Technical Report

Best Practices for File System Alignment in Virtual Environments

Abhinav Joshi, Eric Forgette, Peter Learmonth, Jon Benedict, NetApp
January 2011 | TR-3747

ABSTRACT

This document provides guidelines for preventing, detecting, and correcting file system misalignment issues for virtual machines hosted on VMware® ESX™, Microsoft® Hyper-V™, and Citrix XenServer infrastructures.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 EXECUTIVE SUMMARY

File system misalignment is a known issue in virtual environments and can cause performance issues for virtual machines (VMs). This technical report provides an overview of the storage layers in a virtualized environment, provides details on the proper alignment of guest file systems, and describes the performance impact misalignment can have on the virtual infrastructure. It also provides guidance on how the issue can be prevented by following proper steps when provisioning storage for the VM. For existing VMs, this report provides guidelines on how to determine and correct misalignment issues.

Note that this issue is not unique to NetApp® storage arrays and can occur with any storage array from any vendor. The other storage vendor white papers highlighting the need for alignment are listed in the references section later in the report. VMware has identified that this also can be an issue in both VMware vSphere™ 4 and VMware Infrastructure 3 environments. For more details, refer to the following VMware documentation:

- VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions
- Performance Best Practices for VMware vSphere 4.0

2 NETAPP STORAGE AND VIRTUAL ENVIRONMENTS DATA LAYOUT CONCEPTS

In any server virtualization environment using shared storage, there are different layers of storage involved for the VMs to access storage. This section explores the different ways shared storage can be presented for the different hypervisors and also highlights the different layers of storage involved.

Note: The scope of this report is limited to VMware ESX, Microsoft Hyper-V, Red Hat Enterprise Virtualization, KVM (as implemented by Red Hat), and Citrix XenServer environments.

2.1 VMWARE

VMware vSphere 4 and VMware Infrastructure 3 have four ways of using shared storage for deploying virtual machines:

- **VMFS** (Virtual Machine File System) on a Fibre Channel (FC) or iSCSI logical unit number (LUN) attached to the ESX host.
- **NFS** (Network File System) export mounted on an ESX host.
- **RDM** (Raw Device Mapping) is the primary method of presenting a VM direct access and ownership of a LUN; the guest formats the RDM LUN as it would for any disk.
- **LUNs directly mapped by the guest OS** (operating system) by using an iSCSI software initiator where the guest OS supports it.

For both the VMFS and NFS options, the files that make up a VM are stored in a directory on the LUN or NFS export and each VM has a separate directory. Each virtual disk of the VM is made up of two files:

- **<vmname>-flat.vmdk.** The monolithic file containing the actual disk image of the guest VM
- **<vmname>.vmdk.** A text descriptor file that contains information about the size of the virtual disk as well as cylinder, head, and sector information for the virtual BIOS to report to the guest OS

Both VMFS and NFS mounted by ESX are referred to as datastores. For the NFS option, there is no VMFS layer and the VM directories are directly stored on the NFS mount presented as a datastore.

Table 1 contains the different layers of storage involved for each of the shared storage options. The check mark (✓) indicates that alignment must be certain at this layer of guest OS, or hypervisor file system, or the NetApp storage array blocks.

For example, there are three layers involved for the VMFS option—the guest OS, VMware ESX, and the NetApp storage array. Alignment must be certain at all three levels, as indicated in Table 1.

Table 1) Layers of storage for VMware shared storage options.

Layers of Storage	VMware Shared Storage Options			
	VMFS-Formatted LUN	NFS Export	RDM LUN	LUNs Directly Mapped by the Guest OS
Guest OS	✓	✓	✓	✓
VMware ESX	✓	N/A	N/A	N/A
NetApp Storage Array	✓	N/A	✓	✓

2.2 MICROSOFT HYPER-V

Microsoft Windows® Server 2008 Hyper-V has three ways of using shared storage for deploying VMs:

- **NT file system (NTFS)-formatted LUNs** (FC or iSCSI) are attached to the Hyper-V parent partition as physical disks. The VMs are represented by Virtual Hard Disk (VHD) files hosted on the LUNs. There are three different types of VHDs:
  - **Fixed-size VHD.** This type of VHD allocates the full amount of storage configured at VHD creation and does not expand over time. It offers the lowest performance overhead of all three VHD variants and is the NetApp recommended best practice.
  - **Dynamically expanding VHD.** This type of VHD does not allocate the full amount of storage configured at the time of VHD creation and expands over time as new data is added to the VM's disk. This type of VHD differs mostly in performance because of the impact associated with having to grow the VHD file each time data is added to the VM's disk.
  - **Differencing VHD.** This type of VHD is not created at the time the VM is created. It is created as it is needed, for example, when a Hyper-V snapshot is made of an existing VM. A differencing VHD points to a parent VHD file, which can be any type of VHD, and functions similar to a dynamically expanding VHD.
- **Pass-through disks** are LUNs that are attached to the Hyper-V parent partition but assigned directly to a VM and formatted with the child OS file system.
- **LUNs are mapped directly to the child OS** by using an iSCSI software initiator where the child OS supports it.

Table 2 contains the different layers of storage involved for each of the shared storage options. The check mark (✓) indicates that alignment must be certain at this layer of guest or hypervisor file system or the NetApp storage array blocks.

Table 2) Layers of storage for Microsoft Hyper-V storage options.

Layers of Storage	Hyper-V Shared Storage Options		
	NTFS-Formatted LUN	Pass-Through Disks	LUNs Directly Mapped to the Child OS
Child OS	✓	✓	✓
Hyper-V Parent Partition	✓	N/A	N/A
NetApp Storage Array	✓	✓	✓

2.3 CITRIX XENSERVER

The Citrix XenServer host accesses containers named Storage Repositories (SRs) in which Virtual Disk Images (VDIs) are stored. A VDI is a disk abstraction that, when attached to a XenServer host, appears as a physical disk drive to the VM. The interface to storage hardware provided on the XenServer host allows VDIs to be supported on a large number of different SR substrate types. VDIs may be:

- Files on a local disk
- Files on an NFS mount
- Logical Volumes (LVs) within a LUN
- A raw LUN itself directly attached to the VM

When hosting shared SRs on a NetApp storage controller, the different options to provision storage are:

- **NetApp managed LUNs** hosted on a NetApp storage array are accessible through the NetApp Data ONTAP® SR type. They are hosted on NetApp storage running a version of Data ONTAP 7.0 or greater. LUNs are allocated on demand through the XenServer management interface and mapped dynamically to the XenServer host through the XenServer host management framework (using the open iSCSI software initiator) while a VM is active. All the thin provisioning, FlexClone®, and data deduplication capabilities in the NetApp storage controllers are available through the NetApp Data ONTAP adapter. TR-3732: Citrix XenServer and NetApp Storage Best Practices provides further details.
- **NFS** export mounted as SR on the Xen Master Server using XenServer host. In this option, the VHD format is used to store VDIs on an NFS mount exported from a NetApp storage array. There are two types of VHDs:
  - **Sparse VHD.** This is the default type of VHD created when using the NFS SR. It does not fully provision the storage upfront at the time of creating the VHD.
  - **Chained VHD.** This type of VHD allows two VDIs to share common data. When you clone an NFS-based VM, the resulting VMs share the common on-disk data at the time of cloning.
- **LUNs with the Logical Volume Manager (LVM) layer** provide shared storage using an LVM layered over either an FC or iSCSI LUN hosted on a NetApp storage controller and accessed through FC HBAs or iSCSI hardware or software initiators.

Table 3 contains the different layers of storage involved for each of the options. The check mark (✓) indicates that alignment should be certain at this layer of guest, or hypervisor file system, or NetApp storage array blocks.

Table 3) Layers of storage for Citrix XenServer shared storage options.

Layers of Storage	Citrix XenServer Shared Storage Options		
	NetApp Managed LUN	NFS Export	LUNs with the LVM Layer
Guest OS	✓	✓	✓
Citrix XenServer Control Domain	N/A	N/A	N/A
NetApp Storage Array	✓	N/A	✓

2.4 RHEV AND KVM

Although Red Hat still supports the Xen hypervisor, this report covers only disk alignment needs as they relate to deploying Red Hat's distribution of the KVM hypervisor. Also note that Red Hat supports the following two deployments of the KVM hypervisor, and this affects the virtual disk format:

- **A thin deployment** using a stateless, small footprint hypervisor directly on the bare metal; this is referred to as RHEV-H, and is part of the Red Hat Enterprise Virtualization platform
- **A thick or full OS deployment** using the RHEL 5.4 (or higher) operating system as the hypervisor

Note: This report refers to the thin hypervisor as RHEV-H and the thick hypervisor as KVM.

The thin hypervisor supports the following means of using shared storage:

- Raw disk file (on an LV) on an FC or iSCSI LUNs
- QCOW2 (QEMU Copy on Write) for thin-provisioned disks on NFS
- Raw disk files for preprovisioned disks on NFS

For block-based shared storage, the Storage Pool Manager layers the LUN with a Logical Volume Group. The individual disk files are stored as raw disk files on LVs as follows:

- Thin-provisioned LVs are created as 512MB, and grow as necessary in 512MB increments.
- Preallocated or preprovisioned LVs are created as the size specified for the disk.

For NFS-based datastores, the files that make up a virtual machine are stored in individual directories. Each directory includes at least two files:

- **`<disk_image_UUID>.`** This is the raw or QCOW2 disk image file.
- **`<disk_image_UUID>.meta.`** This is a text file that describes the disk image but does not contain any disk geometry information.

Note: Snapshots generated by RHEV management (RHEV-M) are also stored in the same directory as the parent image. Snapshots have a different UUID from their parents and also have a separate meta file. Additionally, snapshots generated by RHEV-M are all of the QCOW2 disk type.

A VM under RHEV is comprised of two files: the disk image itself and a meta file. The meta file is a flat file that contains a number of variables and values. You can review disk geometry by running `parted` or `fdisk` against the disk image. Note that disk images and meta files under RHEV are not designed to be interacted with directly but through the RHEV-M console.

The full OS hypervisor (using RHEL 5.4 or higher) supports the following means of using shared storage:

- QCOW2 (QEMU Copy on Write) for thin-provisioned disks on NFS
- QCOW2 (QEMU Copy on Write) for thin-provisioned disks on an FC or iSCSI LUN formatted with GFS2
- Raw disk files for preprovisioned disks on NFS
- Raw disk files for preprovisioned disks on an FC or iSCSI LUN formatted with GFS2

Table 4 contains the different layers of storage involved for each of the shared storage options. The check mark (✓) indicates that alignment needs to be certain and verified at that particular layer.

Table 4) Layers of storage for RHEV and KVM storage options.

Layers of Storage	RHEV and KVM Storage Options		
	Logical Volume	NFS Export	GFS2 Formatted LUN
Guest OS (RHEV)	✓	✓	✓
Guest OS (KVM)	N/A	✓	✓
NetApp Storage Array	✓	N/A	✓

3 FILE SYSTEM ALIGNMENT

As emphasized in the previous section, there are multiple layers of storage involved. Each layer is organized into blocks, or chunks, to make accessing the storage more efficient. The size and the starting offset of each block can be different at each layer. Although a different block size across the storage layers doesn't require any special attention, the starting offset does. For optimal performance, the starting offset of a file system should align with the start of a block in the next lower layer of storage. For example, an NTFS file system that resides on a LUN should have an offset that is divisible by the block size of the storage array presenting the LUN. Misalignment of block boundaries at any one of these storage layers can result in performance degradation.

This issue is not unique to NetApp storage arrays and can occur for storage arrays from any vendor. VMware has also identified that this can be an issue in virtual environments. For more details, refer to the following VMware documentation:

- VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions
- Performance Best Practices for VMware vSphere 4.0

3.1 CONCEPTS AND ISSUE

Disks use geometry to identify themselves and their characteristics to the upper-layer operating system. The upper-layer operating system uses the disk geometry information to calculate the size of the disk and partitions the disk into predetermined addressable blocks. Just as with physical disks, LUNs report disk geometry to the host (physical host, virtualization host, or the VM, depending on the mode of usage) so that it can calculate space and partition the LUN into addressable blocks.

The Cylinder-head-sector article on Wikipedia provides background information on cylinder-head concepts.

Historically, hard drives (LUNs) presented the OS with a physical geometry that would be used to partition and format the disk efficiently. Disk geometry today is virtual and fabricated by the disk firmware. Operating system partitioning programs such as fdisk use the emulated disk geometry to determine where to begin a partition. Unfortunately, some partitioning programs invoked during OS setup create disk partitions that do not align with underlying block boundaries of the disk. Also, the more notable tools, such as GNU fdisk, found on many Linux® distributions; and Microsoft DiskPart, found on Windows 2000 and Windows 2003, by default also create disk partitions that do not align with underlying block boundaries of the disk.

By default, many guest OSs, including most versions of Windows, attempt to align the first sector on a full track boundary. The installer/setup routine requests the Cylinder/Head/Sector (CHS) information that describes the disk from the BIOS (PC firmware that manages disk I/O at a low level), or, in the case of many VMs, an emulated BIOS. The issue is that the CHS data hasn't actually corresponded to anything physical, even in physical machines, since the late 1980s. At larger LUN sizes, usually 8GB or more, the sectors per track (S number) is always reported as 63, so the partitioning routine sets a starting offset of 63 sectors in an attempt to start the partition on a track boundary.

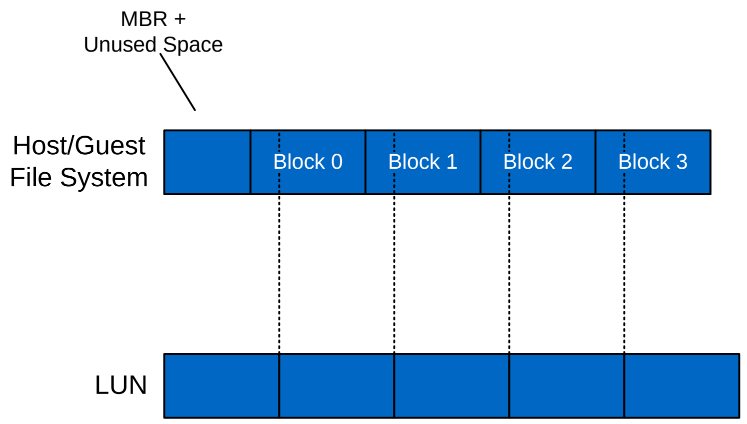
While this may be correct for a single physical disk drive, it does not line up with any storage vendor's logical block size. A block is the smallest unit of data that can be used to store an object on a storage device. In order to make sure of optimal storage capacity and performance, data should reside in the blocks. Physical disk blocks always have 512 (usable/visible) bytes, but for efficiency and scalability reasons, storage devices use a logical block size that is some number of physical blocks, usually a power of 2. For example, NetApp Unified Storage Architecture arrays have a logical block size of 4K, that is, 8 disk blocks. EMC Symmetrix storage arrays have a logical block size of 64KB.

Write operations can consume no less than a single 4KB block and can consume many 4KB blocks depending on the size of the write operation. Ideally, the guest/child OS should align its file system(s) such that writes are aligned to the storage device's logical blocks. The problem of unaligned LUN I/O occurs when the partitioning scheme used by the host OS doesn't match the block boundaries inside the LUN, as shown in Figure 1. If the guest file system is not aligned, it might become necessary to read or write twice as many blocks of storage than the guest actually requested because any guest file system block actually occupies at least two partial storage blocks. As a simple example, assuming only one layer of file system and that the guest allocation unit is equal to the storage logical block size (4K or 4,096 bytes), each guest block (technically an allocation unit) would occupy 512 bytes of one block and 3,584 bytes (4,096 – 512) of the next. This results in inefficient I/O because the storage controller must perform additional work such as reading extra data to satisfy a read or write I/O from the host.

