

**University of South Florida St. Petersburg**

---

**From the Selected Works of Berrie Watson**

---

May 1, 2014

# Virtualizing Linux DSpace to VMWare.pdf

Berrie Watson, *University of South Florida St. Petersburg*



Available at: <https://works.bepress.com/berriewatson/2/>



# VIRTUALIZING DSPACE

Virtualizing Linux CentOS to VMWare: A Physical to Virtual  
Migration and step-by-step Guide to Virtualizing a Physical Linux  
Server Hosting DSpace Software

Berrie Watson  
[Email address]

## Virtualizing Linux CentOS to VMWare: A Physical to Virtual Migration

A step-by-step Guide to Virtualizing a Physical Linux Server Hosting DSpace Software

---

This guide is intended for some of us veteran Windows admins that have been thrust into supporting virtualized Linux. I tried to use standard easy to use tools that emulate processes used on the Windows side, such as cloning from boot disks, but along the way I have found some of the Linux commands to be very powerful, although not familiar. Linux veterans will likely find shortcuts to this process.

### *Why use this Guide instead of VMWare Converter?*

One main issue in the virtualization process is that when installing Linux on a physical machine, only the drivers needed for physical hardware are included in the Kernel. This results in a "kernel panic" if drivers for drives change type (sda, sdb, etc for SCSI instead of IDE) so booting fails after conversion- the drivers simply aren't present. Also at issue is LVM the Linux Logical Volume Manager. If drivers or kernel specific to these disks are not available, (also likely due to drivers not loaded into the RAMDISK) then the volumes aren't mounted. This conversion solves these issues by copying files from LVM to standard volumes- pre-formatted on the target VM by pre-installing Linux.

### **You will need:**

A Linux boot disk, such a Knoppix, or the Red Hat Boot Disk to boot your virtual server in order to copy partitions and adjust the system to boot.

A flash drive, to use to copy kernel & files from a running VM to the cloned VM for booting.

An external USB enclosure, to mount your physical drive to the VM. VMWare Workstation for the conversion and to easily mount USB drives. Later the VM can be ported to ESX.

### **Safeguard Your Data**

Depending on your threshold for risk, to protect your original physical drive (or array, etc) you can use a UNIX cloning utility such as g4l to copy your physical drive to a SATA disk that is large enough to hold all of the partitions of the original. Optional, but to follow the first rule of computing.. 'do no harm'... I chose to make a physical copy of the server first onto another drive, then use that drive for the conversion. I used

d4l, which booted the system from CD and performed a sector-by-sector copy of the entire drive very easily. I then booted from the copied disk.

## Prepare your Physical Disks

On your physical server, discover the drive and partition sizes:

Fdisk -l, fdisk -l (drive), blkid, df -h

Write down the volumes, sizes, mount points:

Partition	start-end	size	type	type	mount
/dev/sda1	1-13	104391	83	Linux	/boot
/dev/sda2	14-523	4,096,575	83	Linux	/home
/dev/sda3	524-1033	4,096,575	83	Linux	/usr
/dev/sda4	1034-53309	419,906,970	5	extended	
/dev/sda5	1034-53309	419,906,938+	8e	Linux LVM	

My particular server had an LVM Volume, which complicated the conversion because the LVM wouldn't mount and my system would not boot. Through this process I converted the information to non-LVM partitions through issuing DD 'block copy' commands.

## Prepare Your Virtual Linux Server

Create a Linux Virtual Server by booting from the CentOS DVD on a New VM and adjusting/creating custom partitions to match your physical system. I created different vmdk files for each component of the file system to make resizing easier later, and avoided LVM volumes:

[Picture of VM Settings with Multiple Drives Here]

Drive One: /usr, /home /boot

Drive Two: / (root)

Drive Three: swap

It isn't important to perform a full install, since you will be wiping this installation out later. Just get Linux installed with all of the partitions, sizing your VM partitions to handle all of your physical disk sizes with some padding for resizing later (I upped mine to 200MB for /boot and 20GB for /home & /usr,):

