# Chapter 12

# Disaster Recovery and Backup

Disaster recovery (DR) takes many forms, and the preceding chapter on DRLB covers a small part of DR. Actually, DRLB is more a preventative measure than a prelude to DR. However, although being able to prevent the need for DR is a great goal, too many disasters happen to rely on any one mechanism. In this chapter, we categorize disasters and provide solutions for each one. You will see that the backup tool to use will not dictate how to perform DR, but it's the other way around. In addition to DR, there is the concept of business continuity (BC) or the need to keep things running even if a disaster happens. Some of what we discuss in this chapter is BC and not truly DR. However, the two go hand in hand because BC plans are often tied to DR plans and handle a subset of the disasters.

**What Is the Goal of Disaster Recovery?**
The goal of DR is to either prevent or reduce downtime caused by either man or nature.

## Disaster Types

There are various forms of well-defined disasters and ways to prevent or workaround these to meet the defined goal. There is no one way to get around disasters, but knowing they exist is the first step in planning for them. Having a DR or BC plan is the first step toward prevention, implementation, and reduction in downtime. At a conference presentation, I asked a room of 200 customers if any of them had a DR or BC plan. Only two people stated they had a DR or BC plan, which was disconcerting but by no means unexpected.

**Best Practice for DR and BC**
Create a written DR and BC plan.

Stating in writing the DR and BC plan will, in the case that it is needed, help immensely because there will be absolutely no confusion about it in an emergency situation. For one customer, the author was requested to make a DR plan to cover all possible disasters. Never in the customer's wildest dreams did they think it would need to be used. Unfortunately, the "wildest dream" scenario occurred, and the written DR plan enabled the customer to restore the environment in an orderly fashion extremely quickly. It is in your best interest to have a written DR plan that covers all possible disasters to minimize confusion and reduce downtime when, and not if, a disaster occurs.

**Best Practice for DR and BC**
Plan for failure; do not fail to plan.

Yes, this last best practice sounds like so many other truisms in life, but it is definitely worth considering around DR and BC, because failures will occur with surprising frequency, and it is better to have a plan than everyone running around trying to do everything at once. So what should be in a DR and BC plan? First, we should understand the types of disasters possible and use these as a basis for a DR and BC plan template. Granted, some of the following examples are scary and

unthinkable, but they are not improbable. It is suggested that you use the following list and add to it items that are common to your region of the world as a first step to understanding what you may face when you start a DR or BC plan. A customer I consulted for asked for a DR plan, and we did one considering all of these possibilities. When finished, we were told that a regional disaster was not possible and that it did not need to be considered. Unfortunately, Katrina happened, which goes to show that if we can think it up, it is possible. Perhaps a disaster is improbable, but nature is surprising.

Disasters take many forms. The following list is undoubtedly not exhaustive, but it includes many different types of potential disasters.

- **Application failure**

  An application failure is the sudden death of a necessary application, which can be caused by poorly coded applications and are exploited by denial-of-service (DoS) attacks that force an application to crash.

- **VM failure**

  A VM failure could be man-made, by nature, or both. Consider the man-made possibilities such as where a security patch needs to be applied or software is to be added to the VM. By nature could be the failure of the VM due to an OS bug, an unimplemented procedure within the virtualization layer, or an application issue that used up enough resources to cause the VM to crash. In general, VM failures are unrelated to hardware because the virtualization layer removes the hardware from the equation. But it does not remove OS bugs from the equation.

- **ESX Server failure**

  A machine failure can be man-made, by nature, or even both. For example, a man-made failure could be the planned outage to upgrade firmware, hardware, the ESX OS, or the possible occurrence of a hardware failure of some sort that causes a crash. Another example is if power is inadvertently shut off to the server.

- **Communication failure**

  A communication failure is unrelated to ESX, but will affect ESX nonetheless. Communication can be either via Fibre Channel or Ethernet. The errors could be related to a communication card, cable, switch, or a device at the non-ESX side of the communication. An example of this type of failure is a Fibre or network cable being pulled from the box or a switch is powered off or rebooted.

- **Rack disaster**

  Rack failures are extremely bad and are often caused by the rack being moved around or even toppling over. Not only will such an incident cause failures to the systems or communications, but it could cause physical injury to someone caught by the rack when it topples. Another rack failure could be the removal of power to fans of and around the whole rack, causing a massive overheat situation where all the servers in the rack fail simultaneously.

- **Datacenter disaster**

  Datacenter disasters include air conditioning failures that cause overheating, power spikes, lack of power, earthquakes, floods, fire, and anything else imaginable that could render the datacenter unavailable. An example of this type of disaster is the inadvertent triggering of a sprinkler system or a sprinkler tank bursting and flooding the datacenter below. It may seem odd, but some datacenters still use water and no other flame prevention system. Use of halon and other gasses can be dangerous to human life and therefore may not be used.

- **Building disaster**

  Like datacenter disasters, these disasters cause the building to become untenable. These include loss of power or some form of massive physical destruction. An example of this type of disaster is what happened to the World Trade Center.

- **Campus disaster**

  Campus disasters include a host of natural and man-made disasters where destruction is total. An example of this type of disaster are tornados, which will strike one place and skip others but will render anything in its path rubble.

- **Citywide disaster**

  Citywide disasters are campus disasters on a much larger scale. In some cases, the town is the campus (as is the case for larger universities). Examples range from earthquakes, to hurricanes, to atomic bombs.

- **Regional disaster**

  Regional disasters include massive power outages similar to the blackout in the New England area in the 2003 and hurricanes such as Katrina that cover well over 200 miles of coastline.

- **National disasters**

  For small countries such as Singapore or Luxembourg, a national disaster is equivalent to a citywide disaster and could equate to a regional disaster. National disasters in larger countries may be unthinkable, but it is not impossible.

- **Multinational disaster**

  Again because most countries touch other countries and there are myriad small countries all connected, this must be a consideration for planning. Tsunamis, earthquakes, and other massive natural disasters are occurring around us. Another option is a massive planned terrorist attack on a single multinational company.

- **World disaster**

  This sort of disaster is unthinkable and way out of scope!

## Recovery Methods

Now that the different levels of disasters are defined, a set of tools and skills necessary to recover from each one can be determined. The tools and skills will be specific to ESX and will outline physical, operational, and backup methodologies that will reduce downtime or prevent a disaster:

- **Application failure**

  The recovery mechanism for a failed application is to have some form of watchdog that will launch the application anew if it was detected to be down. Multiple VMs running the same application connected to a network load balancer will also help in this situation by reducing the traffic to any one VM, and hence the application, and will remove application from the list of possible targets if it is down. Many of these types of clusters also come with ways of restarting applications if they are down. Use of shared data disk clustering à la Microsoft clusters is also a possible solution.

- **VM failure**

  Recovery from a VM failure can be as simple as rebooting the VM in question via some form of watchdog. However, if the VM dies, it is necessary to determine why the problem occurred, and therefore this type of failure often needs debugging. In this case, the setup of some form of shared data disk cluster à la Microsoft clusters will allow a secondary VM to take over the duties of the failed VM. Any VM failure should be investigated to determine

the cause. Another mechanism is to have a secondary VM ready and waiting to take over duties if necessary. If the data of the primary VM is necessary to continue, consider placing the data on a second VMDK and have both VMs pointing to the second disk. Just make sure that only one is booted at the same time. Use DRLB tools to automatically launch this secondary VM if necessary.

