# Installing
# Linux
# With
# RDM Custom Task

## A Whitepaper

## July 13, 2006

**NOTE:**

Before using this information and the product it supports, read the general information in “Notices,” on page **36**.

# Contents

# Figures

# Revision History

| Date | Summary |
|---|---|
| 11/01/2004 | Initial Version |
| 3/13/2005 | Fixed some errors in example scripts. |
| 4/14/2005 | Added changes for RH EM64T/AMD64 support, updated example scripts. |

# Preface

This Whitepaper explains how to use IBM® Remote Deployment Manager (RDM) 4.30 (or newer) to deploy Linux using the Custom task. As examples, this paper describes how to deploy Red Hat Enterprise Linux (RHEL) AS 4 and SuSE Linux Enterprise Server (SLES) 9 & 10. The following steps are required to deploy Linux via the RDM Custom task:

- Set up a network server
- Create a Custom task in RDM
- Copy the distribution files to the network servers
- Add hardware support as required
- Create an automated install file
- Run the RDM Custom task

## *Documentation Conventions*

The '\' character denotes that the subsequent text was too long to fit on a single line in this document, and should be combined with the text on the preceding line. For example:

```
this is a really long \
line of text
```

should be reproduced in the resulting file or document as:

```
this is a really long line of text
```

Be sure to put a space between the segments when concatenating lines.

## *Who should read this whitepaper*

This whitepaper is intended to help skilled RDM administrators create deployment procedures and understand the concepts involved.  To effectively use this whitepaper, the reader should have an extensive knowledge of network environments (topology etc.), RDM environment, and the Linux installation requirements.

Portions of this document can be quite complex and require significant expertise to implement.

## *Further Reading*

In addition to this paper, there are various other sources of information regarding RDM and RDM Custom tasks.  Please refer to vendor documentation if problems are encountered with a particular distribution.

### Guides

The following product documentation is available for RDM:

- *Remote Deployment Manager 4.30 Getting Started* – Step-by-step examples of using several tasks
- *Remote Deployment Manager 4.30 Operations Guide*  – The main reference manual for RDM
- *Remote Deployment Manager 4.30 Installation Guide* – Describes the complete installation process of RDM

### Online help

In general, every window has online help available (except for some message windows or other windows where no help is applicable), either using a **Help** menu or a **Help** button.

### Links

The following links are available for further information:

- Support is available for supported systems (IBM and non-IBM) through e-mail or fee-based telephone support. Telephone support is not available in all countries. For more information about the fee-based telephone support, go to http://www.ibm.com/support or http://service.software.ibm.com/supportline.html. For more information about e-mail support, refer to the RDM home page.

Important:

This whitepaper requires RDM 4.30 (or newer).

### File differences between Windows and Linux

When creating files from this document keep these tips in mind:

1. Always use plain-text word processors.
2. Linux and Windows have different formats for new-lines in text files. Make sure that the resulting file has the Linux formatted new-lines or the scripts won't run. The Linux program "dos2unix" may be used to convert files if needed.
3. Some word processors like to replace hyphens and quotes ("-") to a more pleasing looking ("–") form. These forms are **not** the same in script files. Check to make sure that none of these are in the script files.

# Step 1: Create the Custom task

The Custom task template is a general-purpose template that is used to create custom processes to run on a target system.  Like most RDM tasks, it contains the Command List file, or list of commands that are to be executed as part of the task.