Partition	start-end	size	type	type	mount
/dev/sda1	1-25	200781	83	Linux	/boot
/dev/sda2	26-2575	20,482,875	83	Linux	/home

```
/dev/sda3 524-1033      20,482,875 83   Linux    /usr
```

(swap omitted, I wanted this on a separate disk, see SWAP below)  
Next, insert a USB thumb drive (move USB function from host if necessary), and copy the /boot/kernel, initrd and \*\* other config files\*\* onto this usb drive, for later use. I copied both the default kernel & files as well as one older kernel that was installed on the working VM. That way I had two options for working kernels later.

```
cp /boot/*29.6* /mnt/usb
```

### **Install your disk to an enclosure**

Mount original or copied disk into your USB enclosure. I found that with VMWare Workstation there was no trouble identifying the USB-based disk while booted from Knoppix on the VM.

### **Boot from Linux CD**

Now that you have your Linux VM ready, set it to boot from CD (or .iso), and get it booted from Knoppix. Next, connect your enclosure to the VM (I used VMWare Workstation for this).

Once you have done this you can issue an `fdisk -l`, `df -h`, `blkid` . You should see the disks that you created for your VM as well as the disk & partitions that you cloned to the enclosure. Note that the USB enclosure disks should change from `sda1`, `sda2`, etc to `sdb1`, `sdb2`, etc. since they are not the primary drives.

### **Use DD to Clone Partition Contents**

Having NO partition mounted on either your VM or the USB SATA drive, you can use DD to clone the contents one partition at a time.

```
dd if=/dev/sdb1 of=/dev/sda1  
|--SOURCE --|    |--TARGET--|
```

You can also use this same command to copy from an LVM partition to a non-LVM partition:

```
Vgchange -a y [VolGroupName] (to activate LVM volumes)
```

```
dd if=/dev/mapper/VolGroup00-LogVol00 of=/dev/sdb1
```

A great source for using DD for backup can be found at [www.backuphowto.info/linux-backup-hard-disk-clone-dd](http://www.backuphowto.info/linux-backup-hard-disk-clone-dd).

Afterward, it is a good idea to check all of the ext3 filesystems for errors:

```
fsck.ext3 /dev/sda1
fsck.ext3 /dev/mapper/VolGroup00-LogVol00
etc.
```

After the partition has been increased, and the blocks copied, the next step is to get the filesystem to see the change in disk space:

```
e2fsck -f /dev/hdb2
resize2fs -f /dev/hdb2
```

Running Fdisk would show the original file system sizes, and not the changed ones, as far as fdisk is concerned the file system is already correct as it was (it doesn't see the change in size)

Next, set the boot partition on the copied partition to have the boot flag active. Otherwise the system will not boot.

### **Getting the System To Boot**

I had several issues to overcome to get the system to boot, including a kernel that didn't like LVM partitions, the need to place filesystems on separate vmrk drives, and issues with the swap file. I will go into each of these, although most of it revolved around creating and managing partitions.

### **Note about Mounting**

When booting from a CD, it becomes necessary to mount the various file structures in order to edit items such as /boot/grub/menu.lst, /etc/fstab, and to verify that your data exists.

1<sup>st</sup>, make a directory in order to mount, in /mnt or anywhere:

```
mkdir mounted
```

2<sup>nd</sup>, use the mount command to mount the device/partition:

```
mount -t ext3 /dev/sda1 mounted
```

change directory to mounted, and view your files.

### **Creating Partitions Using fdisk**

```
Fdisk /dev/[drive]
```

```
N (new) (size is starting block, then +XGB)
```

T type, 83 for ext3 filesystem  
W write changes to disk  
New partition becomes sda[#]

```
mkfs -t ext3 /dev/sda5
```

grub /boot/grub/menu.lst look for default 0  
if grub > appears when you boot, it found grub, but grub.conf is wrong (grub.conf is same as menu.lst file)

/etc/fstab 0= means no dump file 2 = means do filecheck after root filesystem (root gets mounted first)