- **Machine failure**

  Hardware often has issues. To alleviate machine failures have a second machine running and ready to take on the load of the first machine. Use VMware High Availability (HA) or other high-availability tools to automatically specify a host on which to launch the VMs if a host fails. In addition, if you know the host will fail due to a software or hardware upgrade, first VMotion all the VMs to the secondary host. VMware HA can be set up when you create a VMware cluster or even after the fact. We discussed the creation of VMware clusters in Chapter 11, "Dynamic Resource Load Balancing." VMware HA makes use of the Legato Automated Availability Management (Legato AAM) suite to manage the ESX Server cluster failover. There is more on HA later in this chapter in the section "Business Continuity."

- **Communication failure**

  Everyone knows that Fibre and network connections fail, so ensure that multiple switches and paths are available for the communications to and from the ESX Server. In addition, make local copies of the most important VMs so that they can be launched using a local disk in the case of a SAN failure. This often requires more local disk for the host and the avoidance of booting from SAN.

- **Rack disaster**

  To avoid a rack disaster, make sure racks are on earthquake-proof stands, are locked in place, and perhaps have stabilizers deployed. But also be sure that your ESX host and switches are located in separate racks in different locations on the datacenter floor, so that there is no catastrophic failure and that if a rack does fail, everything can be brought back up on the secondary server.

- **Datacenter disaster**

  To avoid datacenter disasters, add more hosts to a secondary datacenter either in the same building or elsewhere on the campus. Often this is referred to as a hot site and requires an investment in new SAN and

ESX hosts. Also ensure there are adequate backups to tape secured in a vault. In addition, it is possible with ESX version 3 to VMotion VMs across subnets via routers. In this way, if a datacenter was planned to go down, it would be possible to move running VMs to another datacenter where other hosts reside.

- **Building disaster**

  The use of a hot site and offsite tape backups will get around building disasters. Just be sure the hot site is not located in the same building.

- **Campus disaster**

  Just like a building disaster, just be sure the other location is off the campus.

- **Citywide disaster**

  Similar to campus disasters, just be sure the hot site or backup location is outside the city.

- **Regional disaster**

  Similar to campus disasters, just be sure the hot site or backup location is outside the region.

- **National disasters**

  Similar to campus disasters, just be sure the hot site or backup location is outside the country, or if the countries are small, in another country far away.

- **Multinational disasters**

  Because this could be considered a regional disaster in many cases, see the national DR strategy.

- **World disasters**

  We can dream some here and place a datacenter on another astronomical body or space station.

## Best Practices

Now that the actions to take for each disaster are outlines, a list of best practices can be developed to define a DR or BC plan to use. The following list considers an ESX Server from a single host to enterprisewide with the thought of DR and BC in mind. The list covers mainly ESX, not all the other parts to creating a successful and highly redundant network. The list is divided between local practices and remote practices. This way the growth of an implementation can be seen. The idea

behind these best practices is to look at our list of possible failures and to have a response to each one and the knowledge that many eggs are being placed into one basket. On average for larger machines, ESX Servers can house 20+ VMs. That is a lot of service that could go down if a disaster happens.
First we need to consider the local practices around DR:

- Implement ESX using N+1 hosts where *N* is the necessary number of hosts to run the VMs required. The extra host is used for DR.
- When racking the hosts, ensure that hosts are stored in different racks in different parts of the datacenter.
- Be sure there are at least two Fibre Channel cards using different PCI buses if possible.
- Be sure there are at least two NIC ports for each network to be attached to the host using different PCI buses if possible.
- When cabling the hosts, ensure that redundant cables go to different switches and that no redundant path uses the same PCI card.
- Be sure that all racks are stabilized.
- Be sure that there is enough cable available so that machines can be fully extended from the rack as necessary.
- Ensure there is enough local disk space to store exported versions of the VMs and to run the most important VMs if necessary.
- Ensure HA is configured so that VMs running on a failed host are automatically started on another host.
- Create DRLB scripts to start VMs locally if SAN connectivity is lost.
- Create DRLB scripts or enable VMware DRS to move VMs when CPU load is too high on a single host.

Second, we need to consider the remote practices around DR:

- When creating DR backups, ensure there is safe storage for tapes onsite and offsite.
- Follow all the local items listed above at all remote sites.
- Create a list of tasks necessary to be completed if there is a massive site failure. This list should include who does what and the necessary dependencies for each task.

The suggestions translate into more physical hardware to create a redundant and safe installation of ESX. It also translates into more software and licenses, too.

Before going down the path of hot sites and offsite tape storage, the local DR plan needs to be fully understood from a software perspective, specifically the methods for producing backups, and there are plenty of methods. Some methods adversely impact performance; others that do not. Some methods lend themselves to expansion to hot sites, and others that will take sneaker nets and other mechanisms to get the data from one site to the other.

## Backup and Business Continuity

The simplest approach to DR is to make a good backup of everything so that restoration is simplified when the time comes, but backups can happen in two distinctly different ways with ESX. In some cases, some of these suggestions do not make sense because the application in use can govern how things go. As an example, we were asked to look at DR backup for an application with its own built-in DR capabilities with a DR plan that the machine be reinstalled on new hardware if an issue occurred. The time to redeploy in their current environment was approximately an hour, and it took the same amount of time for a full DR backup through ESX. Because of this, the customer decided not to go with full DR backups.

### Backup

But what is a full DR backup? As stated previously, there are two major backup styles. The first in terms of ESX is referred to as a backup for individual file restoration or a backup made from within the VM. The second is a DR-level backup of the full VM disk image and configuration file. The difference is the restoration method. A standard backup, using agents running within the VM, usually follows these steps for restoration:

1. Install OS onto machine.
2. Install restoration tools.
3. Test restoration tools.
4. Restore system.
5. Test restoration.

A full DR-level backup has the following restoration process:

1. Restore VMDK and configuration file.
2. Register VM into ESX.
3. Boot VM.

As can be seen, the restoration process for a full DR backup is much faster than the normal method, which in many cases makes a DR backup more acceptable, but it generally requires more hardware. But what hardware is really the question, and one that needs to be considered for ESX. A standard ESX stand-alone ESX Server consists of lots of memory and as much disk as can be placed into the server. A standard remote data store–attached ESX Server consists of lots of memory and very little local disk space, and a boot from SAN (BFS) ESX Server usually has no local disk space, which is not a best practice, as outlined in Chapter 3, "Installation." Our best practice for installing ESX outlines a need for a very large `/vmimages` directory and a local VMFS partition, and the main reason for this is to perform safe backups. Figure 12.1 presents backup steps from the point of view of a simple ESX Server connected to an entry-level SAN. Entry-level remote data stores are missing the BC or copy features available on larger remote data stores.

