

Enabling root-on-multipath for SLES 9 zSeries

By John Romanowski, September 2006

For SLES 9, Novell's [How to setup / use multipathing on SLES](#) explains how to multipath FCP scsi disks using Device Mapper and the multipath-tools package (referred to as DM MPIO). But it says "Currently, DM MPIO is not available for either the root or the boot partition, as the boot loader does not know how to handle MPIO."

Here's a method to remove those restrictions and have a multipath root and boot file system on FCP scsi disks using multipath-tools on SLES 9 SP3 zSeries. Use this for root-on-multipath-partition or root-on-LVM-on-multipath. This method supplements Novell's How-to so be sure to read theirs first.

Notice !! Use this method at your own risk. Novell doesn't support SLES 9 with DM MPIO for the root or boot partition.

First I'll show you which scripts and config files to change to enable root-on-multipath. I'll discuss boot-on-multipath, maintenance issues and then show an example of converting a linux system to have root-on-multipath and boot-on-multipath.

Change three shell scripts

Modify a copy of the `/sbin/mkinitrd` script. Mine was `mkinitrd-1.2-27.21`. My modification checks if `dm-multipath` is in your `INITRD_MODULES` list: if it is then the mods populate the initial ramdisk with some multipath-tools commands and your current `/etc/multipath.conf` file and puts code in `/linuxrc` to run `/sbin/multipath` and `/sbin/kpartx` to multipath devices in the initial ramdisk phase of booting. The modified `mkinitrd` issues the eye-catcher message "Including: `udev dm/ lvm2 ROOT_ON_MPIO`" so you'll know the `initrd` was built with the `root-on-multipath` feature enabled.

```
cd /root ; cp /sbin/mkinitrd mkinitrd_MPIO
```

Edit `mkinitrd_MPIO` and insert these two sections of code:

First, about halfway in the file find the two original lines that read

```
# DEBUG: cp_bin /bin/ls $tmp_mnt/bin  
echo -ne "Shared libs:\t"
```

and insert this code between those two lines:

```
# ----- ROOT_ON_MPIO modification: -----  
# Using module dm-multipath triggers our mods.  
# set a flag, add a feature name and copy commands and  
# config file into initrd.  
if has_module dm-multipath ; then  
  ROOT_ON_MPIO=1  
  features=({features[@]} ROOT_ON_MPIO)  
  add_module dm-round-robin  
  cp_bin $root_dir/sbin/multipath $tmp_mnt/sbin  
  cp_bin $root_dir/sbin/dmsetup $tmp_mnt/sbin          #used by vgscan  
  cp_bin $root_dir/sbin/mpath_* $tmp_mnt/sbin  
  cp_bin $root_dir/sbin/kpartx $tmp_mnt/sbin  
  cp_bin $root_dir/sbin/devmap_name $tmp_mnt/sbin    #for udev.rules  
  cp -a $root_dir/etc/multipath.conf $tmp_mnt/etc  
fi  
# -----
```

Second, avoid `if [-n "$root_lvm"]` and find the two later lines that read

```
if [ -n "$root_dm" ] ; then
```

```
# Name of the volume containing the root filesystem
```

and insert this code between those two lines. I used tab before each pipe, |, symbol being very ignorant of shell scripting but seeing this note in mkinitrd's `cat_linuxrc()` 'Note that here documents can only be indented with tabs, not with spaces'. I tried to comply.

```
# ----- ROOT_ON_MPIO modification: -----
  if [ -n "$ROOT_ON_MPIO" ] ; then
    cat_linuxrc <<-'EOF'
    |echo "ROOT_ON_MPIO -----"
    |multipath -v0
    |sleep 1
    |# Map the partitions of each multipath /dev/mapper device
    |for i in $(ls /dev/mapper); do
    |  if [ $i != "." ] ; then
    |    if [ $i != ".." ] ; then
    |      if [ $i != "control" ] ; then
    |        echo "... $i "
    |        kpartx -a -v /dev/mapper/$i
    |      fi
    |    fi
    |  fi
    |done
    |# Path events from now to when boot.multipathd runs are lost.
    |echo "done - if root is a partition ignore -unable to find vg mapper- "
    |EOF
  fi
# Note: original code assumed root_dm meant only LVM2
# -----
```

During booting, the original `/linuxrc` code then runs LVM2's `vgscan` and `vgchange`. Harmless for root-on-multipath-partition, needed for root-on-LVM-on-multipath.

