Using Network Appliance™ Snapshot™ Technology with VMware® ESX Server

Network Appliance | Brian Casper | March 2005 | TR 3393

TECHNICAL REPORT

Network Appliance, a pioneer and industry leader in data storage technology, helps organizations understand and meet complex technical challenges with advanced storage solutions and global data management strategies.

Abstract

This document discusses the techniques for backing up and restoring virtual machines and application data used by virtual machines in a Vmware ESX server environment using Network Appliance Snapshot technology.

Table of Contents

1. Executive Summary

VMware ESX server is a virtual infrastructure software technology that enables server consolidation and management for Intel® and AMD platforms. Network Appliance Snapshot technology allows instantaneous point-in-time copies to be made of virtual storage devices. This technical report will describe methodologies that may be utilized to make efficient use of Snapshot copies with virtual machines in a SAN ESX server environment.

2. Background

Utilizing NetApp storage systems in virtual server environments meets the flexibility challenges that arise from provisioning, backup and recovery operations, and disaster protection. This technical report will discuss how to meet these challenges as they pertain to the Vmware ESX Server environment. Although the platform utilized for the development of this work was based on the ESX 2.5 and Data ONTAP™ 7.0 releases, the processes are intended to be applicable in whole or part to other software versions.

Starting with a bare metal server, the ESX operating environment is installed. Management of the ESX server is accomplished via a web-based management user interface (MUI) or through direct login via ssh or telnet to a service console. All of the physical hardware on the server platform is pooled and shared among the ESX server OS and one or more virtual machines (VMs). Each VM operates as an independent entity and contains a single instance of a guest OS. ESX supports a wide variety of guest OS types, including many flavors of Microsoft® Windows®, Linux®, and NetWare™.

Each physical server contains hardware that must be managed and shared by the ESX software. Hardware such as network adapters (NICs) may be dedicated to a specific VM, shared among multiple VMs, or indirectly shared by the ESX service console and all VMs via a logical Ethernet switch mechanism. Memory and CPU resources may also be shared or dedicated depending on the objectives of the virtual infrastructure environment.

In the case of dedicated hardware, the ESX server partitions the adapter at the hardware level and binds it to a specific server entity. If the server entity is the ESX server, no VMs will have access to that physical hardware. This is useful, for example, to provide a network interface that is dedicated strictly to managing the ESX server over the network and connecting to the VM consoles. If the binding is specific to a VM, only that VM can use that piece of hardware. For example, you might dedicate a network interface to a specific server that has very high throughput requirements. Other methods of resource management, such as network traffic shaping, are beyond the scope of this paper.

When sharing hardware between the ESX server and VMs, VMware has utilized a two-part driver model. The lower layer executes within the ESX server kernel and interacts directly with the physical hardware. The ESX server then provides a virtual device interface for each piece of physical hardware. At the next layer, the VM makes this virtualized hardware visible to the guest OS running inside the VM. From the perspective of the guest OS, there is nothing special about this hardware and it will be treated the same as if it were running on a single physical server in a non-ESX environment.

FCP

The shared hardware example described above is the most interesting from a storage perspective. In an FCP environment, the underlying HBA devices may be based on either QLogic or Emulex, and the appropriate HBA drivers for these adapters are included with the ESX software. They are installed automatically as part of the ESX OS and load during the boot sequence. Included with these drivers is a VMware Active/Passive multipath layer for FCP fault tolerance and static load balancing on a per-LUN basis. It is not possible to dedicate an FCP HBA to a VM, so the VM will never be required to load or maintain HBA driver software.

Any LUN devices that are mapped from a filer and become visible to the ESX server are then presented to the VM as a virtualized SAN device. This is accomplished in one of two ways. The LUN may be formatted with a VMFS and virtual disk files can be created, or the entire LUN may be mapped directly to a VM via the pass-through device method. In either case, within the guest OS the disk hardware appears to be a locally attached DAS device. At the guest OS level, all SAN storage that is managed by the ESX server is connected to a virtual LSI-logic or Buslogic SCSI adapter. From the perspective of the guest OS, the storage is treated as local disk and there is no view of the QLogic or Emulex adapter. Accordingly, no multipath software or HBA driver software is required inside the guest OS.

iSCSI

Special purpose hardware adapters, such as transfer offload engines (TOE) or HBA- and NIC-based protocol accelerators designed for the iSCSI protocol, are not presently supported by VMware, so there is no direct method from the ESX server OS to act as an initiator using the iSCSI protocol. Additionally, you cannot download and compile a software iSCSI initiator and load it into the ESX server OS to provide standard virtualized storage to a VM.