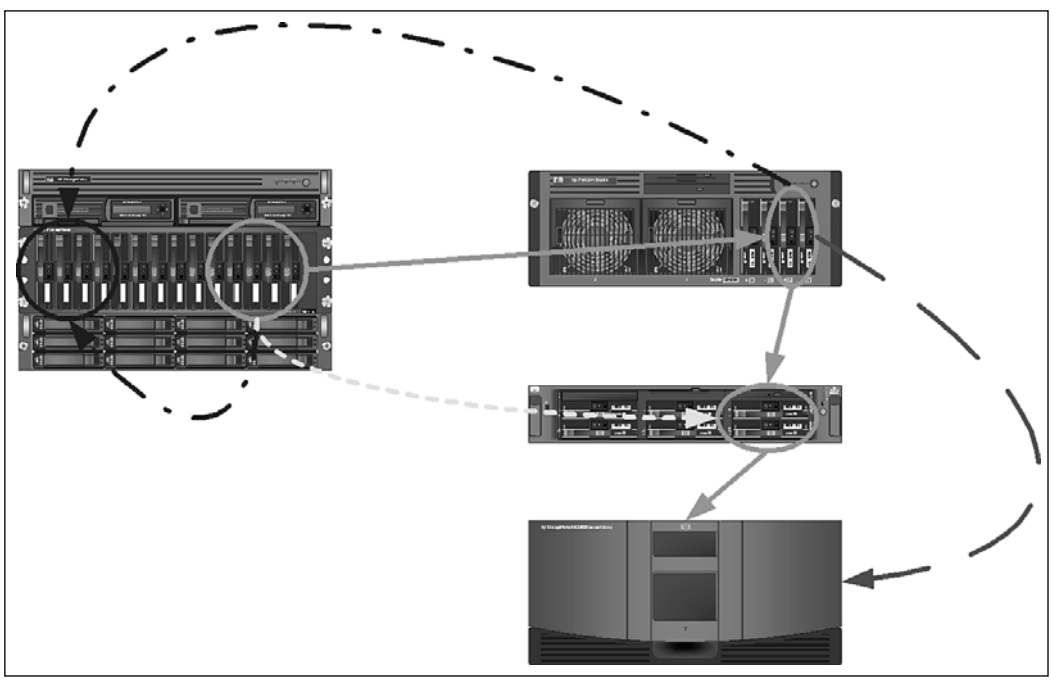

**Figure 12.1** Entry-level backup options
*Visio templates for image courtesy of Hewlett-Packard.*

Whether there is an entry-level remote data store, SCSI-attached storage, or local storage, the steps are similar for creating backups and for DR. There are few

ways to create backups, and the methods are similar no matter where the data will eventually reside. DR backups can be made many ways using an equally different number of tools. Built into ESX is the first method, the second is to use VMware Consolidated Backup (VCB), and one of the other tools is to use Vizioncore's ESXRanger. All can be used to eventually place the data on a tape device, local storage, remote storage, or a remote hot site. File restore backups can be made using VCB and other third-party backup agents.

The simplest form of backup is the single ESX Server approach, as outlined in Figure 12.1. Now it does not matter whether the VMFS data stores are SAN, iSCSI, or even local SCSI for this method. This method works quite well with low-cost, no-frills data store solutions like an entry-level SAN without BC features.

In Figure 12.1, there are several backup paths and destinations shown that are intrinsic within the ESX version 3 software. All but one of these paths exist in earlier versions of ESX. In addition, some third-party backup solutions provide the same functionality as the intrinsic tools but in a more graphical function.

### Backup Paths

Path 1, designated by the solid gray lines, represents a common backup approach for earlier versions of ESX. This approach still provides a level of redundancy that protects a system from catastrophic remote storage failures. This is a full DR-level backup:

- VMs are exported from the remote or local VMFS datastores to a placeholder on the local machine. This placeholder is usually an ext3 file system, but should be another VMFS designated as a special repository just in case the SAN fails. It would not be available for normal usage.
- The exported VM files are in turn sent to a remote backup server through a secure network copy mechanism.
- All but the *most important* VM files are then deleted from the ESX Server. By keeping the *most important* VM files locally, we have taken a simple and useful precaution in the event of failed remote storage.
- The remote backup server sends the data to tape storage.

Path 2 is very similar to Path 1, designated by the short-dash light-gray line. This patch is only available with ESX version 3 and represents the use of the VCB. This can either be a full DR-level backup or per-file backup if the VCB proxy server, Windows 2003, understands the file system to be mounted by the VCB tools:

- The remote storage data stores are mounted onto a VCB proxy server.

- The VCB proxy server will mount the VM from the remote datastore and either export the VM to another location on the VCB in a monolithic full DR backup, or provide a means to access the VMDKs for a per-file backup. (Once more, per-file backup is only available if the VCB Proxy understands the underlying file system of the VMDK.)
- When the VMDK is exported or the per-file backup is finished, the proxy server unmounts the VMDKs of the VM.
- The data is then sent to tape or a tape server from the VCB proxy.

Path 3 has two branches designated by the dash-dot black lines, and is used to clone VMs from one remote storage location to another; this is a poor man's BC copy that more expensive SANs can perform automatically. The branches can be explained as follows:

- **Upper path:** VMs are exported to a local file system.
- **Upper path:** VMs are imported or copied to a new file system on the remote storage.
- **Upper path:** All but the *most important* VMs are deleted from the local file system.
- **Lower path:** VMs are exported from one file system of the remote storage directly to another file system on the remote storage using service console-based VCB or other service console-based backup tools, including Vizioncore's ESXRanger Pro product.

Path 4 is designated by the long-dash, dark-gray line and is not a recommended path but is there for use in rare cases. This is an alternative to Path 1 in that instead of sending the files to a remote backup server, the data would be sent directly to a tape library using the service console. This path is never recommended because there is a need for an Adaptec SCSI HBA, and when the SCSI HBA has issues, the entire ESX Server may need to be rebooted and may crash.

#### Modification to Path 3

As can be seen, even for a single-server ESX Server, there are a myriad of paths for backup and BC. Now let's look at a slightly different configuration. This configuration, outlined in Figure 12.2, describes a system where the remote storage has the capability of doing a business copy or LUN snapshot.

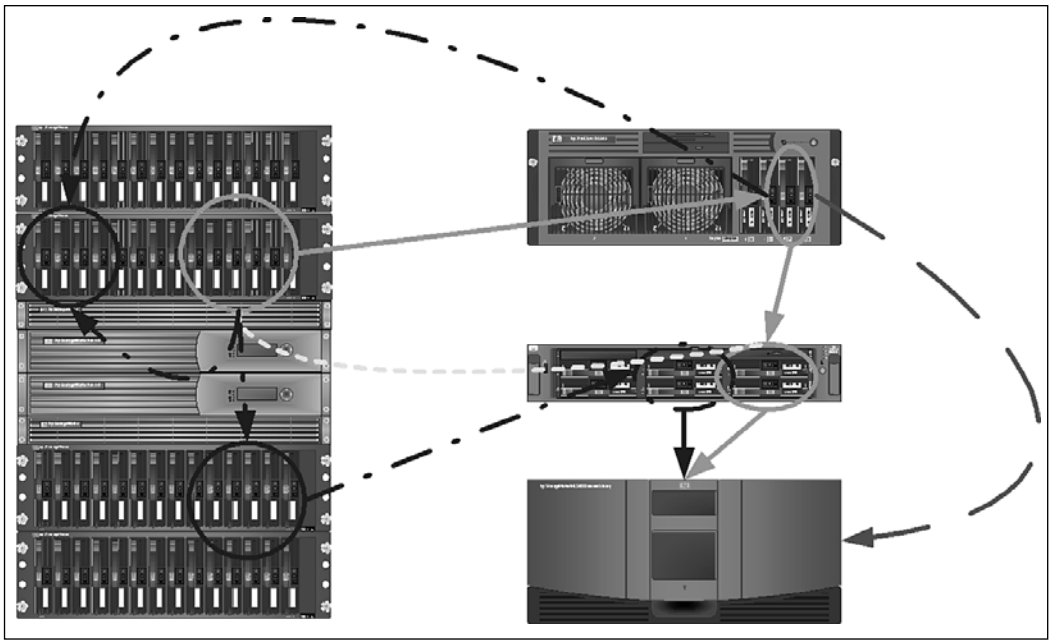

**Figure 12.2** Enterprise class SAN backup options
*Visio templates for image courtesy of Hewlett-Packard.*

Figure 12.2 gives us a change to one of our existing backup paths described previously. Specifically, that change is a modification of Path 3.