Save the file. You'll use `/root/mkinitrd_MPIO` instead of `/sbin/mkinitrd`.

```
*****
```

Make a backup copy and modify the `/etc/init.d/boot.device-mapper` script. Change the start of the script to add my comment after `#!/bin/sh`:

```
#!/bin/sh
```

```
# -----Modified for ROOT_ON_MPIO
```

and after the line that reads:

```
/sbin/devmap_mkno d.sh
```

insert these five lines:

```
#ROOT_ON_MPIO: Clean up and/or re-populate the /dev/mapper dir.
```

```
# That's essential if using multipath root device since the
```

```
# initrd's /dev/mapper directory from ramdisk is gone and
```

```
# neither hotplug nor running /sbin/multipath recreate's it on SLES9.
```

/sbin/dmsetup mknodes

It's safe to use this script even if you're not doing root-on-multipath.

Make a backup copy and modify the **/etc/init.d/boot.multipath** script. Change the start of the script to add my four bold lines after "#! /bin/sh":

```
#!/bin/sh
# -----Modified for ROOT_ON_MPIO
returnRC() {
  return $1
}
```

and find the line in the start) case that reads:

```
$PROGRAM -v 0
```

and insert this code after that line:

```
savedRC=$?
```

```
# ROOT_ON_MPIO: Map the partitions of each /dev/mapper multipath device.
# No harm kpartx-ing a partition name.
# Running script twice will kpartx an LVM vol (if any): not harmful.
for i in $(ls /dev/mapper); do
  if [ $i != "." ]; then
    if [ $i != ".." ]; then
      if [ $i != "control" ]; then
        echo "... kpartx of /dev/mapper/$i"
        kpartx -av /dev/mapper/$i
      fi
    fi
  fi
done
echo
# trying to preserve PROGRAM's exit code for rc_status
returnRC $savedRC
```

Save the file. It's safe to use this script for multipath without doing root-on-multipath.

The original script clears the `/dev/disk/by-name/*` directory and says it'll re-populate the directory as a side-effect of running the multipath command. My experience is the directory remains empty if you do root-on-multipath which runs multipath in the initrd phase of booting before this script runs. The hotplug events from the initrd phase don't recur when this script runs and recreate the directory? I can live with an empty `/dev/disk/by-name/` directory; there's probably a way to recreate it but I lack the time to pursue it.

Change configuration files

```
/etc/sysconfig/kernel
```

Put `dm-multipath` in your `INITRD_MODULES` list. Having `dm-multipath` in the list

triggers my ROOT_ON_MPIO feature in the mkinitrd_MPIO script. Also follow Novell's MPIO How-to and add any driver modules needed for your storage system. Example:

```
INITRD_MODULES="dasd_eckd_mod dasd_fba_mod dm-multipath"
```

/etc/zipl.conf

Change your kernel parameter root= to point at your root-on-multipath.

For root-on-multipath-partition, use /dev/mapper/<name>p<number> Don't use /dev/disk/by-name/, it might be empty.

Example: root's on lun002 (an alias I assigned in multipath.conf) partition 1:

```
parameters = "root=/dev/mapper/lun002p1 selinux=0 TERM=dumb elevator=cfq"
```

For root-on-LVM-on-multipath use the normal LVM path /dev/vgname/lvname

Example: root's in LVM volume group sys , volume name root:

```
parameters = "root=/dev/sys/root selinux=0 TERM=dumb elevator=cfq"
```

Optionally add **linuxrc=trace** to those parameters to see what's happening during the /linuxrc phase of booting. linuxrc=trace does a set -x in /linuxrc.

