VMware® Infrastructure 3

Advanced Technical Design Guide

~and~

Advanced Operations Guide

Two books in one!

Ron Oglesby
Scott Herold
Mike Laverick

Chapter 12: ESX Command-Line Configuration (including installation)

About three and half years ago when I first picked up ESX I knew nothing about the command-line environment of Linux. Although I grew up on DOS and Netware, my command-line skills had weakened about the time I started to use Windows 95 and NT4. So the old adage holds true: use it or lose it. It's odd how some of the things I used to teach students, like .bat files and text manipulation, come in handy with ESX on the command-line, too. As they say, "What goes around comes around."

Anyway, at the end of 2003, I bought a secondhand book on Redhat Linux 9.2 (the version used by the Service Console in ESX 2.0) from EBay, and began to learn the basics you need to mange any operating system; how to navigate the file system, how to copy files around, how to edit a text file, how to fdisk and format a disk. I still would not regard myself as a Linux guru, after all my operating system is ESX, not Linux, but I know enough to get around and do my job. What I don't know – I can learn by using this thing called the Internet. I frequently ask my buddies on the forum about Linux stuff which I would regard myself as being a bit "fuzzy" on. So the moral of the story is, if you're a Windows guy – then have no fear. Commands are just words you type to carry out actions. What you don't know – you can learn because you're not stupid. But perhaps it would be a good idea to brush up your command-line skills before jumping head first in the ESX Service Console.

There are two main reasons to come to grips with the command-line environment of ESX: the Service Console. Firstly, it can be an excellent troubleshooting tool. Indeed, it might be your only option for troubleshooting if you cannot create a connection between the VI Client on your workstation and your ESX host. Secondly, learning the command-line environment could be useful for automation purposes – scripting laborious tasks especially in the area of ESX host installation and configuration.

There are a couple of caveats surrounding this chapter. Firstly, although the Service Console is based around Redhat Linux I do not intend to write a complete guide to Redhat Linux; rather we intend to drill down into very specific

ESX host tasks that largely have nothing to do with Linux. I would recommend that if you are a complete novice with Linux then purchase a book on Redhat Linux to become familiar with the basic tasks such as moving, copying, and deleting files. There are lots of free tutorials on the Linux command-line from the web – and in no time at all you will be restarting services and killing processes just as you would with any other operating system. A more complete and longer guide to the command-line is available for free from www.rtfm-ed.co.uk. This chapter was taken from that original guide and improved to make automating the ESX process more readable and focused.

Secondly, while the command-line environment surrounding ESX is very rich, it may not be the appropriate environment for certain scripting tasks. For example, if your goal is to write a script that manipulates and manages a VM, writing an ESX script will not be the best way to begin. As we saw in the previous chapter, the location of your VMs could be constantly changing because of VMotion, DRS, or HA events. In this case you would be best placed to investigate the programming environment of VirtualCenter's Software Development Kit (SDK).

The structure of this chapter is taken from the overall structure of the operations guide. We will begin with a look at automating the installation of ESX using scripts. At the end of the installation we will look at the commands used to create and configure vSwitches and iSCSI and NAS based storage. The goal will be an attempt to configure an ESX host from a blank machine to the point that it could be added to VirtualCenter and used to create VMs. At the end of the chapter there will be a roundup of some of the most useful commands used for troubleshooting and interacting with VM's from the command-line.

If you are reading this book in preparation for the VI-3 test, don't worry. We have heard no reports stating you need a strong familiarity with the command-line. This chapter is merely here for your knowledge – as a little knowledge is dangerous thing.

Try not to feel too intimidated by this chapter. The generation that grew up with graphical interfaces feels put off by command-line environments. Remember, commands are merely words that carry out instructions. Personally, I see both graphical and command-line tools as dangerous in the wrong hands. There is nothing intrinsically safer about a graphical environment – in fact, fa-

miliarity is often their weakest element. When was the last time you clicked OK, when you should have clicked cancel?

Scripted Installation of ESX

There are many reasons to automate the installation of ESX. Firstly, installing to blades using CDs can be a challenge. Secondly, automated installs guarantee consistency; this is especially useful if you have datacenter policies for partitioning and for ensuring consistently named port groups which are required for VMotion, DRS, and HA. Lastly, automating the ESX install will allow you to quickly rebuild a system if you are in a DR scenario.

There are many ways of automating ESX's installation. ESX uses the "Anaconda" installer popular with many Linux distributions. It can be automated with a configuration file containing "kickstart" (KS) commands. The automated install can use the source code via the physical CD-ROM, Floppy boot disk or remotely by FTP, NFS, or Pre-Execution Environment (PXE). I've decided to put the emphasis on PXE booting because in my experience people doing automated deployments often desire "diskless" installation where neither CD or floppy media is required – and wish to build an environment where the build they wish to deploy can be selected from a menu.

Another way of approaching this is to use your hardware vendor's ESX deployment tools such as IBM RDM or HP RDP. For those people with HP hardware an excellent site which covers the usage of HP RDP for ESX 2.x and 3.x is:

http://www.brianshouse.net/hp/

We will use a popular PXE virtual appliance downloadable from VMware's Virtual Appliance directory. This will save you considerable time configuring all the appropriate services and files required to set up a PXE boot server in a Linux environment. I decided to use the Ultimate Deployment Appliance v1.3 (UDA) as PXE server for a number of reasons. Firstly, it is very small to download. Secondly, it offers a way of serving up the files from the ESX ISO without manually copying files. Thirdly, it has an easy to use GUI which allows you to reconfigure the appliance with little Linux knowledge. Lastly, although

the appliance is not geared up to work with ESX 3, with a few modifications it can be re-engineered to do so – and it can also be used to deploy a whole range of other operating systems including Windows, Linux (Fedora, Ubuntu, Suse), and Solaris X86. UDA was built to run on "free" virtualization with either VMware Server or VMware Workstation and a Windows server is needed to host the ISO files.

You can download UDA and read information about its development here:

http://www.vmware.com/vmtn/appliances/directory/232

Additionally, we can use ESX's own web-based wizard to generate a "base" kickstart.cfg file which we can modify with additional parameters to customize your installation.

GOTCHA:

Later in this chapter we will see how we can modify the kickstart script to handle the post-configuration of the ESX host. There are some limits to what we can achieve through the command-line. There are two notable restrictions. It is not possible to enable VMotion on a VMkernel port group using the ESX host "esxcfg-vswitch" command.

Additionally, although we can set the name of the license server to the ESX host, we cannot set the Host Edition level. So an ESX host remains unlicensed until enabled for Starter or Standard Edition in the VI Client.

Configuring the Ultimate Deployment Appliance (UDA)

LATE TO PRESS:

I have been liaising with Carl Thijssen, the creator of UDA. Carl has agreed to create templates for ESX, and introduce some features which I feel will greatly improve his appliance. UDA 1.4 is currently in beta but promises to include such features as:

- *A kickstart creator for ESX 3*
- *Local mounted ISOs which will dispense with the need for a Windows File Server*
- *The ability to set which NIC is used for the installation*
- *To hard-code kickstart scripts to servers by MAC Address*

I hope to add a PDF to vi3book.com covering the new features of UDA 1.4 when it is fully released.

UDA contains all the services and daemons required to do a PXE installation. It has a static IP address (10.0.0.104) and contains a DHCP scope for issuing DHCP request to PXE clients. Using its graphical web-interface it allows you to mount ISO images held on Windows/SMB shares and create a menu of pre-defined scripted installations using the Fedora Core 5 template.

Its configuration for our purpose involves 7 stages:

- Share out a Windows Folder containing ESX ISO images
- Download, Power On, Reset Passwords
- Reconfigure IP and DHCP Scope
- Setup Mount Point and Select ISO to Mount
- Create a Fedora Core 5 Template

Windows Shared Folder

This server should be in the same IP range as the UDA and ESX hosts. It could be the very same machine that is running the UDA appliance:

1. On a Windows File server setup a Windows share which contains the ESX 3 ISO.

   Note:

   You could also add additional ISO's supported by UDA if you so wish.

2. Click the Permissions button in the Sharing Tab.
3. Remove Everyone/Read, and Set the Permissions to be Full Control for an individual user account.

   Note:

   This account must be "local" to the Windows File Server. In other words, it cannot be an Active Directory or NT4 Domain User account.

Download, Power-On, and Reset Passwords

1. Download the Appliance and Extract the VMDK files.
2. Using VMware Workstation or VMware Server build a new VM using the "Existing Disks" option.
3. Power on the UDA VM.
4. Login to the Appliance with root using test as the password.
5. Change the password of root with the passwd command.
6. Open a web-browser on http:/10.0.0.1

   Note:

   You will need to give yourself a temporary 10.x.y.z address to communicate to the UDA if you don't use this range on your existing network.

7. Click the Web-Interface link and login as admin using admin as the password.
8. Click the password link, and reset the admin password.

   Note:

   You will find you have to logon again with the new password.

Reconfigure IP and DHCP Scope

TIP:

I found having my ESX hosts, my Windows File Server, and the UDA all on the same network made troubleshooting network problems much easier.

1. In the web-appliance, click the Network link.
2. Change your IP address, subnet mask, and default gateway as appropriate.

   Note:

   After clicking OK you will be disconnected, and you will have to connect under the new IP address.

3. Click the DHCP server link; replace the IP data with information valid for your network. Below is a sample of my DHCP file:

```
ddns-update-style ad-hoc;
if substring ( option vendor-class-identifier,
0, 9) = "PXEClient"
{
  filename "pxelinux.0" ;
  next-server 192.168.3.150 ;
}
subnet 192.168.3.0 netmask 255.255.255.0 {
option routers 192.168.3.150;
option domain-name-servers 192.168.3.130 ;
range 192.168.3.20 192.168.3.30 ;
max-lease-time 300;
}
```

   Note:

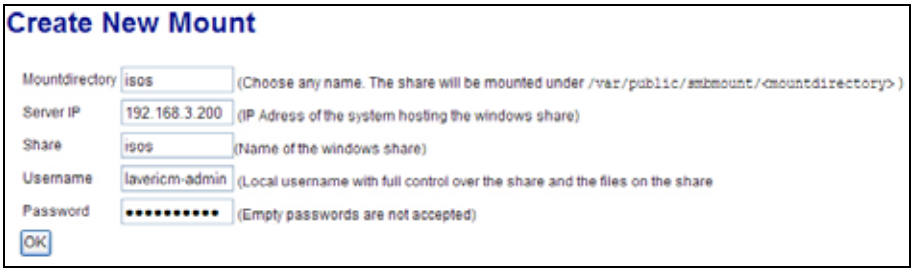
   Next-server is the IP address of the UDA appliance, not the Windows file server.

4. Click OK. UDA should stop and restart DHCP service.

Setup Mount Point and Select ISO to Mount

1. Click the **Mount Points link** at the top of the page.
2. Choose **Create New Mount** – fill in the form to access the share. Figure 12.1 shows my settings.

Figure 12.1



Note:

I found that UDA prefers a RAW IP rather than hostname or FQDN, even when name resolution was setup using a host's file on the UDA itself. After clicking OK you should receive an OK-DONE message. If not re-check your IP, share names, and user-name and password. Clicking Mount Point again will show you the shares you have currently mounted.

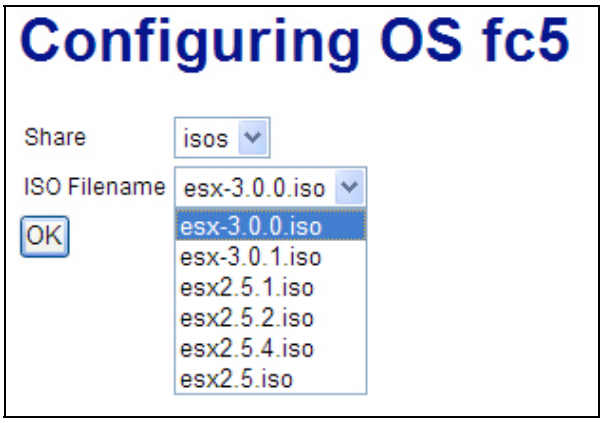
3. Next in the web-admin tool, **click the OS link.**

Note:

As you can see, currently UDA 1.3 does not yet have predefined template for ESX 3. I found that Fc5 – Fedora Core 5 worked just fine.

4. Click the **Configure link for Fedora Core 5**, and in the following page **select an ISO from your mounted share**. Figure 12.2 shows my list.

Figure 12.2



5. After clicking **OK**, you should find that in the OS page it now reads "**Mounted**" in green.

Create a New Template

1. Next click the Template link.
2. Click Create New Template.
3. Type in a friendly templateID name.

   Note:

   This must be exactly 5 characters in length, so something like ESX01 or RTFM1 will work.

4. Choose from the OS type Fc5 (Fedora Core 5).
5. Type in a description such as: Dell: esx1.vi3book.com

Enabling ESX Scripted Installer

UDA is ready to do its work – but the problem is the KS template it created is only valid for Fedora Core 5. We need to replace it with a KS file which is valid for installing ESX 3.

As mentioned earlier, ESX has its own wizard for creating the kickstart.cfg file. It is a web-based tool, but it is not enabled by default on the ESX host. You must first edit a configuration file (struts-config.xml) on the ESX server to en-

able it. If you don't make this change, the following message appears when you try to access the web-page:

"Scripted Install is disabled

Message: Your ESX Server is not configured to support scripted installations. To support scripted installations, please refer to the VMware Web Access Administrators Guide."

1. Open a command-line view on an ESX 3.x host and change in the directory where the struts-config.xml is located:

   ```
   Cd /usr/lib/vmware/webAccess/tomcat/apache-
   tomcat-5.5.17 /webapps/ui/WEB-INF/
   ```

2. Make a backup of the configuration file with

   ```
   cp struts-config.xml struts-config.xml-backup
   ```

3. Edit the configuration file with

   ```
   nano –w struts-config.xml
   ```

4. **Comment out the line** that begins

   ```
   <action path="/scripted install ….  Dis-
   abled.jsp
   ```

   Note:

   You comment things out by beginning the comment <!—and ending the comment with -->. If you read the line just above, it will tell you why we are doing this. Basically this stops the warning message outlined above.

   ```
   <!–Note : Please comment the line below, if
   enabling Scripted Install Functionality.  -->
   ```

5. **Next remove the comments to enable the line that begins just below** with

   ```
   <!---
   <action path="/scriptedInstall"
   ```

```
type="com.vmware.webcenter.
scripted.ProcessAction">
```

and ends with

```
<forward name="scriptedInstall.form7"
path="/WEB-INF/jsp/ scriptedInstall/form7.jsp"
/>
</action>
-->
```

6. **Save the file** and **exit nano.**
7. **Restart the webAccess service** with

```
service vmware-webAccess restart
```

Creating a base ESX KS.cfg File

Depending on your selection, the web-based wizard displays 6 pages, each with a specific purpose. The layout and design of these web-pages are not completely intuitive – but they do work. For example, you are asked about licensing twice on three separate pages rather than just one (one page for the EULA, another for the license mode and third for your license server name and port number). However, this said, the scripted installer web-pages does a great job of creating a base kickstart file which we can modify for our purposes:

- Page 1: Deployment Method, Time zone, root password
- Page 2: Input IP Settings of ESX Host
- Page 3: Accept EULA
- Page 4: Disk Partition Scheme and Licensing Mode
- Page 5: Licensing Server Information
- Page 6: Download completed kickstart file

1. Open your web-browser to the URL of your ESX Host.
2. Click the hyperlink called Log in to the Scripted Installer.

3. In Page 1 of 6, Change the installation method to be Remote.
4. In the Remote Server URL type: http://<ip>/fedora/fc5/

   Note:

   The IP address I am specifying here is of the UDA appliance (not the Windows file server), so adjust appropriate to your IP scheme. Also the path here must be completed with /fedora/fc5/ as this is the default location that UDA will look for your kickstart script – the final / is also required by the ESX scripted installer tool.
5. For Network Method, select Static IP.

   Note:

   This IP and directory location is where the physical ESX host will PXE boot to. As ESX is based on the Anaconda Installer and so is Fedora Core 4 and 5 we can reuse the work done by the UDA.
6. Choose No, to the option to "Create a default network for VMs."

   Note:

   Set your VLAN and Time Zone according to your network and geographical location.
7. Set a default root password.
8. Click Next.
9. In Page 2 of 6, enter the IP settings for the ESX host.
10. In Page 3 of 6, tick to Accept the EULA.
11. In Page 4 of 6, create the partition table for the appropriate disk.

    Note:

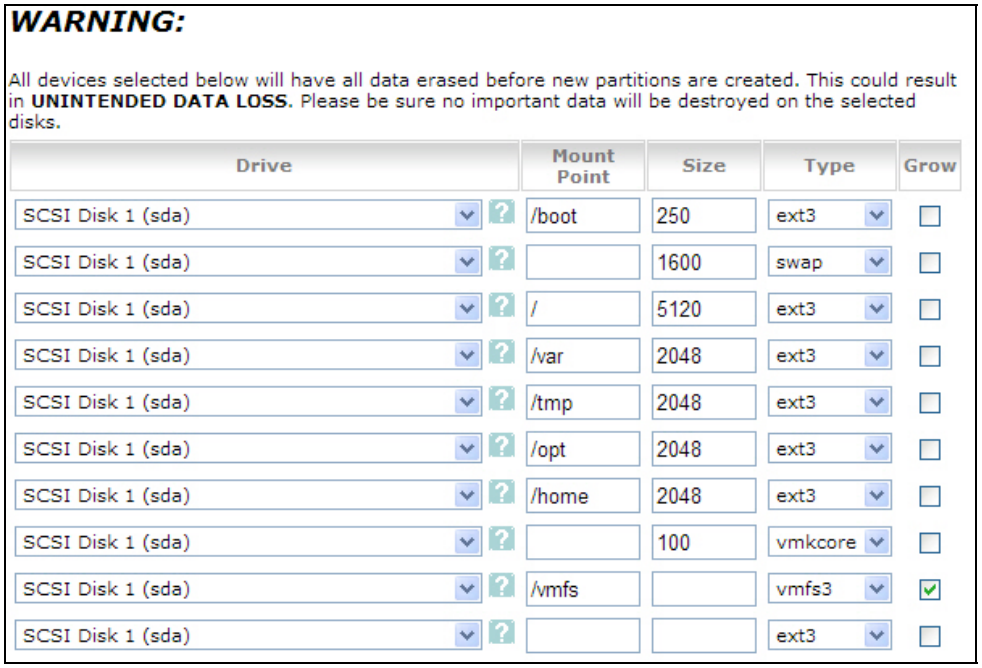
    If you are unsure what partitions to create check back to the first chapter about installing ESX which discussed appropriate partitioning schemes.
12. Under "drive" column ensure you select the right type of device to partition. In my experience my old Dell PowerEdge Servers, which merely have an Adaptec SCSI controller, always see the local disk or LUN as /dev/sda. In contrast my HP DL385 Proliant server with a Smart Raid Array 6i sees this device as /cciss/c0d0.