Path 3 has two branches designated by the dash-dot black lines and is used to clone VMs from one remote storage location to another; a poor man's BC copy that the more expensive SANs can perform automatically. Path 3 and the modification (essentially creating Path 5) work this way:

- **Upper path:** VMs are exported to a local file system.
- **Upper path:** VMs are imported or copied to a new file system on the remote storage.
- **Upper path:** All but the *most important* VMs are deleted from the local file system.
- **Middle path (previously lower path):** VMs are exported from one file system of the remote storage directly to another file system on the remote storage using service console-based VCB or other service console-based backup tools, including Vizioncore's ESXRanger Pro product.
- **Lower path (this is the new path or Path 5):** If a SAN is capable of doing a LUN snap or LUN-to-LUN copy, it is now possible to copy the complete LUN

using just the SAN software. ESX is unaware of this behavior. However, the caveat is that this produces a crash-consistent backup, whereas the other methods do not.

The LUN snap or LUN-to-LUN copy procedure starts by mirroring the data from one LUN to another LUN, when the mirror is fully built and consistent, creating an instantaneous copy of a LUN with no impact on the ESX Server that would snap it off from the primary LUN. The LUN-to-LUN copy would be crash consistent and happen while VMs are running. Now here is the tricky part; because the new LUN houses a VMFS, and thus all the VMDKs, another ESX Server is required to access the new LUN so that the VMDKs can be exported to a local disk and from there be dropped to tape. This backup processing ESX Server would in general not be used for anything else and may belong to its own farm. For ESX versions earlier than 3.0, a method to transfer the VM configuration files would need to be added, too, but that is a small chunk of data to transfer off the production ESX Servers compared to the VMDKs and could be set up to be transferred only when the files change.

A crash-consistent backup is one in which the data written to the VMDKs has been halted as if the VM has crashed. Therefore, all backups that are crash consistent have the same chance of being rebooted as if the VM has crashed. One solution to this problem is to quiesce the VMDKs prior to the LUN mirror copy being completed, which would require some way to communicate from the SAN to the ESX Servers. The creation of a snapshot, which VCB does, will quiesce the VMDK so that it is safe to copy. Unfortunately, disk quiescing only happens in Windows, NetWare, and a few other types of VMs, but not necessarily Linux. It also requires VMware Tools to be installed. It is possible, however, to write your own quiescing tools to be run when the VM is placed into snapshot mode.

#### Additional Hot Site Backup Paths

Figure 12.3 demonstrates multiple methods to create a hot site from the original ESX environment. A hot site is limited in distance by the technology used to copy data from one location to another. There are three methods to copy data from site to site: via Fibre, via network, and via sneaker. In addition to the four existing paths, we can now add the paths covered in this section.

Path 6, depicted by the leftmost dash-dot-dot gray line, is the copying of local remote storage to similar remote storage at a hot site. Path 6 picks up where Path 3 ends and adds on the transfer of a full LUN from one remote storage device to another remote storage device using Dark Fibre, Fibre Channel over IP, or standard networking methods. After the LUN has been copied to the remote storage, Path 2 or Path 3 could once more be put into use at the hot site to create more backups.

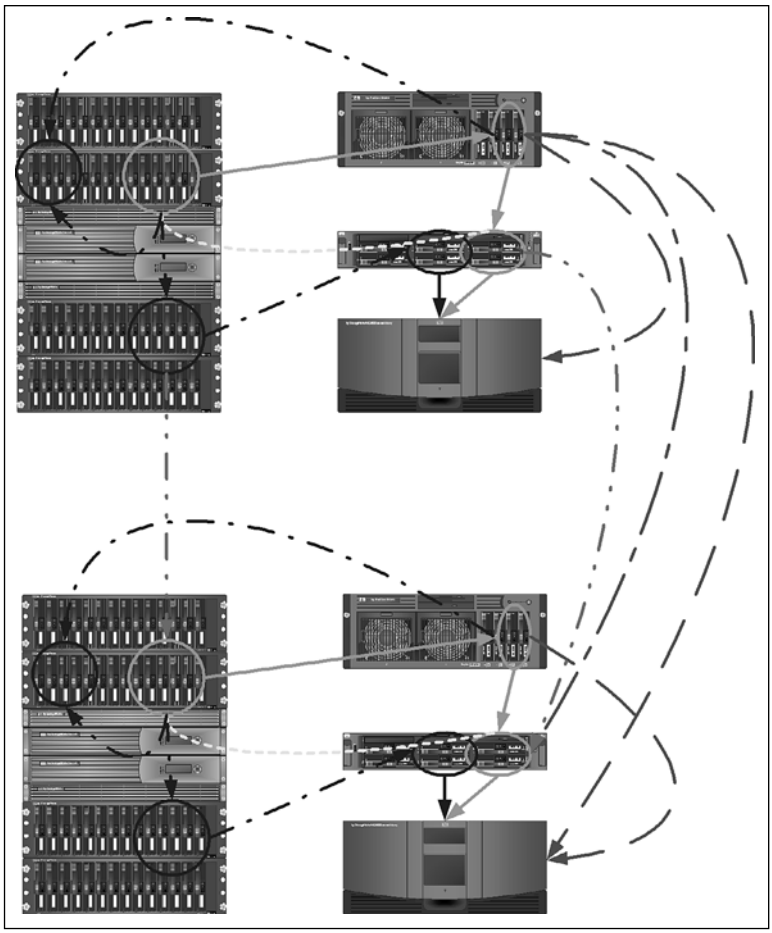

**Figure 12.3** Enterprise class SAN hot site and remote backup options
*Visio templates for image courtesy of Hewlett-Packard.*

Path 7, depicted by the rightmost dash-dot-dot gray line, is the copying of the local ESX storage to a remote location using a secure network copy mechanism. Path 7 picks up where Path 1, 2, or even 3 leave off and uses standard network protocols to get the data onto a backup server on the hot site. From the backup server at the hot site, the VMs could then be restored to the hot site datastores waiting to be enabled in case of failure. Or from there they could be sent to a remote tape device.