By default, many guest operating systems, including Windows 2000, Windows 2003, and various Linux distributions, start the first primary partition at sector (logical block) 63. The reasons for this are historically tied to disk geometry. This behavior leads to misaligned file systems because the partition does not begin at a sector that is a multiple of 8.

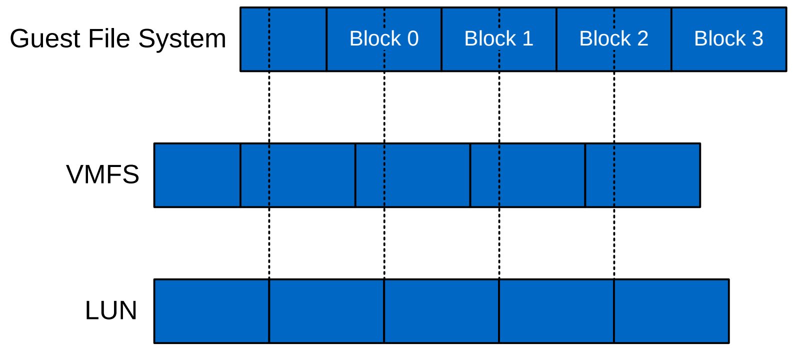
Note that Windows Server 2008 and Windows Vista® default at 1,048,576, which is divisible by 4,096, and do not require any adjustments. Also, RHEL 6 does not require any adjustments because it aligns its partitions properly by default.

Figure 1) Misaligned file system.



The misalignment issue is more complex when the file system on the virtualization host contains the files (for example, `vmdk` or `vhd`) that represent the VM virtual disks, as shown in Figure 2. In this case, the partition scheme used by the guest OS for the virtual disks must match the partition scheme used by the LUNs on the hypervisor host and the NetApp storage array blocks.

Figure 2) Guest OS and VMFS file system not aligned with the storage array blocks.



VMWARE

VMs hosted on VMFS involve two layers of alignment, both of which should align with the NetApp storage blocks:

- VMFS
- The file system on the guest `vmdk` files inside the VMFS

The default starting offset of VMFS2 is 63 blocks, which results in misaligned I/O. The default offset of VMFS3, when created with the Virtual Infrastructure Client or VMware vSphere Client, is 128 blocks by default, which does not result in misaligned I/O. Datastores migrated from VMFS2 to VMFS3 as part of an ESX/VI3 upgrade are not realigned; VM files need to be copied from the old datastore to a newly created datastore. In addition to properly aligning VMFS, each VM guest file system needs to be properly aligned as well. (See section 4.1.)

RDM and LUNs directly mapped by the guest VM do not require special attention if the LUN type on the LUN matches the guest operating system type. Note that alignment is automatic for LUNs created and connected inside the guest VM using NetApp SnapDrive® software. For more information on LUN type, see the LUN Multiprotocol Type section in the “Data ONTAP Block Access Management Guide” or the “Commands Manual Page Reference” document, which can be downloaded from the NetApp NOW™ (NetApp on the Web) site.

While NFS datastores do not require alignment themselves, each VM guest file system needs to be properly aligned. (See section 4.1.)

For all of these storage options, misalignment at any layer can cause performance issues as the system scales.

MICROSOFT HYPER-V

VHDs hosted on NTFS-formatted LUNs that are attached as physical disks on the Hyper-V parent partition involve two layers of alignment, both of which should align with the NetApp storage blocks:

- The NTFS file system on the physical disk
- The file system on the child VM hosted on the physical disk

Pass-through disks and LUNs directly mapped by the child VM do not require special attention if the LUN type of the LUN matches the guest operating system type. For more information on LUN type, see the LUN Multiprotocol Type section in the “Data ONTAP Block Access Management Guide” or the “Data ONTAP Commands Manual Page Reference,” which can be downloaded from the NetApp NOW site.

For all of these storage options, misalignment at any layer can cause performance issues as the system scales.

CITRIX XENSERVER

For all the storage options in Citrix XenServer discussed in section 2.3, there is only one layer of alignment involved—the file system on the guest VM should align with the NetApp storage blocks. Misalignment can result in performance issues.

RHEV AND KVM

For VMs hosted on an NFS datastore (RHEV and KVM), the NFS export itself is properly aligned by default. However, all partitions on the guest disk must be properly aligned. (See section 4.5.)

When creating a LUN for use as a datastore in RHEV, choose LUN protocol Linux. The RHEV Storage Pool Manager layers the LUN with an LV Group, which needs no further intervention.

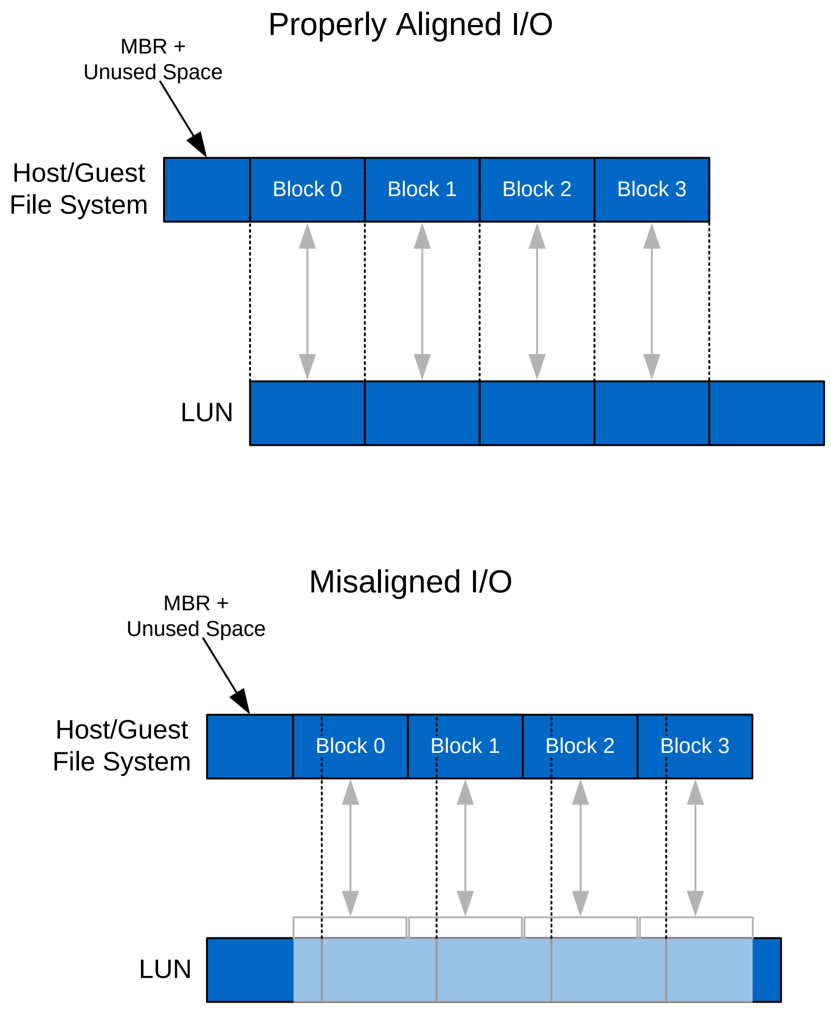
When creating a LUN for use as a datastore in KVM, choose LUN protocol type Linux. If you are layering LVM2 on top, NetApp recommends layering it on the entire device (for example, `/dev/vda`). If partitioning the device is required (for example, `/dev/vda1`), the partition must start at sector 64, 128, or another sector number evenly divisible by 8.

In all cases, the guest disk image must be properly aligned. All partitions on the guest operating system must be properly aligned. (See section 4.5.)

3.2 IMPACT

Misalignment can cause an increase in per-operation latency. It requires the storage array to read from or write to more blocks than necessary to perform logical I/O. Figure 3 shows an example of a LUN with and without file system alignment. In the first instance, the LUN with aligned file systems uses four 4KB blocks on the LUN to store four 4KB blocks of data generated by a host. In the second scenario, where there is misalignment, the NetApp storage controller must use five blocks to store the same 16KB of data. This is an inefficient use of space, and performance suffers when the storage array must process five blocks to read or write what should be only four blocks of data. This results in inefficient I/O, because the storage array is doing more work than is actually requested.

Figure 3) Impact for aligned and misaligned file systems.



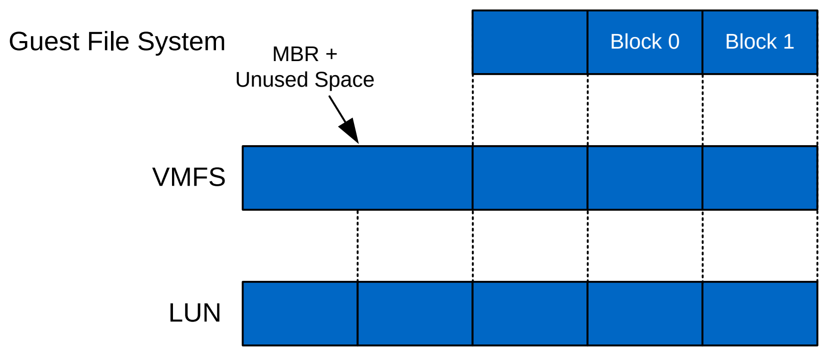
4 FILE SYSTEM MISALIGNMENT PREVENTION

4.1 VMWARE

VMFS

VMFS-based datastores (FC or ISCSI) should be set to the LUN type VMware and created using the Virtual Infrastructure Client or VMware vSphere Client. This results in the VMFS file system being aligned with the LUN on the storage controller. If you use `vmkfstools`, make sure that you first partition the LUN using `fdisk`. This allows you to set the correct offset.

Figure 4) Guest VM file system and VMFS file system are aligned with the storage array blocks.



Use the following procedure to detect misaligned VMFS partitions. For instructions on VMware VMFS partition alignment using `fdisk`, see the article [VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions]().

Use cold migration or storage VMotion™ to migrate VMs hosted on misaligned VMFS partitions to aligned VMFS partitions.

1. Run the following `vmkfstools` command to get the vmhba device from the VMFS label (mount point).

```
vmkfstools -P /vmfs/volumes/vmprod

VMFS-3.31 file system spanning 1 partitions.
File system label (if any): vmprod
Mode: public
Capacity 214479929344 (204544 file blocks * 1048576), 12256804864 (11689 blocks) avail
UUID: 47bd0f4d-8c6c0b00-202b-000423c3e841
Partitions spanned (on "lvm"):
        vmhba0:1:1:1 (for VI3)
        or
        naa.60a98000486e5352424a476547365454:1 (for vSphere 4)
```

Note: `vmkfstools` returns the partition, which ends in :1, rather than the whole disk or LUN device, which ends in :0.

2. Use `fdisk` to check the starting offset of the VMFS partition.

   For VI3

```
fdisk -lu /vmfs/devices/disks/vmhba0:1:1:0

Disk /vmfs/devices/disks/vmhba0:1:1:0: 214.7 GB, 214748364800 bytes
255 heads, 63 sectors/track, 26108 cylinders, total 419430400 sectors
Units = sectors of 1 * 512 = 512 bytes
```

```
                        Device Boot  Start    End     Blocks     Id  System
/vmfs/devices/disks/vmhba0:1:1:0p1  128 419425019 209712446  fb  Unknown
```

For vSphere 4

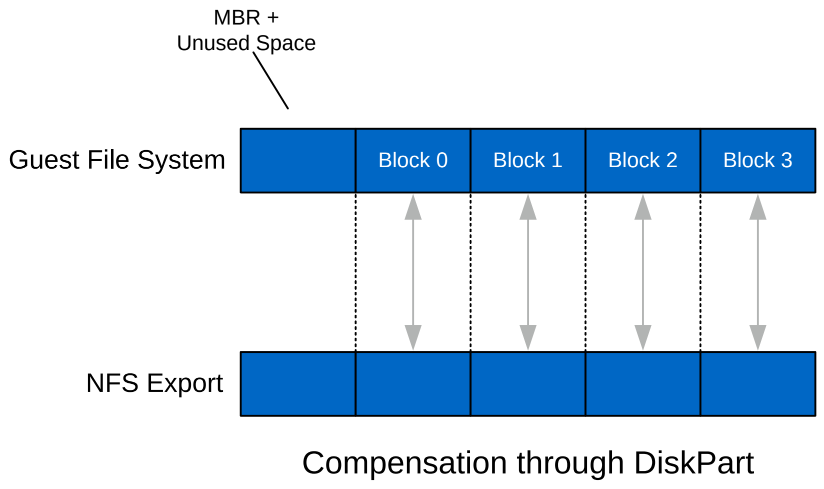
```
fdisk -lu /vmfs/devices/disks/naa.60a98000486e5352424a476547365454
last_lba(): I don't know how to handle files with mode 8180

Disk /vmfs/devices/disks/naa.60a98000486e5352424a476547365454: 214.7 GB, 214748364800
bytes
255 heads, 63 sectors/track, 26108 cylinders, total 419430400 sectors Units = sectors
of 1 * 512 = 512 bytes

                                               Device Boot      Start         End
Blocks   Id  System
/vmfs/devices/disks/naa.60a98000486e5352424a476547365454p1          128   419425019
209712446   fb  VMware VMFS

Note that with fdisk you specify the whole disk/LUN, not the partition reported from
vmkfstools.  In other words, use the NAA ID, but not the trailing colon or partition
number.
```

Note: Start is listed as 128 sectors (physical disk blocks). The default in the GUI in vCenter™ prior to 2.0 and from the command line in any version of ESX is 63.

NFS

There is no VMFS layer involved with NFS, so only the alignment of the guest VM file system within the VMDK to the NetApp storage array is required. Enable this using the procedure described in section 4.4.

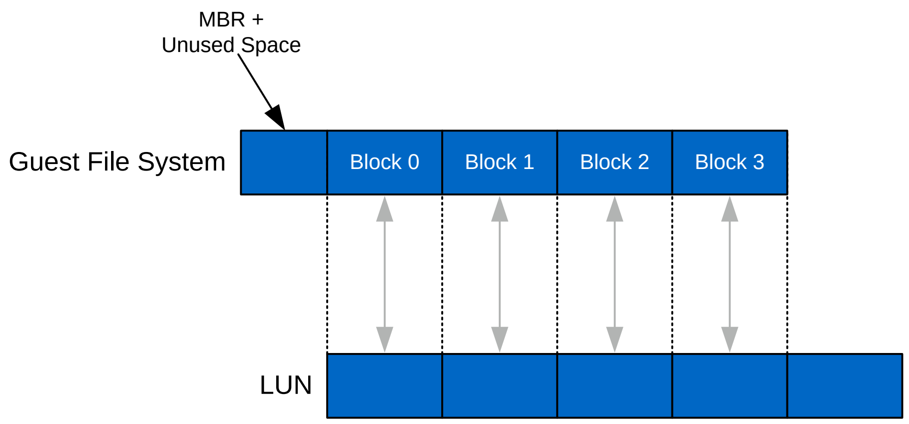
Figure 5) NFS: Guest VM file system aligned with the storage array blocks.



RDM

For RDM, selecting the LUN type of the intended guest OS when creating the LUN enables the file system on the LUN to align with the blocks on the NetApp storage array. Note that correct alignment is automatic for LUNs created and connected inside the guest VM using NetApp SnapDrive software.

Figure 6) RDM: Guest VM file system aligned with the storage array blocks.



LUNS DIRECTLY MAPPED TO THE GUEST OS

For LUNs directly mapped to the guest OS using the iSCSI software initiator, selecting the LUN type of the intended guest OS when creating the LUN enables the file system on the LUN to align with the NetApp storage array blocks, as shown in Figure 6. Note that correct alignment is automatic for LUNs created and connected inside the guest VM using NetApp SnapDrive software.

4.2 MICROSOFT HYPER-V

NTFS-FORMATTED LUNS

For NTFS-formatted LUNs attached to the Hyper-V parent partition as physical disks hosting VHDs, selecting the correct LUN type is very important. Use LUN type Hyper-V for NetApp storage systems running Data ONTAP version 7.3.1 and higher.