GOTCHA:

As well as reading the important warning here – be cautious if you are doing the install with a SAN connected to the ESX host. The way local storage device is recognized can be very different from what you might expect. I have seen cases where the internal disk or LUN is listed *after* all the SAN LUNs and the device name is /dev/sdh rather than /dev/sda

Figure 12.3 shows the layout of my partition table. In this case I opted to create a local VMFS partition in case I decided to use VM Clustering on this host.

Figure 12.3



Note:

The tick next to the "Grow Option" indicates that this partition will be created with the remainder of the disk or LUN. Personally, I would not recommend creating VMFS using the scripted installation. Firstly, as mentioned earlier in this book only the VI Client can correctly cope with the "disk alignment" issue. Secondly, although a scripted install will create a VMFS partition in the LUN, it does not format it. You would be forced to manually format this partition with vmkfstools –C command.

13. You might find it easier to create VMFS volumes at the end of the install processes automated with the %post section of the kick-start.cfg file.

14. Click Next.

15. In Page 4 of 6, type in the name of your license server and what port it is listening on (the default is 27000) and select your license type, in my case Standard.

    Note:

    This page appears because we choose to leave the default of "Use License Server" in the License Mode section.

16. On Page 6 of 6 click the Download Kickstart File button.

Modifying the Base Kickstart File

When viewing the KS file in Windows I recommend using the old DOS editor (edit.com) or WordPad. You will want to avoid notepad as it does not lay out the KS file in a readable way. Of course, if you're using Linux on your desktop then Vi or nano will suit your purposes perfectly well. The KS file contains five possible sections. We can customize it beyond the defaults asked in the web-access scripted install tool on the ESX host:

- Command
- %packages
- %pre
- %post
- %vmlicense_text

The command section contains the main instructions to complete the installation. Some of these are vanilla and would be found in Linux installations, and some are specific to ESX, including the network, clearpart, part, vmaccepteula, and vmlicense sections. %Packages are used to install additional components. Scripts can be called in %Pre and %Post processed run just before and after an installation. Of these, the %Post is the most important – as we can call a script here that will run VMkernel commands to handle such things as the creation of vSwitches, iSCSI Software Initiator and NAS configuration. Lastly, %vmlicense

is only used in a host-based licensing scenario. If you are using this format you can append the contents of a host LIC file to the KS script below the %vmlicense section.

Many of the commands you might be looking for can be viewed in the default kickstart script that is left on an existing ESX host after a manual installation. It is held in the /root directory and is called Anaconda.cfg. A complete list of KS changes is perhaps too lengthy to dwell on here. If you are looking for a comprehensive list of all the entries possible in the KS file visit:

https://www.redhat.com/docs/manuals/enterprise/RHEL-3-Manual/sysadmin-guide/s1-kickstart2-options.html

GOTCHA:

There are some edits we need to make to the base kickstart file to make it work correctly. If you have more than one disk or LUN, or your ESX host can see LUNs on a SAN, you will have to instruct Anaconda where the MBR record should be written. This is done by modifying the # Bootloader section.

Additionally, if you want to stop a prompt for which network card should be used for the kickstart installation we need to edit a configuration file in UDA.

1. Open your ks.cfg file in WordPad, Vi or nano.
2. Locate the # BootLoader section.
3. If your install LUN is presented to anaconda as /dev/sda then modify:

   ```
   bootloader --driveorder=sda --location=mbr
   ```

4. If your install LUN is presented to anaconda as /cciss/c0d0 then modify:

   ```
   bootloader --driveorder=cciss/c0d0 --
   location=mbr
   ```

5. Save the ks file, and select all the text and copy.

6. Back in the web-interface of UDA, click Template Link, and Choose Edit Configuration file.

7. Select all the text and paste your ESX based KS file – replacing the default Fedora Core 5 KS file generated by UDA.

8. To stop kickstart prompting the operator to select which NIC to use for the installation:

    a. Open an SSH session to the UDA

    b. Login as root with your password

    c. cd /var/public/tftproot/pxelinux.cfg

    d. nano –w default

    e. add the ksdevice=eth0 instruction to the end of each of your templates like so:

    ```
    append ks=http://192.168.3.150/kickstart/esx01.cfg
    initrd=initrd.fc5 ramdrive_size=8192 ksdevice=eth0
    ```

GOTCHA:

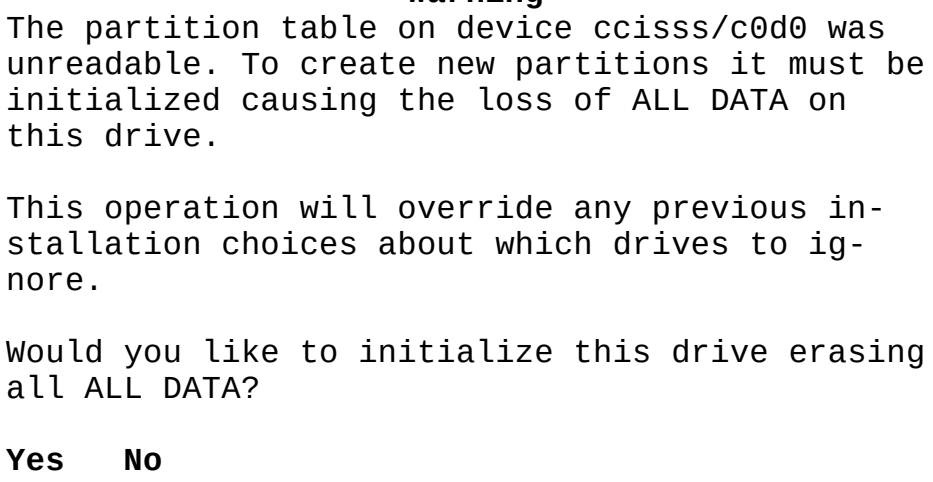
This reference to eth0 merely tells the kickstart installer which NIC to use for the installation. It has nothing to do with selecting the NIC used for the Service Console vSwitch. Currently the only way to adjust this is using the %post section and esxcfg commands to reassign the NIC interfaces.

Note:

The configuration of UDA is now completed so you are ready to test the PXE ESX installer. I recommend you choose a server you are willing to sacrifice if your installation fails, and for safety reasons detach the SAN cables. Most servers will use F12 to trigger the PXE boot process without any need for changing the boot order in the BIOS.

Secondly, if you have a blank disk or LUN with no existing partition table – the installer will ask if it is OK to initialize the LUNs it sees. This can also happen if your SAN is attached. This can be very dangerous if /dev/sda is not a local LUN but a SAN-based LUN that contains data. Figure 12.5 shows the warning box.

Figure 12.4



Additional Notes:

If you wish to edit by hand the various files used by UDA you will find them located here:

- **Your Template files:** /var/public/www/kickstart/
- **UDA Built-in Templates:** /var/public/www/templates
- **UDA PXE Menu:** /var/public/tftproot/message.txt
- **Default Help Message:** /var/public/tftproot/help.txt
- **pxelinux.cfg File:**/var/public/tftproot/pxelinux.cfg/pxelinux.cfg

GOTCHA:

If you shutdown or reboot the UDA appliance it does not automatically remount your shares. Remember if you have just powered on your UDA to request a remounting of the shares.

Configuring Networking

Generally, everything we have created in the VI Client can be also achieved from the command-line. The only exceptions are:

- Enabling VMotion on a VMkernel Port Group
- Settings advanced policy settings such as load-balancing, traffic shaping, and security settings

There are two main reasons to be familiar with the networking commands; firstly, for troubleshooting Service Console connectivity, and secondly for use with the %post section of a kickstart script.

Networking involves the use of three main commands:

- esxcfg-vswitch - handles generic vSwitch operations
- esxcfg-vswif0 - handles Service Console networking
- esxcfg-vmknic - handles VMKernel networking

These are quite sophisticated commands used in a particular order to achieve the results you are looking for. esxcfg-vswitch is the main command – and it has a mix of parameters in lower and upper-case. Lower-case parameters manipulate the switch, whereas upper-case switches manipulate the portgroup. So to add a vSwitch it is –a and upper-case –A that creates a portgroup. In one way this is nice because it's easy to remember, but beware; it is incredibly easy to forget this subtle distinction and accidentally create a switch rather than a portgroup attached to a switch.

GOTCHA:

Lastly, one important thing to mention about the esxcfg commands is that if you use them – and then check your changes in the VI Client – you may find that the VI Client will show out-of-date information. In order for the VI Client to display any changes made with the esxcfg commands, you may need to restart the hostd daemon on the ESX host if the various refresh options in the VI Client refuse to update your administrative views.

This is easy to do with:

```
service mgmt-vmware restart
```

Unfortunately, every time you restart the hostd daemon you may find you will be disconnected from your ESX host. I recommend you become familiar in-

terpreting with the esxcfg-vswitch –l command, so you can make many changes in one go – and then restart the hostd service at the very end of the process.

Viewing your Switches & Service Console Networking

1. To view your switches type the command:

   ```
   esxcfg-vswitch –l
   ```

   Note:

   This shows me I have one vSwitch (vSwitch0) using one NIC (vmnic0) with one portgroup called "Service Console" which is not using VLAN.

2. To view your Service Console network settings type:

   ```
   esxcfg-vswif –l
   ```

   Note:

   Nothing to state here – but I think it's interesting that it doesn't show me my all important default gateway settings which would have been nice. If you want to know your default gateway settings and DNS settings you will find these held in /etc/sysconfig/network and /etc/resolv.conf respectively.

3. To View your network card's vmnic name, PCI settings (b:s:f), driver, link, speed, duplex, and description:

   ```
   esxcfg-nics –l
   ```

Creating a vSwitch (Internal)

1. To create a new switch type:

   ```
   esxcfg-vswitch –a vSwitch1
   ```

2. Then add a portgroup

   ```
   esxcfg-vswitch –A internal vSwitch1
   ```

   Note:

Remember to use lower-case -a for adding a switch, and upper-case -A for adding a portgroup. As stated earlier in the network chapter, I recommend using lower-case and no space for port-group names.

3. If you run the command **esxcfg-vswitch –l** you will see this information:

Figure 12.5

```
vSwitch Name
vSwitch0

      PortGroup Name
      Service Console

vSwitch Name
vSwitch1

      PortGroup Name
      internal
```

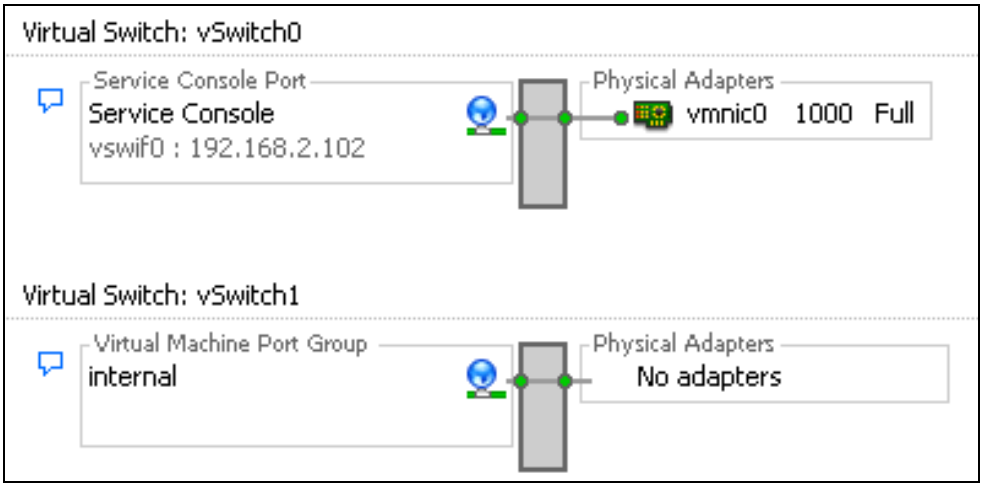
4. **If you wish to see this reflected in the VI Client** then type:

```
service mgmt-vmware restart
```

Note:

This is how my switch configuration looks now:

Figure 12.6



Creating a vSwitch (Single NIC)

1. In this case we will create a new switch and port group in one line with:

   ```
   esxcfg-vswitch –a vSwitch2 –A production
   ```

2. Next patch an NIC to the vSwitch with:

   ```
   esxcfg-vswitch -L vmnic1 vSwitch2
   ```

   Note:

   Again this is a case-sensitive option. –l lists switches, whereas –L links nics to switches.

3. If you run the command **esxcfg-vswitch –l** you will see that vmnic1 has been added to vSwitch2

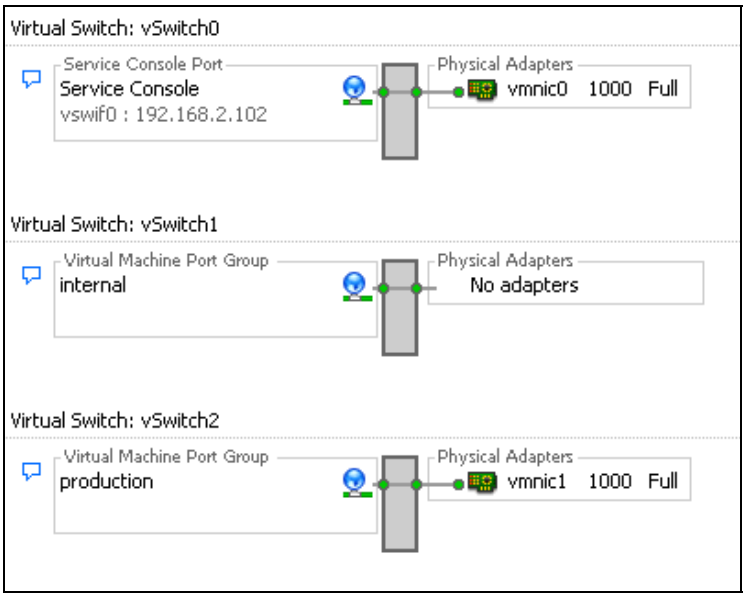
4. **If you wish to see this reflected in the VI Client** then type:

   ```
   service mgmt-vmware restart
   ```

   Note:

   This is how my switch configuration looks now:

Figure 12.7



Creating a vSwitch (Multiple NICs)

Note:

It is very easy to create a NIC-Team, just re-run the previous command with a different NIC using:

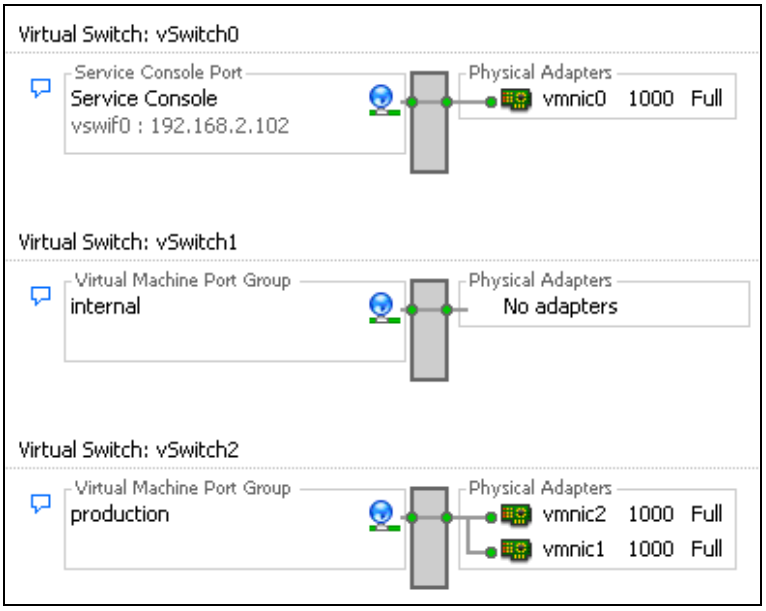
1. esxcfg-vswitch -L vmnic2 vSwitch1
2. If you run the command esxcfg-vswitch –l you will see that two NICs have been added to vSwitch1
3. If you wish to see this reflected in the VI Client then type:

   ```
   service mgmt-vmware restart
   ```

   Note:

   This is how my switch configuration looks now:

Figure 12.8



Deleting a Switch from vSwitch

I'm now running out NICs for the next part of this chapter. So I am going to delete my vSwitch2 to free up my NICs.

1. Type the command:

   ```
   esxcfg-vswitch –d vswitch2
   ```

   GOTCHA:

   Notice how it doesn't ask "are you sure?", but then again neither does the VI Client.

Creating PortGroups for VLAN Networking NIC Team

1. We have to create a switch, create a portgroup for each VLAN, allocate our NICs and then set the VLAN ID:

   ```
   esxcfg-vswitch –a vSwitch2
   esxcfg-vswitch –A accounts vSwitch2
   esxcfg-vswitch –A rnd vSwitch2
   esxcfg-vswitch –A sales vSwitch2
   esxcfg-vswitch –L vmnic1 vSwitch2
   ```

```
esxcfg-vswitch –L vmnic2 vSwitch2
```

2. **The next part is to set the VLAN id** for each network (account, rnd, and sales):

```
esxcfg-vswitch –v 10 –p accounts vSwitch2
esxcfg-vswitch –v 20 –p rnd vSwitch2
esxcfg-vswitch –v 30 –p sales vSwitch2
```

Note:

In this case –v is used to set the VLAN ID, and –p is used to modify the properties of an existing portgroup.