Path 8 is depicted by the long-dash, short-dash line and uses the service console of the ESX Server to create the backup described in Path 7. This is not recommended because you do not want to tie up service console resources with a lengthy network traversal. Path 8 picks up where part of Path 1 finishes and would

initiate a copy of the data not only to the local backup server but also to the remote backup server. This path is not recommended.

Path 9 is depicted by the rightmost long-dash line and uses the service console to write directly to the tape server at the remote site. This path is not recommended because it will tie up service console resources and requires an IP-based tape device at the hot site. Path 9 would pick up where part of Path 1 finishes and would initiate a remote backup using an IP-based network. However, instead of going to a disk like Path 8 does, it would go directly to a tape device.

Path 10 is not depicted on the diagram, but it would be the taking of backup tapes from the main site to the hot site to be restored or stored in secure storage. This is often referred to as a sneaker-net approach.

### Summary of Backup

No matter how the data gets from the primary site to the hot site, the key is to get it there quickly, safely, and in a state that will run a VM. To that end, the long-dash and long-dash, short-dash paths outlined on all the diagrams should be avoided because they are not necessarily very fast, safe, or consistent. The green or solid line and short-dashed paths provide the most redundancy, whereas the dash-dot paths provide the least impact, and the dash-dot-dot paths provide the better methods to create hot sites or move data from primary site to hot site short of sneaker net. If all these paths fail, however, remember your tapes sitting offsite in a vault and the combination to get access to them quickly. Above all, *always* test your backups and paths whether local or using a hot site. Always remember, backup and DR go hand in hand.

## Business Continuity

BC with ESX can be accomplished in several fashions. Just like backups, BC has multiple paths that it can take. Some of these paths are automated, whereas others require human intervention. One of the ideas behind BC is to provide a way for the business to continue uninterrupted and the VMware cluster technology provides a part of this by the implementation of VMware DRS and VMware HA. Where DR-level backups are generally geared toward the re-creation of the environment, BC is the application of clustering technology to prevent the need for such time-consuming restoration. This is achieved using VMware HA but also through the use of preconfigured hot sites that can come online in a fraction of the full restoration time. Our methods discussed earlier implement hot sites by using backup means. However, hot servers or servers in the datacenter that do not do anything until they are needed are other options. VMware HA covers the latter case. VMware HA is a high-availability-based solution that will, when an ESX

Server crashes, boot the VMs running there onto other ESX Servers either randomly (according to VMware DRS rules) or by using defined placement and boot order rules. The HPSIM VMM plug-in module also provides a similar HA capability by specifying an alternative host on which to run the VMs. Although VMware HA is only available for ESX 3, all versions can benefit from the HPSIM VMM alternative host method.

Outside of VMware HA, a myriad of other BC options are available. These include having as many redundant components as possible in different places within the datacenter, building, or campus, and there are multiple paths to all these devices from the ESX Servers. This leads to a much higher cost in and availability of hardware, but it will be the difference between a short service interruption and an absolute disaster. Consider the case of an ESX Server crashing with smoke pouring out of a vent. If you had invested in VMware HA, the software would automatically boot the VMs on another system. However, if you purchased a HP C-class blade in a RAID blade configuration, the ESX Server would fail over using a complete hardware solution. This leads to the question of which is better, hardware or software solutions. And the answer is, as always, it depends on the cost benefit. This same HP C-class blade has one limitation, the designated RAID blade must be in the same chassis of the failing node, and they must both share disks on a disk blade. This limits the amount of processing power to the blade chassis; and what happens if the chassis itself fails?

Many sites keep identical preinstalled hardware locked in a closet to solve some of these problems. However, it is your disaster to recover from, so think of all the solutions and draft the plan for both DR and BC appropriate for you.

## The Tools

Now that the theory is explained, what tools are available for performing the tasks? Although each family of enterprise class remote storage has its own names for the capability to make LUN-to-LUN copies, refer to the SAN compatibility documentation to determine what is and is not supported, because it might turn out that the hot-copy mechanism for your SAN is not supported by any version of ESX. For example, HP SANs Business Copy and Continuous Access are supported, as is EMC SANs Hot Copy.

Beyond the remote storage–to–remote storage copies, many other tools are available from VMware and various vendors. These, in most cases, require some form of agent to be used to create the backup and place it on a data store associated with the tool. All these tools must first place a running VM in a delta mode (snapshot or REDO), which changes the file to which disk writes occur so that the

backup software can make a copy of the primary VMDK, as in Figure 12.4. When the backup finishes, the delta is committed, and once more the primary VMDK is now used. The delta file grows over time as more data is written to it and grows in 15MB chunks to reduce SCSI-2 Reservation conflicts. However, because the delta file is really a FIFO and not a true VMDK, it is much slower and therefore dictates that a backup should occur when the VM is relatively inactive. And finally, the longer a VM is in a delta mode, the slower it will run, and the larger the delta file, which implies more locks. Each SCSI disk associated with a VMDK should be backed up separately to reduce the overall time spent in delta mode. Now, because we discussed with snapshots in Chapter 8, "Configuring ESX from a Host Connection," it is possible to have a tree of delta files for every VM. In this case, most tools will not work, including VMware Consolidated Backup and Vizioncore's ESXRanger Pro products. They require all deltas or snapshots to be deleted first. These situations may require a specialized backup script to be used. In addition, these tools will not back up templates, and those also require a specialized backup script.

Figure 12.4 depicts delta mode processing for ESX version 3 and earlier versions. In previous versions, delta mode was really REDO mode. Deltas in ESX 3 are now created by the snapshot mechanism.