Note: Use LUN type windows_2008 for NetApp storage systems running Data ONTAP 7.2.5 through 7.3; for NetApp storage systems running Data ONTAP version 7.2.4 and earlier, use LUN type Linux.

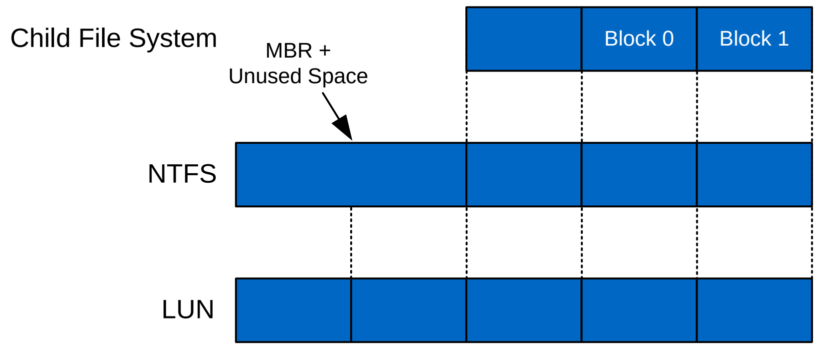
For Data ONTAP version 7.3 and earlier, the LUN type windows_2008 is available only through the Data ONTAP CLI. Therefore, the LUNs for Hyper-V parent partition and Windows Server 2008 child VMs must be created through the LUN setup command on the Data ONTAP CLI.

Use NetApp SnapDrive 6.0 and higher to provision LUNs. This enables the windows_gpt LUN type to be selected and makes alignment certain. However, the alignment of the file system of the child VM to the file system of the underlying physical disk is still required. Enable this using the procedure described in section 4.4.

Fixed-Size VHDs

Proper alignment cannot be guaranteed and there is a performance penalty for dynamically expanding and differencing VHDs. Therefore, NetApp recommends using fixed-size VHDs within your Hyper-V environment whenever possible; this avoids using dynamically expanding and differencing VHDs unless a good reason is found for their use. TR-3702: NetApp Storage Best Practices for Microsoft Virtualization contains further details on this recommendation.

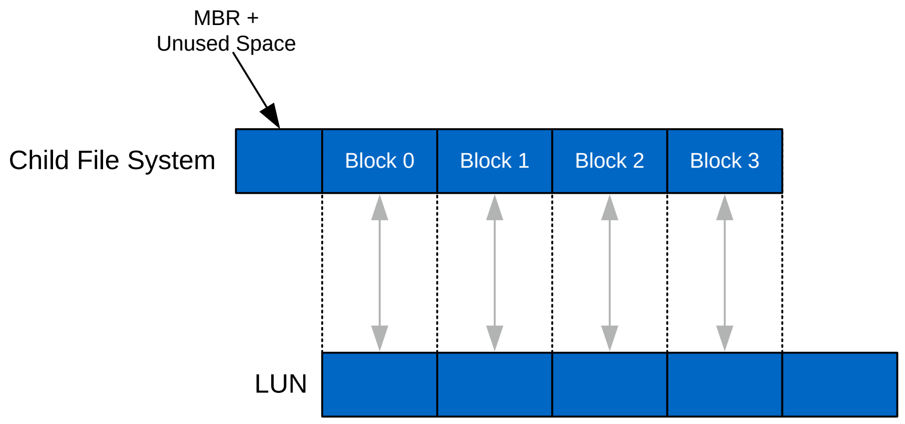
Figure 7) Child VM file system and NTFS file system are aligned with the storage array blocks.



PASS-THROUGH DISKS

For pass-through disks, selecting the LUN type of the intended child OS when creating the LUN enables the file system on the LUN to align with the NetApp storage array blocks.

Figure 8) Child file system aligned with the storage array blocks.



LUNS DIRECTLY MAPPED TO THE GUEST OS

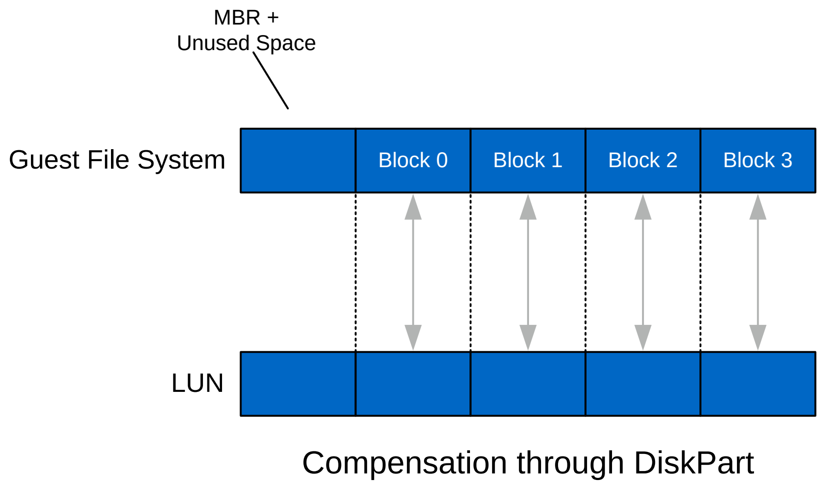
Select the LUN type of the intended child OS when creating the LUN for LUNs directly mapped to the child OS using the iSCSI software initiator. This enables the file system on the LUN to align with NetApp storage array blocks as shown in Figure 8. Note that correct alignment is automatic for LUNs created and connected inside the child VM using NetApp SnapDrive software.

4.3 CITRIX XENSERVER

NETAPP-MANAGED LUNS

Only the alignment of the guest VM file system within the VDI to the NetApp storage array is required for LUNs provisioned using the NetApp SR. Enable this using the procedure described in section 4.4.

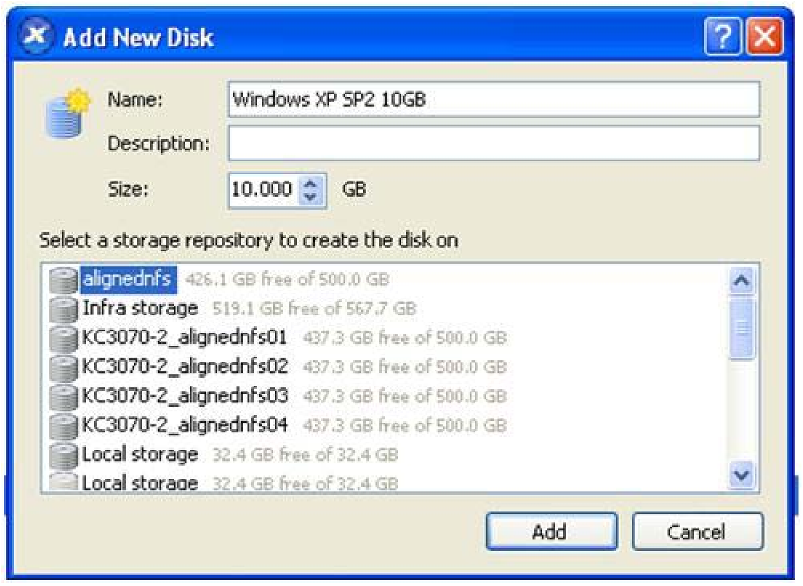
Figure 9) LUNs: Guest VM file system aligned with the storage array blocks.



NFS EXPORT

For the VHD on NFS storage option, NetApp recommends first thick-provisioning the VHD using the following steps.

1. In XenCenter, click the newly created NFS SR in the Resources pane.
2. Click the Storage tab.
3. Click the Add Disk button and enter details for the size of VDI you want. Make sure the newly created NFS SR is highlighted and click Add.


Add New Disk

Name: Windows XP SP2 10GB

Description:

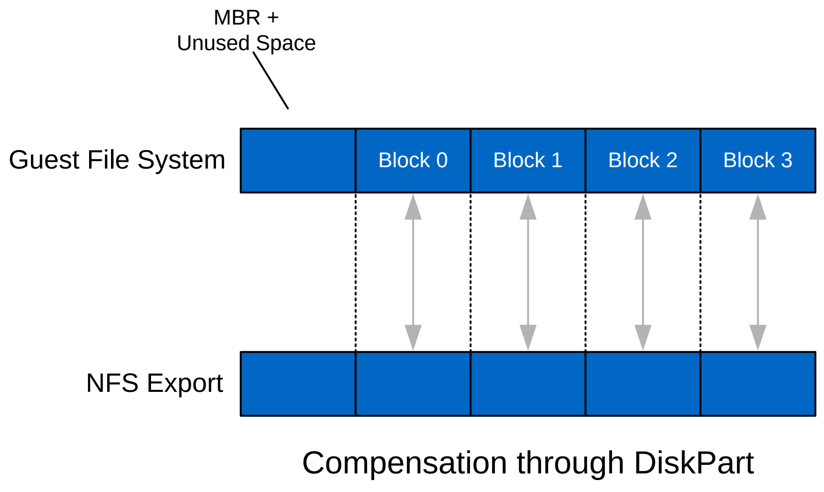
Size: 10.000 GB

Select a storage repository to create the disk on

alignednfs	426.1 GB free of 500.0 GB
Infra storage	519.1 GB free of 567.7 GB
KC3070-2_alignednfs01	437.3 GB free of 500.0 GB
KC3070-2_alignednfs02	437.3 GB free of 500.0 GB
KC3070-2_alignednfs03	437.3 GB free of 500.0 GB
KC3070-2_alignednfs04	437.3 GB free of 500.0 GB
Local storage	32.4 GB free of 32.4 GB
Local storage	32.4 GB free of 32.4 GB

Next, align the guest VM file system within the VHD with the NetApp storage array blocks. Enable this using the procedure described in section 4.4.

Figure 10) NFS: Guest VM file system aligned with the storage array blocks.



Note: If you copy the VHD file and enable disk alignment, you must use the `rfwm` command available in the Master XenServer and then use `sr-scan` to scan the SR to sync with XenCenter.

4.4 GUEST/CHILD VM ALIGNMENT PROCEDURE

There are several options to prevent misalignment when provisioning VMs. The following sections describe these options in detail.

USING DISKPART TO FORMAT WITH THE CORRECT STARTING PARTITION OFFSET (WINDOWS GUEST VMS)

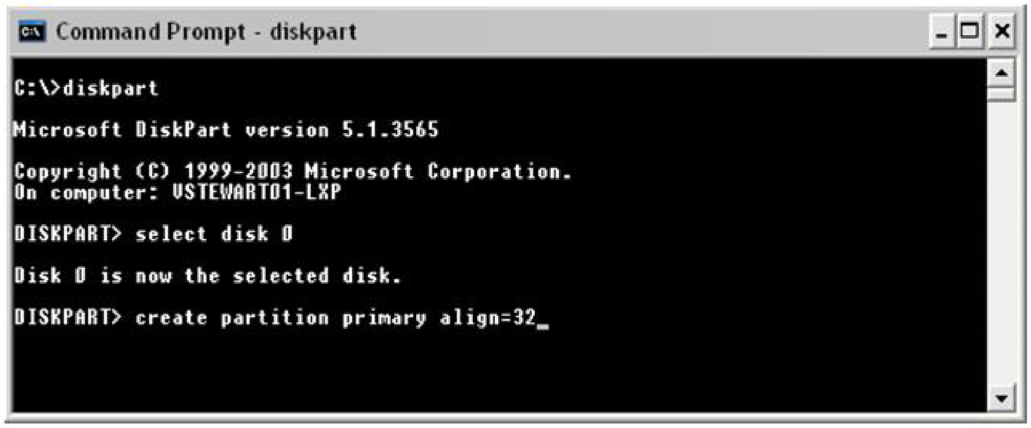
This procedure works for Windows VMs hosted on any hypervisor, including VMware ESX (`vmdk` files hosted on VMFS or NFS datastores), Citrix XenServer (fixed-size `vhd` files hosted on NFS SR), Microsoft Hyper-V (fixed-size VHDs), as well as RHEV and KVM.

Note: This procedure is not required for Windows Server 2008 VMs, Windows Vista VMs, or RHEL 6 VMs, which are aligned by default.

Aligning Boot Disk

Virtual disks to be used as the boot disk can be formatted with the correct offset at the time of creation by connecting the new virtual disk to a running VM before installing an operating system and manually setting the partition offset. For Windows guest operating systems, you might consider using an existing Windows Preinstall Environment boot CD or alternative tools like Bart's PE CD. To set up the starting offset, follow these steps.

1. Boot the VM with the WinPE CD.
2. Select Start > Run and enter DiskPart.
3. Enter Select Disk0.
4. Enter `create partition primary align=32`.



```
C:\>diskpart

Microsoft DiskPart version 5.1.3565

Copyright (C) 1999-2003 Microsoft Corporation.
On computer: VSTEWART01-LXP

DISKPART> select disk 0

Disk 0 is now the selected disk.

DISKPART> create partition primary align=32_
```

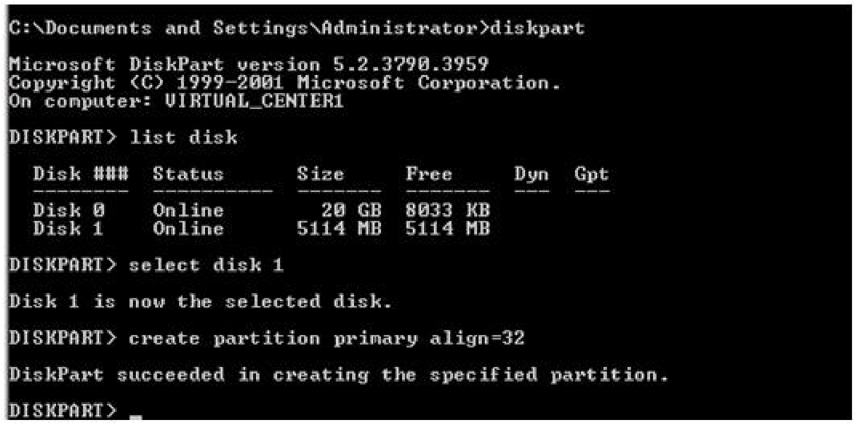
5. Reboot the VM with WinPE CD.
6. Install the operating system as normal.

Aligning Data Disk

To format virtual disks to be used as the data disk with the correct offset at the time of creation, use DiskPart in the VM.

Attach the data disk to the VM. Check that there is no data on the disk.

1. Select Start > Run.
2. Enter `diskpart`.
3. Enter `list disk` to determine the disk # for the new data disk.
4. Enter `select disk <disk_number>` (for example, `select disk 1`).
5. Enter `create partition primary align=32`.