3. **If you run the command esxcfg-vswitch –l** you will that VLAN ID column has values beneath it

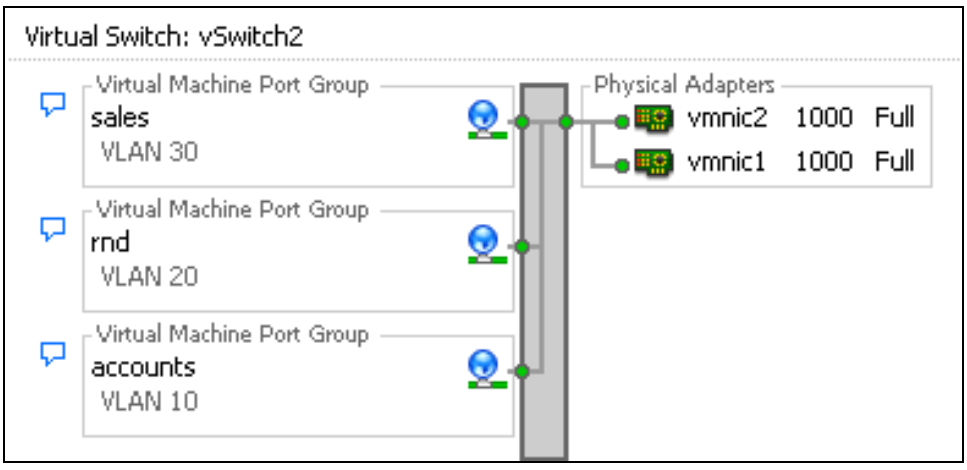
4. **If you wish to see this reflected in the VI Client** then type:

```
service mgmt-vmware restart
```

Note:

This is how my switch configuration looks now:

Figure 12.9



Creating Vmkernel Switches

You will need a good name for the portgroup – because the esxcfg-vswitch command just shows as any ordinary switch. In this example I will create a

vSwitch good for VMotion. Remember that the command-line tools do not support enabling a VMKernel portgroup for VMotion.

- You need to use **esxcfg-vmknic –l** to see it:

1. First create the Switch, Portgroup, and Assign an NIC:

   ```
   esxcfg-vswitch –a vSwitch3
   esxcfg-vswitch –A “vmotion” vSwitch3
   esxcfg-vswitch –L vmnic3 vSwitch3
   ```

2. Next use the esxcfg-vmknic command to add in a VM Kernel NIC and set the IP and Subnet Mask:

   ```
   esxcfg-vmknic –a “vmotion” –i 10.0.0.1 –n
   255.0.0.0
   ```

3. Then set the vmkernel default gateway with:

   ```
   esxcfg-route 10.0.0.254
   ```

   Note:

   Remember that VMware does not support VMotion events across routers. However, you may need a default gateway if this vmkernel portgroup was being used to access iSCSI or NAS based storage.

4. If you run the command esxcfg-vswitch –l you will see the VM Kernel port has been added

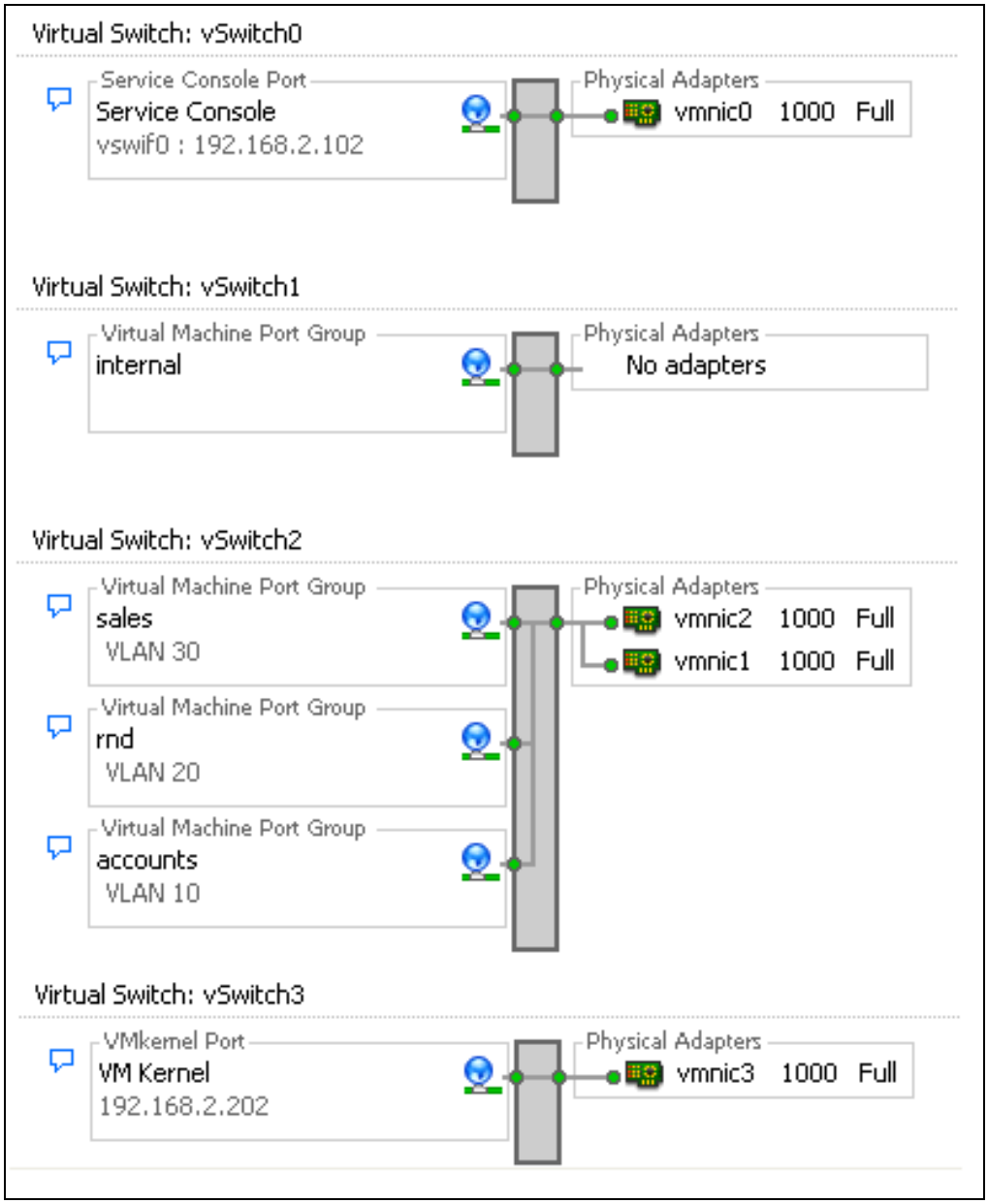
5. If you wish to see this reflected in the VI Client then type:

   ```
   service mgmt-vmware restart
   ```

Note:

This is how my switch configuration looks now:

Figure 12.10



Changing your Service Console IP Settings

You can do this through a putty session but it is likely you will be disconnected unless you have more than one vswif interface (with a different IP address). It is entirely possible to have two vswif interfaces on two separate IP addresses – and use one connection to change the IP address of the other. You do run the risk of losing SSH connectivity if you screw up – so perhaps doing this through

your ILO is a safer-bet and less hard work. You might also wish to include a new DNS entry for your ESX host before you make this change…

1. To view your current IP and Netmask type:

   ```
   esxcfg-vswif –l
   ```

2. To **change your IP and Subnet Mask** type:

   ```
   esxcfg-vswif –i 192.168.3.203 –n 255.255.255.0
   vswif0
   ```

   Note:

   You change your default gateway by editing

   ```
   nano –w /etc/sysconfig/network
   ```

   and then restart your networking with

   ```
   services network restart
   ```

3. Your DNS settings are located in **nano –w /etc/resolv.conf**

Setting the Speed and Duplex of NICs

We have always been able to change the speed and duplex of the vmkernel NICs but in the past the only way to change the Service Console speed and duplex was by editing the /etc/modules.conf file. You can now change both the Service Console and the vmkernel NICs through the GUI. There could be a chicken-egg/catch22 situation here though. If your Service Console NIC has the incorrect speed/duplex you may be unable to connect with the VI Client to change it. That's when knowing the Service Console commands come in handy. It can also be useful to reassign a 100Mps card to a Service Console and then fix its speed/duplex.

1. Work out which NIC has been assigned to the portgroup "Service Console" with:

   ```
   esxcfg-vswitch –l
   ```

2. To view your current speed & duplex:

```
esxcfg-nics –l
```

3. To set the speed/duplex of vmnic0 to 100Mps/Half-Duplex type:

```
esxcfg-NIC’s -s 100 -d half vmnic0
```

4. To reset to auto-negotiate:

```
esxcfg-NIC’s -a vmnic0
```

Recreating your vswif0 Interface

Recreating your ESX vswif0 interface can be useful in a couple of scenarios. I have seen “in place upgrades” sometimes breaks the Service Console network. Secondly, if a novice uses the VI Client to change the Service Console network it may become inoperable. Thirdly, during the installation of ESX you must select the NIC used for vSwitch0. If you select the wrong NIC you might find you have no network communication to the ESX host via the VI Client. In this case being able to modify IP settings with esxcfg-vswif –i and –n is useful, and being able to unlink and link an NIC is essential as well.

1. Logon locally to the ESX host or use your ILO card
2. Create a new switch:

```
esxcfg-vswitch -a vSwitch0
```

3. Create a new portgroup:

```
esxcfg-vswitch -p "Service Console" vSwitch0
```

4. Assign an NIC:

```
esxcfg-vswitch –L vmnic0 vSwitch0
```

5. Assign a vswif interface and set its ip/sn:

```
esxcfg-vswif –a vswif0 -p "Service Console" i
192.168.3.101 –n 255.255.255.0
```

Removing an NIC from vSwitch

Removing an NIC from a vSwitch is relatively easy; it uses the case-sensitive –U switch to unlink an NIC from a switch:

```
esxcfg-vswitch –U vmnic vSwitch2
```

Deleting a PortGroup from vSwitch

Deleting a portgroup is done with the –D switch. Perhaps you accidentally created a portgroup with –A when you should have used –a:

```
esxcfg-vswitch –D production vSwitch
```

Managing the ESX Firewall

Managing the firewall by the VI Client is really easy – it's a tick box style interface. If you use the GUI interface and then query with the command-line tool you get "friendly" information about what is enabled. If you purely use the command-line tool you will just get TCP port numbers and directions (outgoing and incoming). The GUI tool also has some handy "built-in" friendly names for popular applications, vendor specific agents like CommVault Dynamic/Static and many others. In ESX 2.x it was possible to set 3 levels of security (high, medium and low). We can achieve a similar configuration with ESX 3.x:

- High: incoming/outgoing blocked
- Medium: incoming blocked, outgoing not blocked
- Low: firewall off, no blocking of incoming/outgoing traffic

Viewing your current Firewall Settigs

1. Type the command

   ```
   esxcfg-firewall –q outgoing
   esxcfg-firewall –q incoming
   ```

Note:

You can also use esxcfg-firewall –q on its own. This gives you lots of stuff… most useful at the bottom:

```
Incoming and outgoing ports blocked by default.
Enabled services: CIMSLP ntpClient aam VCB
swISCSIClient CIMHttpsServer snmpd sshClient
vpxHeartbeats LicenseClient sshServer
CIMHttpServer
```

Changing Your Security Level

If you wanted to weaken your security to medium you could use the following:

1. Type the command

   ```
   esxcfg-firewall --allowOutgoing –blockIncoming
   ```

   Note:

   You should get warnings like so:

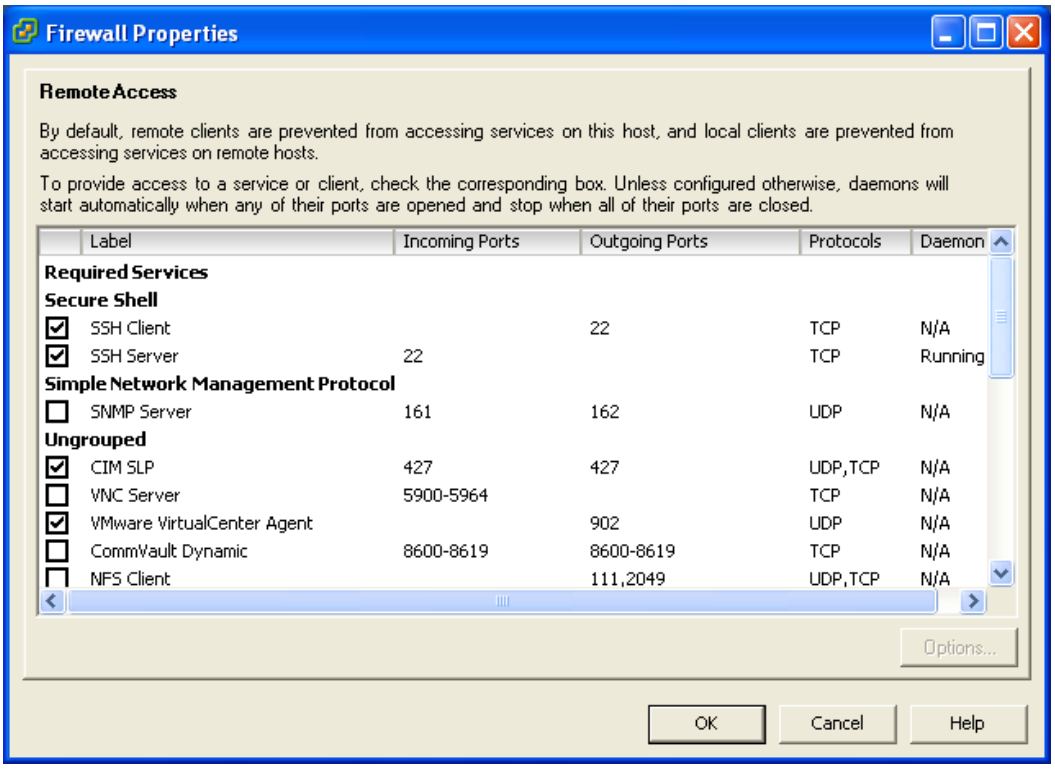
   ```
   2006-07-06 14:39:53 (1965) WARN :  Setting
   firewall default firewall/blockOutgoing to 0

   2006-07-06 14:39:53 (1965) WARN :  Setting
   firewall default firewall/blockIncoming to 1
   ```

Enabling a Single Service/Client/Agent

If you want to SSH from one ESX host to another ESX host or SCP from one ESX host to another you need to enable the SSH Client on port 22. If you do this via the GUI and then do an esxcfg-firewall –q you will see friendly information like so:

Figure 12.11




```
Enabled services: AAMClient CIMSLP LicenseClient
sshServer CIMHttpsServer CIMHttpServer sshClient
vpxHeartbeats
Dgfdfgdsdsdsd
Opened ports:
```

To do the same from the command-line you would type the command:

```
esxcfg-firewall -e sshClient
```

Note:

To disable, type:

```
esxcfg-firewall -d sshClient
```

Enabling non-Standard Ports

This is useful if there is an application or service which is not listed for use with –e or –d. Alternatively you might work in an environment where all port numbers have been changed to non-standard ports as part of your datacenter policies. It is possible to open a specific port by number, transport (udp/tcp) and direction (in/out). For example, to enable port 22 outbound from the server type the command:

```
esxcfg-firewall -o 22,tcp,out,ssh
```

Note:

The ssh at the end is a friendly label. If I run esxcfg-firewall –q again at the bottom it states:

```
Incoming and outgoing ports blocked by default.
Enabled services: AAMClient CIMSLP LicenseClient
sshServer CIMHttpsServer
IMHttpServer vpxHeartbeats
Dfsdfdfdfsdfsdf
Opened ports:
ssh                 : port 22 tcp.out
```

Typical vSwitch Errors

- As stated earlier, one of the most common mistakes with the networking esxcfg command is getting your case-sensitivity muddled up! Below is a quick summary of the command-line switches used in this chapter and their purpose:
- -a add vSwitch
- -A add portgroup
- -l list vSwitches
- -L link vmnic
- -d delete vSwitch
- -D delete portgroup

Another common error is trying to change properties of vSwitch that are in use. Typically the errors look like this:

“Failed to remove vswitch: vSwitch3, Error: PortGroup "Legacy eth0" on VirtualSwitch "vSwitch3" is still in use: 1 active ports, vswif0” or “Legacy vmnic2, Error: Unable to delete portgroup "Legacy vmnic2,” for the following reasons: 1 active ports”

In the first example, the operator tried to delete the vSwitch that is hosting our Service Console. To delete it we would need to add a new Service Console port and connect with it to then delete the old Service Console Connection. The second error happens if you have powered on VMs which are actively connected to the vSwitch portgroup. Powering those VMs off, or configuring them for a different vSwitch portgroup, will normally allow the command to work.

Configuring Storage

This section assumes you have already set up NAS and iSCSI storage correctly, **and** that you have created a VM Kernel vSwitch/Port Group outlined in the previous module. You are now ready to connect to iSCSI or NAS-based storage. This module repeats some of the vmkfstools command listed earlier. Sorry about this repetition but this gives more detail than a simple explanation of the commands.

Configuring NAS Storage

Remember that you must have a VM Kernel Switch to do this. There must be connectivity to the NAS via the Service Console NIC because authentication is driven through the Service Console NIC. My NFS server is called nfs1.vi3book.com, and it has one export on it /ISOs.

Mounting NAS Exports/Shares

Type the command:

```
esxcfg-nas -a ISO’s -o nfs1.vi3book.com -s ISO’s
```

Note:

The command should respond with:

```
Connecting to NAS volume: ISO's
ISO's created and connected.
```

Note:

-a is used to add followed by the friendly label, -o to specify the nfs server, and –s to specify the export/share.