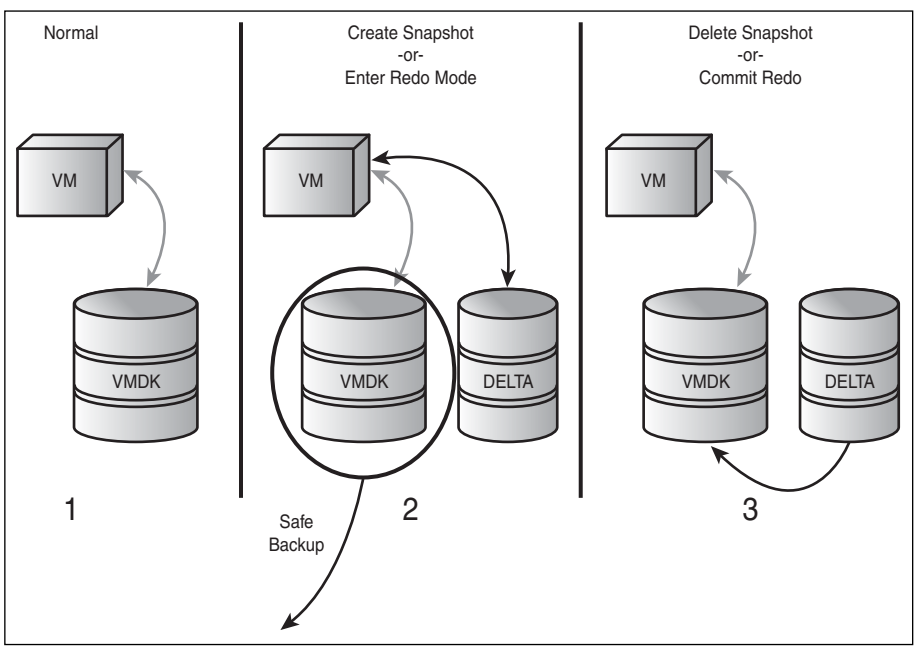


**Figure 12.4** Delta mode processing

Once in delta mode, if the disk was not quiesced, a crash-consistent backup is created, which implies that a boot of a restored VMDK will boot as if the VM had crashed. Quiescing disks is limited to Windows VMs and only those that have snapshots. In addition, LUN-to-LUN or remote storage–to–remote storage copies also produce crash-consistent backups. A crash-consistent backup, depending on how it was achieved, should not be restored to anything other than a VMFS because the VMDKs are sparse files or monolithic files that have a beginning marker, some data, and an end marker and not much else. Restoration of such files to non-VMFS can result in loss of data. However, if the VM is first exported, resides on an NFS data store, or is in 2Gb sparse disk format, it can be restored to any file system and imported back into ESX with no possibility of data loss. With Fibre connections, it is not even necessary for the host that places the VM into delta mode to be the host that actually does the backup. This generally requires extra scripting, but it is possible for a host that is running the VM to place the VM into delta mode and then signal a backup host that the backup can be made. When the backup is completed, another signal is made to the running host to commit the backup. In this way, the backup host offloads the work from the running ESX Server. The signals could even be reversed so that the backup host does all the work and calls the running host only as necessary. Following here is an example script of this behavior for version 2.5. As for ESX version 3, this script is necessary only when the VMware Consolidated Backup proxy server is not in use or other tools such as Vizioncore's ESXRanger Pro or HPSIM's VMM are not in use.

## Simple Backup Scripts

The following are by far the simplest backup scripts that can be written using the tools intrinsic to ESX. For ESX version 3, the script uses VCB, and for ESX version 2, the script uses `vmsnap.pl`. Although the latter script exists in ESX 3, it points the user to the VCB functionality and does nothing else. Some VM configurations, however, will still need similar functionality. These are for those VMs that have snapshots disabled.

### ESX version 3

The VCB tool has a command-line component for the ESX service console and a Windows 2003 Enterprise server. The VCB command-line tools will export VMs, create snapshots, and access the contents of and complete virtual disks for a VM. VCB provides a way to access and backup all VMs in an organized fashion for all ESX Servers and VMs using the VCB proxy server and any version 3 ESX Server.

In addition to implementing VCB, ESX version 3 implements a form of VMware Workstation version 5 snapshot capability. VCB makes exclusive use of the snapshot functionality to make copies of VMDKs before a backup of the snapshot occurs. Under the hood, this keeps delta mode to a minimum and results in a copy of the VMDK that can then be exported or mounted to produce a valuable backup from what is now referred to as a proxy server. The proxy server in the preceding figures is not only the backup server but also forwards the backup images to another server, which talks to a tape or hot site. For example, in Figure 12.1, the proxy server is the second machine from the top on the right side of the image. This proxy server is a Windows machine or another ESX Server that has access to any of the storage devices used by ESX so that it can use the VCB commands to either mount the VMDK to the proxy server to aid in backup or export the VMDK to another location to create a backup.

If the VMDK is a physical mode raw disk map (RDM), a snapshot cannot be made, and the only method of backup of the RDM is to use the traditional methods available: backup from within the host or SAN copies. A virtual mode RDM still works as expected. If the VMDK is an independent disk, snapshots will not work, and the traditional methods to make backups must be used. ESX version 2.5.x and earlier use the independent disk VMDK format, which requires the REDO mode capabilities provided by `vmsnap.pl`. Because snapshots are used by VCB, the disabling of the snapshot features will keep VCB from working.

ESX version 3 also includes a method of scheduling automatic snapshots to make a point-in-time backup of the guest that can then be explicitly exported using VCB, and then dropped to tape, media, or copied across to a hot site after the snapshot is fully built. This functionality will work with running and stopped VMs, which is an advance over ESX version 2.5.x. Automating a snapshot offers a method to save old data from being overwritten until the snapshot is deleted or merged with the previous snapshot. Currently, VCB-based backup tools will not work if there are existing snapshots, so this functionality has limited usage, except perhaps to automatically trigger an action to occur. No alarm will trigger when a snapshot is made, but using the open source VI Perl interface it is possible to poll for this state and take an appropriate action. It is a complex method that most tools that hook into VCB already do for you.

The other option is to use VCB with the provided scripts or write your own to back up the VMs using VCB exclusively. The scripts provided are for TSM and Veritas Networker and BackupExec. The scripts run from a Windows proxy server to access the VMs using remote VCB tools that also exist on every ESX Server. The VCB scripts provide pre- and post-backup work to be done in relation to the

backup of the VM in question. However, these scripts are not useful if the VM's VMDKs are in independent mode.

VCB consolidates the need for licenses to just the proxy server. That way, if there are 300 VMs running, there is no need for 300 VM or 10 or so ESX Server licenses of the backup client. This cost savings can be significant, and the single location to produce backups, the proxy server, aids in operational issues by placing the burden of backup there so that backups no longer impact the performance of the VM or ESX Server and cuts down locking to a minimum.

The following script works from the ESX 3 service console. This code locks the `BACKUP` volume, and if the lock exists, does not perform the backup. When the lock exists, it uses the service console `vcbMounter` command to run the backup:

```
#!/bin/sh

PATH=/usr/sbin:/bin:/usr/bin
export PATH

BACKUP=/vmfs/volumes/BACKUP/vcb

if [ -e $BACKUP/.lock ]
then
         exit
fi
for x in `/usr/bin/vmware-cmd -l ¦ sort -r`
do
         /bin/touch $BACKUP/.lock
         y=`/bin/basename $x .vmx`
         #### Keep two backups, uncomment the following two lines
         #/bin/rm -rf $BACKUP/${y}.bak
         #/bin/mv $BACKUP/$y $BACKUP/${y}.bak
         /usr/sbin/vcbMounter -a name:$y -r $BACKUP/$y
done
/bin/touch $BACKUP/.done
```

By uncommenting the two lines inside the `for` loop, it is possible to keep two copies of the backups. This is a simplistic approach to backups using two common ESX commands (`vmware-cmd`, `vcbMounter`). Note that normally the `vcbMounter` com-

mand takes a host, username, and password, but these are hard-coded into the `/etc/vmware/backuptools.conf` file, which for security reasons should just have read-only permissions for the root user.

### ESX Version 2

The `vmsnap_all` and `vmsnap.pl` scripts provide a scripted mechanism to create a crash-consistent backup image that can be placed on any file system, because the process followed is to place the VM into REDO mode, export the VM, and commit the REDO. Because the VM was exported and not just copied, any file system will work. Many vendor tools make use of the `vmsap.pl` functionality local to the ESX Server to make backups. However, `vmsnap_all` and `vmsnap.pl` will not work on non-running VMs, and those need to be exported and backed up using a traditional script.

The only change to the ESX v3 script presented above for earlier versions is to change the `vcbMounter` command to use `vmsnap.pl` with the appropriate arguments to send backups to local storage.