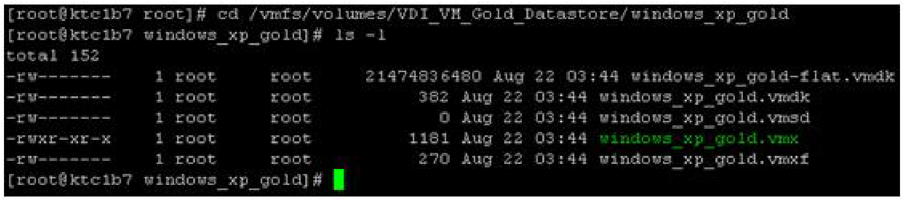
```
C:\Documents and Settings\Administrator>diskpart

Microsoft DiskPart version 5.2.3790.3959
Copyright (C) 1999-2001 Microsoft Corporation.
On computer: VIRTUAL_CENTER1

DISKPART> list disk

  Disk ###  Status      Size     Free     Dyn  Gpt
  --------  ----------  -------  -------  ---  ---
  Disk 0    Online        20 GB  8033 KB
  Disk 1    Online      5114 MB  5114 MB

DISKPART> select disk 1

Disk 1 is now the selected disk.

DISKPART> create partition primary align=32

DiskPart succeeded in creating the specified partition.

DISKPART> _
```

6. Enter `exit` to exit the DiskPart utility.
7. Format the data disk as you normally do.

USING FDISK FROM ESX SERVICE CONSOLE FOR WINDOWS GUEST VMS

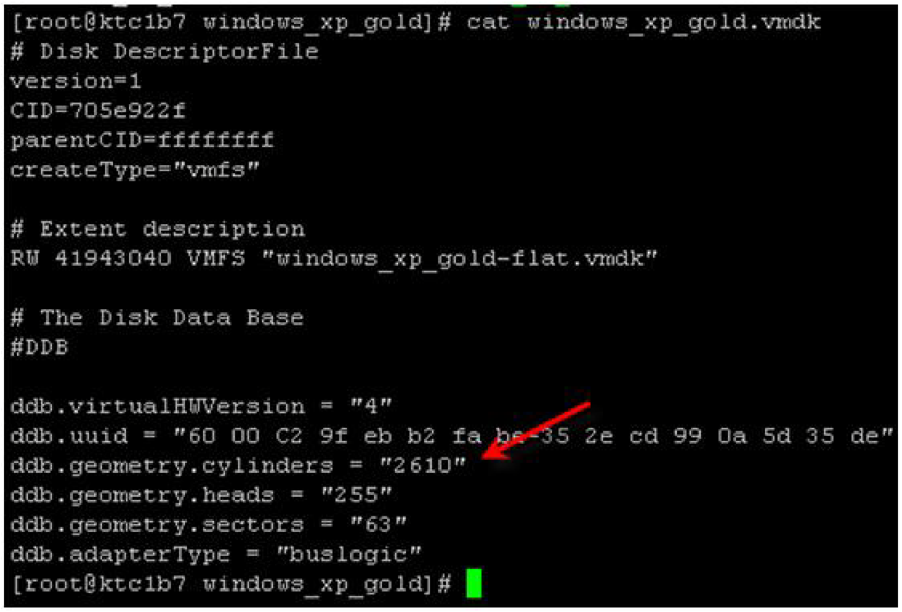
This procedure works for VMware `vmdk` files hosted on VMFS or NFS datastores, for Windows VMs. Note that this procedure is not required for Windows Server 2008, Windows Vista VMs, or RHEL 6 VMs, which are aligned by default. To set up the starting offset using the `fdisk` command in the ESX service console, follow these steps.

1. Log in to the ESX service console.
2. Enter `cd /vmfs/volumes/vdi_gold /windows_xp_gold` to change directory to the VM directory.
3. Enter `ls –l` to list the contents of the VM directory.



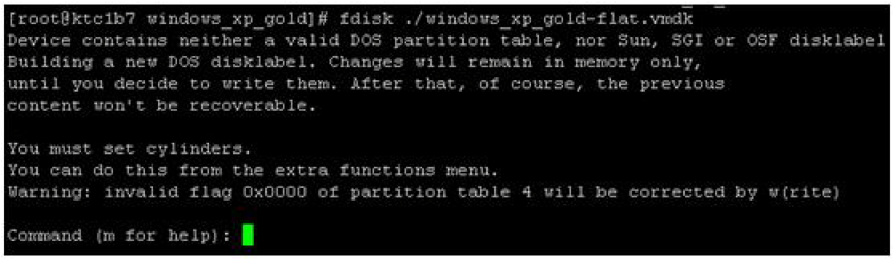
```
[root@ktc1b7 root]# cd /vmfs/volumes/VDI_VM_Gold_Datastore/windows_xp_gold
[root@ktc1b7 windows_xp_gold]# ls -l
total 152
-rw-------    1 root     root     21474836480 Aug 22 03:44 windows_xp_gold-flat.vmdk
-rw-------    1 root     root             382 Aug 22 03:44 windows_xp_gold.vmdk
-rw-------    1 root     root               0 Aug 22 03:44 windows_xp_gold.vmsd
-rwxr-xr-x    1 root     root            1181 Aug 22 03:44 windows_xp_gold.vmx
-rw-------    1 root     root             270 Aug 22 03:44 windows_xp_gold.vmxf
[root@ktc1b7 windows_xp_gold]#
```

4. Enter `cat windows_xp_gold.vmdk` to get the number of cylinders from the vdisk descriptor. (This number differs depending on several factors involved with the creation of your `.vmdk` file.)



```
[root@ktc1b7 windows_xp_gold]# cat windows_xp_gold.vmdk
# Disk DescriptorFile
version=1
CID=705e922f
parentCID=ffffffff
createType="vmfs"

# Extent description
RW 41943040 VMFS "windows_xp_gold-flat.vmdk"

# The Disk Data Base
#DDB

ddb.virtualHWVersion = "4"
ddb.uuid = "60 00 C2 9f eb b2 fa be 35 2e cd 99 0a 5d 35 de"
ddb.geometry.cylinders = "2610"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.adapterType = "buslogic"
[root@ktc1b7 windows_xp_gold]#
```

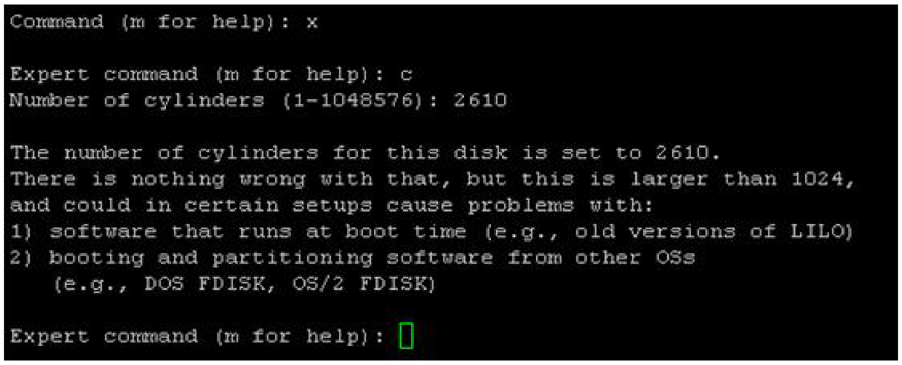
5. Enter `fdisk ./windows_xp_gold-flat.vmdk` to run `fdisk` on the `windows_xp_gold-flat.vmdk` file.



```
[root@ktc1b7 windows_xp_gold]# fdisk ./windows_xp_gold-flat.vmdk
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

You must set cylinders.
You can do this from the extra functions menu.
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
```

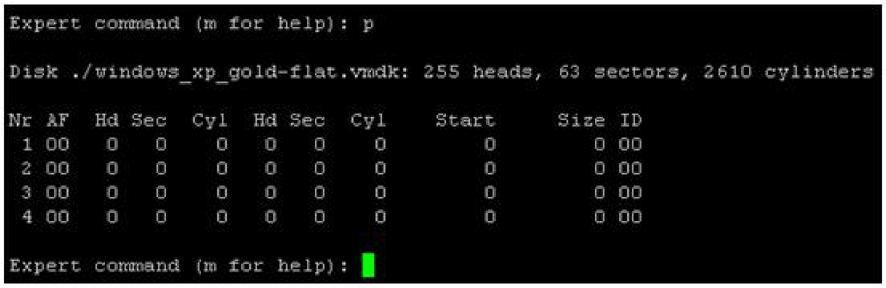
6. Enter x to set the number of cylinders and press Enter.

   You must set the number of cylinders.
7. Enter c and press Enter.
8. Enter the number of cylinders that you found in step 4 and press Enter.



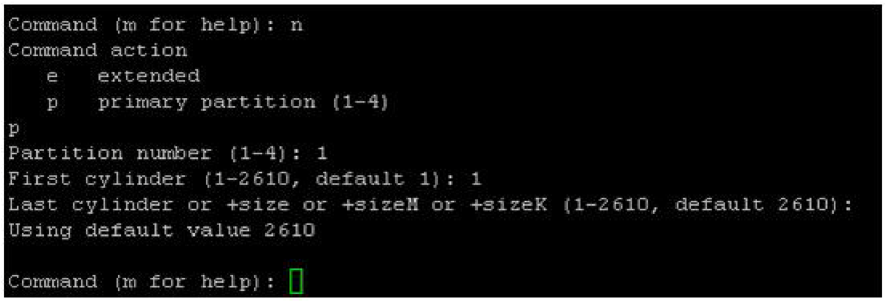
```
Command (m for help): x

Expert command (m for help): c
Number of cylinders (1-1048576): 2610

The number of cylinders for this disk is set to 2610.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Expert command (m for help):
```

9. Enter p at the expert command screen to look at the partition table, which should be blank.



```
Expert command (m for help): p

Disk ./windows_xp_gold-flat.vmdk: 255 heads, 63 sectors, 2610 cylinders

Nr AF  Hd Sec  Cyl  Hd Sec  Cyl    Start     Size ID
 1 00   0   0    0   0   0    0        0        0 00
 2 00   0   0    0   0   0    0        0        0 00
 3 00   0   0    0   0   0    0        0        0 00
 4 00   0   0    0   0   0    0        0        0 00

Expert command (m for help):
```

10. Enter r at the prompt to return to regular (nonextended) command mode.
11. Enter n and then p to create a new partition when asked for the partition type.
12. Enter 1 for the partition number, enter 1 for the first cylinder, and press Enter for the last cylinder question to use the default value.

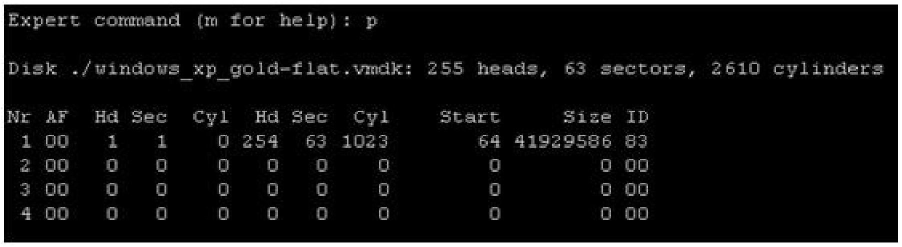


```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-2610, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-2610, default 2610):
Using default value 2610

Command (m for help):
```

13. Enter x to go into extended mode to set the starting offset.
14. Enter b to set the starting offset and press Enter.
15. Enter 1 for the partition and press Enter.
16. Enter 64 and press Enter.

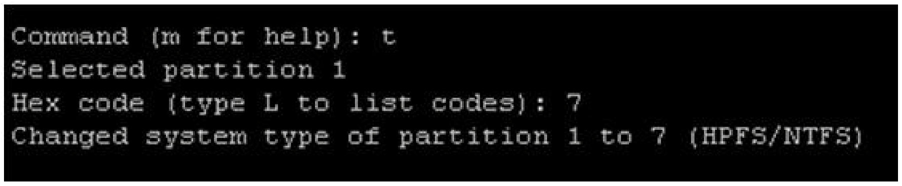
Note: In this procedure, 64 is used as an example. You may choose any value as long as it is divisible by 8. The reason the logical block address needs to be divisible by 8 is because each sector is 512 bytes in size. Eight multiplied by 512 results in 4,096 (or 4KB).

17. Enter p to check the partition table.



```
Expert command (m for help): p

Disk ./windows_xp_gold-flat.vmdk: 255 heads, 63 sectors, 2610 cylinders

Nr AF  Hd Sec  Cyl  Hd Sec  Cyl    Start      Size ID
 1 00   1   1    0 254  63 1023       64 41929586 83
 2 00   0   0    0   0   0    0        0        0 00
 3 00   0   0    0   0   0    0        0        0 00
 4 00   0   0    0   0   0    0        0        0 00
```

18. Enter `r` to return to the regular menu.
19. Enter `t` to set the system type to HPFS/NTF.
20. Enter 7 for the hexcode.

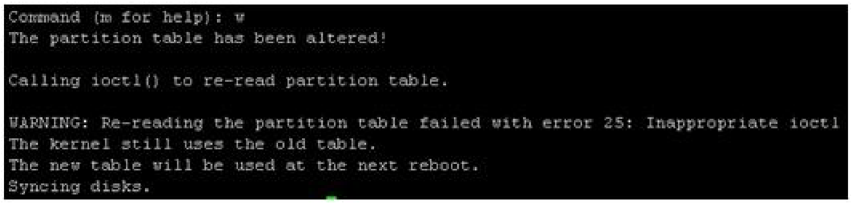


```
Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 7
Changed system type of partition 1 to 7 (HPFS/NTFS)
```

21. Repeat steps 11 through 20 if additional partitions are required.

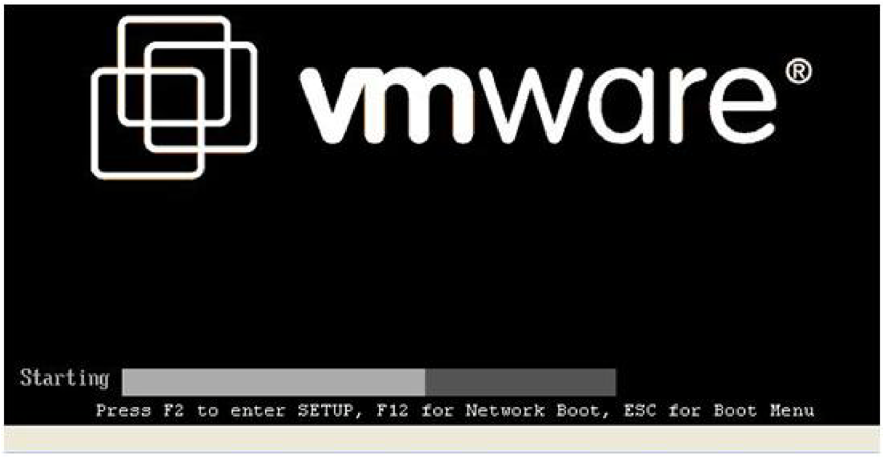
    Make sure that the starting offset (dealt with in steps 13 through 16) is divisible by 8.

    For additional partitions, be sure to increase the partition number.
22. Enter `w` to save and write the partition. Ignore the warning because it is normal.



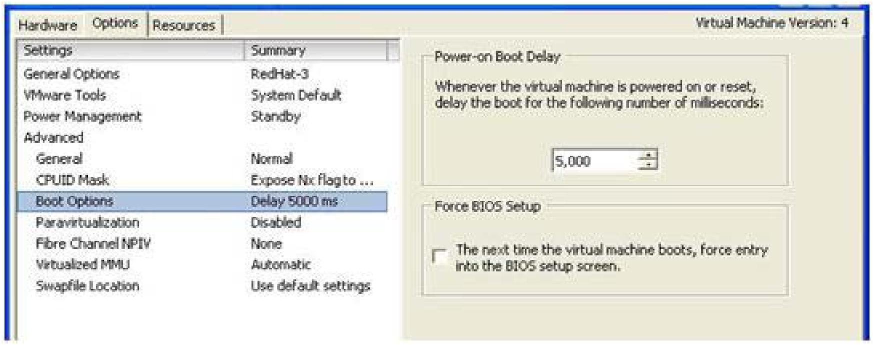
```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 25: Inappropriate ioctl
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
```

23. Start the VM and run Windows setup. Make sure to press Escape to bring up the boot menu and select CD ROM drive to boot from the CD.



```
Starting
        Press F2 to enter SETUP, F12 for Network Boot, ESC for Boot Menu
```

If you miss the boot menu, the VM might appear to hang as a black screen with only a blinking cursor. Press ctrl-alt-insert to reboot the VM and try again to catch the boot menu by pressing Escape. If you have trouble catching the boot process, you can insert a boot delay in the VM settings. In the VI Client, right-click the VM, then > Edit Settings > Options > Advanced / Boot Options.



Virtual Machine Version: 4

Settings	Summary
General Options	RedHat-3
VMware Tools	System Default
Power Management	Standby
Advanced	
General	Normal
CPUID Mask	Expose Nx flag to ...
Boot Options	Delay 5000 ms
Paravirtualization	Disabled
Fibre Channel NPIV	None
Virtualized MMU	Automatic
Swapfile Location	Use default settings

Power-on Boot Delay

Whenever the virtual machine is powered on or reset, delay the boot for the following number of milliseconds:

5,000

Force BIOS Setup

☐ The next time the virtual machine boots, force entry into the BIOS setup screen.