Listing NAS Exports/Shares

```
esxcfg-nas –l
```

Note:

You should receive a response like so:

```
ISO's is ISO's from nfs1.vi3book.com mounted
nas-VM's is nas-VM's from nfs1.vi3book.com mounted
templates is templates from nfs1.vi3book.com mounted
```

Restoring NAS Connections

If the NAS become unavailable you may have to restore the mount by force. This sometimes happens to me if I start-up my ESX hosts before the NAS is up.

1. Type the command:

   ```
   esxcfg-nas –r
   ```

2. Followed by:

   ```
   esxcfg-nas –l
   ```

Using vmkping to carry out tests

Vmkping is a tool that tests your vmkernel switch IP configuration. I'm mentioning it here rather than in the vSwitches section (where it might have been a

more appropriate reference) because it can also reveal information about your NAS connectivity.

1. Type the command:

   ```
   vmkping –D –v
   ```

   Note:

   This runs vmkping in a **d**ebug mode with **v**erbose information. It pings the following interfaces:

   IP address allocation to VM Kernel network(s)

   VM Kernel Default Gateway(s)

   NAS mount(s) and iSCSI systems

Connecting to iSCSI Storage (Software)

My iSCSI box is a Fedora Core 5 with iSCSI Enterprise Target installed and configured. Its name is iscs1.vi3book.com iSCISi has 2 LUNs available, it supports dynamic discovery, and its IP is 172.168.3.210. I used the iqn of iqn.2006-06.com.vi3book:iscsi1 on this iSCSI Target. The esxcfg-swiscsi is more limited than the esxcfg-nas command.

- –e enable
- –d disable
- –q query if the adapter is enabled or disabled
- –s force a scan
- –k forcibly remove iscsi sw stack

Enabling the iSCSI Adapter

1. Type the command:

   ```
   esxcfg-swiscsi –e
   ```

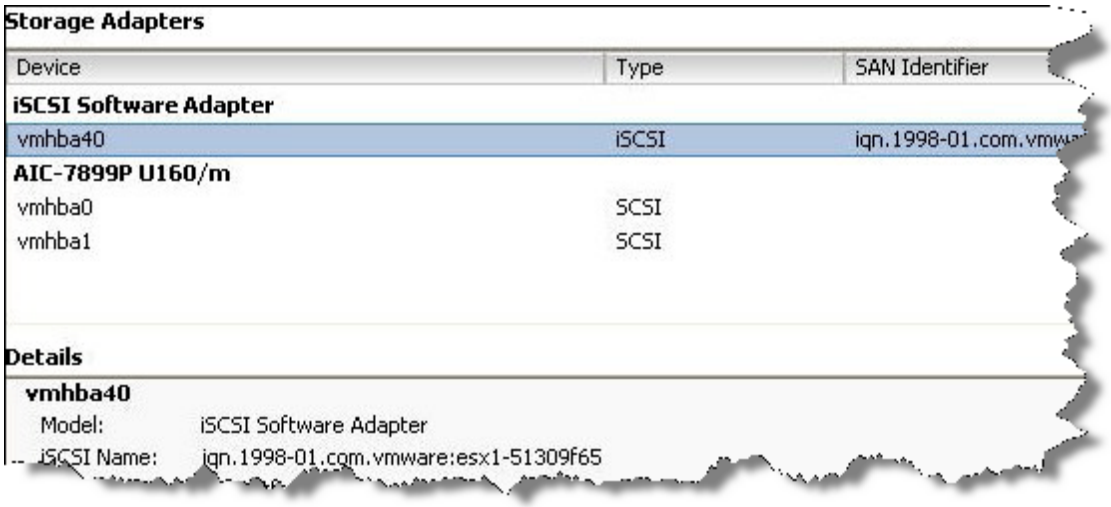
   Will return:

```
Allowing software iSCSI traffic through
firewall...
Enabling software iSCSI...
/usr/sbin/vmkload_mod
/usr/lib/vmware/vmkmod/iscsi_mod.o
Using /usr/lib/vmware/vmkmod/iscsi_mod.o
Module load of iscsi_mod succeeded.
```

Note:

This will enable the iSCSI Software Adapter like so:

Figure 12.12



Setting the iSCSI Target IP with Discovery Mode

Note:

Remember that the Software iSCSI adapter does NOT support static discovery, so to add in an IP for the iSCSI Target for Dynamic Discovery (referred to as the SendTargets method) we use the –D and –a switches.

1. Type the command:

```
vmkiscsi-tool -D -a 172.168.3.210 vmhba40
```

Note:

-D sets the mode of Discovery, rather than static. –a is used to add an iSCSI Target, followed by the IP address of iSCSI Device and the HBA that will be used.

2. To list the targets configured use:

```
vmkiscsi-tool -l -T vmhba40
```

Note:

This should report the IQN name and the Target IP address and port number

Forcing a Rescan of the iSCSI Adapter

1. You might wish to list the current LUNs/Disk currently visible with:

```
ls –l /vmfs/devices/disks
```

Note:

I have two internal disks on an internal SCSI Controller (vmhba0:0 and vmhba0:1).

I also have JBOD with six disks inside (vmhba1:1-6).

2. **Force the rescan** with:

```
esxcfg-swiscsi –s
```

Note:

This should produce the following result:

```
Scanning vmhba40...
Rescanning vmhba40...done.
On scsi2, removing:.
On scsi2, adding: 0:0 0:1.
```

3. If we run the ls command again we can see the 20GB LUNs from my iSCSI box:

Using the vmkiscsi command

Note:

The vmkiscsi command has lots of switches. Here are some of its options:

- To view the discovery settings**:**

  ```
  vmkiscsi-tool -D vmhba40
  ```

  Will return:

  ```
  =========Discovery Properties for Adapter
  vmhba40=========
  iSnsDiscoverySettable    : 0
  iSnsDiscoveryEnabled     : 0
  staticDiscoverySettable  : 0
  staticDiscoveryEnabled   : 0
  sendTargetsDiscoverySettable  : 0
  sendTargetsDiscoveryEnabled   : 1
  slpDiscoverySettable  : 0
  Discovery Status: Done.
  DISCOVERY ADDRESS          : 172.168.3.210:3260

  Static Discovery not supported for this adapter
  ```

- To list LUNs:

  ```
  vmkiscsi-tool -L vmhba40
  ```

  Will return:

  ```
  Target iqn.2006-06.com.vi3book.com:storage.lvm:
  ------------------------------------------
  OS DEVICE NAME    :vmhba40:0:10
  BUS NUMBER              : 0
  TARGET ID               : 0
  LUN ID                  : 1

  ------------------------------------------
  OS DEVICE NAME          : vmhba40:0:11
  ```

```
BUS NUMBER              : 0
TARGET ID               : 0
LUN ID                  : 11
----------------------------------------
```

- To view physical settings of iSCSI "nic":

  ```
  vmkiscsi-tool -P vmhba40
  ```

  Will return:

```
=========PHBA Properties for Adapter
vmhba40=========
VENDOR                          : VMware
MODEL                           : VMware-Isoft
DESCRIPTION                     : VMware Software
Initiator
SERIAL NUMBER                   :

=========Node Properties for Adapter
vmhba40=========
NODE NAME VALID                 : 1
NODE NAME                       : iqn.1998-
01.com.vmware:esx2-1f199fe7
NODE ALIAS VALID                : 1
NODE ALIAS                      :
esx1.vi3book.com
NODE NAME AND ALIAS SETTABLE: 1
```

Creating a VMFS Volume

Using FDISK to Create Partition

1. Logon to the Service Console as ROOT type:

   ```
   fdisk /vmfs/device/disks/vmhba40:0:0:0
   ```

   Note:

   Under ESX 2.x we would have needed to know how Linux addressed this disk/LUN with /dev/sd. Syntax. This is no longer required.

2. Type N to create a new partition.

3. Type P for a Primary partition.
4. Choose 1 for the Partition Number.
5. Accept the defaults for First Cylinder and Last Cylinder.

   Note:

   This creates a partition which uses all of the free space on the disk. Fdisk defaults to being a Linux type of "83" which describes an ext3 partition. This case is not good for our VMFS partition which use a partition type of "fb"
6. Type P to print out the partition table (make a mental note of the partition number here).
7. Type T to change the File System Type.
8. Type the Hex Code of fb: (the code for a VMFS partition).

   Note:

   This should give the result:

   ```
   Changed system type of partition 1 to fb (Un-
   known)
   ```
9. Hex codes tell the system what file system partition will support – 07 NTFS, 82 Linux Swap, 83 Linux File System, FB for VMFS and FC for VMware Core Dump. At this stage the VI Client would identify the partition as VMFS unformatted.
10. Type W to write your partition table changes to the hard drive – it will give you this status information:

    ```
    The partition table has been altered!

    Calling ioctl() to re-read partition table.
    Syncing disks.
    ```

    Note:

    If you re-run fdisk /vmfs/device/disks/vmhba40:0:0:0 and choose P to display the partition table

Format & Label the VMFS Partition

1. Logon to the Service Console as ROOT.
2. To format the new partition with the VMFS file system type:

   ```
   vmkfstools –C vmfs3 vmhbaA:T:L:V
   ```

In my case the hard disk partition is located on the Adapter 40, TARGET 0, LUN 0 on Volume/Partition 1.

So I would type:

```
vmkfstools -C vmfs3 –S iscsi-lun0
/vmfs/devices/disks/vmhba40:0:0:1
```

Warning:

Please note it is an UPPERCASE C you type here. A lowercase c creates a vmdk file. You can specify –b flag to set a block-size. You would need to do this if you thought any one of your virtual disks were going to be greater than 256GB in size.

We can use 1MB, 2MB, 4MB, or 8MB. When entering a size, indicate the unit type by adding a suffix of m or M. The unit type is not case sensitive. vmkfstools interprets either m or M to mean megabytes.

- 2m allows 512GB max file size
- 4m allows 1024GB max file size
- 8m allows 2048GB max file size

Note:

This should give the result like so:

```
Creating file system on "vmhba40:0:0:1" with blockSize
1048576 and volume label "iscsi-lun0."
Successfully created new volume: 44a7bcf0-8b87cb86-9403-
00065bec0eb6
```

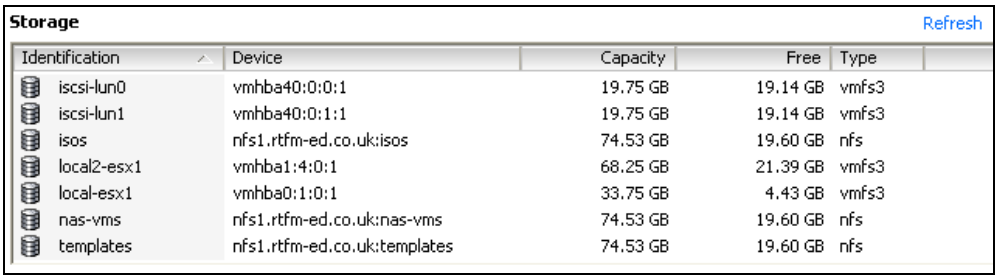
Note:

I repeated this for my other iSCSI LUN 1

```
fdisk /vmfs/devices/disks/vmhba40:0:1:0
vmkfstools -C vmfs3 –S iscsi-lun1
/vmfs/devices/disks/vmhba40:0:1:1
```

Note:

If we re-run ls –l /vmfs/volumes we will see this lots of juicy information. The VI Client looks like this:



Changing the Volume Label

Note:
To do this you need to know the UUID of your VMFS volume.

1. Type the command:

   ```
   ls –l /vmfs/volumes
   ```

   Note:

   The UUID is the value in blue. So the fixed and unchanging volume id is actually something like:

```
/vmfs/volumes/44a38c72-156b2590-be15-
00065bec0eb7
```

2. Type the command:

```
ln -sf /vmfs/volumes/44a38c72-156b2590-be15-
00065bec0eb7
vmfs/volumes/esx1-local
```

Note:

The LN makes symbolic links (a bit like shortcuts), –s makes/changes a "symbolic" link, -f over-writes the existing symbolic link.

Viewing VMFS Volumes/Partition Information

Type the command:

```
vmkfstools -P /vmfs/volumes/local-esx1
```

Note:

This returns information like so:

```
VMFS-3.21 file system spanning 1 partitions.
File system label (if any): local
Mode: public
Capacity 73282879488 (69888 file blocks * 1048576),
41889562624 (39949
locks) avail
UUID: 44a6c956-66056236-c671-00065bec0eb6
Partitions spanned:
        vmhba1:4:0:1
```

Note:

For NAS data store it would look like this:

```
NFS-1.00 file system spanning 1 partitions.
File system label (if any): iso
Mode: public
Capacity 80023715840 (19537040 file blocks * 4096),
25076690944 (6122239 blocks) avail
UUID: ab184e34-6f68911d-0000-000000000000
Partitions spanned:
        nfs:iso
```

Viewing Available Physical Disk Space

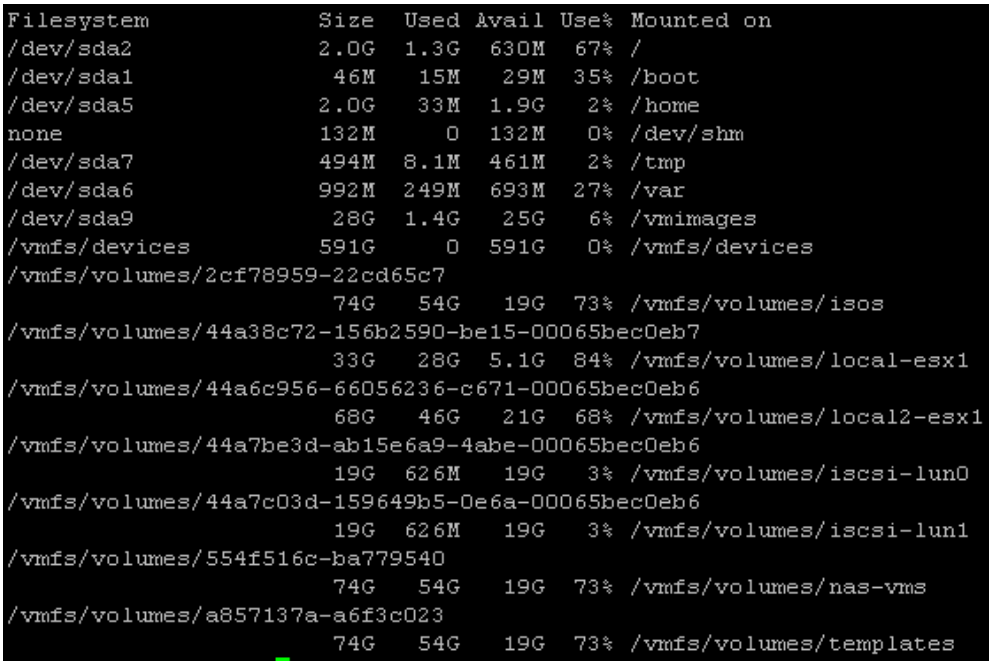
Note:

1. Logon to the Service Console as ROOT
2. Type:

   `df –h (the linux method)`

   or

   `vdf –h (the VM Kernel Method)`



Note:

The partition table for /sda does NOT follow the recommendations for ESX 3.x – this partition table was taken for an upgraded ESX 2.x server.

Note:

If you wish to see how the vmhba syntax relates to /dev/sd you will find that you can still use old ESX 2.x command "vmkpcidivy -q vmhba_devs." However, it has been hugely depreciated – and instead you should use:

```
esxcfg-vmhbadevs –q
```

While returns:

```
vmhba0:0:0 /dev/sda
vmhba0:1:0 /dev/sdb
vmhba1:2:0 /dev/sdc
vmhba1:3:0 /dev/sdd
vmhba1:4:0 /dev/sde
vmhba1:5:0 /dev/sdf
vmhba1:6:0 /dev/sdg
vmhba40:0:0 /dev/sdh
```

```
vmhba40:0:1 /dev/sdi
```

Note

If you want to see all of the information together (vmhba ID, the linux /dev name, LVM id and VMFS volume label) you can use this perl script:

```
#!/usr/bin/perl
@array = `/usr/sbin/esxcfg-vmhbadevs -m`;

foreach (@array)
{
        ($vmk, $cos, $uuid) = split;
        ($tmp, $label) = split (/:/,
`/usr/sbin/vmkfstools -P /vmfs/volumes/${uuid} | grep
"File system label"`);
        print "$vmk \t $cos \t $uuid \t $label\n"; }
```

This produces an output like so:

```
vmhba0:1:0:1 /dev/sdb1 7458-66sb-fsc0-43fs isos
```

Note:

This only shows me my VMFS, not my NAS datastores.

Managing Storage Devices

You can use a couple of commands to handle storage devices. Some of these have already been mentioned like the command esxcfg-swiscsi -s to rescan the iSCSI Adapter(s). For those of you who come from ESX 2.x background it looks like cos-rescan.sh has been discontinued; its functionality has *probably* been integrated into the –s switch.

Forcing a rescan of Fibre Channel Device

To force a rescan of the fibre channel device, type the following command:

```
vmkfstools –s vmhba1
```

Note:

You can also use **esxcfg-rescan vmhba***N* to be rescanned as well which gives this kind of output:

```
Rescanning vmhba1...done.
On scsi1, removing: 2:0 3:0 4:0 5:0 6:0.
On scsi1, adding: 2:0 3:0 4:0 5:0 6:0.
```

Managing SCSI Reservations of LUN's

Occasionally, things go wrong with the SCSI Reservations or locking of LUNs. One example is when, through a configuration error, an SCSI reservation (lock) is put on an LUN, which subsequently doesn't get released. Until an LUN is released you won't be able to manage the file system. Typically, these types of reservation problems happen in VM clustering scenarios where the clustering software inside a VM malfunctions. This SCSI Reservations or locking can be controlled manually by using the –L switch on vmkfstools…

```
-L reserve .
```

This reserves the specified LUN. After the reservation, only the server that reserved that particular LUN can access it. If other servers attempt to access that LUN, they will get a reservation error!

```
-L release .
```

This releases the reservation on the specified LUN. Any other server can access the LUN again.

```
-L lunreset
```

This resets the specified LUN by clearing any reservation on the LUN and making the LUN available to all servers again. The reset does not affect any of the

other LUNs on the device. If another LUN on the device is reserved, it remains reserved.