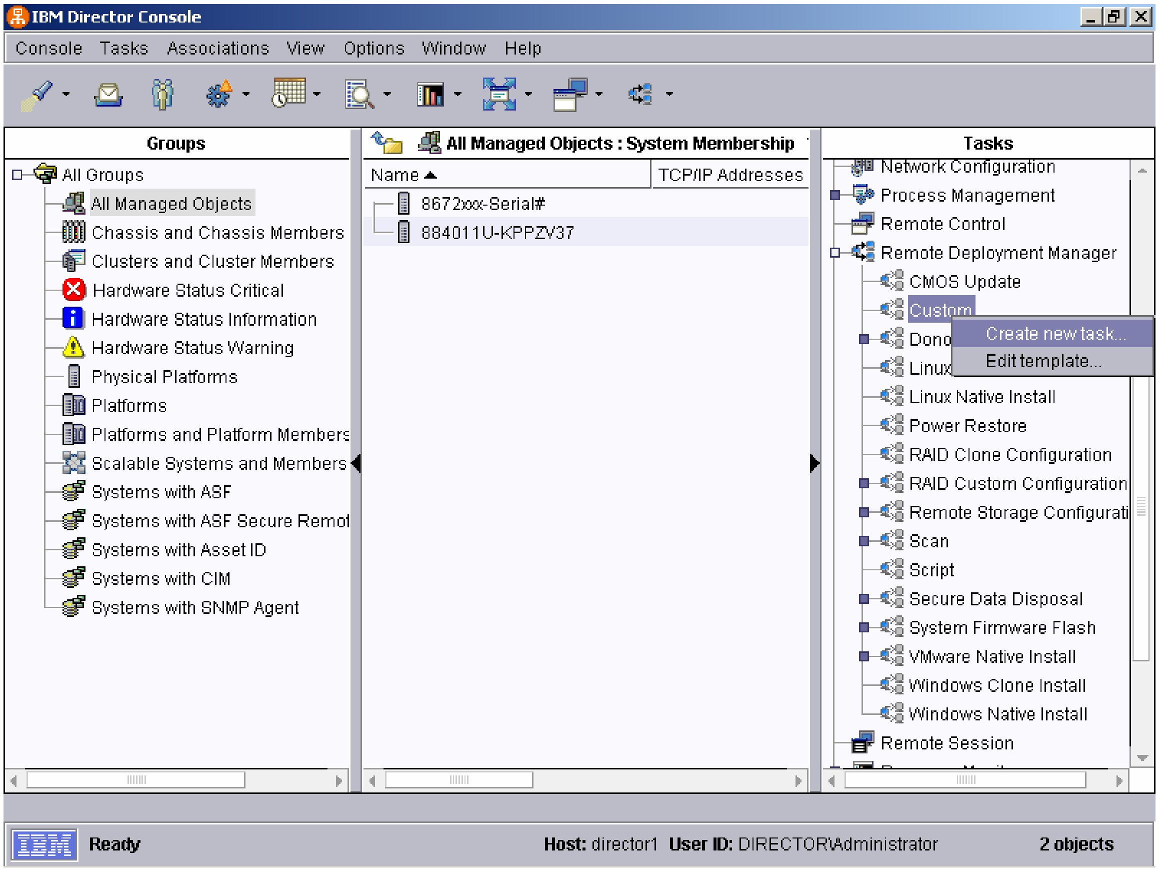


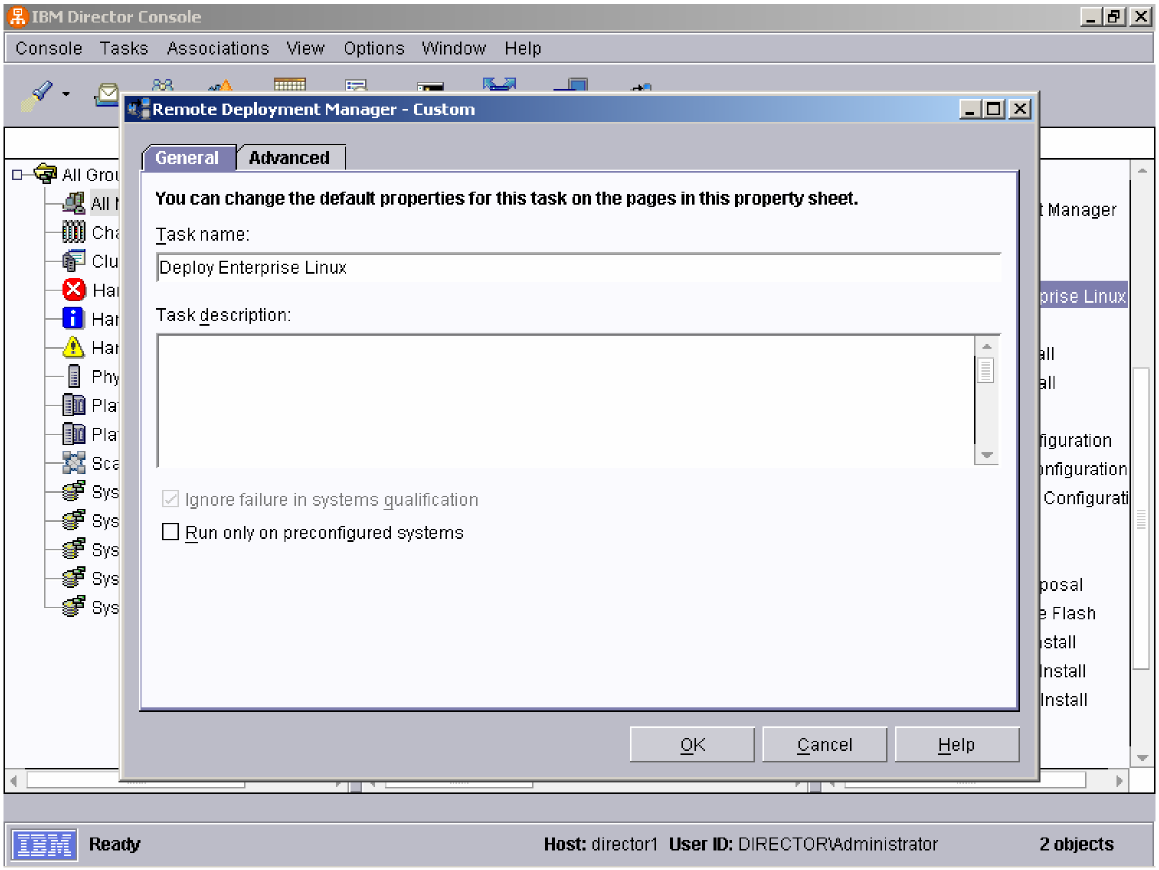
*Figure 1: Custom Task Creation*

a. Expand **Remote Deployment Manager** in the Tasks pane of the Director Console, and right-click **Custom**.  The menu is displayed.