Note: Boot delay is in milliseconds. You should return the boot delay to 0 after the VM boots from its virtual disk normally.

24. Install on the existing partition when the install gets to the partition screen. DO NOT DESTROY or RECREATE! C: should already be highlighted. Press Enter at this stage.

USING FDISK FROM ESX SERVICE CONSOLE FOR LINUX GUEST VMS

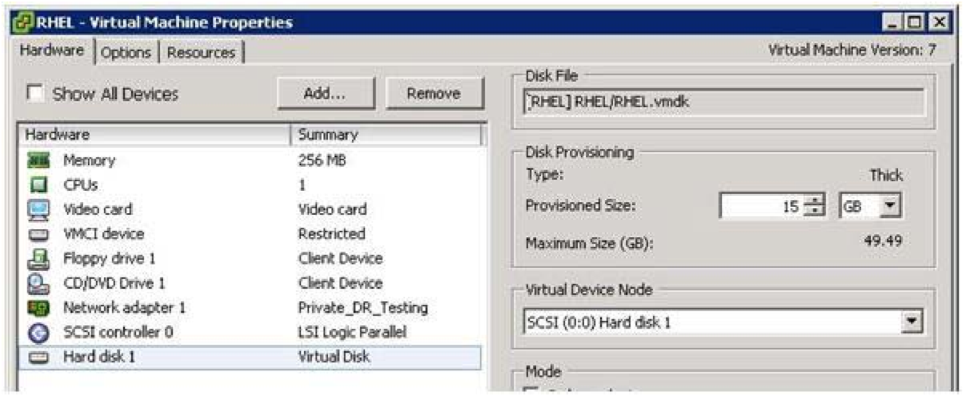
New Linux Template VM

This procedure works for VMware `vmdk` files hosted on VMFS or NFS datastores. For Linux template VMs, NetApp recommends one `vmdk` file with two partitions:

- **Partition 1.** A small 100MB partition that hosts the /boot Filesystem
- **Partition 2.** Occupies the remaining space on the `vmdk` file; hosts the other necessary file systems (/, /var, swap) using the Linux LVM

For both partitions, set the starting offset to be divisible by 8 so that there is alignment to 4K blocks on NetApp storage. To set up the starting offset using the `fdisk` command in the ESX service console, follow these steps.

1. Leverage NetApp RCU 3.0 or higher in VMware vCenter Server to provision a new VMFS/NFS datastore. (This is required only if the template VM is desired in a new datastore.)
2. Create a new template VM using VMware vCenter Server. Create only one virtual disk (`vmdk` file) to host the different file systems (/boot, /, /var, and swap).



RHEL - Virtual Machine Properties

Virtual Machine Version: 7

☐ Show All Devices

Hardware	Summary
Memory	256 MB
CPUs	1
Video card	Video card
VMCI device	Restricted
Floppy drive 1	Client Device
CD/DVD Drive 1	Client Device
Network adapter 1	Private_DR_Testing
SCSI controller 0	LSI Logic Parallel
Hard disk 1	Virtual Disk

Disk File

[RHEL] RHEL/RHEL.vmdk

Disk Provisioning

Type: Thick

Provisioned Size: 15 GB

Maximum Size (GB): 49.49

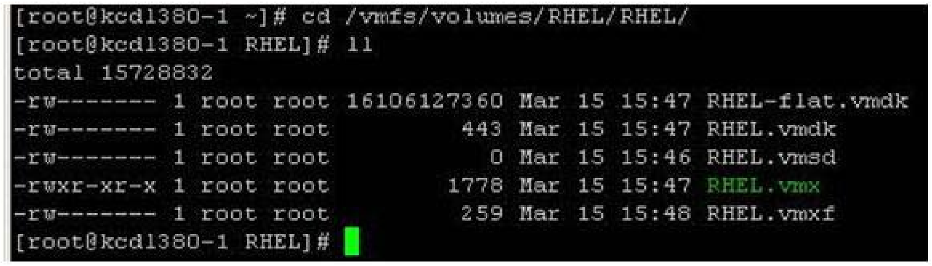
Virtual Device Node

SCSI (0:0) Hard disk 1

Mode

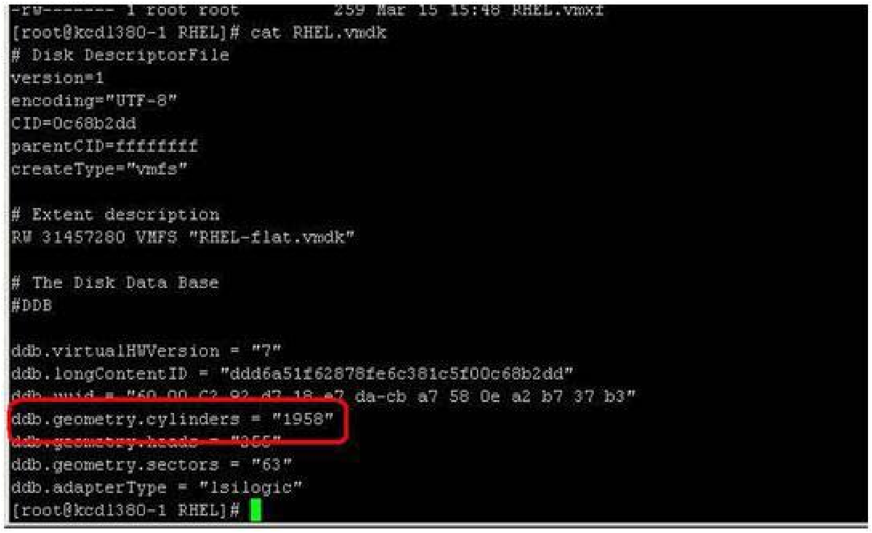
3. Log in to the ESX service console.
4. Enter `cd /vmfs/volumes/RHEL/RHEL` to change directory to the VM directory.

5. Enter `ll` to list the contents of the VM directory.



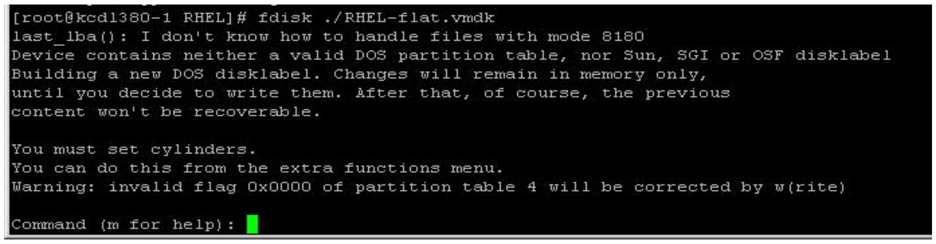
```
[root@kcd1380-1 ~]# cd /vmfs/volumes/RHEL/RHEL/
[root@kcd1380-1 RHEL]# ll
total 15728832
-rw------- 1 root root 16106127360 Mar 15 15:47 RHEL-flat.vmdk
-rw------- 1 root root         443 Mar 15 15:47 RHEL.vmdk
-rw------- 1 root root           0 Mar 15 15:46 RHEL.vmsd
-rwxr-xr-x 1 root root        1778 Mar 15 15:47 RHEL.vmx
-rw------- 1 root root         259 Mar 15 15:48 RHEL.vmxf
[root@kcd1380-1 RHEL]#
```

6. Enter `cat RHEL.vmdk` to get the number of cylinders from the vdisk descriptor. (This number differs depending on several factors involved with the creation of your `.vmdk` file.)



```
-rw------- 1 root root         259 Mar 15 15:48 RHEL.vmxf
[root@kcd1380-1 RHEL]# cat RHEL.vmdk
# Disk DescriptorFile
version=1
encoding="UTF-8"
CID=0c68b2dd
parentCID=ffffffff
createType="vmfs"

# Extent description
RW 31457280 VMFS "RHEL-flat.vmdk"

# The Disk Data Base
#DDB

ddb.virtualHWVersion = "7"
ddb.longContentID = "ddd6a51f62878fe6c381c5f00c68b2dd"
ddb.uuid = "60 00 C2 92 d7 18 e7 da-cb a7 58 0e a2 b7 37 b3"
ddb.geometry.cylinders = "1958"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.adapterType = "lsilogic"
[root@kcd1380-1 RHEL]#
```

7. Enter `fdisk ./RHEL-flat.vmdk` to run `fdisk` on the `RHEL.vmdk` file.

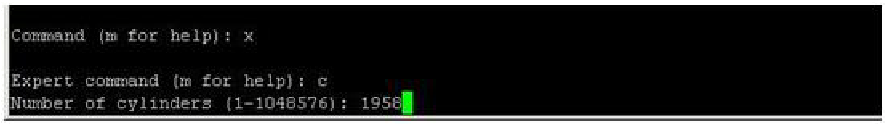


```
[root@kcd1380-1 RHEL]# fdisk ./RHEL-flat.vmdk
last_lba(): I don't know how to handle files with mode 8180
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel. Changes will remain in memory only,
until you decide to write them. After that, of course, the previous
content won't be recoverable.

You must set cylinders.
You can do this from the extra functions menu.
Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
```

8. Enter x to set the number of cylinders and press Enter.

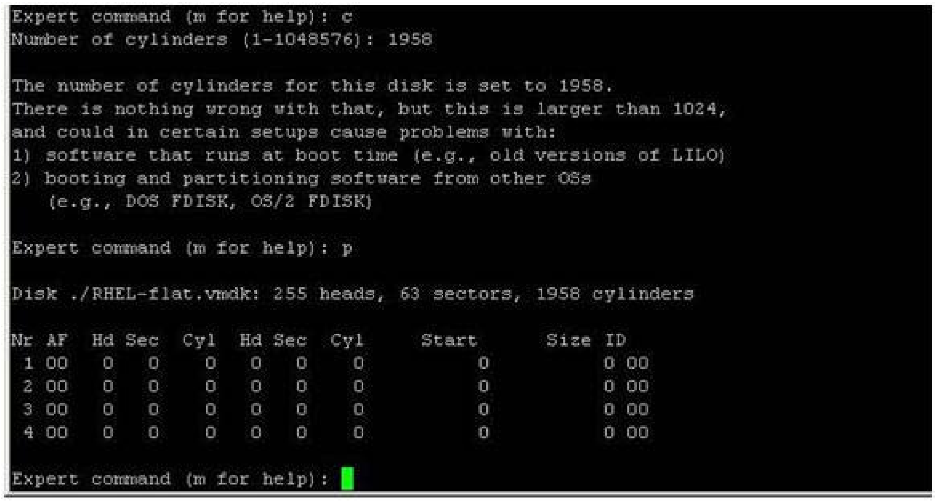
   You must set the number of cylinders.

9. Enter c and press Enter.
10. Enter the number of cylinders that you found from step 6 and press Enter.



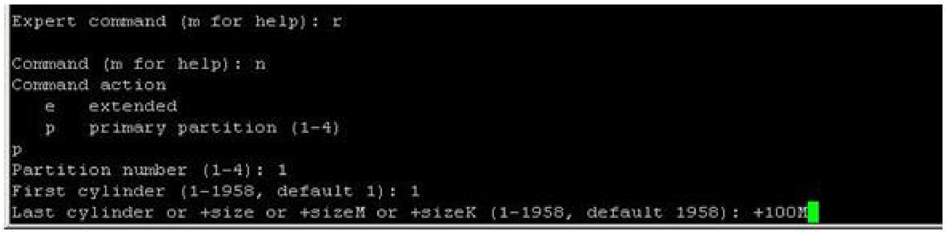
```
Command (m for help): x

Expert command (m for help): c
Number of cylinders (1-1048576): 1958
```

11. Enter p at the expert command screen to look at the partition table, which should be blank.



```
Expert command (m for help): c
Number of cylinders (1-1048576): 1958

The number of cylinders for this disk is set to 1958.
There is nothing wrong with that, but this is larger than 1024,
and could in certain setups cause problems with:
1) software that runs at boot time (e.g., old versions of LILO)
2) booting and partitioning software from other OSs
   (e.g., DOS FDISK, OS/2 FDISK)

Expert command (m for help): p

Disk ./RHEL-flat.vmdk: 255 heads, 63 sectors, 1958 cylinders

Nr AF  Hd Sec  Cyl  Hd Sec  Cyl     Start      Size ID
 1 00   0   0    0   0   0    0         0         0 00
 2 00   0   0    0   0   0    0         0         0 00
 3 00   0   0    0   0   0    0         0         0 00
 4 00   0   0    0   0   0    0         0         0 00

Expert command (m for help):
```

12. Enter `r` at the prompt to return to regular (nonextended) command mode.
13. Enter n and then p to create a new 100MB partition to host the /boot filesystem when asked for the partition type.
14. Enter `1` for the partition number, enter `1` for the first cylinder, and press Enter for the last cylinder question.
15. Enter `+100MB` for the last cylinder question.

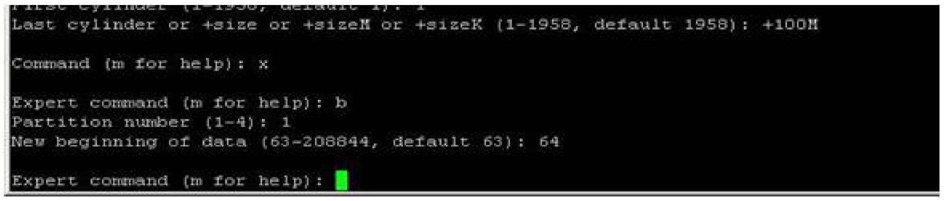


```
Expert command (m for help): r

Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 1
First cylinder (1-1958, default 1): 1
Last cylinder or +size or +sizeM or +sizeK (1-1958, default 1958): +100M
```

Note: During RHEL installation, the default block size of 1,024 bytes is used for the 100MB file system that is mounted as /boot. This does not affect alignment, but results in less efficient writes when compared to using a file system block size of 4,096 bytes or a multiple thereof. However, because /boot is only written to during installation (15MB), upgrade, and configuration changes, there is no need to change this.

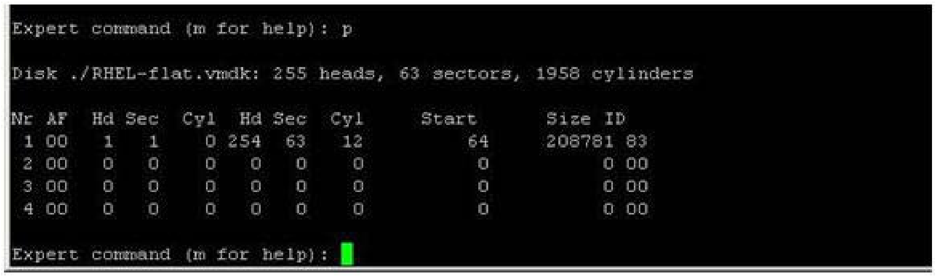
16. Enter x to go into extended mode to set the starting offset.
17. Enter b to set the starting offset and press Enter.
18. Enter `1` for the partition and press Enter.
19. Enter 64 and press Enter.

Note: In this procedure, 64 is used as an example. You may choose any value as long as it is divisible by 8. The reason the logical block address needs to be divisible by 8 is because each sector is 512 bytes in size. Eight multiplied by 512 results in 4,096 (or 4KB).