`-L targetreset .`

This resets the entire target. The reset clears any reservations on all the LUNs associated with that target and makes the LUNs available to all servers again.

`-L busreset .`

This resets all accessible targets on the bus. The reset clears any reservation on all the LUNs accessible through the bus and makes them available to all servers again.

This command uses the device parameter and so uses the following:

`/vmfs/devices/disks/vmhbaA:T:L:V syntax`

Using USB storage Devices

USB device drivers are loaded automatically by default. This can be disabled by modifying /etc/modules.conf and remarking out with # the line which begins alias usb-controller usb-ohci. If someone has done this – then you can load the modules (drivers) manually and access the device as needed using the insmod command.

You will need to think about what file system to use on a removable USB device. You can read but not write to an NTFS partition at the Service Console, and only FAT32 is supported for read/write. Windows does not inelegantly understand the Linux file system of EXT3. However, there are some free EXT3 drivers for Windows (the one I use currently only works on NT4, W2K and WXP, not W2K3).

http://uranus.it.swin.edu.au/~jn/linux/ext2ifs.htm

You might prefer to use EXT3 if you wish to convert virtual disks in a more portable format suitable for transfer to a removable hard-drive. In my case it was very easy to configure. I plugged the USB device into the server. It came

up with a message – and said it allocated the id of /dev/sdc to the disk so all I had to do was mount it with:

```
mkdir /mnt/usbdisk
mount /dev/sdc1 /mnt/usbdisk
```

Note:

If this doesn't happen you might find the information documented below useful:

Systems using the USB-UHCI device driver with the USB 2.0 interface can cause ESX Server to show a false warning message during the boot sequence. The error looks like this on boot-up:

```
Mar 29 11:04:10 VM'server1 rc.sysinit: Initializing USB
controller (usb-uhci): succeeded

Mar 29 11:04:10 VM'server1 modprobe: Hint: insmod errors
can be caused by incorrect module parameters, including
invalid IO or IRQ parameters

Mar 29 11:04:10 VM'server1 modprobe: /lib/modules/2.4.9-
vmnix2/kernel/drivers/usb/usb-ohci.o: init_module: No
such device

Mar 29 11:04:10 VM'server1 modprobe: /lib/modules/2.4.9-
vmnix2/kernel/drivers/usb/usb-ohci.o: insmod
/lib/modules/2.4.9-mnix2/kernel/drivers/usb/usb-ohci.o
failed

Mar 29 11:04:10 VM'server1 modprobe: /lib/modules/2.4.9-
vmnix2/kernel/drivers/usb/usb-ohci.o: insmod usb-ohci
failed

Mar 29 11:04:10 VM'server1 rc.sysinit: Initializing USB
controller (usb-ohci): failed
```

Note:

It is safe to ignore this message, but if you want to configure your system so that this warning does not appear the next time you boot your ESX Server machine, follow these steps. VMware KB Article 1659 outlines the issues involved:

http://www.vmware.com/support/kb/enduser/std_adp.php?p_faqid=1659

If you want to use a USB hard-drive to copy files from the ESX server to it – and you have these errors -then a general work around is to boot to a Knoppix Boot CD and do it that way. Remember if you are copying virtual disks – convert into the correct format first.

If you have no errors like me, this is how you can go about using USB hard-rives and devices.

There are 3 USB host controller interface types (OHCI, UHCI and EHCI). To list what type of USB controller you have, you can use the command below:

```
lspci -v | grep HCI
```

Note:

My server returned:

```
00:0f.2 USB Controller: ServerWorks OSB4/CSB5 OHCI USB
Controller (rev 5) (prog-if 10 [OHCI])
        Subsystem: ServerWorks OSB4/CSB5 OHCI USB Con-
troller
```

so once I know my server I know I need to load the OHCI driver, not the UHCI or EHCI driver.

1. Logon to the Service Console as ROOT.
2. [Optionally] Load up the USB Device Drivers:

```
insmod usbcore
insmod usb-storage
modprobe usb-ochi or modprobe usb-uhci or mod-
probe ehci
```

3. Use the **dmesg | grep usb** to print out a list of active devices and scroll up to locate Initializing USB Mass Storage Driver….

   usb.c: registered new driver usbdevfs

   usb.c: registered new driver hub

   usb-ohci.c: USB OHCI at membase 0xd20c6000, IRQ 18

   usb-ohci.c: usb-01:00.0, Advanced Micro Devices [AMD] AMD-8111 USB

   usb.c: new USB bus registered, assigned bus number 1

   usb-ohci.c: USB OHCI at membase 0xd20c8000, IRQ 18

   usb-ohci.c: usb-01:00.1, Advanced Micro Devices [AMD] AMD-8111 USB (#2)

   usb.c: new USB bus registered, assigned bus number 2

   usb.c: registered new driver hiddev

   usb.c: registered new driver hid

   Note:

   The system assigns an SCSI device ID to the USB device (even though it's likely to be IDE Laptop Disk if it's a portable hard-drive). The critical bits are references to the SDG. This tells me the USB device has been added to the end of all my other SCSI disks (sda, sdb, sdc). This will help me in the next stage which is creating a mount point and mounting the partition on the USB disk.

   Adding additional hard-drives to the system can upset this allocation of sd*n*.

4. **Create a mount point** with:

```
mkdir /mnt/usbdisk
```

5. Mount the first partition on the disk with:

   ```
   mount /dev/sdg1 /mnt/usbdisk
   ```

   Note:

   If you are unsure about the partition scheme on the disk you can use **fdisk –l /dev/sdg** to print the partition table.

6. **List Files** and Start using the disk with

   ```
   ls –l /mnt/usbdisk
   ```

7. **Unmount the USB device when finished** using:

   ```
   cd /
   umount /mnt/usb
   ```

Managing Groups & Users

Creating User and Groups

You can create groups and users local to the ESX host using the groupadd, useradd and set passwords with passwd command. If I wanted to create a group called “esxops” I would use:

```
groupadd esxops
```

To create a user with the login name of herolds, and friendly name of Scott Herold I would use:

```
useradd –c “Scott Herold” herolds
```

By default all users created this way are disabled until you set a password for them. You can do that with:

```
passwd herolds
```

Note:

Without changing the password libraries used in ESX, passwords can be of any length and complexity, even dictionary words.

If you want to set a password during the creation of the user, you can do this by specifying an MD5 hashed version of the password. Below is a sample where the long string in little 'quotes' represents the password of *password*

```
useradd -p '$1$Rg69B9QA$JUtqStBrjNFbyzyP9zTsf0' -c "Mike
Laverick" lavericm
```

If you want to know an MD5 hash for a given password all you have to do is create a user, and set the password. Then cat the contents of /etc/shadow. This will list the username and their corresponding MD5 hash like so:

```
lav-
ericm:$1$Rg69B9QA$JUtqStBrjNFbyzyP9zTsf0:13587:0:90:7:::
bob:$1$ZIx4R9BE$PDniTkVdtrnyOH0uJUEcI.:13589:0:90:7:::
herolds:$1$CHrrCI79$Zt.YgyzbMjhFNuM62pUg30:13589:0:90:7:
::
ron:$1$QYIt/f1V$8iHZEezwJ7WtTwkWMb0xX1:13589:0:90:7:::
```

The MD5 hash is value-highlighted in bold between the two colons. Users Bob, Herolds, and Ron all have the same password of "vmware" but MD5 hash is not a clear text representation of the password. So even if you use the same password the MD5 hash always changes.

MD5 hash is useful in post-configuration options in kickstart scripts for creating users without showing the password in the kickstart script itself.

Configuring SU

By default, anyone who knows the root password can use the SU command. SU can be used to Switch User, and is most commonly used to become root. We can offer some security against that by configuring the special group called "wheel." When the wheel group is configured only its members have the rights to use su – command. Additionally, a log of users who do elevate their rights is logged in /var/log/messages.

Effectively, this means that to use SU you must know both the root password and also be member of the wheel group. Root access can then be quickly granted or denied by changing group membership without forcing a password reset on the root account.

1. Logon to the Service Console as ROOT.
2. Edit the /etc/hosts.allow file with:

   ```
   nano –w /etc/hosts.allow
   ```

3. Add the line:

   ```
   vmware-authd:   <IP>: allow
   ```

   Note:

   Where <IP> is the IP address of your server.

4. We can then add a user to the Wheel group with:

   ```
   usermod -G wheel herolds
   ```

   Note:

   usermod is used to modify existing users – so here user herolds is being added to the wheel group. If you want to add users to groups as you create them you would use:

   ```
   useradd -p '$1$Rg69B9QA$JUtqStBrjNFbyzyP9zTsf0'
   -c "Scott Herold" herolds -G wheel,esxops
   ```

   This creates a user called Scott Herold with a password of password and added to the wheel group. You can add a user to additional groups by separating them with a comma.

   You can view group membership by cat /etc/group. Within this file you will see the groups followed by a list of users separated by commas.

5. Next we will edit the /etc/pa.d/su file to enable the wheel feature

   ```
   nano –w /etc/pam.d/su
   ```

6. remove the # comment on the line:

```
#auth       required
/lib/security/$ISA/pam_wheel.so use_uid
```

7. Save and Exit nano.

Note:

If you login with an ordinary user account and attempt to use su – and you are not a member of the wheel group you receive this message:

```
Password:
su: incorrect password
```

But if you are member of the wheel group – access is granted.

Configuring Sudo

Sudo is a delegation system. If you are familiar with Windows, it is not unlike the "Run As.." option which allows you to run tools and utilities under the context of another user. More specifically, this allows the root administrator in ESX to allow ordinary users to use high-level commands normally only available if you are logged in as root - without having to disclose the root accounts password. This is even more flexible than the su - command which requires you to know the password of root to proceed. Sudo is configured by text file called /etc/sudoers and has its own editor called visudo. The visudo utility is a hybrid of the Linux VI text editor tool, but additionally it locks the /etc/sudoers file (to prevent administrative conflicts during configuration) and ensures that the syntax of the file is correct after you save and exit. This syntax checking is the single biggest reason to use visudo instead of vanilla text editor.

SUDO has a number of features. We can use wildcards to allow us to configure the sudoers file once, and then copy it to many servers. We can create command-alias, which acts as grouping of commands. For example, we could create a command alias called “ESXCMD” that represents all the esxcfg commands. When we come to grant the right to run the esxcfg commands we only have to mention “ESXCMD” once in the sudoers file rather than list every command. Once this has been done we can use our user groups in ESX to say which

commands alias are accessible to our operators. Below is a simple example of how I would give my ordinary ESX operators access to advanced commands:

1. **Logon to the Service Console as ROOT.**
2. Type the command:

   Visudo
3. Press [**I**] to enter **Insert Mode.**
4. Below the entry which begins # Host alias specification add the Host_Alias option:

   # Host alias specification
   Host_Alias ESXSERVERS = 192.168.3.0/24

   Note:
   This IP address which is specified in the CIDR format allows the sudoers file to be portable to other ESX host in the same network ID.
5. Locate the entry which begins # Cmnd alias specification and add some test command alias options:

   # Cmnd alias specification
   Cmnd_Alias ESXCMD = /usr/sbin/esxcfg-NIC's, /usr/sbin/vmkfstools
   Cmnd_Alias NOEXEC = /usr/sbin/esxcfg-vswitch, esxcfg-vswif

   Note:
   Here I am creating two command aliases. ESXCMD will be used to allow a group called esxops access to the commands vmkfstools and esxcfg-NIC's. In contrast NOEXE will be used to stop access to the esxcfg-vswitch and esxcfg-vswif.
6. Below the lines:

   # Uncomment to allow people in group wheel to run all commands
   # %wheel ALL=(ALL) ALL

   Uncomment the # so the %wheel users are not restricted and add another group entry to control the ESX operators.

   # Uncomment to allow people in group wheel to run all commands
   %wheel ALL=(ALL) ALL
   %esxops ESXSERVERS = ESXCMD, !NOEXE

   Note:
   The exclamation mark next to the NOEXE command alias means do not allow access. I created esxops group earlier with the groupadd command:

7. Press [**ESC**] to exit insert mode and enter command mode.
8. Type **:w** to save your changes.
9. Type **:q!** to exit the visudo.

Testing your configuration:

Firstly, make sure that you have a user in the group you gave access to with sudo.

Login as the user:

Type the command **sudo /usr/sbin/vmkfstools**

Type the command **sudo /usr/sbin/esxcfg-NIC's**

If you have been following this guide to the letter these two commands should execute perfectly well. The next two commands should result in an error message:

Type the command **sudo /usr/sbin/esxcfg-vswitch**

Type the command **sudo /usr/sbin/esxcfg-vswif**

```
"Sorry user X is not allowed to execute '/usr/sbin/
esxcfg-vswitch -l' as root on esx1.vi3book.com"
```

User X can use sudo -1 to see which commands they could run.

Note:

In this case I choose to use the wheel group as a way of granting access to SU. This example could be modified to limit the wheel group to set off commands, and deny them access to the SU command.

SSH Login Banners

Many businesses have security policies that enforce a pop-up message warning about unauthorized access to a given system. In some locations such warnings are required if you wish to take legal proceedings against offenders who breach security. Without such a warning offenders can sometimes be acquitted on a mere legal technicality. To enable a banner for SSH access:

1. **Create a message file** with something like:

   ```
   nano –w /etc/ssh/banner.txt
   ```

   Note:

   Cut and paste your business standard message.

2. **Save and Exit nano.**
3. **Then edit sshd_config** to use the banner message:

   ```
   nano –w /etc/ssh/sshd_config
   ```

   locate the #banner entry

   remove the comment and add

   ```
   banner /etc/ssh/banner.txt
   ```

4. **Save and Exit.**
5. **Restart SSH** with:

   ```
   service sshd restart
   ```

   Note:

   The banner message appears after the user has supplied a user-name, but before the user supplies a password.

   Note:

   The sshd_config has an AllowUsers and DenyUsers option if you wish to allow a user Service Console access but not via SSH:

   ```
   AllowUsers lavericm ron herolds
   ```

```
DenyUsers beavers klinek
```

Users beavers and klinek would be able to log on but only at the physical console using say an ILO card.

Setting up LDAP Authentication on ESX

The Service Console is Light-weight Directory Access (LDAP) aware; it is possible to configure ESX to authenticate against Active Directory or any other LDAP complainant directory service. This allows you to centralize access from LDAP system and password resets. To understand why might you want to do this you need to understand how Service Console user accounts work.

Even with VirtualCenter, if you want to give every employee in the IT department access to the Service Console with PuTTy you would have to create many accounts within ESX. This can be a burden. If we have 10 ESX hosts with 10 employees, I would have to create 100 users altogether – ten accounts on each server. This is because the Service Console model for users is essentially a workgroup model. No attempt by default is used to authenticate to an external database of users.

If a user needs to change their password they would have to login 10 times to 10 different ESX hosts, and use either VI Client or the Linux command "passwd" to reset their password. Even if we used the same user name 10 times on 10 different ESX hosts – and the same user name in Active Directory – the user could have potentially 11 different passwords to remember (10 on 10 ESX hosts, and one in Active Directory).

The next disadvantage comes when I want to disable access for a single employee leaving the business. I would have to disable or delete the user 10 times on ten different ESX hosts.

Configuring ESX to authenticate against LDAP would solve *some* of these problems. Active Directory could then be used to reset passwords and disable user accounts. However, it does not fix the first problem – creating the users on each ESX. Nor does the system support deleting the user accounts when they no longer required. There are a number of ways of dealing with this problem. You could buy commercial software which synchronizes named users and groups in your directory service – this software automates the creation and dele-

tion of users as you create and delete them in your LDAP service. Alternatively, a number of talented users on the forum have written scripts that do all this for you for free.

However, these end-user scripts do not come with commercial support, and some large companies might balk at the thought of using scripts that manipulate their LDAP environment for obvious reasons. Steve Beaver, who reviewed this very book, wrote one such excellent script. Steve currently works in Florida, and has co-authored a book about scripting in VMware. You can find Steve's ldap_search tool hosted on Scott's www.vmuguru.com site.

For now we will focus on creating users on the command-line in the Service Console, and reconfiguring ESX to speak to Microsoft Active Directory (Windows 2003 with Service Pack 1).

VMware now provide a handy tool called esxcfg-auth which will allow us to set up the PAM. This is easier than it used to be as the utility edits all the files for us, rather than us having to do that manually.

GOTCHA:

Before you even bother using the esxcfg-auth command, confirm with the ESX host that you can use ping or nslookup against the name of your Active Directory domain. Secondly, verify that the time of the Active Directory server is the same (offset by any time zone difference obliviously) as the ESX host. PAM has a very low tolerance of time difference between the LDAP server and the LDAP Client (the Service Console).

1. Open a session at the Service Console as ROOT.
2. Confirm you have DNS name resolution to the Active Directory domain with

   ```
   nslookup mydomain.com
   ```

3. The syntax of the esxcfg-auth command is:

   ```
   esxcfg-auth --enablead --addomain mydomain.com
   --addc mydc.mydomain.com --enablekrb5 --
   ```

```
krb5realm=mydomain.com --krb5kdc
mydc.mydomain.com --krb5adminserver
mydc.mydomain.com
```

Note:

The switches --enablead, --addomain and --addc are required; if you wish to support Kerberos authentication then you must use the --krb5realm which requires --krb5kdc and krb5adminserver. If you use Kerberos the escfg-auth command will automatically open kerberos and activeDirectorKerberos. Use esxcfg-firewall –q to view these as "Enabled Services."