However, if the backup of a nonrunning VM is made to look the same as the ones produced by `vmsnap.pl`, the `vmres.pl` script can be used to restore them. The following script produces an ESX v2.5 backup of a nonrunning VM that will work with `vmres.pl`:

```
#!/bin/sh
VMX=$1
if [ -f $VMX ]
then
        vmd=`dirname $VMX`
     vmp=`basename $vmd`
     mkdir /vmimages/localbackup/$vmp
     cp $VMX /vmimages/localbackup/$vmp
     cp $vmd/nvram /vmimages/localbackup/$vmp/$vmp.nvram
     cp $vmd/vmware.log.* /vmimages/localbackup/$vmp/
     # Get list of SCSI devices
     for x in `grep scsi $VMX¦grep : ¦grep present¦ grep TRUE ¦ awk
'{print $1}'`
      do
            y=`basename $x .present`
            z=`grep $y.name $VMX ¦ awk '{print $3}'`
```

```
            vmkfstools -e $vmp.$vmp.vmdk $z
     done
fi
```

### ESX Version Earlier Than 2.5.0

Before ESX version 2.5.0, users and vendors had to create their own scripts. These scripts would place a running VM in REDO mode and either copy the data to another VMFS or export to another file system. Often, these VMs were backed up to tape directly from the VMFS, which will produce a tape image that can only be restored to another VMFS, which limits its functionality.

## Local Tape Devices

Other than these methods, ESX does not have the tools necessary to directly control tape libraries or tapes installed by default. Therefore, these are not recommended to be used. Local tape devices and libraries require two things: a specific Adaptec SCSI HBA and software to control the robot or tape. The software is extremely easy to find, but once installed is not really supported by VMware or other support organizations. Either way, when there is a problem with the local tape or tape library device, the ESX Server often has to be rebooted, and although it is possible to remove and reload the appropriate kernel module, some devices are not fully reconfigured unless they are seen at boot time. This is a failing in the COS version, the drivers used, and how the device or devices are allocated. On ESX version 3, all devices are assigned to the VMkernel, which implies a failure of the device for some reason could still require a reboot of the ESX Server. Using tape devices and libraries attached to remote archive servers is the recommended choice if possible.

In Chapter 4, it is suggested that you install `mtx`, which is a tool to manipulate tape libraries. MTX is vital to manipulate tape libraries from within ESX. But it often requires some form of scripting to use it properly. Specifically, there is a combination of tools necessary to properly manipulate tapes and the robotics of the libraries. The combination is MT and MTX. MT is found on the ESX media. Both of these tools use SCSI protocols to communicate with the tape and robot, respectively. Included is a simple script that can be used to manipulate a robot and tape device to flip tapes as necessary. If you *must* roll your own backup solution that uses a local tape device, this is invaluable and is often an inherent part of other solutions.

**Caution**

Do not use this script unless you absolutely have to do backups from the ESX service console.

Using tape devices from the ESX service console could result in the need to reboot the ESX server if there are tape device errors, and the same holds true if using tape devices via pass-thru mode from a VM.

The following script takes two arguments, one to specify the tape device to use and the second is the tape volume to use. This is exactly the arguments the Linux `dump` command issues to a tape-changing script when `dump` is given the `-F` option. Before proceeding, it is best to become familiar with `dump` and other backup tools. There are several references available in the "References" element at the end of this book. In addition to taking arguments, the script uses three files to define its actions, all of which are located in `/usr/local/etc`. Table 12.1 provides the names of the files and what they do.

**Table 12.1**

**Filenames for MTX Script**

| Filename | Usage |
|---|---|
| /usr/local/etc/slot | Defines the current tape library slot being used |
| /usr/local/etc/maxslot | Defines the maximum slots available in the tape library |
| /usr/local/etc/history | A log file that lists the history of tapes used |

The basic use of the following script is to rewind a given tape, eject it if necessary, and then use the robot to unload the current tape, store it in the proper slot, and reload the tape device with a new tape. When the new tape is loaded into the tape device, the script waits until it is available for write before returning to whatever program called it, which could be another script, a tool such as `dump`, or something else:

```
#!/bin/ksh -x
# Manipulate Tape Library and Tape device.

# Get args
dev=$1
```

```
vol=$2
echo "$dev $vol"

# read slot files
SLOT=/usr/local/etc/slot
MAXS=/usr/local/etc/maxslot
HIST=/usr/local/etc/history
mslo=`/bin/cat $MAXS`
slot=`/bin/cat $SLOT`

# Find the device number
d=`echo $dev¦ /usr/bin/wc -c¦/bin/sed 's/[\t ]//g'`
let d=$d-1
dnum=`echo $dev ¦ /usr/bin/cut -c $d`
# let dnum=$dnum-1

# force rewind device
# Rewind and unload tape:
skiptape=1
while [ $skiptape -eq 1 ]
do
      st=`echo $dev¦/bin/sed 's/n//'`
      /bin/mt -f $st rewind
#      /bin/mt -f $st offline
      /usr/local/sbin/mtx unload $slot $dnum
      let slot=$slot+1

      # wrap for now. Be sure we are on even borders however!
      if [ $slot -gt $mslo ]
      then
            if [ $vol -eq 0 ]
            then
                  slot=1
            else
                  # if in a volume 'abort'
                  echo 0 > $SLOT
```

```
            exit 2
      fi
fi

# keep track of where we are
echo $slot > $SLOT
if [ ! -z "$TAPELABEL" ]
then
      echo "$slot      $TAPELABEL      $vol" >> $HIST
fi

# load tape.
/usr/local/sbin/mtx load $slot $dnum

# only release when tape is available for write
rc="false"
i=0
while [ Z"$rc" != Z"true" ]
do
      let i=$i+1
      /bin/sleep 10
      msg=`/bin/mt -f $st status ¦ /usr/bin/tail -1`
      rc=`echo $msg ¦ /usr/bin/awk '/WR_PROT/ {print "skip"}'`
      if [ Z"$rc" = Z"skip" ]
      then
            rc = "true";
      else
            rc=`echo $msg ¦ /usr/bin/awk '/ONLINE/ {print "true"}'`
      fi
      # reached timeout of 120 seconds
      if [ $i -gt 24 ]
      then
            exit 2
      fi
done
rc=`echo $msg ¦ /usr/bin/awk '/WR_PROT/ {print "skip"}'`
```

```
      if [ Z"$rc" != Z"skip" ]
      then
            skiptape=0;
      fi
done

# this is goodness
sleep 20
exit 0
```

In addition to being able to roll your own via the service console, it is also possible to connect a VM to a local tape device. In this case, a VM is attached to a local tape device using the SCSI generic devices available when building a VM. The virtual SCSI device ID must match the physical SCSI device ID in these cases, and the device needs to be multitarget and not multi-LUN; in other words, multiple device IDs should be present and not multiple LUNs for the same device ID. The latter is not supported under any version of ESX. However, a VM attached to the local SCSI tape or tape library should have its SCSI HBA assigned to the VMs, as is the case with ESX version 3, so that there is no contention on the device.

**Best Practices for Local Tape Devices**
Do not use local tape devices; if it becomes absolutely necessary, be sure to understand the impact on the local ESX Server and plan VM deployments accordingly.