It is completely possible, however, to install a software iSCSI initiator inside a guest OS. Packets containing iSCSI traffic may be successfully passed through an ESX server shared network interface. For improved performance, a dedicated NIC may be utilized by the guest OS. iSCSI through a software initiator works perfectly because from the perspective of the ESX server, these packets are nothing more than uninteresting TCP/IP traffic, much like HTTP, NFS, CIFS, SNMP, or any other typical network chatter.

Using iSCSI software initiators inside a guest OS functions exactly like iSCSI software initiators running in any supported OS on dedicated physical hardware. Tools such as SnapDrive™ and SnapManager® for Exchange will interact with a filer with no discrimination for the host type, whether it is dedicated hardware or a VM. Accordingly, this method of employing the iSCSI protocol under an ESX server to communicate with a NetApp filer is completely supported. This approach is recommended to simplify LUN management via SnapDrive and related host-based tools that function perfectly for iSCSI in the guest OS but are not otherwise presently able to manage FCP LUNs.

Service Console

A common misconception is that the ESX server “runs Linux.” This assumption is made because the initial boot process looks much like Linux, and the normal interaction that an administrator has with the server is via the service console. While the service console may actually look like an instance of Linux, it is not the core of the ESX server. The distinction is made clear with this simple fact: The service console is actually a special case of a guest OS that is given extra privileges for communicating with the ESX kernel. A customized version of a Linux OS runs in the service console VM, allowing standard UNIX® administration and techniques to be used for management. All underlying hardware virtualization, partitioning, scheduling, etc, is managed by the ESX kernel with requests being passed to and from the service console via loadable modules present in the Linux kernel of the service console.

It is because of this architecture that standard Linux drivers, like those available for download from an HBA vendor for example, are incompatible with ESX. Instead, VMware must adapt each driver for every piece of supported hardware so that it will execute in this environment. While this may place limitations on the range of available hardware, it also ensures strict compatibility and increases overall availability of the ESX server platform.

The service console and the ESX OS, known as VMNIX, share a common root file system. This enables backup and restore of the whole ESX server to be accomplished much like any other Linux host on dedicated hardware via standard UNIX methodologies and software. It is also possible to use the NFS protocol to share storage to the service console from a Network Appliance filer for the purposes of normal file sharing. However, virtual disks that make up storage managed by the ESX server must reside on VMFS volumes on LUN devices. Presently you cannot place them on an NFS file system.

vmware-cmd facility

VMware provides a mechanism for manipulating the operational state of VMs as a part of the command set included in the service console. The tool, known as the “vmware-cmd” enables key features such as suspend/resume, as well as virtual disk manipulation, which can be used to automate and coordinate Snapshot copies for VMs when invoked via shell scripts. A Perl library is also available for more sophisticated tool creation.

3. Assumptions

The methods described in this technical report assume that the reader has some familiarity with managing Network Appliance filers. The reader should also have some mastery of the concepts of Fibre Channel administration on both the filer and ESX server, as well as familiarity with the setup and management of the ESX server, any guest OS, and the application that is intended to run therein.

4. Best Practices

There are a few rules that should be considered when setting up the virtual machine environment on a NetApp SAN.

On the filer, create a new flexible volume for each VM and keep all LUNs from that VM and only LUNs for that VM in this volume. (Since Snapshot operations occur at the volume level, this will allow all of the LUNs to be copied together and restored as a unit.)
On the filer, create one igroup for each ESX server containing the WWPN for all HBAs in the server. (Use the "vmware" or "linux" type with Data ONTAP versions earlier than 7.1.)
On the filer, create a single LUN for each VM's root disk (C: for Windows, / for UNIX):

- Make the LUN at least a few hundred MB larger than the root disk will be–typically the total LUN should be 5 to 10GB per VM.
- Use the LUN type "vmware." (Use the "linux" type with versions of Data ONTAP earlier than 7.1.)
- Map all LUNs to all of the ESX server HBA WWPNs via the server igroup.
- On the ESX server, create a VMFS volume on the LUN.
- On the ESX server, create a "Persistent" type virtual disk (.dsk or .vmdk file) in the VMFS volume.
- On the ESX server, map the new virtual disk to scsi:0.0 as the boot disk for the guest OS.
- In the VM, store only the system binaries and other system files for the guest OS in this LUN.