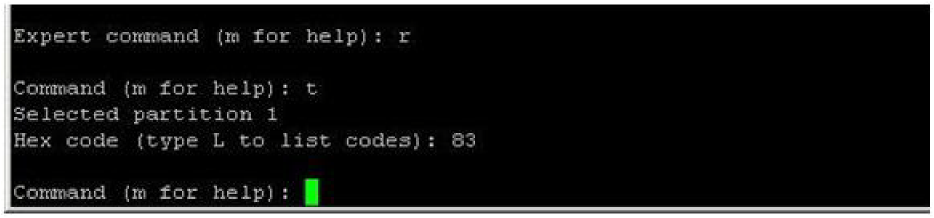
```
Last cylinder or +size or +sizeM or +sizeK (1-1958, default 1958): +100M

Command (m for help): x

Expert command (m for help): b
Partition number (1-4): 1
New beginning of data (63-208844, default 63): 64

Expert command (m for help):
```

20. Enter p to check the partition table.



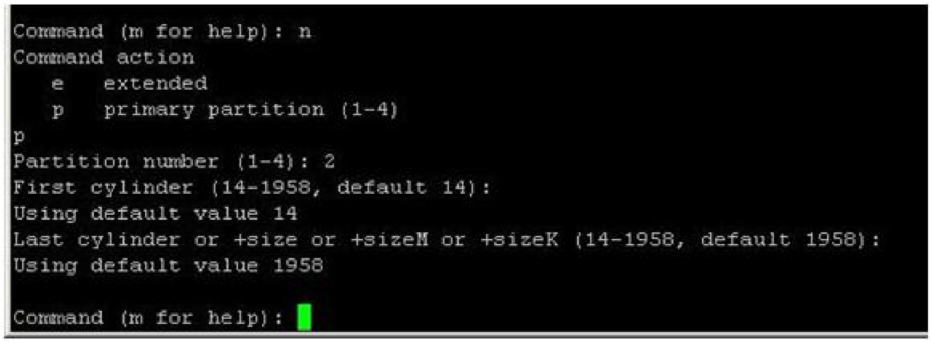
```
Expert command (m for help): p

Disk ./RHEL-flat.vmdk: 255 heads, 63 sectors, 1958 cylinders

Nr AF  Hd Sec  Cyl  Hd Sec  Cyl     Start      Size ID
 1 00   1   1    0 254  63   12        64    208781 83
 2 00   0   0    0   0   0    0         0         0 00
 3 00   0   0    0   0   0    0         0         0 00
 4 00   0   0    0   0   0    0         0         0 00

Expert command (m for help):
```

21. Enter `r` to return to the regular menu.
22. Enter `t` to set the system type to Linux.
23. Enter `83` for the hexcode.



```
Expert command (m for help): r

Command (m for help): t
Selected partition 1
Hex code (type L to list codes): 83

Command (m for help):
```

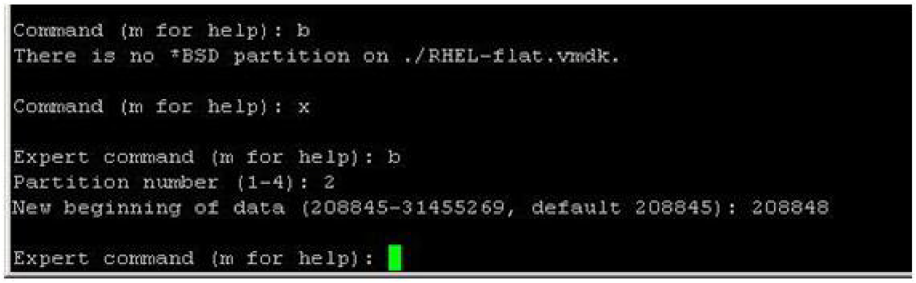
24. Create a second partition on the same vmdk to host the other file systems (/, /var, boot) on LVM by entering n and then p when asked for the partition type.
25. Enter 2 for the partition number, choose default value for the first cylinder, and press Enter for the last cylinder question to use the default value.



```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (14-1958, default 14):
Using default value 14
Last cylinder or +size or +sizeM or +sizeK (14-1958, default 1958):
Using default value 1958

Command (m for help):
```

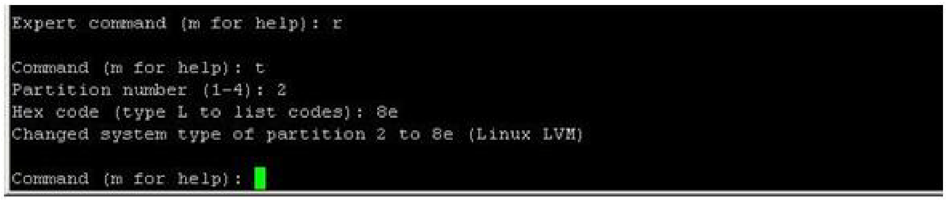
26. Enter x to go into extended mode to set the starting offset.
27. Enter b to set the starting offset and press Enter.
28. Enter 2 for the partition and press Enter.
29. Enter `208848` as the new beginning of data and press Enter.

Note: In this procedure, 208,848 is used as an example. You may choose any value as long as it is divisible by 8. The reason the logical block address needs to be divisible by 8 is because each sector is 512 bytes in size. Eight multiplied by 512 results in 4,096 (or 4KB).



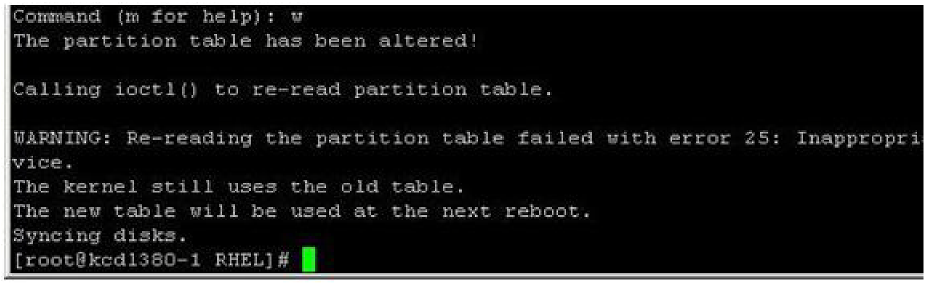
```
Command (m for help): b
There is no *BSD partition on ./RHEL-flat.vmdk.

Command (m for help): x

Expert command (m for help): b
Partition number (1-4): 2
New beginning of data (208845-31455269, default 208845): 208848

Expert command (m for help):
```

30. Enter p to check the partition table.
31. Enter `r` to return to the regular menu.
32. Enter `t` to set the system type to Linux LVM.
33. Enter 8e for the Linux LVM hexcode*.*



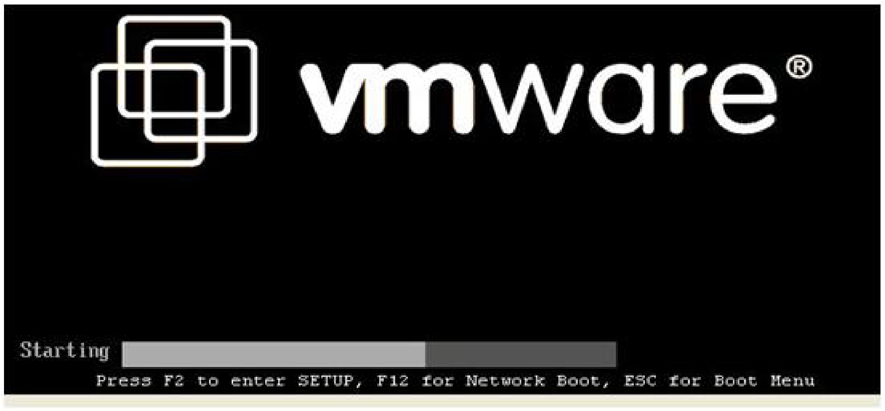
```
Expert command (m for help): r

Command (m for help): t
Partition number (1-4): 2
Hex code (type L to list codes): 8e
Changed system type of partition 2 to 8e (Linux LVM)

Command (m for help):
```

34. Enter w to save and write the partitions. Ignore the warning because this is normal.

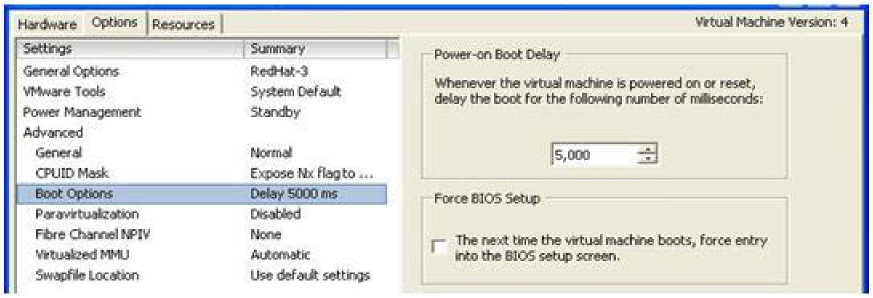


```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.

WARNING: Re-reading the partition table failed with error 25: Inappropri
vice.
The kernel still uses the old table.
The new table will be used at the next reboot.
Syncing disks.
[root@kcd1380-1 RHEL]#
```

35. Start the VM and run Linux setup. Make sure to press Escape to bring up the boot menu and select CD ROM drive to boot from the CD.


vmware®
Starting
Press F2 to enter SETUP, F12 for Network Boot, ESC for Boot Menu

If you miss the boot menu, the VM might appear to hang as a black screen with only a blinking cursor. Press ctrl-alt-insert to reboot the VM and try again to catch the boot menu by pressing Escape. If you have trouble catching the boot process, you can insert a boot delay in the VM settings. In the VI Client, right-click the VM, then > Edit Settings > Options > Advanced / Boot Options.



Hardware | Options | Resources | Virtual Machine Version: 4

Settings	Summary
General Options	RedHat-3
VMware Tools	System Default
Power Management	Standby
Advanced	
General	Normal
CPUID Mask	Expose Nx flag to ...
Boot Options	Delay 5000 ms
Paravirtualization	Disabled
Fibre Channel NPIV	None
Virtualized MMU	Automatic
Swapfile Location	Use default settings

Power-on Boot Delay

Whenever the virtual machine is powered on or reset, delay the boot for the following number of milliseconds:

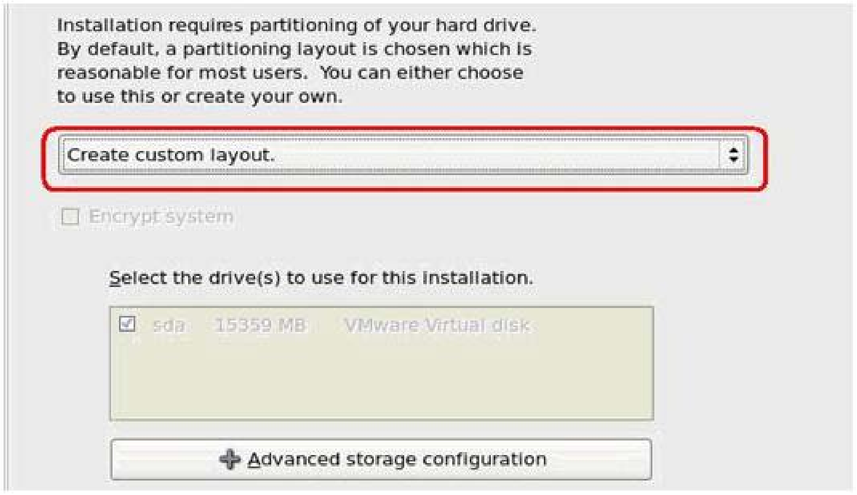
5,000

Force BIOS Setup

The next time the virtual machine boots, force entry into the BIOS setup screen.

Note: Boot delay is in milliseconds. You should return the boot delay to 0 after the VM boots from its virtual desk normally.

36. On the partitioning layout screen, select Create Custom Layout and click Next.



Installation requires partitioning of your hard drive. By default, a partitioning layout is chosen which is reasonable for most users. You can either choose to use this or create your own.

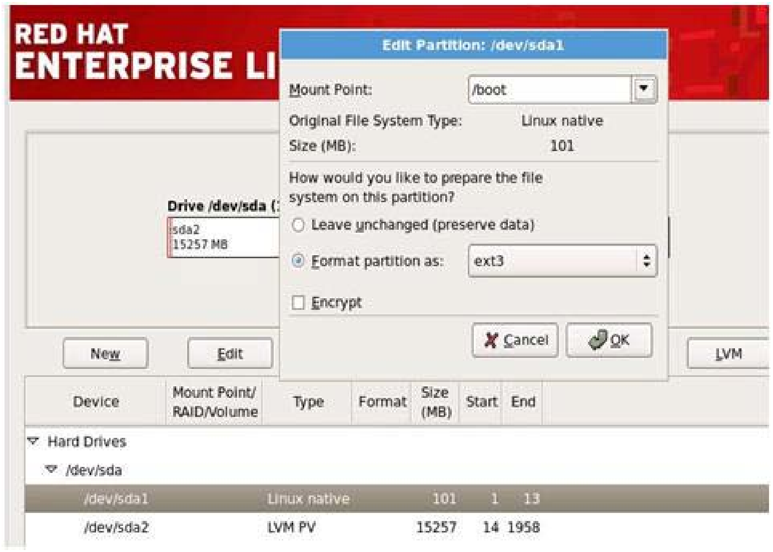
Create custom layout.

Encrypt system

Select the drive(s) to use for this installation.

sda 15359 MB VMware Virtual disk

37. On the screen to specify the partitions for different file systems, select the first partition /dev/sda1 and click Edit.
38. On the Edit Partition dialog box, select /boot for Mount Point. Click the radio button for Format partition as and then select ext3 as the format partition.



Edit Partition: /dev/sda1

Mount Point: /boot

Original File System Type: Linux native

Size (MB): 101

How would you like to prepare the file system on this partition?

( ) Leave unchanged (preserve data)

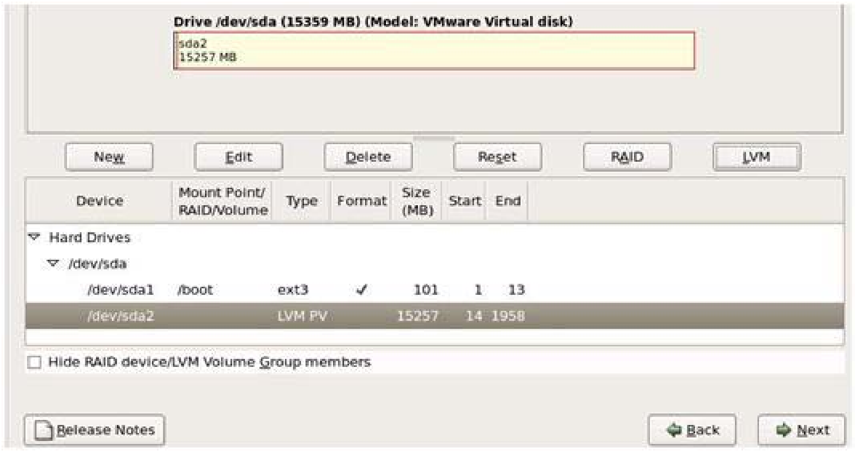
(●) Format partition as: ext3

[ ] Encrypt

Drive /dev/sda (
sda2
15257 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
Hard Drives						
/dev/sda						
/dev/sda1		Linux native		101	1	13
/dev/sda2		LVM PV		15257	14	1958

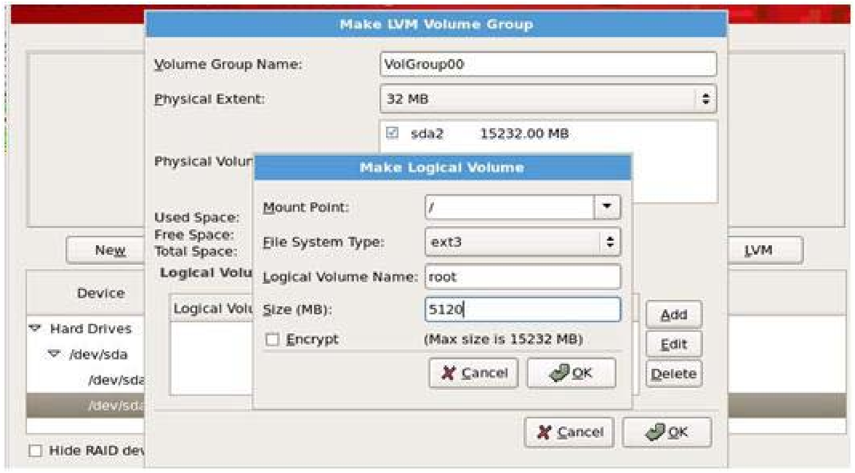
39. Select /dev/sda2 and click LVM to open the Make LVM Volume Group dialog box.