b. Click **Create new task…** on the menu to access the property pages.  There are two pages:

   **1. General**: Contains basic information about the task.

   **2. Advanced**:  Contains the command list, user parameters, and task folder name for the task.


IBM Director Console
Console Tasks Associations View Options Window Help
Remote Deployment Manager - Custom
General
Advanced
OK
Cancel
Help

You can change the default properties for this task on the pages in this property sheet.

Task name: Deploy Enterprise Linux

Task description:

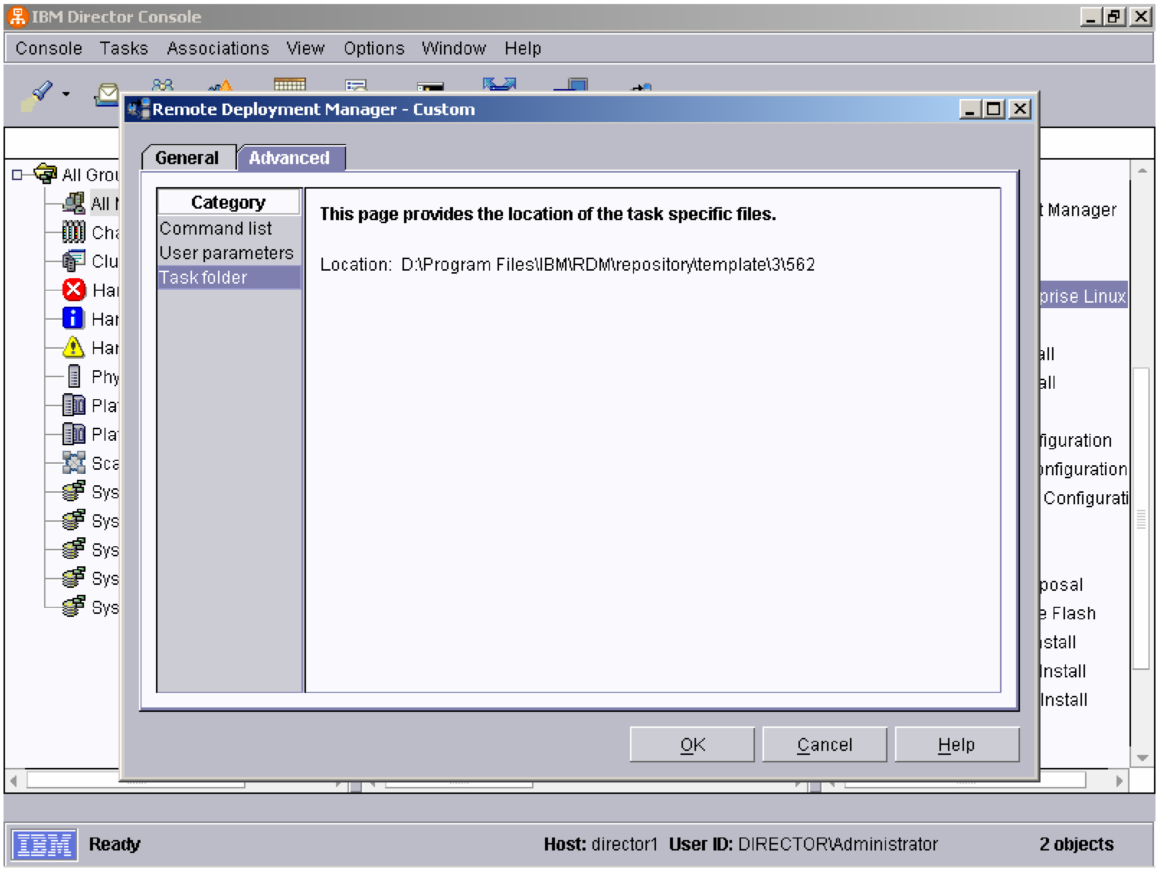
☑ Ignore failure in systems qualification

□ Run only on preconfigured systems

Ready Host: director1 User ID: DIRECTOR\Administrator 2 objects

*Figure 2: Task Configuration*

c. Enter the task information on the General page (e.g., name the task *Deploy Enterprise Linux*).



This page provides the location of the task specific files.