### **Creating & Formatting Your Swap Partition**

I created a 4GB Hard Drive to attach to the VM, with its own VMDK file.  
Then, I noted what the disk designation was: sdc.

```
fdisk sdc  
N (new partition) (size is starting block, then +4GB)  
T type, 82 for Linux swap  
W (write changes to disk)  
Q (quit)
```

To zero out all the data so it can be used as swap space:

```
dd if=/dev/zero of=/dev/sdc1 bs=1024
```

To label the volume as swap:

```
mkswap -L SWAP-sdc1 /dev/sdc1
```

Next, edit the /etc/fstab file, to indicate the newly labeled swap file:

```
LABEL=SWAP-sdc1 swap swap default 0 0
```

Then, activate it:

```
swapon -a
```

The /etc/fstab entry is now made active as the swap drive/partition.

Current Disk Layout:

/dev/sda1	/boot
/dev/sda2	/home

/dev/sda3	/usr
/dev/sdb1	/ (root)
/dev/sdb2	swap

**\*\* Learn Parted to boot from a CD and adjust volume sizes.**

### **Copy your Kernel and Initialization Files from your VM**

Now that you have rewritten your original VM boot partition with one that may not boot, you can replace the Kernel and supporting files. This allowed the VM to boot while retaining all other files/configs needed for the migrated system.

### **Addendum: View, Copy, Adjust LVM Volumes**

For more info, see this valuable page at:

<http://tldp.org/HOWTO/LVM-HOWTO/commontask.html>

### **Mounting LVM Volumes**

If you have a Linux machine with LVM volumes on it, and you boot it from a live CD such as Knoppix and you wonder where your volumes are, just issue (on the LVM enabled system, of course):

```
# vgchange -a y
```

And your volumes will be appear under /dev/\$VOLUME-GROUP-NAME or under /dev/mapper

### **Copying LVM Volumes**

A typical way of copying one volume to another (non- LVM's) is to use "dd":

```
dd if=/dev/hda of=/dev/hdb
```

IF is the Source Drive, OF is the Target Drive. Be sure both volumes are UNMOUNTED.

When you deal with a logical volume in LVM, like this:

/dev/mapper/VolGroup00-LogVol00 --	146G	32G	108G	23%	/
/dev/hda1 --	99M	31M	64M	33%	/boot
none --	252M	0	252M	0%	/dev/shm

You can use exactly the same method:

```
dd if=/dev/VolGroup00/LogVol00 of=...etc
```



**or**

**dd if=/dev/mapper/VolGroup00-LogVol00 of=...etc**

The only catch comes after you have finished cloning and want to look at your handiwork. Since the LVM identifiers are identical on both drives, you will get error messages when you attempt to activate, scan for or access the logical volumes on either disk and will only be able to access one disk. If you really just want to clone the disk for use in another system, that isn't a big deal, since you will only be running one disk at a time.

\*\*\*\*\*

However, if this is a running system, you might like to investigate one of the fantastic features of LVM == snapshotting!

You need a bit of empty space in the volume group. (If there's none free at the moment you need to resize the logical volume AND the filesystem using a rescue/live cd).

Then you just do something like:

```
#lvcreate -s -L 64M -n LogVol00-snapshot /dev/VolGroup00/LogVol00
```

(as the snapshot is 'copy-on-write' the -L size needs only be as big as the changes that occur while using the snapshot ... plus a bit )

This creates a snapshot at /dev/VolGroup00/LogVol00-snapshot that looks the same size as the original.

Then you can use your dd method for imaging.

To remove after:

```
#lvremove /dev/VolGroup00/LogVol00-snapshot
```

If you are trying to make a backup, another way is to create a new volume group and logical volume(s) on the second drive using a different volume group name, and to dd:

```
dd if=/dev/VG00/LV00 of=/dev/VG01/LV00 bs=64M
```

\*\*\* As long as the underlying filesystems are the same size, you can even copy a logical volume to a traditional partition. When you select a partition or a logical volume to copy from/to, you are really copying the contents of the partition (i.e., the filesystem), unlike when you copy one entire drive to another.