Drive /dev/sda (15359 MB) (Model: VMware Virtual disk)
sda2
15257 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
Hard Drives						
/dev/sda						
/dev/sda1	/boot	ext3	✓	101	1	13
/dev/sda2		LVM PV		15257	14	1958

[ ] Hide RAID device/LVM Volume Group members

40. On the Make LVM Volume Group dialog box, click Add to open the Make Logical Volume dialog box.


Make LVM Volume Group

Volume Group Name: VolGroup00

Physical Extent: 32 MB

Physical Volur

[✓] sda2 15232.00 MB

Used Space:
Free Space:
Total Space:

Logical Volu

Logical Volu

Make Logical Volume

Mount Point: /

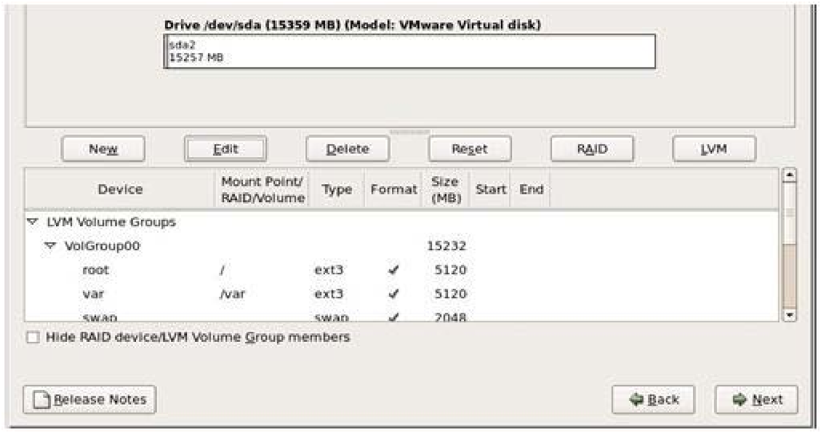
File System Type: ext3

Logical Volume Name: root

Size (MB): 5120

[ ] Encrypt (Max size is 15232 MB)

41. Enter the mount point, file system type, logical volume name, and size. Then click OK. Add the different mount points (/, /var, swap).
42. Click OK to go back to the window that shows the different partitions and click Next.



Drive /dev/sda (15359 MB) (Model: VMware Virtual disk)

sda2
15257 MB

Device	Mount Point/ RAID/Volume	Type	Format	Size (MB)	Start	End
LVM Volume Groups						
VolGroup00				15232		
root	/	ext3	✓	5120		
var	/var	ext3	✓	5120		
swap		swap	✓	2048		

☐ Hide RAID device/LVM Volume Group members

43. The Format warnings pop-up window for partition /dev/sda1 appears. Click Format.
44. Complete the Linux installation by clicking Next and following the standard best practices as you would normally do in your environment.

Adding Additional Disks to Existing Linux VM

If additional disks (for example, a second vmdk) are required, follow these scenarios so that the disks are aligned.

- **Scenario 1: New disk to be used in LVM.** NetApp recommends using `pvcreate` to create a physical volume on the whole disk (for example, `pvcreate /dev/sdb`) rather than a partition on the new disk. This new physical volume on the whole disk can be used in multiple ways:
  - To create a new LVM volume group (for example, `vgcreate VolGroup01 /dev/sdb`) and to create LVs and file systems on this new volume group
  - To add the new physical volume to an existing LVM volume group (for example, `vgextend VolGroup00 /dev/sdb`) and to create LVs and file systems on the existing volume group
- **Scenario 2: New disk not to be used in LVM.** If the new virtual disk will not be used in LVM, NetApp recommends building the file system on the entire disk device instead of creating the file system on a partition. The following example demonstrates creating an `ext3` file system on an entire disk device. This command is run from inside the guest VM.

  ```
  $ mkfs -t ext3 /dev/sdb
  mke2fs 1.40.2 (12-Jul-2007)
  /dev/sdb is entire device, not just one partition!
  Proceed anyway? (y,n) y
  ```

  However, if creating partitions on the new disk is desired, NetApp recommends using `fdisk` or `parted` from inside the guest VM to create partitions on the new disk so that the starting offset is divisible by 8.

4.5 GUEST ALIGNMENT IN RHEV AND KVM

USING PARTED IN RHEV AND KVM TO PROPERLY ALIGN RHEL GUEST VMS

The first method recommended for aligning RHEL guest VMs involves using Kickstart to automate the alignment prior to OS installation for both RHEV and KVM.

This technical report does not discuss how to set up a Kickstart; it discusses only the two sections of a Kickstart file that are relevant to disk alignment. More information on Kickstart, is widely available, including at Red Hat Documentation.

The two sections of the Kickstart file that we are concerned with are the optional %pre section at the end of the file and the section that specifies the disk layout.

After creating or modifying a Kickstart file, add a %pre section such as the following:

```
%pre
parted /dev/sda mklabel msdos
parted /dev/sda mkpart primary ext3 64s 208718s
parted /dev/sda mkpart primary 208720s 100%
parted /dev/sda set 2 lvm on
```

The first `parted` command creates a disk label on disk /dev/sda. The second `parted` command creates a primary partition of just over 100MB that starts at sector 64. The third `parted` command creates a second primary partition starting at the next available alignment-friendly sector of 208720 that takes up whatever disk space is left over. The final `parted` command simply changes the partition type to LVM for the second partition. Keep in mind that additional partitions can be created, provided they each start on alignment-friendly sectors (those that are cleanly divisible by 8). Also, the /dev/sda can be edited to /dev/hda or /dev/vda, depending on how the operating system sees the disk.

The other important section in the Kickstart file is where the disk layout is specified. Notice the differences between the following examples. Note that options in both examples are -- (2 dashes).

EXAMPLE 1

```
zerombr yes
clearpart --linux --drives=sda
part /boot --fstype ext3 --size=100 --ondisk=sda
part pv.2 --size=0 --grow --ondisk=sda
volgroup VolGroup00 --pesize=32768 pv.2
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=1008 --
grow --maxsize=2016
logvol / --fstype ext3 --name=LogVol00 --vgname=VolGroup00 --size=1024 –grow
```

EXAMPLE 2

```
zerombr yes
##clearpart --linux --drives=sda
part /boot --fstype ext3 --onpart hda1
part pv.2 —onpart hda2
volgroup VolGroup00 --pesize=32768 pv.2
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=1008 \--
grow --maxsize=2016
logvol / --fstype ext3 --name=LogVol00 --vgname=VolGroup00 --size=1024 --grow
```

The first difference in Example 2 is that the clearpart line is commented out. (It can also be deleted.) If left in, the line simply wipes out the partitions created in the %pre section that was just created. The second difference is that the two partitions don’t have size directives; the sizes were predetermined in the %pre, thus you only need to specify file system type and partition. Keep in mind that if the %pre section specifies /dev/sda or /dev/vda, the disk layout in the main section needs to follow suit.

The other sections of the Kickstart file remain as normal, as do the typical Kickstart procedures. The %pre section is executed prior to the rest of the Kickstart, resulting in an aligned disk. The disk layout section then applies its parameters to the newly created partitions.

MANUALLY CREATING A PROPERLY ALIGNED VIRTUAL DISK IN KVM

Manually creating properly aligned disk images is simply a matter of using Red Hat's KVM and the QEMU implementation tools. This is not valid for RHEV. The first task is to create a disk image, raw in this case. (You can also use qcow2.)

```
qemu-img create –f raw new_disk.img 10G
```

Next, use `parted` to create properly aligned partitions. In the following example, the layout is appropriate for a RHEL VM:

```
parted /dev/sda mklabel msdos

parted /dev/sda mkpart primary ext3 64s 208718s

parted /dev/sda mkpart primary 208848s 100%

parted /dev/sda set 2 lvm on
```

This creates two partitions, one for /boot and the other for an LVM-managed partition. From there, boot the VM with the RHEL installation DVD. After the Disk Layout window comes up, complete the installation by following steps 36 through 44 from the procedure found in the section Using fdisk from ESX Service Console for Linux Guest VMs.

For a Windows VM, use the following commands:

```
parted /dev/hda mklabel msdos

parted /dev/hda mkpart primary NTFS 128s 100%
```

From there, simply boot the installation DVD and install the image to the newly created partition. Note that this is not required for Windows 2008, Windows Vista, or RHEL 6 because they are automatically aligned.

5 FIXING FILE SYSTEM ALIGNMENT

5.1 DETECTION

Alignment issues can be detected in multiple ways, including:

- Using the NetApp `mbrscan` tool (Windows and Linux guest VMs)
- Using `msinfo32` (Windows guest VM only)
- Using `fdisk` from the ESX service console (Windows and Linux guest VM)
- Using `fdisk` from within a RHEV or KVM VM (Linux guest VMs only)

USING THE NETAPP MBRSCAN TOOL (WINDOWS AND LINUX GUEST VMS)

The NetApp `mbrscan` tool interrogates a VM disk file and reports on the file system alignment. It can effectively check for alignment on VMware `vmdk` and thick type `vhd` files for Citrix XenServer that are partitioned using MBR.

- For VMware VMFS and NFS datastores, this tool can be run from the ESX console. For NFS datastores, it can also be run from any UNIX® or Linux host that has permissions to mount the NFS datastore.
- For Citrix XenServer NFS SR, this tool can be run from domain 0 (dom0).

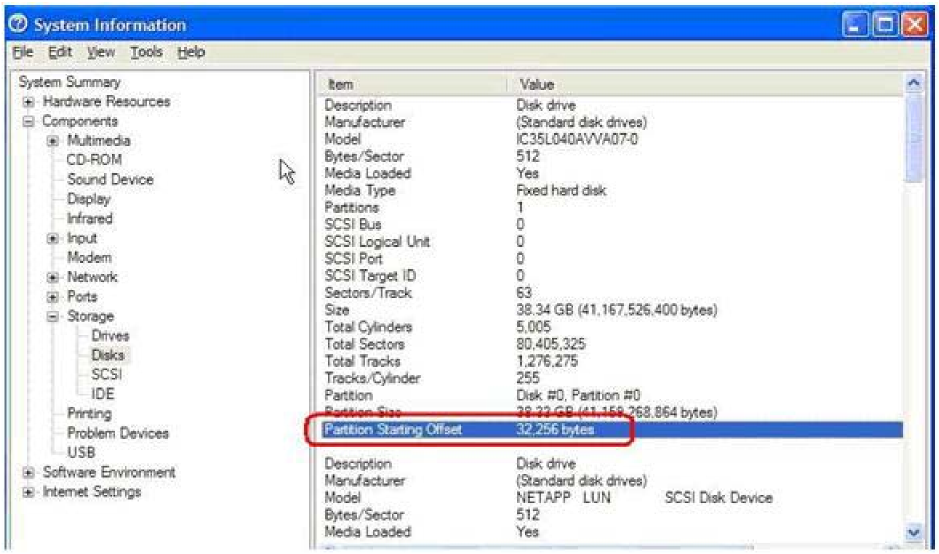
The `mbrscan` tool is available as part of VMware ESX Host Utilities 5.0 or higher (for both VMware Infrastructure 3 and VMware vSphere 4), which can be downloaded from the NetApp NOW site. For the detailed procedure on how to use the `mbrscan` tool, refer to the ESX Host Utilities 5.1 Installation and Setup Guide.

USING MSINFO32 (WINDOWS GUEST VM ONLY)

This procedure works for all of the hypervisors, including VMware ESX, Citrix XenServer, and Microsoft Hyper-V. When aligning the file systems for use with NetApp storage systems, the starting partition offset must be divisible by 4,096. For Windows guest operating systems, verifying this value is easy. Run `msinfo32` on the guest VM by selecting Start > All Programs > Accessories > System Tools > System Information. Next, navigate to Components > Storage > Disks and check the value for Partition Starting Offset. For misaligned VMs, you typically will find that the VM is running with a default starting offset value of 32,256, which is not completely divisible by 4,096; hence, the partition is not aligned as shown in Figure 11.

Note: For Windows LUN type RDM or raw LUNs directly mapped to the VM, 32,256 is reported as the correct starting offset value. This is true because the storage controller compensated for the offset when selecting the correct LUN type.

Figure 11) Using system information to identify the starting partition offset.



Item	Value
Description	Disk drive
Manufacturer	(Standard disk drives)
Model	IC35L040AVVA07-0
Bytes/Sector	512
Media Loaded	Yes
Media Type	Fixed hard disk
Partitions	1
SCSI Bus	0
SCSI Logical Unit	0
SCSI Port	0
SCSI Target ID	0
Sectors/Track	63
Size	38.34 GB (41,167,526,400 bytes)
Total Cylinders	5,005
Total Sectors	80,405,325
Total Tracks	1,276,275
Tracks/Cylinder	255
Partition	Disk #0, Partition #0
Partition Size	38.33 GB (41,158,268,864 bytes)
Partition Starting Offset	32,256 bytes
Description	Disk drive
Manufacturer	(Standard disk drives)
Model	NETAPP LUN SCSI Disk Device
Bytes/Sector	512
Media Loaded	Yes

USING FDISK FROM THE ESX SERVICE CONSOLE (WINDOWS AND LINUX GUEST VM)

For VMware deployments, you can also check the LUN alignment using the `fdisk` tool from the ESX service console.

Enter the following command at the service console command prompt.

```
fdisk -lu /vmfs/volumes/<datastore>/<vm>/<vm>-flat.vmdk
```

This reports the required information, including starting offsets, and then exits. If the Start value is 63 (or any other number not divisible by 8), the partition is not aligned. This procedure works for VMware ESX only. For more information, see the following VMware documentation: VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions.

USING FDISK FROM WITHIN A RHEV OR KVM VM (LINUX GUEST VMS ONLY)

This procedure works for both RHEV and KVM. Once logged into the VM, open a console and determine which disk is used as the primary disk for the VM.

Enter the following command to list the partitions that the kernel sees.

```
cat /proc/partitions
major minor  #blocks  name
253     0   26214400 vda
```

```
253     1     104327 vda1
253     2   26109976 vda2
```

For RHEL 5.4 or later, the disk(s) is likely to be vda, vda1, vda2, and so on. For earlier versions of RHEL or other Linux distributions, the disk(s) is likely to be sda, sda1, sda2, and so on.

Enter the following command to output the partition information.

```
fdisk –lu /dev/vda
Disk /dev/vda: 26.8 GB, 26843545600 bytes
4 heads, 32 sectors/track, 409600 cylinders, total 52428800 sectors
Units = sectors of 1 * 512 = 512 bytes
   Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *          64      208718      104327+  83  Linux
/dev/vda2          208848    52428799    26109976   8e  Linux LVM
```

For each partition, there is a value listed in the Start column. This is the sector number, and each start sector needs to be cleanly divisible by 8, for example, 64 or 128. The default start of sector 63 is misaligned. All subsequent partitions must also be cleanly divisible by 8.

5.2 CORRECTION

Correcting the starting offset is best addressed by correcting the template from which new VMs are provisioned. Aligning the boot disk in existing virtual machines is not required but improves performance under some circumstances. If you have diagnosed performance issues due to misalignment, it might be necessary to align boot disks as well as data disks. In any case, it is essential to align templates and gold image VMs in order to prevent proliferation of misaligned VMs, and to align any new VMs using procedures in this document.