/etc/fstab

Name your multipath root same as in the above zipl.conf instructions. Plus change to /dev/mapper/<name>p<number> the names of any multipath partitions you're mounting.

Example with multipath root and boot LUNs and LVM-on-multipath:

/dev/mapper/lun002p1	/	ext3	acl,user_xattr	1 1
/dev/mapper/lun000p1	/boot	ext3	acl,user_xattr	1 2
/dev/sys/home	/home	ext3	acl,user_xattr	1 2
/dev/sys/opt	/opt	ext3	acl,user_xattr	1 2
/dev/sys/usr	/usr	ext3	acl,user_xattr	1 2
/dev/sys/var	/var	ext3	acl,user_xattr	1 2
/dev/sys/swap1	swap	swap	pri=42	0 0
devpts	/dev/pts	devpts	mode=0620,gid=5	0 0
proc	/proc	proc	defaults	0 0
sysfs	/sys	sysfs	noauto	0 0

/etc/lvm/lvm.conf

For root-on-multipath, if you'll have any LVM volumes on scsi change LVM's filter statement to accept the /dev/mapper directory. Needed because /dev/disk/by-name/ might be empty in the initrd phase of booting and will be empty post-initrd. Example (from my system, yours could be different):

```
filter = [ "a|/dev/dasd.*|", "a|/dev/mapper.*|", "a|/dev/disk/by-name.*|", "r|.*|" ]
```

The Novell MPIO How-to tells you to add /dev/disk/by-name/.* plus make other important lvm.conf changes; make those changes too.

Using a multipath boot partition

The boot partition can be multipath if you can un-multipath its LUN whenever you need to update the LUN's scsi boot loader with zipl. zipl quits with "Could not get disk geometry" when /boot is a multipath device. You can't un-multipath /boot's LUN if it has other partitions that are "busy". You can unmount and un-multipath /boot's LUN if

/boot's isolated to a LUN of its own. Then you're able to unmount /boot, un-multipath its LUN, mount one of its path's /dev/sd?1 at /boot, run zipl, unmount /boot, multipath its LUN and mount the multipath partition at /boot.

(Or avoid the hassle and move your boot partition to a small dasd minidisk. Put the rest of your system, including the root partition, on multipath scsi.)

Example commands for a single-partition LUN, /dev/mapper/lunBp1 mounted at /boot:

```
multipath -l lunB          Display its scsi device names; one is sdb (perhaps).  
umount /boot; dmsetup remove lunBp1; dmsetup remove lunB  
mount /dev/sdb1 /boot      Use an sd? name from multipath -l's response  
zipl -V
```

<some messages not shown here>

Preparing boot device: sdb.

Detected SCSI PCBIOS disk layout.

Writing SCSI master boot record.

```
umount /boot
```

```
multipath -v1 lunB; kpartx -av /dev/mapper/lunB
```

```
mount /dev/mapper/lunBp1 /boot
```

Maintenance Issues

Installing a new Service Pack or applying fixes could replace your modified scripts and make your modified system unbootable. Likewise if you replace /sbin/mkinitrd with mkinitrd_MPIO. While applying patches Yast Online Update might pop up a window saying "Run zipl": be prudent and run mkinitrd_MPIO and then run zipl.

The initial ramdisk contains a copy of your then-current multipath.conf and lvm.conf. If you change those files rerun mkinitrd_MPIO and zipl to update the boot loader.

Example - Convert a dasd + multipath scsi system to all-scsi w/root-on-multipath-partition

An existing SLES 9 64-bit system has its root (/dev/dasda1) and boot (/dev/dasda2) partitions on dasd minidisk 200. The dasd root (/) file system mounts /usr /var /opt and /home from LVM volumes on multipath scsi lunA (lunA is an alias name set in multipath.conf). lunA has an empty partition #1, lunAp1, that's about the same size as the dasd root partition.

Here's how to make the system all multipath-scsi with root-on-multipath-partition and boot-on-multipath partition. The converted system won't use dasd 200. For fallback if the conversion fails you could still boot the original system from dasd 200 and try again.