So if I have a domain called vi3book.com with a domain controller called dc1.vi3book.com I would type:

```
esxcfg-auth --enablead --addomain vi3book.com -
-addc dc1.vi3book.com --enablekrb5 --
krb5realm=vi3book.com --krb5kdc dc1.vi3book.com
--krb5adminserver dc1.vi3book.com
```

Note:

Actually, naming a server like dc1 is quite poor practice. If DC1 was down we would find that users would not be authenticated. It is actually better practice to allow DNS lookups to find "Service Records" (type SRV) which assist in locating the Knowledge Consistency Checker (kdc). A better format for this command would be:

```
esxcfg-auth --enablead --addomain vi3book.com -
-addc vi3book.com --enablekrb5 --
krb5realm=vi3book.com --krb5kdc vi3book.com --
krb5adminserver dc1.vi3book.com
```

4. Next create a user at the Service Console *without a password:*

```
useradd testuser
```

Note:

Remember if you create a user with useradd which does not possess a password it is disabled by default. Next, open "Active Directory Users and Computers" and create a user in the Organization

Unit structure with the *same name as the user created with useradd* – set a password which corresponds with your datacenter policy.

You can easily validate your setup by changing the password in Active Directory and disabling accounts. If you disable testuser in Active Directory the user receives the Access Denied Message.

GOTCHA:

Currently esxcfg-auth only covers Active Directory authentication for console access. It does not work with the VI Client. If you want VI Client authentication to Active Directory you really need VirtualCenter.

Resetting a lost root password

If you forget or lose the root password, it is very easy to reset. There is no need for any special "root" kits or hacking tools. We simply reboot the ESX host and run it in a different run-level. Run-levels allow different levels of functionality to Linux/Unix based OS and are set in /etc/inittab. For example, run-level 3 allows high levels of functionality but does not run any GUI front-end. Whereas run-level 5 would allow high levels of functionality with a GUI front-end (if one had been installed). Run-level 1 runs the VMkernel or the Linux kernel in single user mode without prompting the user for a login or password. It logs you in automatically as root with a challenge. For this reason it's absolutely critical that you physically secure your servers, and password to protect any ILO, RAC or IP-based KVM system from unauthorized access.

To reset a lost root password:

1. Reboot your ESX host.
2. At the GRUB menu move the cursor to stop the timeout operation.
3. Select Service Console Only (troubleshooting mode).
4. Press [A] to pass arguments/parameters to the boot loader.
5. Type [1] to set temporary use of run-level 1 at the end of the prompt that reads

   ```
   mem=272M tblsht 1
   ```

6. then press [Enter].

7. Allow the boot process to continue.
8. At the sh-*n.nnn#* prompt type the command:

   ```
   passwd
   ```

   this will reset your root password as fits your datacenter policies.
9. Type the command:

   ```
   Reboot
   ```

   This will restart your ESX host.

Setting up NTP on ESX

The Network Time Protocol (NTP) client is used to keep the ESX host physical clock in synch with a more accurate time source. This is critical configuration because in 99% of the time VMs receive their time updates via VMware Tools from the ESX host. I say 99% of the time because there some circumstances where the time configuration of a VM is different. A case in point is running Active Directory inside a VM and the domain controller is Primary Domain Controller Emulator (PDC). You should consult the VMware KB article 1318 which outlines the special considerations required to make time synchronization work for AD.

http://www.vmware.com/support/kb/enduser/std_adp.php?p_faqid=1318

As we saw, time can create dependencies with other systems such as authenticating to LDAP service such as Active Directory at the ESX host.

Most ESX hosts are not given access to the Internet for security reasons so it is likely you will need to point them to an internal NTP service which then in turn communicates to the Internet for a publicly accessible list of NTP servers. When you select a pool of NTP servers on the Internet as your time source it is recommended you select one close to your geographic location. Most NTP servers on the Internet will use a geographical DNS sub-zone to indicate this, such as “europe.pool.ntp.org.”

You might find these URLs useful when looking for a publicly accessible NTP service:

http://www.pool.ntp.org/

http://www.ntp.org/

NTP servers are listed by their stratum number. The smaller the number the close your source NTP server is to the most accurate time source currently available from an atomic clock. NTP is UDP-based using port 123, so this will need opening on the connection to the Internet for this work.

Setting up a Simple NTP Server/Client on Linux

In this case the Linux service is going to be an external time source client and also server time to my ESX hosts. My example uses Redhat Linux as the NTP Server/Client.

1. Edit the /etc/ntp.conf file and locate the line:

   ```
   # restrict 192.168.1.0 mask 255.255.255.0
   notrust nomodify notrap
   ```

2. Uncomment the line and adjust the network ID to reflect the ESX host Service Console network. In my case:

   ```
   restrict 192.168.3.0 mask 255.255.255.0 notrust
   nomodify notrap
   ```

3. Locate the section of the ntp.conf which begins

   ```
   # server mytrustedtimeserverip and add the list
   of NTP servers.  Below is my list of publicly
   accessible NTP hosts from ntp.org:
   # server mytrustedtimeserverip
   server 0.uk.pool.ntp.org
   server 1.uk.pool.ntp.org
   server 2.uk.pool.ntp.org
   server 3.uk.pool.ntp.org
   ```

Note:

Here I change the entry to reflect my IP settings.

4. Save the ntp.conf file.
5. Start the ntp daemon and ensure it starts with every subsequent reboot with:

```
service ntpd start
chkroot ntpd on
```

Note:

You can check your configuration with ntpq –p. If you do not receive a positive response, check that your firewall within Linux allows UDP 123 outgoing and that your physical firewall has rules enabled to allow your NTP server access to the internet. Additionally, you might also wish to check DNS name resolution to the external Internet-based NTP servers.

Enabling the NTP Client on an ESX Host

1. Open a Session to the Service Console as ROOT.
2. **Open the ESX firewall to allow the NTP Client** with:

```
esxcfg-firewall -e ntpClient
```

Note:

Confirm this was successful with esxcfg-firewall –q

3. You can enforce time synchronization with the command-line tool:

```
ntpupdate –u ntp1.vi3book.com
```

Note: Positive Responses

In my case

```
19 Mar 10:50:02 ntpdate[3308]: adjust time
server 192.168.3.211 offset 0.010533 sec
```

Note: Troubleshooting Negative Responses

If you fail to receive a positive response – try ping ntp1.vi3book.com. If you receive responses check that the firewall on the NTP Client/Server is open for UDP 123 Inbound. You can confirm this by checking if the service is accessible by using telnet 123 ntp1.vi3book.com from a machine that is on the same network as the ntp.

4. **Edit the /etc/ntp.conf** file and locate the line:

```
# restrict 192.168.1.0 mask 255.255.255.0
notrust nomodify notrap
```

5. Uncomment the line and adjust the network ID to reflect the ESX host Service Console network. In my case this would be:

```
restrict 192.168.3.0 mask 255.255.255.0 notrust
nomodify notrap
```

6. **Locate the section of the ntp.conf which begins:**

```
# server mytrustedtimeserverip
```

and add list of NTP servers. Below is my list of internally accessible NTP hosts for vi3book.com

server mytrustedtimeserverip

server ntp1.vi3book.com

server ntp2.vi3book.com

Note:

We can list more than one NTP server or use DNS round-robin to provide fault-tolerance to the NTP source. You may also wish to add these entries to your /etc/hosts file to cover yourself from potential DNS failures.

7. Save the ntp.conf file.

8. **Start the ntp daemon and ensure it starts with every subsequent reboot** with:

```
service ntpd start
chkroot ntpd on
```

Note:

Again, you can check your configuration with the ntpq –p command.

Renaming an ESX Host

Occasionally, you might need to change the identity of an ESX host. There is a simple method from the VI Client and a more convoluted method from the command-line. The advantage of renaming an ESX host from the command-line is that you do not have to reboot your ESX host to make the change take effect. Remember if you rename an ESX host but do not update DNS then you could experience name resolution issues. Secondly, the old name of your ESX host will still be listed in VirtualCenter. It would be good practice to remove the old host from VirtualCenter, and add in the host under its new name. If you prefer to rename an ESX host with Vi-Client:

1. Logon to the VI Client.
2. Click the Configuration Tab.
3. Under the Software pane select DNS and Routing.
4. Click the Properties…
5. Under "Host Identification" change your name and domain as you deem appropriate.

   Note:

   A reboot is not enforced, but is required for your changes to take effect. This process also regenerates any SSL certificates created for the Web Access Service during installation.

From the command-line the same procedure can be achieved by editing files, and some simple commands to restart services and processes:

1. Logon to the Service Console as ROOT and type:

   ```
   nano –w /etc/sysconfig/network
   ```

2. Edit the **HOSTNAME= entry**

3. Exit nano and type:

   ```
   nano –w /etc/hosts
   ```

4. Edit the entry that reflects the name of your server.
5. If you are changing your Domain Name and DNS settings

   ```
   nano –w /etc/resolv.conf
   ```

   Note:

   You may wish to check you can ping a name in the new domain.
6. The command hostname will tell you current FQDN. We can use hostname to change the current hostname with:

   ```
   hostname esx10.newdomain.com
   ```

   Note:

   This steps saves you having to reboot the ESX host.

Configuring VMs

Understanding Different Virtual Disk Formats

Understanding the available disk formats is important for a whole number of operational tasks. A good example would be converting virtual disks from VMware Workstation or VMware Server into a format recognized by ESX. We can create virtual disks of different formats at the Service Console using vmkfstools with the –d switch. There are in fact eight different formats of virtual disk:

- **zerodedthick** (default)

  This disk is fully allocated at the time the virtual disk is created. This type of disk takes up the space on the disk you specify. So a 10GB zerodedthick disk would take up 10GB of space. Old deleted data is not purged from the disk except when the VM begins

to write to the physical disk. This format is used on SAN and iSCSI based storage.

- **eagerzeroedthick**

  This type of disk is similar to zerodedthick, except space is allocated during the creation of disk. The vmkfstools zeroes out space the disks takes up as it is being created on the command-line. This format must be used in the cluster-in-a-box scenario and takes much longer to create than any other format.

- **thick**

  This disk is fully allocated at the time the virtual disk is created – however stale data on the disk is not over-written. The free space is over-written as needed.

- **thin**

  Space for this format of disk is created on a on-demand, as-needed basis. This format is used if you store your virtual disks on NAS based storage.

- **2gbsparse**

  This format splits the disk up into a series of files no larger than 2GB. Each of the 2gbsparse files are linked together by a smaller text file (metadata) which describes the geometries of the original disk. This format is "safe" for moving to and from other operating systems like Microsoft Windows which historically has certain tools which corrupt files larger than 2GB. This format is sometimes used in VMware Workstation. Actually the size of the 2gbspares file is the amount of data in the virtual disk. So a 10GB virtual disk with 4GB of data would result in 6 files being created (5x2GB disks plus the metadata file). However, the actual space taken up on the physical disk would be only 4GB. This format is frequently preferred for backing up the virtual disk and general archiving purposes.

- **RDM (Raw Device Mapping File)**

  As we saw earlier in Chapter 5 and Chapter 10, an RDM file is not really a virtual disk although it does have the .vmdk extension. The RDM is actually a metadata file which tells the VM how to access

LUNs natively on an SAN or iSCSI system. RDMs can be created through the VI Client. This format uses virtual compatibility, treating the native LUN as if were a virtual disk, and enables features such as VMware's "Snapshot" Feature. Again, RDMs of this format can be used in backup scenarios for backing up VMs while they are powered on. A common misconception is that RDMs offer a performance benefit. They do not. Instead the term "RAW" is used to indicate "native" where the VM formats the LUN with its native files system.

- **RDMP** (RDM in Physical Compatibility "Pass-Through" Mode)

  This is a hybrid of the RDM which access the LUN using "physical compatibility." As we saw in Chapter 10 in the "physical-to-virtual clustering" scenario this allows direct access to an LUN via the ESX host and weakens VMkernel locking procedures in favor of the locking procedures used by the clustering software.

- **RAW**

  This format is not supported in ESX 3.x.x and is there for legacy purposes when it was supported in ESX 2.x.x. This format predates even the RDM format which was introduced into ESX 2.5.x.

You can quite easily work out the format of your virtual disk (Virtual Disk, RDM or RDMP) using either via the VI Client, however the command-line will tell you a great deal more information about your virtual disks. To find out more you can view the metadata or file descriptor information. All virtual disks actually have two files – the metadata file, followed by the virtual disk's file or files. Using the Linux command "cat" we can print to a console the contents of a virtual disk's metadata file.

The print out below this paragraph shows a 2bsparse file set. It will have as its "createType" what is called "twoGbMaxExtentSparse." In the extents description it shows a sparse virtual disk is assembled from a sequence of files listed with a serial number and the word "SPARSE" as the format. As with a conventional physical disk the file will also tell about disk geometries (Cylinders, Heads and Sectors) and what kind of SCSI adapter type it uses.

```
# Disk DescriptorFile
version=1
CID=00000000
```

```
parentCID=ffffffff
createType="twoGbMaxExtentSparse"

# Extent description
RW 4192256 SPARSE "w2k3-s001.vmdk"
RW 4192256 SPARSE "w2k3-s002.vmdk"
RW 4096 SPARSE "w2k3-s003.vmdk"

# The Disk Data Base
#DDB

ddb.adapterType = "lsilogic"
ddb.geometry.sectors = "63"
ddb.geometry.heads = "255"
ddb.geometry.cylinders = "522"
ddb.toolsVersion = "5153"
ddb.virtualHWVersion = "3"
```

This information isn't tremendously exciting or thrilling but it becomes useful when seen used in practical examples.

Creating Virtual Disks

1. Power off your target virtual machine…
2. To create a virtual disk use the vmkfstools command:

   ```
   vmkfstools –a lsilogic -c 10240m
   /vmfs/volumes/local-esx1-storage/vm1/vm1_1.vmdk
   ```

   Note:

   This is lower-case –c which creates a file, upper-case –C creates a vmfs file system. Here I am following the file-naming convention that VC would normally apply. The first disk would be vm1.vmdk, and subsequent disks would be serialized with vm1_1.vmdk and vm1_2.vmdk

   The command actually creates two files vm1_1.vmdk (metadata, of a couple of KB) and an vm1_1-flat.vmdk which is 10GB in size. I could have used 1g instead of 10240m on the command-line.

The –a option allows users to indicate which device driver should be used to communicate with the virtual disk. Failure to set this could cause a question in the virtual machine on power-up depending on what SCSI Adapter controller is used in the VM. Personally, I always use LSILogic as the driver unless it is unsupported in the guest OS I am working with.

3. **Next we need to add this into the virtual machine**. We can do this by **editing the VMX file:**

   ```
   nano –w /vmfs/volumes/local-esx1/vm1/vm1.vmx
   ```

4. add the lines:

   ```
   scsi0:1.present = "true"
   scsi0:1.fileName = "vm1_1.vmdk"
   scsi0:1.deviceType = "scsi-hardDisk"
   ```

   Note:

   scsi0:1 means it is the next disk after the boot disk, on the first virtual SCSI adapter. True means the device is connected, filename indicates the virtual disk (metadata file only), and the device type indicates this virtual SCSI disk, not a virtual IDE drive. It's worth saying that IDE drivers are not supported in ESX 2.x.x or 3.x.x and this can sometimes cause problems for VMware Workstation users who have used that format – more about this topic shortly.

5. **Save** your VMX file **and Exit nano.**

6. Power on your virtual machine with:

   ```
   vmware-cmd –l
   ```

   Note:

   That's a lower-case L for lima. This produces a list of registered VMs on the ESX host. We can use the vmware-cmd command with "start" option to power on a VM:

   vmware-cmd /vmfs/volumes/44a38c72-156b2590-be15-00065bec0eb7/vm1/vm1.**vmx start**

Note:

In the past we used to be able to use the friendly volume label. This is no longer supported as outlined in VMware KB 2122:

http://www.vmware.com/support/kb/enduser/std_adp.php?p_faqid=2122

Creating and Managing RDMs

There are two types of RDM – virtual compatibility mode and physical compatibility mode. These are normally set by radio button options in the VI Client. On the command-line the way these are specified is with two different switches. Remember that the main uses of RDMs are:

- Accessing existing data on a SAN/iSCSI
- Clustering inside VM'sVM's
- Running some (but not all) SAN Management Tools inside a VM

RDM for Virtual Compatibility

1. Before you begin, you may wish to check which LUNs your ESX server has access to with:

   ```
   ls -l /vmfs/devices/disks
   ```

2. Type the command:

   ```
   vmkfstools –a lsilogic -r
   /vmfs/devices/disks/vmhba40:0:0:0
   /vmfs/volumes/esx1-storage1/vm1/vm1_2.vmdk
   ```

   Note:

   This creates an RDM metadata file in /vmfs/volumes/esx1-storage1 using my iSCSI Lun on vmhba40:0:0:0. It creates two files as in the –c example, a metadata file called vm1_2.vmdk and a vm_2-rdm.vmdk file. Usefully, VMware automatically appends –rdm to the end of the file for us. The metadata file MUST be stored on a VMFS partition – it cannot reside on an NAS DataS-

tore. You do have to specify the last 0 in vmhba40:0:0:**0**. This last 0 indicates you wish to use the entire LUN. You cannot create an RDM file to a specific partition WITHIN an LUN.

3. Next we need to add this into the virtual machine. We can do this by editing the VMX file:

   ```
   nano -w /vmfs/volumes/local-esx1/vm1/vm1.vmx
   ```

4. add the lines:

   ```
   scsi0:2.present = "true"
   scsi0:2.fileName = " vm1_1.vmdk"
   ```

RDM for Physical Compatibility

Physical compatibility (required for physical to virtual clustering) is set up in a very similar way. With physical compatibility the SCSI Reservations/Filtering normally imposed by the vmkernel are "loosened" such that other systems that also want to impose SCSI reservations will work (such as clustering systems).

This is sometimes referred to as a Pass-Through RAW Device Mapping. In this case the switch is –z

- vmkfstools –a lsilogic -z /vmfs/devices/disks/vmhba40:0:1:0 /vmfs/volumes/esx1-storage1/vm1/vm1_2.vmdk

  Note:

  In this case the VMware labels the corresponding file vm1_2-rdm**p**.vmdk. We know this is a RAW Disk Mapping file with **p**hysical compatibility.

Viewing RDM Information

In the past it wasn't easy to view the contents of an RDM file, although you were able to use vmkfstools –l to list files. The command would allow you to see which files were virtual disks and which were RDMs.