On the filer, create one or more additional LUNs for application data storage:

- Size the LUN appropriately for the application.
- On the filer, map all LUNs to all of the ESX server HBA WWPNs via the server igroup.
- If you will manage the Snapshot copies of your application data manually, do the following:
  - On the filer, use the LUN type that matches the guest OS type.
  - For each LUN, create a new hard disk device on the ESX server using the "System LUN/Disk" type.
  - On the ESX server, connect the virtual disk to the matching LUN using "Virtual Compatibility" and "Persistent Disk Mode."
  - Store the metadata for the raw device mapping (RDM) in the system disk VMFS.
- If you will snap all application data with your system disk and manage the VM as a unit, do the following:
  - Make the LUN on the filer a few hundred MB larger than the application requirements.
  - On the filer, use the LUN type "vmware." (Use the "linux" type with versions of Data ONTAP earlier than 7.1.)
  - On the ESX server, create a VMFS volume on the LUN.
  - On the ESX server, create a "Persistent" type virtual disk (.dsk or .vmdk file) in the VMFS volume.
  - On the ESX server, map the virtual disk to the next virtual SCSI device port in the VM.

On the ESX server, identify and choose a nonproxy path for each LUN. Set the failover policy for that path to "Fixed." (Note that MSCS systems may function more efficiently with the policy set to "MRU.")
For load balancing, alternate the preferred path for each LUN on the ESX server so that access is distributed across all available HBAs.

In principal, there are two methodologies for handling FCP LUNs used for application data inside of a VM:

1. The first method of management treats the application data storage devices as a separate entity from the system disk storage. The pass-through method is utilized for each nonsystem disk. To effectively create Snapshot copies of application data, the application must be managed within the guest OS to suspend write operations before the copy is created on the filer. Snapshot copies of the system disk can be taken using any of the mechanisms described in the next section, but application data in those copies cannot be guaranteed to be in any better condition than a crash recovery can provide. Manual coordination will be required separately to take consistent Snapshot copies of the application data. Fortunately, with pass-through devices utilizing RDM there is no buffering at the ESX server level, so any method of write flushing that is normally available in the guest OS will have the appropriate effect on the data in the LUN. Examples of such methods are "sync" on a Linux host or VSS in a Windows 2003 host. Most file systems such as NTFS and EXT3 use block journaling to protect against file system corruption, so using them minimizes the risk of losing data during uncoordinated Snapshot operations.

2. Following the second method, the entire VM is treated as an atomic system. All disk operations are inherently cached at some level, whether it is inside of the application, the operating system, or both. The difficult aspect for managing snapshots of LUN devices is that there may be metadata for the file system, as well as blocks for files in that file system, that should be flushed to disk in the filer before the Snapshot copy is taken. Within the guest OS, it is generally possible to ready the application to take a snapshot of its data by entering a state where the I/O is suspended and outstanding writes are flushed. Since the LUN is actually owned by the ESX server, a method of signaling to the VMNIX that this has occurred would be necessary so that any pending writes cached by the ESX server itself could also be flushed. Unfortunately, VMware does not provide such a communications channel today. The simpler method is to copy the entire VM as a whole, including a dump of the active kernel and application memory. In order to facilitate this properly, it is necessary to ensure that all LUN storage is managed by the ESX server via VMFS volumes and virtual disk files. This method will be discussed in detail later in this report in the section covering "warm" Snapshot copies. Using this method, complete snapshot copies can be automated and scheduled entirely within the service console, and no coordination is required with the guest OS.

Making use of iSCSI LUNs via software initiator inside of the guest OS requires no special handling when integrated with a VM infrastructure. As such, this approach will not be covered in detail in this report. If additional information is required, refer to the standard Network Appliance iSCSI documentation and procedures for dedicated servers.

5. Snapshot Operations

Network Appliance Snapshot technology can be utilized underneath a VM in a number of ways. The three most common techniques will be outlined in this section. Each has a different level of impact on the clients of the server and the way in which recovery actions are executed. Sample scripts depict an example of the logic flow required for Snapshot automation.