Location: D:\Program Files\IBM\RDM\repository\template\3\562

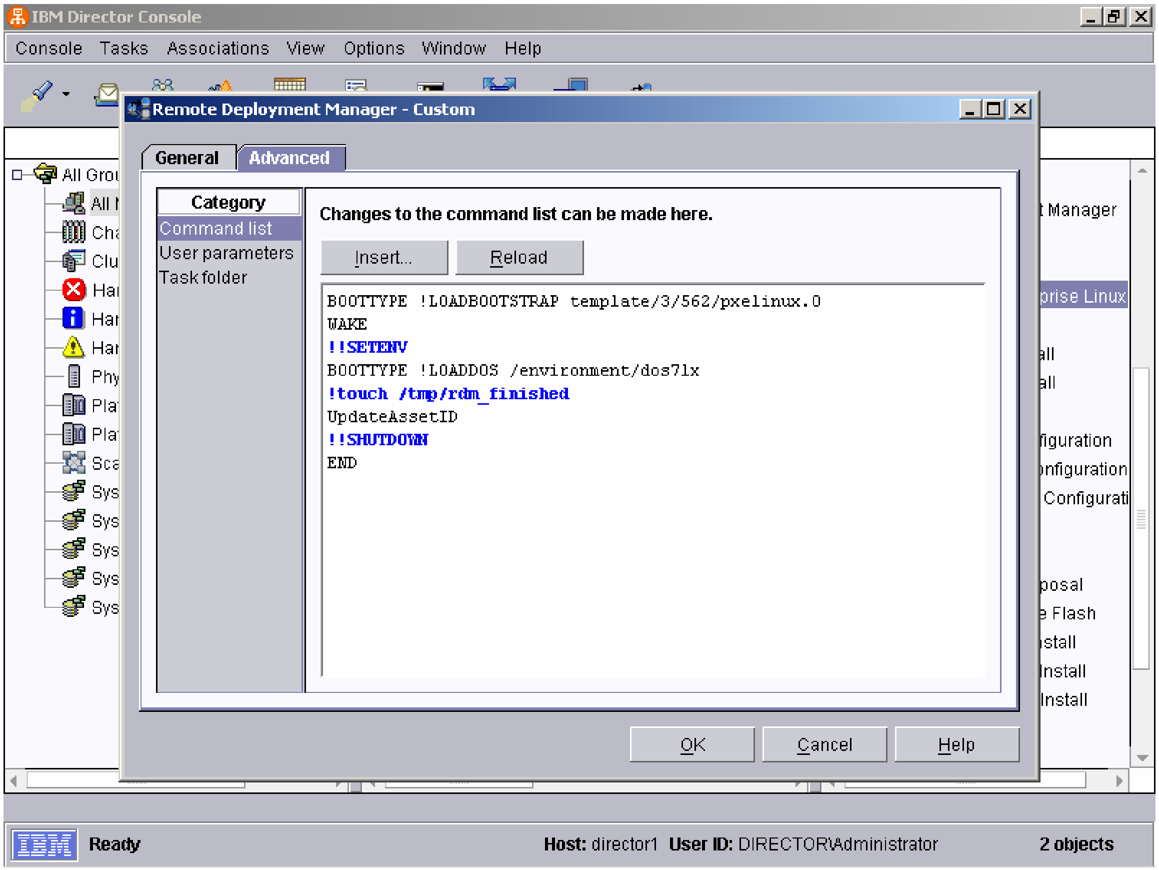
Ready Host: director1 User ID: DIRECTOR\Administrator 2 objects

*Figure 3: Task Folder*

d. Click **"OK"** to save close the task.

e. Right click on the newly created task and select **"Edit Task."**

f. Click on the "**Advanced"** tab, click on the category "**Task Folder**." This will reveal the location of the files associated with this task, and required in later steps.


IBM Director Console
Console Tasks Associations View Options Window Help
Remote Deployment Manager - Custom
General
Advanced
Category
Command list
User parameters
Task folder
Insert...
Reload
OK
Cancel
Help
Ready
Host: director1 User ID: DIRECTOR\Administrator
2 objects

Changes to the command list can be made here.

```
BOOTTYPE !LOADBOOTSTRAP template/3/562/pxelinux.0
WAKE
!!SETENV
BOOTTYPE !LOADDOS /environment/dos71x
!touch /tmp/rdm_finished
UpdateAssetID
!!SHUTDOWN
END
```

***Figure 4: Editing Custom Task Command List***

g. Stay on the "**Advanced"** tab, click on the "**Command List"** and modify the command list this way:

```
BOOTTYPE !LOADBOOTSTRAP template/3/562/pxelinux.0
WAKE
!!SETENV
BOOTTYPE !LOADDOS /environment/dos71x
!touch /tmp/rdm_finished
UpdateAssetID
!!SHUTDOWN
END
```

**NOTE:** The command list is a list of RDM commands that perform the Custom task. Refer to Appendix A, "Command list commands", in the *Remote Deployment Manager 4.30 Operations Guide* for information on commands and syntax within a command list.

Change '3/562' to match the same portion of the Task Folder's location, found in step 1f.

## Step 2: Configure the NFS Server

Configure the NFS Server according to the Linux distribution’s documentation.

