher host systems.

Listing 7-10. *Backing Up a Single VM on a Virtual Server Host*

```
' Filename: VS2005VMBackup.vbs
'
' Purpose: Automates the backup of a single VMs on a host system
'          by first suspending the VM, then running the backup, and finally
'          resuming the VM.
'==========================================================================
Option Explicit
'Declare variables
Dim strVMName, strVMFolder, strToday, strBackupFolder, strBackupFile
Dim strJobName, strDescription, strNTBackupCommand
Dim objVirtualServer, objVirtualMachine, objShell, errBackup

'Load current date (formatted as mm-dd-yyyy) into variable strToday
strToday =  Month(Date) & "-" & Day(Date) & "-" & Year(Date)

'Define Script Variables
strVMName = "W2K_Srv"
strVMFolder = """E:\VMs\W2K_Srv"""
strBackupFolder = "G:\Backup\VMs"
strBackupFile = Chr(34) & strBackupFolder & "\" & strVMName & "_" & strToday _
                & ".bkf" & Chr(34) 'Chr(34) is used to include a quote in the string
strJobName = Chr(34) & strToday & " Full Backup" & Chr(34)
strDescription = Chr(34) & "Starting " & strToday &Chr(34)
strNTBackupCommand = "ntbackup backup " & strVMFolder & " /J " & strJobName & _
                   " /D " & strDescription & " /F " & strBackupFile

'Instantiate Virtual Server Object
Set objVirtualServer = CreateObject("VirtualServer.Application")

'Instantiate Virtual Machine Object
Set objVirtualMachine = objVirtualServer.FindVirtualMachine(strVMName)

'Save VM State
objVirtualMachine.Save()

'Instantiate command shell object
Set objShell = WScript.CreateObject("Wscript.shell")

'Use ntbackup.exe to back up saved VM
errBackup = objShell.Run(strNTBackupCommand,1,True)

'Restart VM
objVirtualMachine.Startup()
```

Since the script saves the state of a running VM before backing it up, and since the save state feature runs transparently to the guest OS, this script can back up any Virtual Server VM, regardless of its installed OS.

At this point, we've covered all the primary flat-file backup scenarios for both Windows and Linux VMs. Next, we'll cover a method for performing backups in real-time: online snapshots.

Taking Online Snapshots

Online snapshots are a feature supported in the VMware line of virtualization applications. With online snapshots, you can create a point-in-time copy of a running virtual machine. With an online snapshot saved, you can revert to the snapshot later. This gives you the ability to create an image of a VM, make any changes you want, and with the click of a button roll the VM back to the time of the last snapshot.

A disadvantage of this feature is that it's GUI-driven. If you want to do a snapshot, you need to do it from the VMware management UI. Because of this, you shouldn't look at snapshots as a way to replace the other backup methodologies already mentioned in this chapter but rather as a feature that can aid in testing and training. In our opinion, whenever backups are left up to humans to manually perform (instead of being automated), it's just a matter of time before a backup is skipped.

Although not necessarily a good choice strictly for backups, here are some reasons to use the snapshot feature:

- **Testing**: Create a baseline VM, and then create a snapshot. After testing is complete, revert the VM to the previously saved snapshot.
- **Demonstrations**: Take a snapshot before a software demonstration. Once the demonstration completes, you can power down the VM and revert to the saved snapshot.
- **Training**: You can walk students through a task and afterward revert to the beginning so the students can practice.

Because of snapshots' limited scope, VMware supports only a single saved snapshot per VM, so you'll need to decide carefully when you plan to perform a snapshot.

Taking Snapshots

Taking a snapshot of a VM is a relatively simple process. Here are the steps to follow:

1. In the VMware management UI, boot up the VM (if it isn't currently started).
2. With the VM running, click the Snapshot menu, and select Save Snapshot.
3. Wait for the snapshot to complete.

The snapshot information will be saved in the same location as the VM's virtual disk files. The snapshot files will have `_REDO` as part of their extension, making them easy to identify.

Recovering a Snapshot

Snapshot recovery is just about as easy as taking the initial snapshot. To recover a snapshot, perform these steps while the VM is running:

1. Click the Snapshot menu, and select Revert to Snapshot.
2. If prompted to confirm that you want to revert to the saved snapshot, click Yes.

In a few moments, the system will be restored to the time of the last snapshot.

Although there's no flexibility in the number of snapshots you can take (one), you can set a few options in regard to how snapshots behave. Let's quickly look at them.