The following table shows the relative differences between the different techniques:

	Cold	**Warm**	**Hot**
Service Impact	Complete shutdown of application and server	Momentary suspension of processing	Imperceptible
Snapshot File System Contents	Closed and unmounted	Point in time Cached I/O completely resumed	IO flushed up to point in time File systems unaware
Recovery Process	Cold boot guest OS	Guest OS resumes from point in time	Guest OS performs system crash recovery

The trade-off between cold, warm, and hot snapshots can best be characterized by the availability requirements for the service that the guest OS provides. If there is time in the normal schedule for a complete shutdown, a cold copy may be the best method. But if, for example, there is a high-availability SLA for a particular server and even a few minutes of downtime is unacceptable, a hot copy may be the best method. On balance with availability is the grace at which a recovery can be performed. These topics will be discussed further in the sections below.

5.1. Cold Snapshot Copies

Taking a cold Snapshot copy is the safest way to guarantee that a system can be completely recovered. It is normal to take a cold copy just before and just after any large configuration changes or maintenance processes to allow for a complete rollback. A cold copy may also be combined with FlexVol™ cloning to replicate an entire VM for a variety of purposes, such as scaling, making a copy of a production system available to test/development, and migrating to tier 2 storage.

Initiation

The process of producing a cold Snapshot copy is as follows:

1. Shut down all applications running in the guest OS.

2. Shut down the guest OS.
3. Power off the VM.
4. “Sync” the ESX within the service console.
5. Initiate a Snapshot copy of the appropriate volume on the filer.
6. Power on the VM.
7. After the guest OS has booted, start up all applications.

Recovery

To recover from a cold Snapshot copy, perform the following actions:

1. Power off the VM.
2. Put the associated LUNs offline.
3. Initiate the volume SnapRestore® operation on the filer.
4. Bring the associated LUNs back online.
5. Power on the VM.
6. After the guest OS has booted, start up all applications.

5.2. Warm Snapshot Copies

A warm Snapshot copy makes use of the capability to suspend a guest OS that is provided by the ESX server. When the suspend action is performed, the program counter for the VM CPU is stopped, all active memory is saved to a .vmss state file inside the VMFS where the boot disk resides, and the VM is paused. At this point in time, a Snapshot copy can be taken of the entire VM, including the memory contents file and all LUNs with the associated active file systems. In that copy, the machine and all data will be frozen at the point in processing when the suspend operation was completed.

Since all of the contents of memory, the active virtual CPU registers, and the associated state are written to a file, the next instruction to process when the system resumes, as well as all memory data structures such as the I/O queue and any applications that were running, is included in this information.

When the Snapshot action is complete, the VM can be restarted and will resume at the exact point where the suspend action was initiated. The guest OS will continue processing, including handling I/O activity with any in memory transactions that were loaded from the .vmss file. Applications and server processes will resume processing from the same point in time. The outward appearance is as if a pause button had been pressed for the duration of the Snapshot activity. To any network client of the guest OS server, it will appear that there was a temporary interruption of network service. Usually this is on the order of 30 to 120 seconds for moderately loaded servers.

Recovery from a snapshot will return the guest OS and all processes to the exact moment in time when the suspend process was initiated. It is important to recover all LUNs that are part of the VM as a unit in order for consistency across the connected file systems to be maintained.

Initiation

The process of producing a warm Snapshot copy is as follows:

1. Suspend the VM.
2. “Sync” the ESX within the service console.
3. Initiate a Snapshot copy of the appropriate volume on the filer.
4. Resume the VM.

The following script shows an example of one way to automate the process. This would be run from a terminal session logged into the service console, or it can be scheduled via `cron`. In this example, the filer is named `F960-1` and the volume for the VM is `esx_vm1`. The `VM_CFG_PATH` refers to the configuration file for the particular VM on which this script will operate, which in this case is an instance of Windows 2003 Enterprise Server. The actual guest OS type does not matter.

```
#!/bin/sh
#
# take-warm.sh
#
```

```
# This script will take a snapshot of a running VM using suspend/resume
# capability provided with the vmware-cmd facility. It will maintain and
# cycle the last 4 snapshots.
#
# NOTE: VMware ESX does not ship with the rsh utility. The Rsh protocol is
# an inherently insecure mechanism for remote computing, especially outside
# of a secure trusted environment. Alternate mechanisms utilizing ssh and
# the perl based ONTAPI library are available for adaptation inside of a
# script like this one which may be developed for production-quality
# implementation. For the purposes of this demonstration, a viable copy of
# the rsh binary was obtained from a running instance of Linux and is used
# herein to facilitate a quick and easily readable code example.
#
# This sample code is provided AS IS, with no support or warranties of any
# kind, including but not limited to warranties of merchantability or
# or fitness of any kind, expressed or implied.
#
# 03.17.2005 Brian Casper, Network Appliance Technical Marketing Engineer
#
# --------------------------------------------------------------------------

PATH=$PATH:/bin:/usr/bin

# filer where the LUNs reside
FILER=F980-1
VOL=esx_vm1
# the VM config file
VM_CFG_PATH=/root/vmware/winNetEnterprise/winNetEnterprise.vmx

echo ' Starting ---> '`date`

# rename and delete old snapshots
rsh -l root $FILER snap delete $VOL old3
rsh -l root $FILER snap rename $VOL old2 old3
rsh -l root $FILER snap rename $VOL old1 old2
rsh -l root $FILER snap rename $VOL new old1

# suspend the running VM
vmware-cmd $VM_CFG_PATH suspend soft

# flush out cached IOs from the ESX server
sync

# take a new snapshot of the volume
rsh -l root $FILER snap create $VOL new

# resume the suspended VM
vmware-cmd $VM_CFG_PATH start soft

echo ' Finished ---> '`date`
```