For RHEL, please refer to the Red Hat Installation Guide section entitled “Preparing for a Network Installation.”

For SLES, please refer to the AutoYast documentation at: http://forge.novell.com/modules/xfmod/docman/?group_id=1532.

These steps contain instructions on installing and configuring the NFS server, as well as copying the distribution’s media onto the NFS server to be shared with the clients as they are being deployed.

SuSE users should also be aware that the file “media.1/info.txt” contains SuSE’s End User License Agreement (EULA). The presence of this file will cause YaST, the SuSE installation program, to display the EULA and wait until someone at the client accepts it. In order to ensure a fully unattended install, this file should be viewed and accepted outside of this installation procedure and the filename should be changed.

## Step 3: Obtain and Configure the bootloader

a. Download the syslinux package from http://syslinux.zytor.com; this is in the “pxelinux” RPM package under Red Hat.

b. If the ‘pxelinux.0’ binary is present in the archive obtained in step 3a, then proceed to step 3f.

c. On a Linux computer with the “make” and “nasm” packages, extract the source code for the “pxelinux” boot loader, which is part of the syslinux package.

   **NOTE:** Prior to RHEL AS Update 5, the Red Hat installer (anaconda) would prompt the user for network information, effectively interrupting an unattended installation, if it encountered a string beginning with 'ip=' on the kernel command line.

   Please see Red Hat BUG ID 134054: ("anaconda ip= and pxelinux IPAPPEND don't coexist peacefully") regarding this issue at:

   https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=134054.

   To avoid this problem, edit the file **“**pxelinux.asm**”**. A patch has been provided for syslinux version 3.11 in Appendix 6). At or near line 2277, change:

```
     mov eax, ‘ip=’
```

to:

```
     mov eax, ‘ip!’
```

d. Run ‘`make clean`’ to delete the products of previous builds.

e. Run ‘`make pxelinux.0`’ to create the binary image.

f. Copy the ‘pxelinux.0’ file to the RDM Task Folder described in step 1f.

g. In the task’s directory, create a sub-directory called ‘pxelinux.cfg’. Inside this directory, create a file named “default” containing the following text for Red Hat:

```
DEFAULT vmlinuz
APPEND initrd=initrd.img root=/dev/ram0 \
ks=nfs:10.1.6.230:/share/redhat/ks.cfg \
ksdevice=link
IPAPPEND 1
```

**NOTE:** The “`ksdevice=link`” directive instructs anaconda to use the network device that has an active network link. If the network device doesn’t support link detection, or if more than one network device is connected during installation, “link” will need to be replaced with the name of the network device to use during the installation, for example “`eth0`”.

Change “10.1.6.230” to reflect the IP address of the NFS server. Likewise, change the “share/redhat” to reflect the directory created in step 2.

For SLES, the file should contain this:

```
DEFAULT linux
LABEL linux
  kernel linux
  append load_ramdisk=1 initrd=initrd \
autoyast=nfs://10.1.6.230/share/suse/file.xml \
install=nfs://10.1.6.230/share/suse/
IPAPPEND 1
```

Be sure to change the server name and path to “file.xml” and the installation source to reflect the network topology and server layout. This document uses “/share/suse” as the directory in which the SuSE source media was installed.

h. For Red Hat, copy the files “images/pxeboot/vmlinux” and “images/pxeboot/initrd.img” off the first CD to the task’s directory. For SuSE, the files are “/boot/loader/linux” and “/boot/loader/initrd”.

   **NOTE:** Please read the Red Hat documentation and assess hardware requirements prior to kernel selection. The pxeboot images mentioned above were utilized during the creation of this paper.

## Step 4: Add Hardware Support (as required)

When deploying to a computer containing a storage or network adapter that is not supported by the Linux distribution, the correct drivers will have to be obtained and added to the installation source files prior to installation. This process has been described in Appendix 1 for Red Hat distributions and Appendix 2 for SuSE distributions.

### *Current Unsupported Devices*

At the time of this writing, the following devices were not supported by the Linux kernel distributed with RHEL AS 3 Update 4. Additional Devices TBD.

- IBM eServer BladeCenter HS20 (Type 8832) - LSI MegaRAID IDE Adapter, requires driver 5.06A.

- IBM eServer BladeCenter HS20 (Type 8840) - LSI MegaRAID IDE Adapter, requires driver 5.06J.

At the time of this writing, the following devices were not supported by the Linux kernel distributed with SLES 8 SP3. Additional Devices TBD.

- IBM eServer HS40 Gigabit Ethernet Adapter, requires driver (e1000) version 5.2.22.1

- IBM eServer BladeCenter HS20 (Type 8832) - LSI MegaRAID IDE Adapter, requires driver 5.06A.