First acquire a tiny second LUN of 50MB to hold the new boot partition; get this LUN online to the current system, as plain /dev/sd?, not multipath. Use **fdisk** to make its partition #1 be the whole disk; put a ext3 file system on it; then mount the LUN and copy /boot to it:

```
mount /dev/sdx1 /mnt  
cp -dr --preserve=all /boot /mnt  
umount /mnt
```

Next move the dasd root partition to lunA's empty partition #1: put an ext3 file system on it and copy dasd root partition to it:

```

mkfs -t ext3 /dev/mapper/lunAp1
mount /dev/mapper/lunAp1 /mnt
cp -dr --preserve=all /bin /dev /etc /lib /lib64 /root /sbin /srv /mnt
cd /mnt; mkdir home mnt opt tmp usr var proc sys
chmod 1777 tmp
Done moving root partition.

```

```

cd /mnt/etc/init.d

```

In /mnt/etc/init.d modify the boot.device-mapper and boot.multipath scripts as described above.

```

cd /mnt/root

```

In /mnt/root create the mkinitrd_MPIO script described above.

Prepare the new system for booting by changing config files in the new scsi root:

Change **/mnt/etc/multipath.conf** to define the new 50MB boot lun as lunB:

```

scsi_id -g -u -s /block/sdx           To display that LUN's wwid

```

Define the LUN's wwid and lunB alias in a **/mnt/etc/multipath.conf** multipath { } section:

```

multipaths {
    multipath {
        wwid      360050768016180762800000000000288
        alias     lunA
    }
    multipath {
        wwid      36005076801618076280000000000020d
        alias     lunB
    }
}

```

Change **/mnt/etc/sysconfig/kernel** to put dm-multipath in the INITRD_MODULES list.

Change **/mnt/etc/zipl.conf** to say root=/dev/mapper/lunAp1

Change **/mnt/etc/fstab** to use the multipath replacements for the dasd root and boot file systems. The LVM mounts don't change. The changed file's first two lines will now say:

```

/dev/mapper/lunAp1 /          ext3    acl,user_xattr  1 1
/dev/mapper/lunBp1 /boot      ext3    acl,user_xattr  1 2
instead of
/dev/dasda1      /          ext3    acl,user_xattr  1 1
/dev/dasda2      /boot      ext3    acl,user_xattr  1 2

```

Our system uses scsi LVM so change **/mnt/etc/lvm/lvm.conf** to add /dev/mapper to the filter statement:

```

filter = ["a|/dev/mapper.*|", "a|/dev/disk/by-name.*|", "r|..*|"]

```

Now run mkinitrd_MPIO and zipl to make the new boot LUN bootable:

```
cd /mnt; mount /dev/sdx1 boot
mount -t proc proc proc
mount -t sysfs sysfs sys
mount -o ro --bind /usr usr      Need some stuff for mkinitrd & zipl
mount -o ro --bind /var var      Needs this too.
cd / ; chroot /mnt
```

```
/root/mkinitrd_MPIO             I've omitted some messages here:
Root device: /dev/mapper/lunAp1 (mounted on / as ext3)
Module list: jbd ext3 dasd_fba_mod dm-multipath dm-round-robin dm-mod dm-snapshot
dasd_eckd_mod sd_mod zfcplib
Kernel image: /boot/image-2.6.5-7.252-s390x
Initrd image: /boot/initrd-2.6.5-7.252-s390x
               <omitted>
DASDs:        0.0.0200(ECKD)
zfcplib HBAs: 0.0.0300 0.0.0301
zfcplib disks:
               0.0.0300:0x5005076801300ef7:0x0002000000000000
               0.0.0300:0x5005076801300ef7:0x0007000000000000
               0.0.0301:0x5005076801200ea5:0x0002000000000000
               0.0.0301:0x5005076801200ea5:0x0007000000000000
Including:      udev dm/lvm2 ROOT_ON_MPIO
initrd updated, zipl needs to update the IPL record before IPL!
```

```
zipl -V                        I've omitted some messages here:
```

```
Using config file '/etc/zipl.conf'
Target device information
Device.....: 08:30
Partition.....: 08:31
Device name.....: sdx
Type.....: disk partition
Disk layout.....: SCSI disk layout
               <omitted>
kernel parmline...: 'root=/dev/mapper/lunAp1 selinux=0 TERM=dumb elevator=cfq' at 0x1000
initial ramdisk...: /boot/initrd at 0x800000
Preparing boot device: sdd.
Detected SCSI PCBIOS disk layout.
Writing SCSI master boot record.
Syncing disks...
Done.
```

```
exit                            Exit the chroot environment
```

```
cd /mnt; umount boot proc sys usr var
```

```
cd /; umount /mnt
```

```
shutdown -h now                  Shutdown and let's boot off scsi
```