An example of output from the execution of this script is as follows:

```
[root@ibmx345-rtp01 snap_tools]# ./take-warm.sh
 Starting ---> Sun Mar 20 17:21:03 EST 2005
deleting snapshot...
suspend(soft) = 1
creating snapshot...
start(soft) = 1
 Finished ---> Sun Mar 20 17:21:39 EST 2005
```

Total execution time for this example is 36 seconds of real time, which is comprised of the time required to roll the Snapshot activity, copy the guest OS RAM image to disk, suspend the VM, take the new copy, load the RAM contents from disk, and resume operations. Service time on a more heavily loaded server is likely to be longer; however it is not likely to exceed the two-minute barrier for even excessively loaded machines.

Recovery

To recover from a warm snapshot, perform the following actions:

1. Power off the VM.

2. Take the associated LUNs offline.
3. Initiate the volume SnapRestore operation on the filer.
4. Bring the associated LUNs back online.
5. Power on the VM.

Utilizing this method of recovery will produce a guest OS that will restart and continue from the exact moment in time when the suspension was initiated. Recovery of the application to the state that may have existed after that point in time, such as additional transactions to a database, should be performed as a postprocess to the Snapshot recovery action.

5.3. Hot Snapshot Copies

Virtual disks that contain the boot partition for a guest OS should be configured as “Persistent” files inside of a VMFS. In that state, all writes that occur are immediately applied to the virtual disk to maintain a high level of consistency in the file system. VMware provides a facility to place a persistent virtual disk into a hot backup mode for the purposes of taking Snapshot copies at the disk subsystem layer by adding a REDO log file. When the REDO log file mode is initiated, all outstanding writes are flushed to the virtual disk file and all future writes are written as log data into the REDO file. Much like putting a database into hot backup mode, the guest OS root disk or any other virtual disk is placed into hot backup mode with this operation.

Note: It is not possible to add a REDO file to a RDM pass-through device configured in physical compatibility mode, so they should be avoided if the intention is to fully automate the Hot Snapshot mechanism. Instead, create a VMFS in each LUN and provision a virtual disk for each LUN as a separate SCSI device to the VM or use virtual compatibility mode.

Once the REDO log has been activated, it is safe to take a snapshot of the LUN containing the VMFS. After the snapshot operation is complete, another command can be issued which will commit the REDO log transactions to the underlying virtual disk file. When the commit activity is complete, all log entries will have been applied and the REDO file will be removed. During this operation, a marginal slowdown of processing will occur, but all operations will continue to function. It is possible to stack a secondary REDO file behind primary REDO in order to smooth the application of log entries via staging. In most cases, however, the Snapshot process is nearly instantaneous and the time between create and commit for the REDO is minimal. Maintaining as short a window as possible will prevent excessive transactions from queuing in the REDO log, which will minimize the external impact of the commit phase of the operation.

An artifact of creating the Snapshot copy with a REDO log engaged is that an empty REDO file or one that contains some small amount of write transaction data may occur between the times when the REDO file was added and when the copy takes place. When recovering from this type of copy, as the VM transitions to the running state, it will find the REDO file and prompt the user to decide whether the REDO should be applied or skipped. It is wise to skip the application of these transactions because they are data that was created at a point in time after the decision was made to take a Snapshot copy. There is no guarantee that the extraneous data is complete or particularly useful since the REDO file is in a “fuzzy” state when the copy is taken.