## Step 5: Create RDAgent Script

a. Copy the ‘rdagent’ file (typically located in the “environment\etc” directory in the RDM repository) to the directory containing the Linux installation files.

b. Create the following file, named “postscript” in the “/share/redhat” or “/share/suse” directory on the NFS file server:

   **NOTE:** If you are installing RHEL AS 3 prior to Update 5, you <u>must</u> use the script in Appendix 5 – the following script <u>will not</u> function properly. Please refer to the comment contained in the script below (above the IP_LINE assignment for more information).

```
#!/bin/sh

echo Running RDM post-install script

########################################
# Setup variables
########################################

export DEV=eth0

if [ ."$RDM_CHROOT" = ."" ]; then
   DEST_DIR=/
else
   DEST_DIR=/mnt/sysimage
fi

if [ ."$SOURCE_DIR" = ."" ]; then
   SOURCE_DIR=/mnt/source
fi

########################################
# Parse kernel command line for network
# information
########################################


for CMD in $(cat /proc/cmdline); do
   if echo $CMD | grep ^ip 2>&1 > /dev/null; then
      # If you are using RHEL AS 3 prior to Update
      # 5, then you must comment out the following
      # line and uncomment the one below it
      IP_LINE=$(echo $CMD | cut -f 2 -d = )
      #IP_LINE=$(echo $CMD | cut -f 2 -d \! )

      CLIENT=$(echo $IP_LINE  | cut -f  1 -d :)
      SERVER=$(echo $IP_LINE  | cut -f  2 -d :)
      GATEWAY=$(echo $IP_LINE | cut -f  3 -d :)
      NETMASK=$(echo $IP_LINE | cut -f  4 -d :)
   fi
done
```

```
########################################
# Set environment variables for rdagent
########################################

TARGET=$(ifconfig  $DEV | grep HWaddr  | sed 's/^.*HWaddr //' | \
sed 's/://g')

export server_ip client_ip uuid bootmac ip
export server_ip=$SERVER
export client_ip=$CLIENT
export ip=${CLIENT}:${SERVER}:${GATEWAY}:${NETMASK}

bootmac=$(echo $TARGET | sed -e 's/://g')

export RDRASLEVEL=0
export RDSTATUS=RDAGEN000I
export RDRETPATH=/
export RDRETFILE=none

${SOURCE_DIR}/rdagent /e

if ( test -f ./_rdm.bat) ; then
   . ./_rdm.bat
fi

########################################
# Process RDM commands
########################################

while [ ! -f /tmp/rdm_finished ] ; do
   ${SOURCE_DIR}/rdagent
   if ( test -f ./_rdm.bat) ; then
      . ./_rdm.bat
   fi
   rm -rf ./_rdm.bat
done
```

This script calls RDAgent, which checks in with the RDM server to update the status and receive work items from the command list.  This is required so that RDM can know when the installation is complete.

c. Run “`chmod +rx postscript rdagent`” in the redhat directory on the NFS server to mark this script executable.

## Step 6: Create the Automated Install File

Most modern distributions of Linux provide a method to perform unattended (or scripted) installations. This process is typically driven by a configuration file containing installation parameters; for Red Hat, this is called a “kickstart” file and for SLES this is called an ‘autoyast’ file. This file may be created by hand or via the use of a program (‘redhat-config-kickstart’ for RHEL and ‘autoyast’ for SLES).

**NOTE:** This file must conform to the format set forth by the Linux vendor, otherwise installation problems will occur.

**NOTE:** When installing applications that are not included with the SuSE or Red Hat distribution, be sure to include all dependencies in this file. For example, Director Agent version 4.2 requires that the package “compat-glibc++” be installed; to tag this package for installation, add the appropriate package name to the “%packages” section or add a package group that includes it.

For installing with a Red Hat kickstart file, make sure you specify the installation source, as with this line:

```
nfs --server 10.1.6.230 --dir /share/redhat
```

In this example, the Linux server’s address is 10.1.6.230.

Also, you will need a section to call the RDAgent, such as this:

```
%post --nochroot
/mnt/source/postscript
```

During the Red Hat installation, the source media – in this case the NFS server – is mounted on ‘/mnt/source’. Likewise, the harddrive that is being installed to is on ‘/mnt/sysimage’.

Also, the “`--nochroot`” option exposes the entire system, including the installer to the script, instead of limiting the visible directories to those of the installed system.

If you try to use a Non-NFS based network install, such as FTP or HTTP, you will need to download the postscript install file to the local computer and execute it from there.

**NOTE**: The RHEL kickstart file does not support a mechanism for configuring more then one Network Interface Card (NIC) during an unattended installation. For computers with multiple NICs, the additional

NICs will need to be configured via a post install script.  Here is an example of how a second Ethernet card (“eth1”) could be configured:

```
FILE=/mnt/sysimage/etc/sysconfig/network-scripts/ifcfg-eth1
echo <<EOF >  $FILE
BOOTPROTO=static
IPADDR=9.44.55.66
NETMASK=255.255.240.0
NETWORK=9.44.48.0
ONBOOT=yes
EOF
```

**NOTE:** This example is for target machines that will be using a static IP setups.  If the NIC is being configured via DHCP, then set “`BOOTPROTO=dhcp`”, and do not specify the IPADDR, NETMASK, and NETWORK keywords. When done modifying the kickstart file, put it in the directory on the NFS server created in Step 2.