There is now a switch to query the metadata file which then reports information:

1. Running:

   ```
   vmkfstools –q /vmfs/volumes/esx1-
   storage1/vm1_1.vmdk
   ```

   Will return:

   ```
   Disk vm1_1.vmdk is a Non-passthrough Raw Device
   Mapping
   Disk Id: vml.0100000000202020205649525455 41
   Maps to: vmhba40:0:0:0
   ```

2. Running:

   ```
   vmkfstools –q /vmfs/volumes/esx1-
   storage1/vm1_2.vmdk
   ```

   Will return:

   ```
   Disk vm1_2.vmdk is a Passthrough Raw Device
   Mapping
   Disk Id: vml.0100010000202020205649525455 41
   Maps to: vmhba40:0:1:0
   ```

   Note:

   You can also cat the contents of the metadata file as well – but the VM MUST be powered off first!

3. To cat the contents, type:

   ```
   cat /vmfs/volumes/esx1-storage1/vm1_2.vmdk
   ```

   Which returns:

   ```
   # Disk DescriptorFile
   version=1
   CID=0a8fee69
   parentCID=ffffffff
   createType="vmfsRawDeviceMap"
   ```

```
# Extent description
RW 8388608 VMFSRDM "vm1_2-rdm.vmdk"

# The Disk Data Base
#DDB
ddb.toolsVersion = "7172"
ddb.adapterType = "lsilogic"
ddb.geometry.sectors = "63"
ddb.geometry.heads = "255"
ddb.geometry.cylinders = "522"
ddb.virtualHWVersion = "4"
```

Exporting & Importing Virtual Disks

Those of you with some ESX 2.x experience will know importing (into a VMFS volume) and exporting (out of a VMFS volume) virtual disk has two main functions.

Firstly, it serves as a safe method of converting disks into a format that can be taken to/from "foreign" file systems such as NTFS and ESX3. Historically, these file systems have tools that could corrupt files bigger than 2GB in size. In the past, these two formats have been referred to as "Monolithic" and "COW" (copy-on-write). COW is no longer a favored term as it is technically inaccurate – instead we refer to it as the "Sparse" format. That is more or less still the same. However there is a new switch in vmkfstools (-d) that will allow you to control the format of the disk. The –d switch is not always required; it depends on whether you are exporting or importing.

Another purpose is for transporting virtual disks safely to/from other vPlatforms such as Workstation on Linux or Windows or VMware Server (nee GSX) on Linux or Windows. Additionally, perhaps you don't have VirtualCenter; you still have a stand-alone ESX servers – and you still use the import and export method of ESX 2.x.

GOTCHA:

For those of you from an ESX 2.x background you should be aware that the old vmkfstools –e switch is depreciated and the way you now do an export is confusingly with the –i switch with the new –d 2gbsparse switch.

Exporting "Zeroedthick" Virtual Disks (nee Monolithic) into the "Sparse" (nee COW) Format

In my case I am using /vmimages partition but you could use any EXT3 location or even export directly to a mount NFS or SMB share.

1. Type the command:

   ```
   vmkfstools -i /vmfs/volumes/esx1-
   storage/vm1/vm1.vmdk -d 2gbsparse
   /vmimages/vm1.vmdk
   ```

   Note:

   You should get a response like this:

   ```
   Destination disk format: sparse with 2GB maxi-
   mum extent size
   Cloning disk '/vmfs/volumes/local-
   esx1/vm1/vm1.vmdk'...
   Clone: 3% done.
   ```

   Note:

   The disk is now in a portable format… you could tar it up and copy somewhere else for backup purposes – connect with WinSCP on your workstation and bring the files down to your PC. If you have VMware Workstation on your PC you could configure a virtual machine to use this disk.

Importing Virtual Disk from the Sparse Format

For those of you familiar with ESX 2.x this process hasn't changed much at all. What is different is that VMFS now stores directories, and the VM's configuration files are held in a directory. The question now is: where do you restore the disk to? The easiest way to do this is first to create a VM with a virtual disk that is very small; this will then set-up up the directory structure for the configuration files. Then delete the small virtual disk. Next trigger the vmkfstools import to the vmfs volume and directory making sure you create a vmdk file with the same name of the smaller virtual disk created and deleted a moment ago.

1. Type the command:

```
vmkfstools -i /vmimages/workstationdisk.vmdk
/vmfs/volumes/esx1-storage/workstation1/ work-
station1.vmdk
Destination disk format: VMFS thick
Cloning disk '/vmimages/media.vmdk'...
Clone: 10% done.
```

Copying Files from one ESX host to another

SCP (Secure Copy) is the utility to use. This example copies a vmdk in its sparse format. Remember to use ESX as SSH client; this must be enabled in the ESX firewall first from the location that starts the SCP. You can use esxcfg-firewall –e sshClient. You will also need to do this if you want to ssh from one ESX host to another.

1. Type:

   ```
   scp /vmimages/vm1*.vmdk esx2.vi3book.com:
   /vmimages
   ```

   Note:

   Don't forget the colon: after the name of your server. If you do not specify a user (lavericm@esx2.vi3book.com) then SCP assumes you are using the root account at the destination.

   Note:

   It will warn you that the first time you copy the destination host

   ```
   " The authenticity of host 'esx2.vi3book.com'
   (192.168.3.102)' can't be established.
   RSA key fingerprint is
   92:4b:ba:b5:ca:31:6f:e7:8c:2d:00:6e:cf:c6:b6:ea
   ."
   ```

   This is because the certificate of the corresponding machine is untrusted and cannot be verified. To get rid of these first time messages you need to assign your own certificates or acquire them from a 3rd party.

2. Choose **Yes** to continue.

Note:

The system will then warn you that:

```
" Warning: Permanently added "esx2.vi3book.com
," 192.168.3.102' (RSA) to the list of known
hosts."
```

3. At the prompt which reads "**root@esx2.vi3book.com password**:" type the **other servers root password**, in my case **********

Note:

In this screen grab I copied the tar file for upgrade of esx 2.x to 3.x The system will give you a status bar like this:

```
vmware-esx-3.0.0.tar.gz 67% 309MB 9.2MB/s 00:15
```

Note:

By default root access is not enabled, unless you weaken security in the /etc/ssh/sshd_config file.

Renaming Virtual Disks

Renaming virtual disks used to be a simple process. You could just use the mv command and edit your VMX file accordingly to point the VM to the new file name. Now virtual disks have two files - their companion vmdk metadata file and the –flat.vmdk which has made it slightly more complicated. Fortunately, we now have a special vmkfstools switch to rename files and keep the metadata in synch with the –flat file. I guess you could manually edit the metadata file – but why bother when vmkfstools does it all for you? This tool is especially useful when a virtual disk is in the 2gbsparse format which is assembled from many files – and where manually renaming them would be a burden.

1. Before my rename the metadata in the vm1.vmdk looked like this:

```
# Disk DescriptorFile
version=1
CID=a896726b
parentCID=ffffffff
createType="vmfs"
```

```
xdgkdkfdlkfjldkjflkjsdf
# Extent description
RW 4194304 VMFS "vm1-flat.vmdk"
Xdgkdkfdlkfjldkjflkjsdf
# The Disk Data Base
#DDB
ddb.adapterType = "lsilogic"
ddb.geometry.sectors = "63"
ddb.geometry.heads = "255"
ddb.geometry.cylinders = "261"
ddb.thinProvisioned = "1"
ddb.virtualHWVersion = "4"
ddb.toolsVersion = "7172"
```

2. Type the command:

```
vmkfstools -E /vmfs/volumes/esx1-
storage1/vm1/vm1.vmdk m1-os.vmdk
```

3. After the rename the metadata looks like this:

```
# Disk DescriptorFile
version=1
CID=a896726b
parentCID=ffffffff
createType="vmfs"
xdgkdkfdlkfjldkjflkjsdf
# Extent description
RW 4194304 VMFS "vm1-os-flat.vmdk"
Xdgkdkfdlkfjldkjflkjsdf
# The Disk Data Base
#DDB
ddb.adapterType = "lsilogic"
ddb.geometry.sectors = "63"
ddb.geometry.heads = "255"
ddb.geometry.cylinders = "261"
ddb.thinProvisioned = "1"
ddb.virtualHWVersion = "4"
ddb.toolsVersion = "7172"
```

Note:

Next you would manually edit the VMX file to reflect your changes.

Resizing Virtual Disks

According to VMware documentation on the forum, the –x switch is only supported for making virtual disks larger – not smaller. This has been the case in ESX 2.x. This surprised me – because in the past I have successfully used –x to make a virtual disk smaller. This however, was way back in the mists of time, circa ESX 2.0 (around 2004). So perhaps they withdrew it as a function as it is very easy to corrupt files when you make virtual disks smaller. If you want to clone a big virtual disk to a small one perhaps you could use a disk cloning utility like Symantec Ghost or PowerQuest DriveImage Pro.

While it is relatively easier to increase the size of the VMDK file, Windows does not give you an easy way to reduce or increase the size of a partition. Therefore 3rd Party tools may be required to complete the procedure – especially if the partition is the boot partition. There are five main methods:

- Partition Tools – you can use Power Quest Partition Magic on a Client based OS, to resize the partition or Power Quests Volume Manager on Server based OS.
- If you have Windows 2003 (or Windows 2000 with the Resource Kit) you can go through a procedure using Microsoft's Diskpart utility. It's not a very friendly process and is quite convoluted. Its main advantage is that it is cheap – as you do not need 3rd party tools to achieve it. Its disadvantage is that it is not available for NT4, as far as I am aware. For more information about this you could visit Dominic Rivera's rather useful vmwareprofessional.com website.
- DR method – Backup OS System State to another location – Create a new OS disk file and restore the data.
- Ise the Knoppix Boot CD (http://www.knoppix.org) and the qparted disk repartitioning tool.
- Drive Cloning Method using Symantec Ghost or PowerQuest Drive Image Pro – Create a new disk of the appropriate size – and restore the data using a disk to disk restore method.

My personal favorite is the last one – as it can be used to make virtual disks safely larger and smaller. I've also used it to convert VMware Workstation IDE

drives into SCSI Disks. My second favorite method is using the Knoppix Boot CD, mainly because it is free and handles many different file systems.

It's worth knowing the alternatives in case you cannot get your preferred method to work. You should expect a number of reboots within the guest operating system – at least one created by the Partitioning tool, and the one called for by the guest operating system (especially Windows). This can be somewhat annoying!

GOTCHA:

Backup up the VM in case the following procedure fails!

Note:

I noticed that Knoppix is susceptible to disconnections from the Remote Console especially when you are shutting down the VM after booting from the ISO.

1. Shutdown the VM.
2. Type the following command:

   ```
   vmkfstools –X 6144m –force /vmfs/volumes/local/
   vm1.vmdk
   ```

   Note:

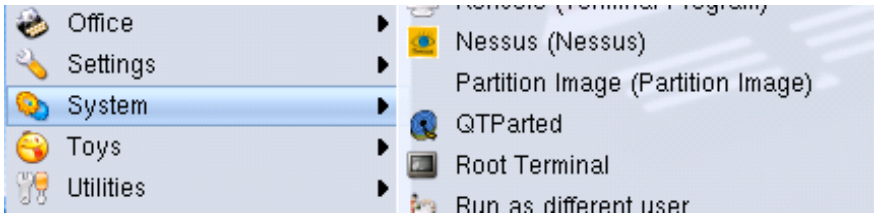
   If you follow this command with ls –l –h you will see it is bigger. Again I could have specified 6g instead of 6144m. It's the –flat file that changes size – the metadata file stays the same but has new geometry information
3. Start-up the VM, and check that it boots properly

   Note:

   You may get "error loading operating system" messages if the vmdk file has become corrupted.

4. Connect to the ISO which contains the Knoppix Boot CD - Using the Console attach to this ISO file making sure you enable X Connected at Power On.
5. Power on the VM.
6. Change the boot order on the VM to give the CD-ROM the highest priority. Allow the system to boot to Knoppix CD or use the [ESC] key to select the CD-ROM as a the boot device.
7. Click the Knoppix KDE button.

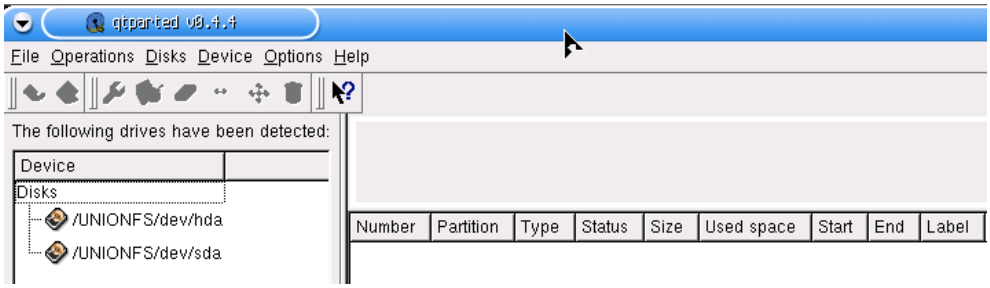
8. Choose System Tools.



9. Choose Qparted.

Note:

Qparted will display the partition/disk layout of the virtual machine.

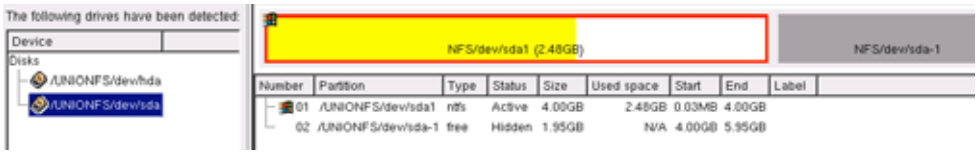


In this case, the IDE CD-ROM and single SCSI boot disk (sda.)

10. Select /inionfs/dev/sda

Note:

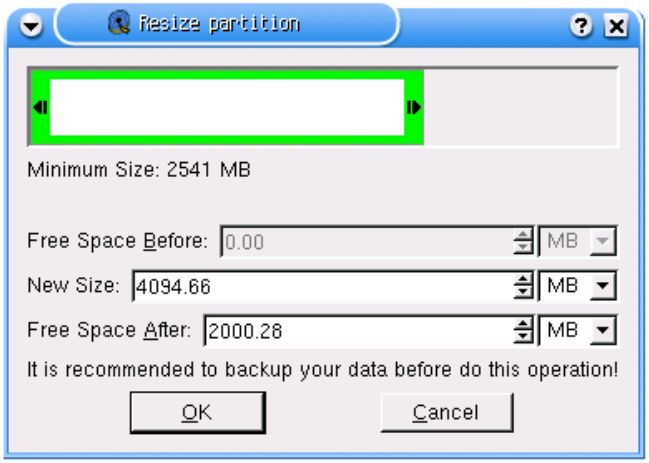
This then shows the partition table on that virtual disk.



In this case a single C drive of 4GB with free space on the hard-drive of 2GB added by using vmkfstools –X 6096mg.

11. Select Partition No 01

12. Right-click and choose &Resize.
13. In the dialog box drag to increase or decrease the size of Partition 01.



14. Click OK
15. Click File, and choose &Commit.

Note:

Confirm the warnings by clicking the Yes button – and watch the dialog boxes go by - and Confirm the Success dialog by clicking OK.

16. Shutdown the Virtual Machine.
17. Disconnect the CD-Rom.

    Note:

    This will allow the VM to boot to the VMDK file rather than to the CD-Rom.

18. Reboot the Virtual Machine.

    Note:

    In Windows 2003 you will see check disk events – this appears to be normal. Additionally, you will find Windows 2003 will detect the hard-drive as if it was a new device and request a reboot.

Converting Workstation VMs to ESX VMs using VMware Converter

In the past we used very long and convoluted manual methods to convert Workstation virtual disks. VMware released a beta tools some months ago called "VMware Importer" used mainly to covert Microsoft virtual disks and VMware Server/Workstation disks into the ESX format. This import functionality has been rolled up into the recently released "VMware Converter" tool. The VMware Converter tool is free in its non-enterprise format, called "Starter Mode."

GOTCHA:

Currently VMware Converter is fully supported for Windows VMs including the 64-bit versions of Windows 2003 and XP. However, converter only has experimental support for Linux. This is what the release notes currently say:

"Support for the following guest operating systems is Experimental. VMware Converter 3 can clone source images containing these operating systems, but the destination virtual machine may or may not work without additional configuration after import. In particular, if the source image contains unsupported hardware, you may need to modify the configuration of the destination virtual machine before using it"

- Linux
- Windows NT 3.x
- Windows ME
- Windows 98
- Windows 95
- MS-DOS"

It's worth mentioning that the only operating system listed here that is officially supported on ESX 3.x.x is Linux.

1. Download and install VMware Converter your PC where your VMware Workstation resides.
2. Load VMware Converter, and click the button labeled Continue in Starter Mode.
3. Click the Import Machine icon.
4. First select the source VM.
5. Choose Next.
6. Choose Stand-alone machine.
7. Click the Browse button and locate the VMX file of the VMware Workstation VM.
8. Retain the default of Import all disks and Maintain Size.
9. Then select the destination location.
10. Choose VMware ESX server or VirtualCenter virtual machine.
11. Enter the name of your VirtualCenter server, and user account details.
12. Type in the name of the new VM, and Select the folder in VirtualCenter to hold the VM.
13. Select an ESX host or DRS/HA Cluster to location for your VM.

14. Select a VMFS or NAS datastore.
15. Select the Network options.
16. Enable Install VMware Tools.

    Note:

    The option to customize the VM runs a guest customization wizard which allows you retain your original VM, and sysprep the converted VM.

17. Click Next and Finish.

    Note:

    You can watch the status of your conversion from the converter windows in a percentage. Additionally, the "Task Progress" tab will give you an overview of the steps the converter is completing.

GOTCHA:

When you first power on the converted workstation you will be asked to deal with the UUID value. This is because the VM has been cloned to a new physical location outside the scope of VirtualCenter. If you ran the guest customization wizard you should choose to generate a new UUID; if you intend to decommission the old VMware Workstation VM you should opt to keep the UUID. This is the difference between a clone to copy, and a clone to move a VM.