From the linux's console detach dasd 200 so it's not online when we run mkinitrd_MPIO to make an initrd that doesn't try to activate dasd 200;

```
#cp detach 200
```

We can boot off any path to our boot LUN; the boot LUN's paths are
0.0.0300:0x5005076801300ef7:0x0007000000000000
0.0.0301:0x5005076801200ea5:0x0007000000000000

```
#cp set loaddev clear port 50050768 01300ef7 lun 00070000 00000000
```

```
#cp ipl 300                      I've omitted many output messages
```

```
HCPLDI2816I Acquiring the machine loader from the processor controller.
HCPLDI2817I Load completed from the processor controller.
HCPLDI2817I Now starting the machine loader.
MLOEVL012I: Machine loader up and running (version 0.18).
MLOPDM003I: Machine loader finished, moving data to final storage location.
```

Linux version 2.6.5-7.252-s390x (geeko@buildhost) (gcc version 3.3.3 (SuSE Linux)) #1 SMP Tue Feb 14 11:11:04 UTC 2006

We are running under VM (64 bit mode)

<omitted>

Activating DASDs:

0.0.0200:0

Configuring device 0.0.0200

<Since we detached dasd 200 we get "no device 0.0.0200">

No device 0.0.0200

<omitted>

Waiting for /dev/mapper/control to appear: ok

ROOT_ON_MPIO -----

<omitted>

... lunA

add map lunAp1 : 0 1512862 linear /dev/mapper/lunA 62

add map lunAp2 : 0 102920 linear /dev/mapper/lunA 1512924

add map lunAp3 : 0 8861412 linear /dev/mapper/lunA 1615844

... lunB

add map lunBp1 : 0 32067 linear /dev/mapper/lunB 63

done - if root is a partition ignore -unable to find vg mapper-

Reading all physical volumes. This may take a while...

Found volume group "sys" using metadata type lvm2

Unable to find volume group "mapper"

< /linuxrc exits to the kernel and you see this next message >

kjournald starting. Commit interval 5 seconds

EXT3-fs: mounted filesystem with ordered data mode.

VFS: Mounted root (ext3 filesystem) readonly.

Trying to move old root to /initrd ... /initrd does not exist. Ignored.

Unmounting old root

Trying to free ramdisk memory ... okay

Freeing unused kernel memory: 116k freed

INIT: version 2.85 booting

System Boot Control: Running /etc/init.d/boot

<Other scripts and your modified boot.device-mapper and boot.multipath run>

System initializes; you ssh to it and make an initrd that doesn't reference the unneeded dasd 200; then reboot:

/root/mkinitrd_MPIO

Need some extra steps to run zipl against one of multipath boot LUN's /dev/sd?'s

multipath -l lunB Display its scsi device names; one is sdb (perhaps).

umount /boot; dmsetup remove lunBp1; dmsetup remove lunB

mount /dev/sdb1 /boot Use an sd? name from multipath -l's response

zipl -V

umount /boot

multipath -v1 lunB; kpartx -av /dev/mapper/lunB

mount /dev/mapper/lunBp1 /boot

reboot

Done. The system's now all FCP scsi with root-on-multipath and boot-on-multipath.