SuSE and other United Linux distributions have a manner to embed the script file into the XML based YaST control file.  The XML file needs to have a “scripts” section inside the “configure” section.  The following script entry will run RDAgent in the same way as the Red Hat example above.

```
<scripts>
 <chroot-scripts config:type="list" >
   <script>
    <filename>chroot.sh</filename>
    <interpreter>shell</interpreter>
    <source>
    <![CDATA[#!/bin/sh
for cmd in $(cat /proc/cmdline) ; do
  if echo $cmd | grep ^install ; then
    NFS_SERVER=$(echo $cmd | cut –f 3 –d /)
    NFS_DIR=$(echo $cmd | cut –f 4- -d /)
  fi
done
SOURCE_DIR=/RDM_AGENT
RDM_CHROOT=true
export SOURCE_DIR NFS_SERVER NFS_DIR RDM_CHROOT
mkdir $SOURCE_DIR
mount ${NFS_SERVER}:${NFS_DIR}
/${SOURCE_DIR}/postscript
umount /RDM_AGENT
]]>
    </source>
   </script>
  </chroot-scripts>
  </scripts>
```

### *RHEL AS EM64T/AMD64 Notes*

The 64-bit version of Red Hat does not have access to 32-bit libraries during the install, but they are available after the install as part of compatibility packages. The RDAgent program needs the 32-bit versions to work properly, therefore if you are installing to a 64-bit x86-64 machine, you need to add the following lines immediately after the %post line:

```
mkdir /lib
ln –s /mnt/sysimage/lib/ld-linux.so.2 \
/lib/ld-linux.so.2
ln –s /mnt/sysimage/lib/libc.so.6 /lib/libc.so.6
```

and add the following to the %packages section of the kickstart file:

```
@compat-arch-support
@compat-arch-development
```

## Step 7: Install IBM Director Agent

In order to install an application, such as IBM Director Agent, the installation media needs to be copied on to a directory on the NFS server. For example, create a directory “/share/IBMDirAgent”.

Also, a script should be created to set up the environment, answer any prompts, and install the application.  For IBM Director Agent, this script is called “IBMDirectorAgent4.11-1.sh.”

Finally, this script must be called from the installer.  To do this, a line must be added to the “%post” section of the kickstart file.  If the script is not written with this environment in mind, it will need to be called with chroot to install to the hard disk (otherwise, it will attempt to install to the install media).  Also, because the install media is mounted in /mnt/source, it will not be accessible unless it is mounted.

For IBM Director Agent, these lines should be added:

```
mkdir /mnt/sysimage/mnt/source
cp –Ravp /mnt/source/IBMDirAgent /mnt/sysimage/tmp
chroot /mnt/sysimage bash –c “cd /tmp/IBMDirAgent; \
./IBMDirectorAgent4.11-1.sh”
```

## Step 8: Replicate to remote D-Servers

If RDM is configured to use one or more remote D-Servers, the Custom task's directory must be replicated to the remote D-Servers.