Since the Snapshot copy is taken of a “running” process, but without any provision for dumping the contents of RAM in the manner that the warm Snapshot copy maintains, recovery of a hot Snapshot copy will appear to the guest OS as if it is cold booting after a total loss of power. Normal crash recovery procedures should be followed (i.e., chkdsk/fsck) to ensure the consistency of the boot or other data disks is maintained. Applications running during a hot Snapshot operation will also require media or transaction recovery as appropriate for the software product. If possible, it is best to prepare application data where possible for the hot Snapshot operation, using the same process that would be followed on a dedicated (i.e., nonvirtual) server.

The outward appearance during a hot Snapshot operation is an imperceptible server slowdown. At worst, it will appear that network congestion or an overloaded CPU may be causing general server sluggishness. At best, there is no noticeable impact. It is important to recover all LUNs that are part of the VM as a unit in order for consistency across the connected file systems to be maintained. This is why it is recommended to keep all LUNs from each VM in a single FlexVol volume on the filer.

Initiation

The process of producing a hot Snapshot copy is as follows:

1. Add a REDO log for each virtual disk.
2. Place guest OS applications into “hot backup” mode.*
3. “Sync” the ESX server within the service console.
4. Initiate a Snapshot copy on the filer.
5. Commit the REDO change log for each virtual disk.
6. Return guest OS applications to regular run mode.*

*These steps may be performed manually as required.

The following script shows an example of one way to automate the portion of the process that can be handled within the service console. This can be run from a terminal session logged into the service console, or it can be scheduled via `cron`, etc. In this example, the filer is named `F960-1` and the volume for the VM is `esx_vm1`. The `VM_CFG_PATH` refers to the configuration file for the particular VM on which this script will operate, which in this case is an instance of Windows 2003 Enterprise Server. The actual guest OS type does not matter.

```
#!/bin/sh
#
# take-hot.sh
#
# Example code to take a snapshot of a running VM using REDO log capability
# provided with the vmware-cmd facility. It will maintain and cycle the
# last 4 snapshots.
#
# NOTE: VMware ESX does not ship with the rsh utility. The Rsh protocol is
# an inherently insecure mechanism for remote computing, especially outside
# of a secure trusted environment. Alternate mechanisms utilizing ssh and
# the perl based ONTAPI library are available for adaptation inside of a
# script like this one which may be developed for production-quality
# implementation. For the purposes of this demonstration, a viable copy of
# the rsh binary was obtained from a running instance of Linux and is used
# herein to facilitate a quick and easily readable code example.
#
# This sample code is provided AS IS, with no support or warranties of any
# kind, including but not limited to warranties of merchantability or
# or fitness of any kind, expressed or implied.
#
# 03.17.2005 Brian Casper, Network Appliance Technical Marketing Engineer
#
# ---------------------------------------------------------------------------

PATH=$PATH:/bin:/usr/bin

# filer where the LUNs reside
FILER=F980-1
VOL=esx_vm1
# the VM config file
VM_CFG_PATH=/root/vmware/winNetEnterprise/winNetEnterprise.vmx

echo ' Starting ---> '`date`

# rename and delete old snapshots
rsh -l root $FILER snap delete $VOL old3
rsh -l root $FILER snap rename $VOL old2 old3
rsh -l root $FILER snap rename $VOL old1 old2
rsh -l root $FILER snap rename $VOL new old1

# set up the REDO log file on the boot drive
vmware-cmd $VM_CFG_PATH addredo scsi0:0

# flush out cached IOs from the ESX server
sync

# take a new snapshot of the volume
rsh -l root $FILER snap create $VOL new

# apply the REDO and get rid of the log file
# <freeze>=0 <level>=0 <wait>=1
vmware-cmd $VM_CFG_PATH commit scsi0:0 0 0 1

echo ' Finished ---> '`date`
```

A sample output from the execution of this script is as follows:

```
[root@ibmx345-rtp01 snap_tools]# ./take-hot.sh
 Starting ---> Sun Mar 20 18:07:30 EST 2005
deleting snapshot...
addredo(scsi0:0) = 1
creating snapshot...
commit(scsi0:0 0 0 1) = 1
 Finished ---> Sun Mar 20 18:07:33 EST 2005
```

Total execution time for this example is 3 seconds of real time, which is the time required to roll the Snapshot operation, create the REDO log file, take the new Snapshot copy, commit the REDO log, and resume operations. During these 3 seconds, the server was up and running and able to service client requests. Service time on a more heavily loaded server is likely to be longer and will vary based on the total load generated during the REDO period.