## Vendor Tools

A small number of vendors produce backup tools specifically for ESX. Some are designed specifically for ESX, whereas others can be used with ESX. Pretty much any current backup tool that has a Linux client will work with ESX in some form. However, tools such as VizionCore vRanger and HPSIM VMM have specific code for use with ESX. No matter which tool is chosen, there is a first step for all the backup tools. See Chapter 5, "Storage with ESX," and open up the firewall to allow the appropriate ports to be used before performing any backup and installing any necessary agents. The next step is to install the appropriate agents, if any, and then to perform a test backup to verify everything is set up properly.

### ESXRanger

ESXRanger has a graphical mechanism for creating backups and a series of dialog boxes to use when scheduling backups to run. It can also make multiple versions of backups so that different VM states can be restored as necessary. For ESX versions earlier than version 3, ESXRanger uses the `vmsnap.pl` functionality to perform all backups. For ESX version 3, ESXRanger (see Figure 12.5) uses snapshots and implements its own form of VCB. However, it will work with the legitimate VCB.

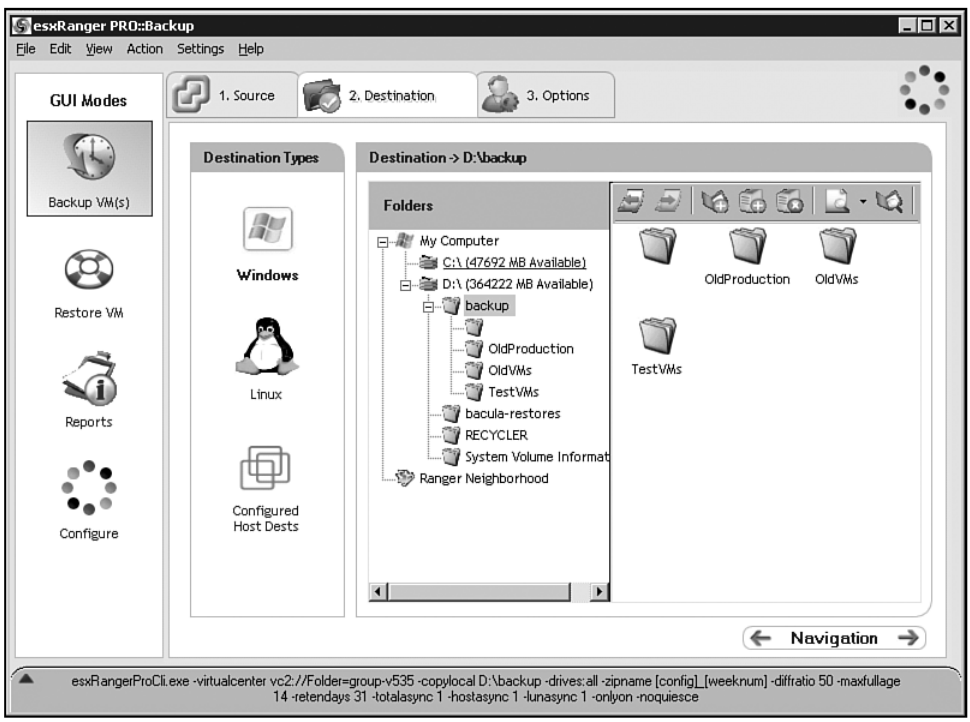


**Figure 12.5** ESXRanger

### HPSIM VMM

In addition to being able to schedule backups of a VM, it is possible to set up a failover host for when an ESX Server that is a part of DRLB dies. This functionality, discussed in the preceding chapter, makes this web-based tool extremely powerful. To use VMM, the ESX Server must be registered in HPSIM (see Figure 12.6) and the VMM component properly licensed and available. Without this, HPSIM only monitors the hardware of the ESX Server.

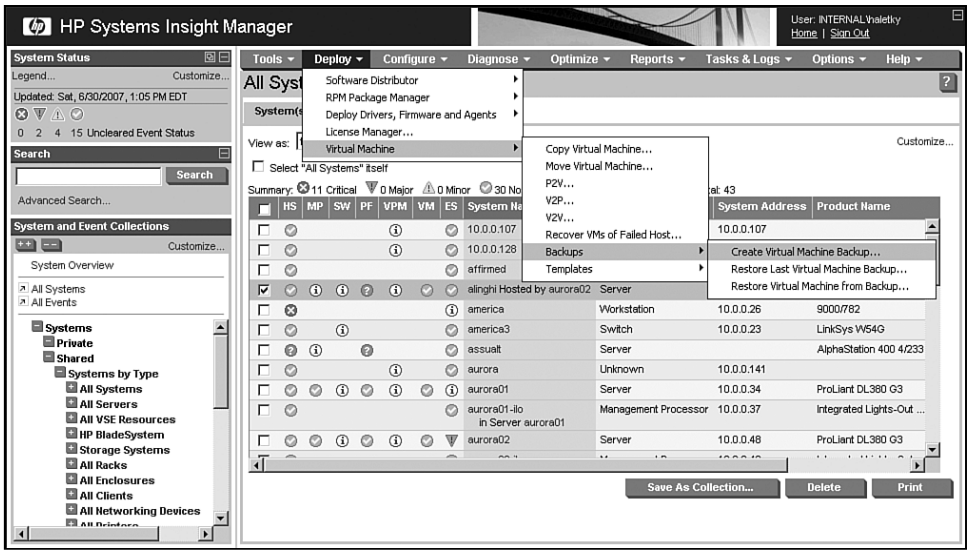


**Figure 12.6** HPSIMM VMM

## Other Tools

Another set of tools requires additions to make them work; unfortunately, the specific scripts for the tools may not exist prepackaged by VMware or any other vendor. VMware and vendors offer suggestions on their use, but the final scripting could be left to the implementer. Each of these tools has a method to schedule a backup that is initiated via an agent running on the backup client (VCB proxy server), and these agents will make a call out to a local script to do anything that is necessary to do prior to the backup occurring. In these cases, it is best to use the VMware-provided scripts to create a safe backup of the VMDK before letting the agent transfer the data to the backup server. And finally, the backup agents call another local script to complete any tasks necessary after the backup has occurred. In addition to often requiring specific scripting, these tools require different versions of the application to be used for different versions of ESX. All of these tools work from within the VM, and most will work from the service console if they are at the proper version of the tool for the version of ESX, and all will work with VCB given proper pre- and post-backup processing scripts. Examples of these other tools include the following:

- **HP Data Protector**

  There are no prepackaged VCB scripts for use by HP Data Protector.

- **Legato Networker**

  VCB for ESX version 3 has a prepackaged set of scripts that provide the pre- and post-processing required to safely back up the files by Networker.

- **Veritas NetBackup**

  VCB for ESX version 3 has a prepackaged set of scripts that provide the pre- and post-processing required to safely back up the files by NetBackup.

- **Tivoli Storage Manager**

  VCB for ESX version 3 has a prepackaged set of scripts that provide the pre- and post-processing required to safely back up the files by TSM.

## Conclusion

It is imperative when using ESX that you create successful backups. Successful backups are ones that are tested to see whether they restore running, usable VMs. The goal of this chapter was to present what is available to the users of ESX and the advanced capabilities of VMware clusters in terms of BC. I may have used ESXRanger and HPSIM VMM to create backups in addition to the intrinsic tools, but they are by no means the only methods available. Just be sure that a good DR and BC plan exists that covers all the possible disasters and that the plan is implemented and tested as often as necessary.