The following list contains examples of situations in which misaligned boot disks can become an issue:

- The boot disk is also the data disk, as is the case for VMs that have only one virtual disk.
- Oversubscription of memory within a VM, triggering swapping or paging within the VM. This can be alleviated by aligning the virtual disk that contains the page file or swap partition (this is often the boot disk), increasing the memory allocated to the VM, or both.
- Using the balloon driver for memory management when physical memory is oversubscribed can trigger VMs to swap or page.

Note: ESX swapping a VM into the `.vswp` file is not affected by alignment of the VM boot disk, but is affected by alignment of VMFS.

CORRECTION USING FDISK OR DISKPART

Use either of the correction procedures described in section 4.4 to create a new aligned virtual disk. Attach this new aligned virtual disk to the VM and copy the contents from the existing misaligned virtual disk to the new disk. Finally, detach and destroy the misaligned virtual disk after verifying the contents and integrity of the data on the new aligned virtual disk. If the misaligned virtual disk is the boot partition, follow these steps.

1. Back up the VM system image.
2. Shut down the VM.
3. Attach the misaligned system image virtual disk to a different VM.
4. Attach a new aligned virtual disk to this VM.
5. Copy the contents of the system image (for example, C: in Windows) virtual disk to the new aligned virtual disk. You can use the following tools to copy the contents from the misaligned virtual disk to the new aligned virtual disk:

- Windows xcopy
- Norton/Symantec™ Ghost: Norton/Symantec Ghost can be used to back up a full system image on the misaligned virtual disk and then be restored to a precreated, aligned virtual disk file system.

For VMware RDM and LUNs directly mapped to the guest OS, create a new LUN using the correct LUN type. Next, map the LUN to the VM and copy the contents from the misaligned LUN to this new LUN.

For Microsoft Hyper-V NTFS-formatted LUNs mapped to the Hyper-V parent partition using incorrect LUN type but with aligned VHDs, create a new LUN using the correct LUN type and copy the contents from the misaligned LUN (VHDs) to this new LUN. However, if the VHDs are also misaligned in addition to the NTFS-formatted LUNs being mapped to the Hyper-V parent partition, first create a new LUN using the correct LUN type and copy the contents from the misaligned LUN (VHDs) to this new LUN. Then perform the steps described at the beginning of section 4.4 to create new aligned VHDs on the new LUN. Copy the contents from the existing VHDs to the new VHD. If the VHD is a boot partition, follow the steps described earlier in this section. For pass-through disks and LUNs directly mapped to the child OS, create a new LUN using the correct LUN type, map the LUN to the VM, and copy the contents from the misaligned LUN to this new LUN.

For Citrix XenServer LUNs created using NetApp Data ONTAP Adapter, create a new LUN, map the LUN to the VM, and copy the contents from the misaligned LUN to this new LUN.

CORRECTION USING MBRALIGN

The NetApp mbralign tool properly aligns a guest file system to the NetApp storage array blocks. This tool works for VMware `vmdk` files hosted on VMFS or NFS datastores and is run on the ESX service console. The following high-level steps provide direction for correcting the misalignment:

1. Shut down the VM.
2. Run the mbralign tool on each vmdk.
3. Start up the VM.
4. After the VM starts correctly and is verified intact, delete the backup vmdk created by mbralign.

FIXING ALIGNMENT FOR MISALIGNED VMS HOSTED ON VMWARE ESXI

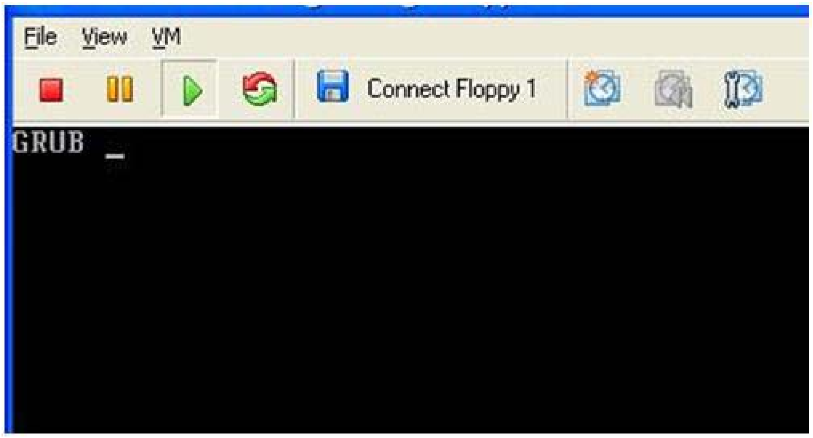
In order to leverage the NetApp mbralign tool for correcting the file system partition alignment for misaligned VMs hosted on ESXi hosts, NetApp recommends leveraging a standby VMware ESX host and using a supported version of the mbralign tool to fix the VM alignment issues. This standby VMware ESX host should have access to the NFS/VMFS datastores hosting the misaligned VMs. Also, it is not required that this ESX host be part of the HA/DRS cluster hosting production VMs.

The mbralign tool, along with detailed instructions, is available for download here, on the NetApp Tool Chest, or on the NetApp NOW site.

FIXING GRUB FOR LINUX VMS

After using mbralign to align a Linux VM, GRUB is usually no longer able to find the stage 1.5 boot loader. The VM stops with a black screen and the word GRUB appears, as shown in Figure 12.

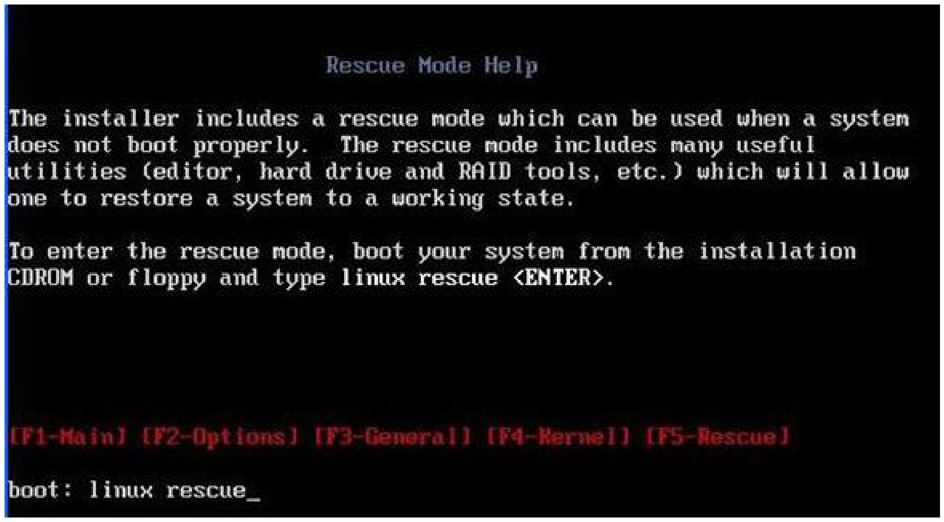
Figure 12) A VM hung at GRUB.



```
GRUB _
```

To fix this issue, follow these steps.

1. Boot the VM from a Linux Rescue CD/ISO. (Disk 1 of your Linux distribution usually works.)
2. At the GRUB hang display in the VM remote console, click in the display area to make sure it is active.
3. Press ctrl-alt-insert to reboot the VM.
4. As soon as you see the VMware BIOS splash, press Escape once. You should have a boot menu. If you have trouble catching the boot menu, you can add a boot delay by beginning at step 23 near the end of the procedure Using fdisk from ESX Service Console for Windows Guest VMs.
5. Select CD-ROM.
6. At the Linux boot screen, enter the command `linux rescue` to boot Linux in rescue mode.

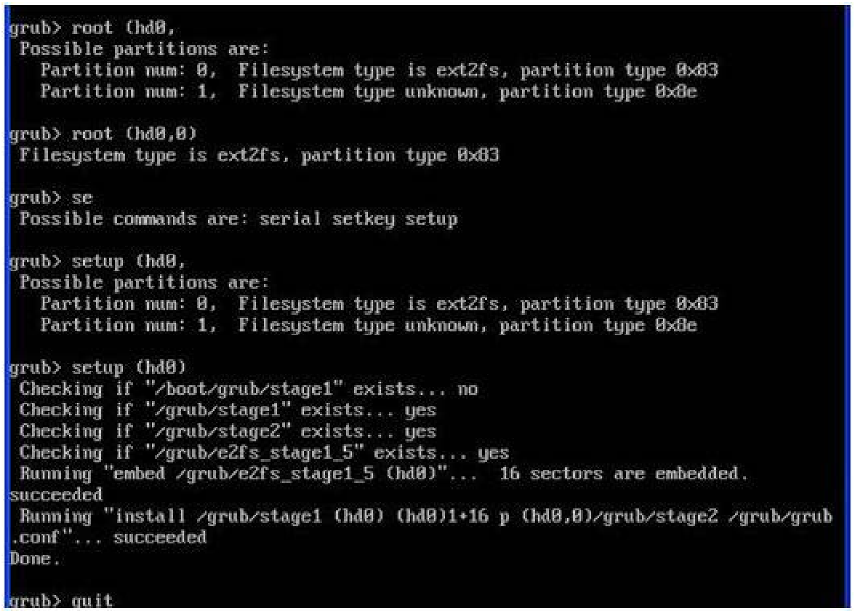


```
                         Rescue Mode Help

The installer includes a rescue mode which can be used when a system
does not boot properly.  The rescue mode includes many useful
utilities (editor, hard drive and RAID tools, etc.) which will allow
one to restore a system to a working state.

To enter the rescue mode, boot your system from the installation
CDROM or floppy and type linux rescue <ENTER>.



[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]

boot: linux rescue_
```

7. Take the defaults for Anaconda (the blue/red configuration screens).
   Networking is optional.
8. Enter `grub` to launch GRUB.
9. Select the correct disk and fix the GRUB stages as follows:

```
grub> root (hd0,0)
grub> setup (hd0)
grub> quit
ctrl-d
```

Note: GRUB provides a form of command completion by pressing tab, which also lists hard disks and partitions, where appropriate. The following screenshot illustrates this effect. Also note that if you have multiple disks, hd0 is probably your boot disk but you must know how Linux and its components were installed.



```
grub> root (hd0,
 Possible partitions are:
   Partition num: 0,  Filesystem type is ext2fs, partition type 0x83
   Partition num: 1,  Filesystem type unknown, partition type 0x8e

grub> root (hd0,0)
 Filesystem type is ext2fs, partition type 0x83

grub> se
 Possible commands are: serial setkey setup

grub> setup (hd0,
 Possible partitions are:
   Partition num: 0,  Filesystem type is ext2fs, partition type 0x83
   Partition num: 1,  Filesystem type unknown, partition type 0x8e

grub> setup (hd0)
 Checking if "/boot/grub/stage1" exists... no
 Checking if "/grub/stage1" exists... yes
 Checking if "/grub/stage2" exists... yes
 Checking if "/grub/e2fs_stage1_5" exists... yes
 Running "embed /grub/e2fs_stage1_5 (hd0)"...  16 sectors are embedded.
succeeded
 Running "install /grub/stage1 (hd0) (hd0)1+16 p (hd0,0)/grub/stage2 /grub/grub
.conf"... succeeded
Done.

grub> quit
```

10. After logging out (ctrl-d), Linux rescue shuts down and reboots. Linux should come right up.

CORRECTION FOR RHEV- AND KVM-BASED VIRTUAL MACHINES

Currently, there are no supported tools available for correcting alignment issues within RHEV- or KVM-based VMs. While it is possible to attach a properly aligned disk to a VM and use tools such as dd or rsync to relocate blocks or files, it is a laborious and time-consuming exercise. For now, the following steps present the most reliable way to alleviate alignment issues on RHEV- and KVM-based VMs.

1. Perform a backup of any important local data.
2. Destroy the original VM.
3. Create a new VM using a properly aligned disk. (See the sections Using parted in RHEV and KVM to Properly Align RHEL Guest VMs and Manually Creating a Properly Aligned Virtual Disk in KVM.)
4. Restore data from backup.

CORRECTION VERIFICATION

You can verify the alignment correction by following the procedure to collect the Data ONTAP performance counters on LUNs, as described in section 5.1.

6 CONCLUSION

File system misalignment is a known issue in virtual environments and can cause VM performance issues. Follow the steps outlined in this report so that the issue is prevented during the storage provisioning and VM creation phase of the project. For existing VMs, follow the steps outlined in this report to detect and correct any existing misalignment issues. Note that the misalignment issue is not unique to NetApp storage and can occur with any storage array. For more details, refer to the following VMware articles:

- VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions
- Performance Best Practices for VMware vSphere 4.0

7 FEEDBACK

Send an e-mail to xdl-vgibutmevmtr@netapp.com with questions or comments concerning this technical report.

8 REFERENCES

- Cylinder-Head-Sector
  http://en.wikipedia.org/wiki/Cylinder-head-sector
- Data ONTAP Block Access Management Guide
  http://now.netapp.com/AskNOW/search?query=storage+management+guide&search_collections=docs&search_collections=kbase&search_collections=bugs&search_collections=tools&action=search
- Data ONTAP Commands Manual Page Reference
  http://now.netapp.com/AskNOW/search?query=Commands+Manual+Page+Reference+Document&search_collections=docs&search_collections=kbase&search_collections=bugs&search_collections=tools&action=search&x=11&y=8
- Designing and Optimizing Dell/EMC SAN Configurations
  www.dell.com/downloads/global/power/ps4q04-20040149-Mehis.pdf
- Disk Performance May Be Slower Than Expected When You Use Multiple Disks in Windows Server 2003, in Windows XP, and in Windows 2000
  http://support.microsoft.com/kb/929491
- EMC: CLARiiON Integration with VMware ESX Server
  www.vmware.com/pdf/clariion_wp_eng.pdf
- ESX Host Utilities 5.1 Installation and Setup Guide
  http://now.netapp.com/NOW/knowledge/docs/hba/esx/esxhu51/pdfs/install.pdf
- How to Align Exchange I/O with Storage Track Boundaries
  http://technet.microsoft.com/en-us/library/aa995867(EXCHG.65).aspx
- IBM Technical Report: Storage Block Alignment with VMware Virtual Infrastructure and IBM System Storage N Series
  ftp://service.boulder.ibm.com/storage/isv/NS3593-0.pdf
- mbralign v1.4 Frequently Asked Questions (FAQ)
  http://communities.netapp.com/docs/DOC-2932
- Partition Design
  http://technet.microsoft.com/en-us/library/bb738145(EXCHG.80).aspx
- Performance Best Practices for VMware vSphere 4.0
  www.vmware.com/pdf/Perf_Best_Practices_vSphere4.0.pdf
- Predeployment I/O Best Practices: SQL Server Best Practices Article
  http://technet.microsoft.com/en-us/library/cc966412.aspx
- Red Hat Enterprise Linux 5 Deployment Guide
  http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html/Deployment_Guide/index.html
- Red Hat Enterprise Virtualization for Servers
  https://access.redhat.com/knowledge/docs/manuals/Red_Hat_Enterprise_Virtualization_for_Servers/
- TR-3702: NetApp Storage Best Practices for Microsoft Virtualization
  http://media.netapp.com/documents/tr-3702.pdf

- TR-3732: Citrix XenServer and NetApp Storage Best Practices
  http://media.netapp.com/documents/tr-3732.pdf
- An Updated Version of the Disk Partition Tool for Windows Server 2003 Is Available
  http://support.microsoft.com/kb/923076
- Using EMC Celerra Storage with VMware vSphere and VMware Infrastructure
  www.emc.com/collateral/software/technical-documentation/h5536-vmware-esx-srvr-using-emc-celerra-stor-sys-wp.pdf
- VMware ESX Server Performance Tuning Best Practices for ESX Server 3
  www.vmware.com/pdf/vi_performance_tuning.pdf
- VMware Infrastructure 3 Recommendations for Aligning VMFS Partitions
  www.vmware.com/pdf/esx3_partition_align.pdf

NetApp®
www.netapp.com