Recovery

To recover from a hot Snapshot operation, perform the following actions:

1. Power off the VM.
2. Take the associated LUNs offline.
3. Initiate the volume SnapRestore operation on the filer.
4. Bring the associated LUNs back online.
5. Power on the VM.

Utilizing this method of recovery will produce a guest OS that will begin with a cold boot when the VM is powered on. The first action may be an fsck or chkdisk of the system disk and any other disks that are configured in the VM. With the use of journaling file systems such as NTFS or EXT3, the likelihood of file system corruption is minimized during the boot recovery phase.

Once the guest OS has completed the boot process, appropriate data checks should be completed for the data files that are used by any application on the system before the application is started. Recovery of the application to the state that may have existed after that point in time, such as additional transactions to a database, should be performed as a postprocess to the Snapshot recovery action.

6. Special SnapRestore Operations

The scenarios described in the preceding sections are useful for restoring an entire VM as a unit. In some instances, it may be necessary to restore only part of a VM or to gain access to a specific file inside of a file system that is owned by a guest OS that lives inside a LUN. The next section will discuss methods of obtaining access to the data inside a LUN for these special purposes. In any of these cases, Snapshot technology, combined with the ability to create clones of LUNs that do not require 100% space overhead, will be utilized to simplify the recovery process.

LUNs Backed by Snapshot Copies via LUN Cloning

The first step in the process is to create a new LUN on the filer. In this example, we will use one of the Snapshot copies created during the steps outlined above. The procedure is as follows:

1. Obtain a list of available Snapshot copies.
2. Create a new temporary LUN using the copy as backing store.
3. Map the LUN to an ESX server for recovery.
4. Perform the “Rescan SAN” operation on the ESX server or service console.

The first step can be accomplished by using the filer CLI:

```
F980-rtp01> snap list esx_vm1
Volume esx_vm1
working...

 %/used %/total date name
---------- ---------- ------------ --------
 1% ( 1%) 0% ( 0%) Mar 20 22:21 new
 6% ( 5%) 1% ( 1%) Mar 20 22:08 old1
 10% ( 5%) 2% ( 1%) Mar 20 22:07 old2
 10% ( 0%) 2% ( 0%) Mar 20 22:07 old3
```

Next, we will create a clone of the LUN:

```
F980-rtp01> lun clone create /vol/esx_vm1/the_clone -o noreserve -b /vol/esx_vm1/winlun new
```

This procedure has now created an exact duplicate of the primary LUN that utilizes all of the same blocks on the filer as the original LUN. Note that the `-o noreserve` option has been selected since the intention is to use this LUN for an immediate recovery need, which upon completion will be destroyed. Using this option will not provide any space guarantee for the temporary LUN if a future Snapshot copy is taken while this temporary LUN exists. Also, the "new" copy is locked from deletion until this clone LUN is destroyed. If the requirement is to keep the LUN clone around permanently, then the `-o noreserve` option should not be used, and the `lun clone split start` *`lunpath`* command should be issued to create a new standalone copy of the LUN and unlock the Snapshot copy.

When a VMFS volume is created on a LUN, the ESX server generates a unique serial number for the VMFS. Present limitations in the ESX server do not allow more than one copy of a VMFS to be mapped to a single instance of ESX at a time. If this happens, access to the VMFS will be restricted and the LUN will be placed into "Private (deprecated mode)" on the ESX server. To work around this, two options are available:

Swap the LUNs and present only the clone copy. The following process can be used as a guideline:

1. Stop application and other access to all files in the file system.
2. Unmount the file system or remove the drive letter if possible and stop the volume at the guest OS level.
3. Take the primary LUN on the filer offline.
4. Map the clone LUN to the ESX server at the same LUN ID as the primary LUN.
5. Remount the file system or restart the volume at the guest OS level.
6. Copy the required files out of the temporary storage to other storage on the VM.
7. Restore the original storage:
    - Unmount fs/remove drive letter/stop volume on the clone,
    - Unmap the clone,
    - Bring the primary LUN back online.
    - Mount fs/add drive letter/start volume.
8. Destroy the clone LUN.