Manually Converting Workstation VMs to ESX VMs

If VMware Converter doesn't work for you, this manual method always works without failure. One reason might be that you are trying to convert a guest operating system which is only experimentally supported with the VMware Converter tool. This manual process of conversion does take a significant number of steps. An alternative to this process is using a free utility called IDE2SCSI which is available on the sanbarrow.com website as part of the MOA project.

This long title actually explains an occasional problem some users have bringing virtual disks from VMware Workstation into ESX server. In contrast to ESX, VMware Workstation supports 4 different formats which include IDE or SCSI and the disks can either be single "growable" disk (type 0) or a "growable" disk split in 2GB chunks (type 1, or what ESX user would called 2gbsparse). Other formats can be created from the command-line using the VMware Workstation utility vmware-vdiskmanager.exe which include:

- Type 0 - single growable virtual disk
- Type 1 - growable virtual disk split in 2Gb files
- Type 2 – pre-allocated virtual disk
- Type 3 – pre-allocated virtual disk split in 2Gb files

By "growable" VMware means the files dynamically take up more space on the disk as data is written to them. In contrast a "pre-allocated" disk is more like the ESX default format where a 10GB virtual disk uses 10GB of space. Growable disks are popular in VMware Workstation where local disk space on a PC or laptop may be limited. Additionally, one thing to beware of is that the default and recommended virtual disk format is IDE in VMware Workstation 5.x. VMware Workstation sets IDE whether you choose a Typical or Custom when creating the VM. This can cause further problems when bringing a Workstation disk into ESX.

These are barriers or problems some people experience in bringing VMware Workstation VM'sVM's into an ESX environment:

- IDE format is not supported in ESX but is a default in VMware Workstation
- ESX does not appear to like the Type 0 disk from VMware Workstation
- Virtual disks must be copied to ESX in the 2gbsparse format, and converted into one of the other formats (eagerzerodthick, eagerthick, or thick)
- Occasionally, the guest operating system lacks a driver for lsilogic or buslogic

This procedure is a good example of using VMware Workstation command-line tools together with ESX to move a VM into ESX – it also draws upon our capacity to access information in various virtual disk metadata files. I've had to use these occasionally with Virtual Appliances which are created in an IDE/preallocated format which means they cannot immediately be used on the ESX platform – and where VMware Converter refuses to work with that guest operating system.

Such a virtual appliance is the Ultimate Deployment Appliance (UDA) covered earlier in this chapter. A VM with this kind of disk IDE and Single Growable would have vmdk metadata like so:

```
# Disk DescriptorFile
version=1
CID=a9a0b01d
parentCID=ffffffff
createType="monolithicFlat"

# Extent description
RW 4194304 FLAT "uda13-flat.vmdk" 0

# The Disk Data Base
#DDB

ddb.virtualHWVersion = "4"
ddb.geometry.cylinders = "4161"
ddb.geometry.heads = "16"
ddb.geometry.sectors = "63"
ddb.adapterType = "ide"
ddb.toolsVersion = "0"
```

VMware KB 1881 is a knowledge base article which outlines the basics of manual conversion. It is an old KB article but it is still relevant. These are the stages it outlines, and I have added a few more to make it 100% accurate and complete:

- Duplicating the IDE Type 0 disk to a IDE Type 1 format
- Gather information about the IDE Type 1 disk to carry out step 3
- Adding a SCSI disk to the VM using the Type 1 format

- Editing the VMware Workstation VMX file to add a SCSI adapter
- Powering on the VM to confirm access to the SCSI Device
- Editing either the Type 1 or Type 2 disk to make it into a 2GB Sparse SCSI disk
- Transfer metadata information from the SCSI disk to the IDE Type 1 disk created in Step 1
- Uploading the disk created at Step 6 to an ESX host
- Create a new VM in ESX
- Use vmkfstools to import the 2gbsparse disk copied at step 7

Step 1: Duplicating the IDE Type 0 disk to a IDE Type 1 format

1. We begin first by backing up the virtual disk/machine.
2. Next we need to create growable 2GB sparse VMDK using the VMware Workstation vmware-vdiskmager utilities; in the case of UDA1.3 the following would:
3. Open a command-prompt and type:

   ```
   cd\Program Files\VMware\VMware Workstation
   ```

4. Type the command:

   ```
   vmware-vdiskmanager.exe -r <path>\uda13.vmdk -t
   1 <path>uda13-growable2gb.vmdk
   ```

   Note:

   The command-line tool will give you progress information. The –r switch indicates we are converting the disk, the first path indicates the source vmdk file, and the –t is the type switch indicates we need a type 1 Mware Workstation disk (a vmdk in the growable 2gbsparse format).

Step 2: Gather information about the IDE Type 1 disk to carry out step 3

1. Type the contents of the growable 2gbsparse disk with:

   ```
   type <path>uda-growable.vmdk
   ```

2. This should print out information like this to the command-line:

   ```
   # Disk DescriptorFile
   version=1
   CID=a9a0b01d
   parentCID=ffffffff
   wcreateType="twoGbMaxExtentSparse"

   # Extent description
   RW 4192256 SPARSE "uda13-growable-s001.vmdk"
   RW 2048 SPARSE "uda13-growable-s002.vmdk"

   # The Disk Data Base#
   DDB
   ddb.virtualHWVersion = "4"
   ddb.toolsVersion = "0"
   ddb.geometry.cylinders = "4161"
   ddb.geometry.heads = "16"
   ddb.geometry.sectors = "63"
   ddb.adapterType = "ide"
   ```

3. To create a similar sized virtual disk in the SCSI format we need to sum the size of the growable disk by the number sectors. These are numbers I have formatted in bold. Basically add up all these values for each of the sparse files. In my case that's 4192256+2048=4194304.

Step 3: Creating a SCSI disk to the VM using the Type 1 format

1. Open a command-prompt andy type:

   ```
   cd\program Files\VMware\VMware Workstation
   ```

2. Type the command:

```
vmware-vdiskmanager.exe -c -s  4194304 -a lsi-
logic -t 1 <path>uda13-scsidisk.vmdk
```

Note:

The switch –c is used to create a new virtual disk, and –s sets the size of the disk sectors. The –a switch sets which adapter type the disk will use (lsilogic/buslogic). If you are doing this for Windows beware that different Windows flavors support different drivers as part of the i386 code. For example, Windows 2000 and NT4 support the Buslogic Driver whereas Windows XP and 2003 support the LSIlogic Driver. You may have to be ready to supply a driver to the Windows operating system. In my case I chose lsilogic as I prefer it – the Fedora Core 5 operating system is the guest OS of UDA 1.3 – and support LSIlogic adapter.

This should take a the fraction of the time our clone/covert did previously, as it merely has to create a file with the correct geometries but does not have to copy any data.

Step 4: Adding in the SCSI Disk, Power On and Check SCSI Disk is found

1. In VMware Workstation click Edit Virtual Machine Settings.
2. Click Add.
3. Click Next, Select Hard Disk and Click Next.
4. Choose Use an Existing Virtual Disk.
5. Browse to the SCSI disk created in Step 3.
6. Click Next and Finish.
7. Using a text editor (not Windows NotePad!) edit the VMX file for the VM and add an SCSI Device by adding after scsi0:0.present = "TRUE"

```
scsi0.virtualDev = "lsilogic"
```

8. Power On the VM.

   Note:

   You can check the SCSI disk is present by using fdisk /dev/sda in Linux. If you are using Windows, the operating system should detect this new device and plug and play it. Use Device Manager and Disk Management to check that the Windows operating system can see the new SCSI disk.

   Regardless of your operating system, this disk is un-partitioned and will not be mountable by driver letter or mounting point.

9. Shutdown the VM.

10. Remove both the IDE and SCSI Disk (but don't delete them).

Step 5: Modify the Metadata of the IDE "Growable 2GBSpare Disk"

1. In a text editor (not Notepad!) open the .vmdk file of the SCSI disk.

2. In a text editor (not Notepad!) open the .vmdk file of the Growable 2GB disk.

3. Select and copy all the text in the SCSI disk below the # Disk Data Base Line, in my case:

```
ddb.virtualHWVersion = "4"
ddb.geometry.cylinders = "261"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.adapterType = "lsilogic"
```

4. Paste this over the equivalent lines in the Growable 2GB Disk.

5. Close and Save your changes to the metadata of the Growable 2GB Disk.

Step 6: Transfer to the ESX Host, Create VM and Covert Virtual Disk

1. Using WinSCP or a similar tool, copy the Growable 2GB Sparse files to the ESX Host.

   Note:

   The location of /vmimages is a good place – but check you have enough space in whatever partition you choose before copying as you will want to avoid filling the / partition.

2. Create a new VM but with a 1MB virtual disk – Choose Custom, and make sure you select the right SCSI Controller type for your SCSI virtual disk (lsilogic or buslogic).

   Note:

   This will create the directory structure for the VM. We actually don't need the 1MB virtual disk, but it is impossible to create a VM with the VI Client without some kind of virtual disk.

3. Logon to the Service Console as ROOT.

4. Delete the 1MB virtual disk with the rm command, for example:

   ```
   rm /vmfs/volumes/virtualmachines/uda13/uda13.
   vmdk –f
   ```

   Note:

   The –f forces rm to delete both the metadata file and the –flat.vmdk file as well.

5. Use vmkfstools to import the "growable 2gbsparse" disk with:

   ```
   vmkfstools –i /vmimages/uda13-growable.vmdk
   /vmfs/volumes/virtualmachines/uda13/uda13.vmdk
   ```

6. Power on your VM in VMware ESX server.

Deleting Virtual Disks

There is a special command for deleting virtual disks – however we could easily use the Linux command:

```
rm namofvirtualdisk*.vmdk -f
```

This would delete the metadata and also the –flat.vmdk file… but vmkfstools can do it too with the command:

```
vmkfstools -U /vmfs/volumes/local/vm1.vmdk
```

Note:

It deletes both the metadata and the –flat file – and doesn't currently ask "Are you sure???."

Using VMware-cmd

VMware-cmd works by manipulating the VMX and sending instructions to the VM via the VMX file. There is lots you can do with this command beyond what is feasible in this document. For further information, look at the Scripting Guide for ESX. It also returns 1 for positive results, and 0 for failures – so you can use it for shell scripts with if statements:

Listing Registered VMS

To list Registered VMs, type:

```
vmware-cmd –l
```

Note:

List paths and names of VMX files, useful for when you have to type long paths to the VMX file. You can highlight an entry in the list and copy it to your

command-line. Unfortunately, vmware-cmd no longer supports volume names – instead you have to use the UUID path to the VMX file.

I will give one example of vmware-cmd with the UUID. After that I will replace the string with <UUID> because it is so long!

Registering and Un-registering a VM

To register or un-register a VM, type:

```
vmware-cmd -s unregister /vmfs/volumes/<UUID>/vm1/
vm1.vmx
```

Note:

This also includes register as a command. S stands for set. This un-registers a VM from an ESX host (stand-alone).

Unfortunately, it does NOT un-register it from VirtualCenter. Instead you are left with an "orphaned" VM that needs right-clicking and "Remove from the Inventory."

This happens because what vmware-cmd –s is manipulating is what used to be called the vm-list. It's not manipulating the VC database.

Note:

Why is the command still useful? If you had a server failure which you had to rebuild, you then need to "register" the VMX file with an ESX server to be able to manage it. If you had 40 VMX you wouldn't want to do that by hand using the GUI and the datastore browser utility.

Power Options on a VM

If you use trysoft option, the Guest OS will be shut down gracefully. The trysoft is a "mode" option – it tries to run the normal scripts but uses a hard shutdown/startup if the VM is not behaving properly. There are two other "modes" – soft which runs scripts but never does a hard start or stop, or "hard" which powers off/on a VM as if you had hit the power switch. Sometimes people use trysoft first, and then if the VM refuses to power off, they follow it with hard power off.

1. To power on a VM type:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   start trysoft
   ```

2. To power off a VM type:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   stop trysoft
   ```

3. To suspend a VM type:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   suspend
   ```

   Note:

   There isn't much in the way of status/progress here. The old-style "Remote Console" from ESX 2.x days used to give you a dialog box with a status bar.

4. To resume a suspended VM type:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   start trysoft
   ```

   Note:

   There is no resume switch on vmware-cmd – just a power-up which retrieves the suspend file, and resumes the machine.

5. To reset a VM type:

```
vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx reset trysoft
```

Note:

This is a soft reboot of the virtual machine.

6. To find out the **Power Status** of a VM use:

```
vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx getstate
```

Note:

It will return "On"; "Off"; "Suspend" and "Stuck" if the VM is waiting for interaction.

Finding the Heartbeat

To find the heartbeat, type:

```
vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx getheartbeat
```

Note:

You should see a number. Repeat the command. If the number increments the machine is alive. If it stays the same it is dead; it doesn't have a heartbeat.

Finding the Status Of Devices

The Vmware-cmd tools are able to find the configuration of the VMX file, and also change entries in the file. To find out the status of the CD-ROM you would use:

```
vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx getconfig
ide0:0.deviceType
```

Note:

The CD-ROM is an emulated (but not Virtualized) IDE 0:0 channel. This is how the CD-ROM is addressed in a Windows Environment. A safe way of learning the VMX variables is just print the VMX file to the console with:

```
cat /vmfs/volumes/esx-storage1/vm1/vm1.vmx | more
```

Note:

Cat is the same as the "type" command in Windows. It merely prints the contents of files to the command-line.

As long as you know the name of the variable in the VMX file you can change anything you like from the command line. For example, there is a variable called ide0:0.fileName = which controls the path for the CD-ROM, be it physical or an ISO image. There is a variable called ide0:0.deviceType = which controls whether it is the physical or ISO image (atapi-cdrom for the physical and cdrom-image if it is an ISO file). In this example we will attach an ISO to the CD-ROM and Connect the CD-ROM to the VM. You may wish to have VC Console open while you're doing this – to see visually the effect of your changes.

I assume that initially the CD-ROM is connected to the physical CD:

1. First, Disconnect the current device with:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   disconnectdevice ide0:0
   ```

2. Change the system to use an ISO file instead of a physical CD with:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   setconfig ide0:0.deviceType cdrom-image
   ```

3. Then set the image file to use with:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   setconfig ide0:0.fileName
   /vmfs/volumes/ISO's/w2k3.iso
   ```

4. To Refresh the OS - Reconnect the device with:

   ```
   vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
   connectdevice ide0:0
   ```

Note:

Now that the CD is set as an ISO file, you merely need to do stages 2 and 3 to switch from one ISO file another. The whole process could be put in SH script like so:

```
vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
disconnectdevice ide0:0

vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
setconfig ide0:0.deviceType cdrom-image

vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
setconfig ide0:0.fileName
/vmfs/volumes/ISO's/w2k3ent.iso

vmware-cmd /vmfs/volumes/<UUID>/vm1/vm1.vmx
connectdevice ide0:0
```

Scripted Installation Revisited

Now that we have a good knowledge of the main configuration commands, we can use this to further automate our scripted installations using the %post section of the kickstart.cfg file. We can use %post section to create a script file that executes when the ESX first boots. This script file contains the esxcfg commands. We can then use the %post section to set the permissions on this file to give it the Read, Write, and eXecute attributes. What is critical is that the post-script file must be executed *after* the ESX host has loaded the VMkernel. If the VMKernel is not loaded, or if the esxcfg post-script executes *before* the VMKernel has fully-loaded, it will fail.

I would like to thank the http://www.brianshouse.net/hp/ website. I lifted some of this content from his samples used with the HP RDP product.

Below is a basic sample from a %post section of a kickstart.cfg. A fuller version can be downloaded for free from http://www.vi3book.com/downloads/ks.cfg. If you do open this file in Windows make sure you use WordPad and not notepad, as notepad will corrupt the file. Anything between <<EOF1 and EOF can contain Linux or VMkernel commands. All this sample below shows is how to create an internal switch with a Virtual Machine Port group called internal.

```
%post
cat > /tmp/esxcfg.sh <<EOF1
#!/bin/sh
```

This first part creates a file called esxcfg.sh inserting the line #!/bin/sh which is associated with the Linux scripting SHell. Everything between <<EOF1 and EOF is included in the esxcfg.sh file. These act as text delimiters to indicate the beginning and of the file. The cat command is the similar to the type command in Windows and allows us to redirect the contents (>) of the script into the esxcfg.sh file.

```
# Configure ESX Server
# Create vSwitch1 with a port group of Internal
esxcfg-vswitch -a vSwitch1
esxcfg-vswitch -A internal vSwitch1
EOF
```

This part adds a vSwitch called vSwitch1 with a port group called internal. No network card is added with the esxcfg-vswitch –L command.

```
# Make esxcfg.sh eXcutable
chmod +x /tmp/esxcfg.sh
```

Chmod is used to modify permissions in the Linux file system. As you might gather this changes the attributes of esxcfg.sh to make it executable.

```
# Backup original rc.local file
cp /etc/rc.d/rc.local /etc/rc.d/rc.local.bak
# Make esxcfg.sh run from rc.local and make rc.local re-
set itself
cat >> /etc/rc.d/rc.local <<EOF
cd /tmp
/tmp/esxcfg.sh
mv -f /etc/rc.d/rc.local.bak /etc/rc.d/rc.local
EOF
```

You can see the rc.local file as akin to the old autoexec.bat of DOS days – it's one of the last scripts to run in the boot process. By adding our script to the end of the file we can guarantee that it will execute properly. Another alternative to rc.local is the script held in /etc/rc.d/rc3.d/S99local - both files will do

the job. As mentioned before, the critical thing here is that the script must execute **after** the VMkernel has loaded as the Service Console (based on Redhat Linux) would not know how to process our esxcfg- style commands.

Conclusion

In this chapter I have not covered every single command, utility or script available. I have tried to cover the most popular and the most useful. We now have a series of very quick ways of installing and rebuilding ESX either from a PXE boot server or with a USB key. Additionally, we now have a good knowledge of the networking commands which will help you in troubleshooting service console connectivity. Try to see this chapter as a good jumpstart to automating ESX host configurations or troubleshooting. There is plenty more to learn about the Service Console. Dive in and explore!