Setting Snapshot Options

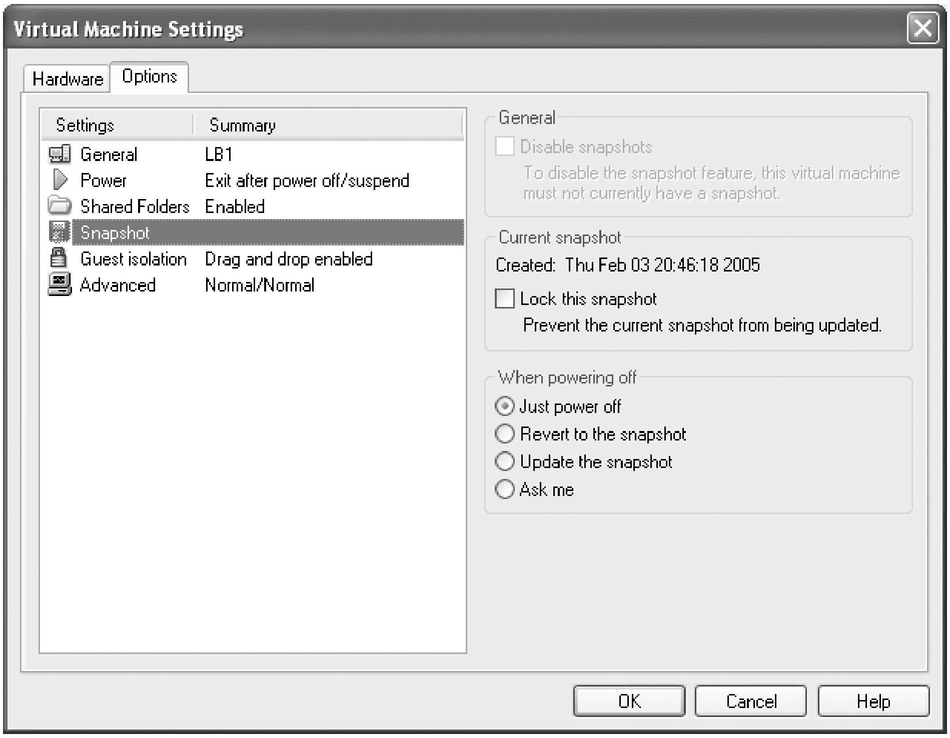
VMware allows you to set the following options for snapshot behavior:

- Disable all snapshots (not available if a current snapshot is saved).
- Lock the current snapshot, preventing any future snapshots accidentally taken from overwriting it.
- Select how powering down the VM affects the snapshot. VMware will allow you to select from the following power-down behavior options:
  - **Just Power Off**: Do nothing to last snapshot (default setting).
  - **Revert to the Snapshot**: Roll back VM's hard disks to their state as of the last snapshot.
  - **Update the Snapshot**: Apply the current VM saved data to the snapshot (rolls snapshot forward to current point in time).
  - **Ask Me**: Causes a dialog box to be displayed when the VM is powered down, allowing user to decide how to handle the snapshot.

To configure these options, follow these steps:

1. With the VM open in the VMware UI, click the VM menu, and select Settings.
2. In the Virtual Machines Settings dialog box, click the Options tab.
3. Now click the Snapshot setting (see Figure 7-12). Make any desired changes to the snapshot, and then click OK.

■ **Tip** If you want to exclude a particular virtual disk from a snapshot, configure the disk as an independent disk. You can access this setting through the virtual disk's advanced properties in the VMware UI.


Virtual Machine Settings

Hardware | Options

Settings	Summary
General	LB1
Power	Exit after power off/suspend
Shared Folders	Enabled
Snapshot	
Guest isolation	Drag and drop enabled
Advanced	Normal/Normal

General

☐ Disable snapshots

To disable the snapshot feature, this virtual machine must not currently have a snapshot.

Current snapshot

Created: Thu Feb 03 20:46:18 2005

☐ Lock this snapshot

Prevent the current snapshot from being updated.

When powering off

[✓ Just power off]
Revert to the snapshot
Update the snapshot
Ask me

Figure 7-12. *Configuring VM snapshot options*

Again, while snapshots are useful for training, testing, and demonstration purposes, they shouldn't be looked at as a substitute for VM backups. Now that you've examined all methods for securing VM data, next we'll cover the general procedures for recovering VMs.

Performing a Full System Recovery

The entire point of backing up data is to protect your organization's data from loss stemming from corruption, system or disk failure, or a natural disaster. Although understanding how to reliably back up virtual machine data is critical, you shouldn't overlook the process for restoring the data.

To prepare for recovery, a best practice is to ask yourself, can I rebuild my production systems from scratch and have them configured exactly the same as they are now? Many administrators answer "no" to this question, so if that's your answer, you're not alone. To fully rebuild your systems following a disaster, you need to know exactly how the systems were configured. This configuration information includes the following:

- Host name
- TCP/IP configuration
- Disk drive configuration, disk configuration, and partitions (including SCSI IDs)
- Installed applications
- Installed hardware

With this information in hand, it's much easier to successfully recover lost data or systems. The method you choose to back up your VMs will ultimately dictate the techniques you'll use for recovery. In the next two sections, we'll present the methodologies for recovering VMs backed up locally while online, as well as recovering VMs that were backed up directly from the host via flat-file backups.