Map the clone to another physical ESX server. The following process can be used as a guideline:

1. Map the clone to a different ESX server that has no access to the primary LUN.
2. Rescan SAN with the ESX MUI or service console.
3. Assign the virtual disk device in the VMFS to a new or existing VM of like guest OS type.
4. Mount the file system inside the guest OS.
5. Share the file system over the network to the original VM that needs access to the original files.
6. Copy the required files.
7. Remove the virtual disk device from the surrogate VM.
8. Unmap the clone LUN from the ESX server.
9. Destroy the clone LUN.

This process is simpler if the LUN is accessed via the RDM method because there are no restrictions on where the clone LUN can be mapped. To access the LUN directly by the guest OS, which owns the primary storage, follow this process:

1. Map the clone to the same ESX server with a new LUN ID.
2. Rescan the SAN with the ESX MUI or service console.

3. In the VM configuration, create a new hard disk using the “System LUN/Disk” type.
4. Mount the file system inside the guest OS.
5. Copy the files back into the original location.
6. Remove the virtual disk device from the VM.
7. Unmap the clone LUN from the ESX server.
8. Destroy the clone LUN.

7. Other Considerations

In addition to the standard Fibre Channel methodologies that have been discussed in this technical report, other Network Appliance technologies can be used to access and protect storage when used with an ESX server. A creative systems architect may devise any number of solutions that meet business needs by combining many of the features available. A few of the more popular solutions can be built using the technologies described in this section.

SnapMirror®

The underlying technology behind a SnapMirror is Snapshot. Although the discussion in this technical report is centered around Snapshot copies, once a valid copy has been created it is also possible to replicate that data to another filer via the standard SnapMirror process for LUNs. SnapMirror enables disaster recovery architectures to be realized for ESX server scenarios. Once the LUNs have been brought online at the remote location, a new VM on the remote ESX server can be build from the existing virtual disks. The guest OS in this new VM will boot into the exact image of the original VM at the primary location.

Included in the mirrored image are the hostname, IP address, and other unique server and application identifiers. Special consideration should be given to customizing the configuration at the remote site just like the normal process that would be followed for dedicated servers. However, because the OS, the application, and the application data are typically stored in a FlexVol volume that is mirrored as a unit, starting the host and application may actually be as simple as performing a power-on operation if the network configuration matches.

NFS, CIFS, and Multiprotocol Access

Like iSCSI, the content of all network traffic that traverses the shared network adapters is transparent to the ESX server. Network storage shared via the NFS or CIFS protocols may be mounted by a guest OS just as it would on an OS running on dedicated hardware. In some instances, the process of taking Snapshot copies of application data may be simplified when utilizing NAS protocols since the file system ownership is maintained by the filer. Accordingly, all of the well-known benefits provided by NetApp filers utilizing NAS protocols can be leveraged by the guest OS on an ESX server. The multiprotocol capability of the filer to simultaneously provide NAS and SAN storage can be employed in an ESX architecture granting access to the best of both storage worlds.

8. Summary

Snapshot technology makes quick and efficient work of creating backups of virtual machines running in the VMware ESX server environment. Using simple tools, the process can be automated to allow for trouble- and worry-free data protection.

This paper is not intended to be a definitive implementation guide. Many factors are not addressed in this document. Expertise may be required to solve logistical problems when the system is designed and built. NetApp has not tested this procedure with all the combinations of hardware and software options available on all ESX or guest OS variants. There may be significant differences in your configuration that will alter the procedures necessary to accomplish the objectives outlined in this paper. If you find that any of these procedures do not work in your environment, please contact the author immediately. Comments on this technical report are welcome. Please contact the author here.

9. References

A guide to Managing NetApp Filers in SAN Environments:
Data ONTAP 7.0 Block Access Management Guide (NOW access required).

FlexVol and FlexClone™:
Flexible volume and data replication technology for optimized utilization and simplified management.

Related Best Practices:

Best Practices Guide for Data Protection with Filers Running FCP

More information about All Aspects of the Installation of the ESX Server Software:

VMware ESX Server 2 Installation guide

More Information about All Aspects of the Operation and Configuration of the ESX Server:

VMware ESX Server 2 Administration guide

Information Regarding the "Manage ONTAPI SDK," Which Provides Perl Library Access to Manage Filers via XML:

http://www.netapp.com/solutions/dfms/manage-ontap.html

NetApp

Network Appliance, Inc.
495 East Java Drive
Sunnyvale, CA 94089
www.netapp.com