Restoring Online VM Backups

The process for restoring data from an online VM backup is nearly identical to the data recovery process for physical systems. If you needed to restore a few files or folders, you'd use either your enterprise backup product or a utility such as `dump` or Windows Backup to recover the data. The processes involved in recovering files using `dump` and Windows Backup were outlined earlier in this chapter.

When you're faced with completely restoring a VM from an online backup, the recovery process is slightly more difficult. One of the major advantages of virtualization applications is that they emulate nearly all VM system hardware (see Chapter 1). This means that if a VM's host system crashes, you can recover the VM's backup data to a new VM running on a different host system. With emulated hardware, it's likely that the VM's OS won't even notice the change.

■ **Caution** VMware has long noted that its hardware emulation doesn't support moving a Linux VM from a host with an Intel CPU to a host with an AMD CPU. Doing so will cause an irrecoverable kernel panic at startup. This problem doesn't occur when moving VMs with other installed operating systems between hosts with different CPUs.

Let's further discuss the scenario of a VM's host crashing. This is equivalent to one of your servers or workstations crashing. When this happens, the general recovery process involves the following steps:

1. Reconfigure the system's hardware as close to the failed system's configuration as possible. This will include installed devices and hard disks. The number of disks as well as the size of each disk should be equal to or larger than the number of disks on the original system.

2. Power on the rebuilt system (VM), and reinstall the operating system. During installation, use the same host name and TCP/IP settings assigned to the original system.

3. Repartition and format the system's hard disks to their original specifications.

4. If backup agent software was installed previously, reinstall the backup agents.

5. Use your local or enterprise backup software to perform a complete full system restore from the VM's backup data. Note that all the examples used in this chapter involved exclusively performing full backups. Since this isn't practical in many environments, your recovery may involve restoring a full backup and several incremental (or a differential) backups following the recovery of the full backup.

6. When the recovery finishes, restart the system.

Although we used the term VM throughout the previous steps, keep in mind that the same steps apply to host system recovery. You'd start by reinstalling any necessary hardware, and then you'd reinstall the OS, re-create the host system's disk partitions, and then perform the full system restore. Following the full system restore of the host, you'd finally need to restore the data for each VM running on the host.

Restoring Flat-File VM Backups

Flat-file VM backups are much easier to restore than online backups, since the flat-file backup includes the VM's configuration files as well as its virtual disk files. Since virtual disk files are always dynamically changing, you should have to restore just the most recent flat-file backup of a VM in order to fully recover it. Following the restore of the VM files, you then power on the VM and validate its operation. You may notice that some devices, such as the CD-ROM or floppy drive, may not be present. This is likely because of the VM's devices being mapped to static drive letters on the host system that no longer exists. For example, if a VM was configured to use the D drive on the host as the CD-ROM drive and, when the host is recovered, the CD-ROM is assigned the letter G, the VM may not recognize the CD-ROM drive. To correct this, just update the VM's settings.

Summary

As you can see, with sound reliable backups, VM recovery is truly the easiest part of the data protection process. When protecting VMs, one practice to avoid is to never test your backups. Microsoft once stated that a backup administrator is only "…two failed recoveries away from a pink slip." When data is needed and you can't recover it, you may be the one facing the ultimate responsibility for the data loss. (Blaming the hardware vendor or power company doesn't always work!)

To aid in recovery, many organizations stage recovery servers that already have a base OS installed. You can apply the same philosophy to VMs. Since VMs exist as merely collections of files, it's easy to save a virtual machine base OS installation to a DVD. With this approach, you'd simply need to copy the base VM back onto a host server and then rename the VM to that of the original. This can save you significant time when you need to recover data while under the gun. Another approach with staged-recovery VMs is to create a base VM and run the Windows `sysprep` tool on it. After `sysprep` runs, just shut down the VM and back it up to another location. The next time the VM starts, the Windows mini-setup will run, and you'll then be prompted to provide a name and TCP/IP information for the system.

When taking backup and recovery seriously, it's important to periodically practice restoring your backup data. Just because a backup program told you that a backup was successfully completed doesn't mean the backup data will restore successfully. Many of us have had to learn this lesson the hard way. You shouldn't have to suffer the same fate as those before you!

This chapter marks the end of the general VM management section of the book. In the remaining chapters, you'll be exposed to the many other aspects of virtualization in IT today. If you're still thirsty for information on VMs, then turn to Chapter 11 or Chapter 14 for additional examples and advanced virtual machine configuration topics.