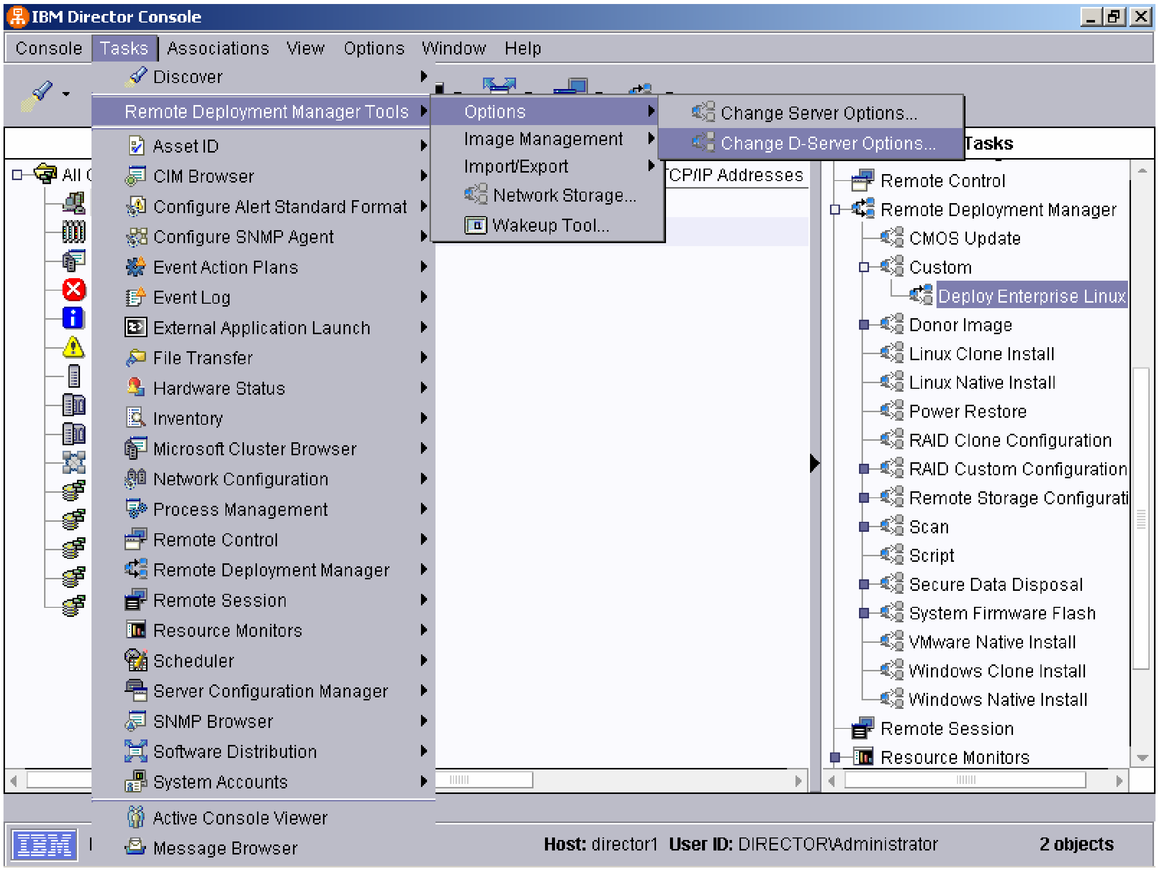


*Figure 5: Remote D-Server Replication*

If this is the first replication to any remote D-Servers, enable replication by going to the Tasks item on the menu bar, then to RDM, Options and Change D-Server options. Select each remote D-Server from the list on the left, open the Replication tab and make sure "Replication Enabled" is checked.

Now replicate the files by selecting the Tasks item on the menu bar, then RDM, Image Management, and then Replicate Files. Select Folders and select the Tasks' folder from Section 1 and click OK. Click on the D-Server button, then select the D-Servers you will be using for this deployment, and then click on OK. Finally, click on OK to exit.

## Step 9: Start a Deployment Task

Finally, you should be ready to kick off a deployment.  Starting this task should be exactly like starting any other deployment tasks.  Scan in a target computer, if you have not already done so, and drag it onto the task.  Select “Run Tasks” and the deployment should begin.

# Appendix 1: Red Hat Driver Image Augmentation

### *Red Hat Driver Disk Overview*

Red Hat driver diskette packages are typically comprised of the following six files:

- module-info or modinfo

  The module-info file contains module definitions and a version tag. Module definitions are indexed by module name and consist of a module type identifier and one or more property strings (the first of which being the module descriptive name).

  **NOTE:** RHEL AS requires that this file be named “modinfo”.

  The following is an example of the module definition entry for the 3Com EtherLink II adapter (note that the indented strings must be preceded by a tab character):

  ```
  3c503
          eth
          "3Com EtherLink II"
          io "Base I/O address"
          irq "IRQ level"
          xcvr "Transceiver (0 = BNC; 1 = AUI)"
  ```

- modules.cgz

  The modules.cgz file is a gzip-compressed CPIO archive that contains the binary object driver files arranged according to target kernel version. For example, the driver module archive distributed in the Red Hat Linux 7.3 pxeboot image has one directory, 2.4.18-3BOOT, which contains drivers that are used during installation.

- modules.dep

  The modules.dep file contains module dependency information. Each entry begins with a module name followed by a colon (‘:’) and a space-delimited list of modules on which the specified module depends. For example, version 2.05.00 of the LSI Logic MPT Fusion SCSI driver module (mptscsih.o) depends on the MPT base driver module (mptbase.o), so the following line would be placed in the modules.dep file:

```
mptscsih: mptbase
```

- pcitable

  The pcitable file contains list of modules and the associated PCI vendor, device, subvendor and subdevice identifiers for all the devices supported by the driver. For example:

```
Vendor  Device  Module  Description
---------------------------------------------------------------------

0x10b7  0x7646  "3c59x" "3Com Corporation|3cSOHO100-TX Hurricane"
0x10b7  0x9000  "3c59x" "3Com Corporation|3c900 10BaseT [Boomerang]"
0x10b7  0x9001  "3c59x" "3Com Corporation|3c900 Combo [Boomerang]"
0x10b7  0x9004  "3c59x" "3Com Corporation|3c900B-TPO [Etherlink XL TPO]"
```

- modules.pcimap

  This file serves the same function as the pcitable file mentioned above and contains many of the same fields.

- rhdd-6.1/rhdd-6.2

  The rhdd files indicate that the diskette or diskette image is a Red Hat driver diskette.

There are two sets of drivers that are used during a Red Hat installation -- the drivers required for transmission and retrieval of the driver diskette image, operating system source files and kickstart file (pre-install) and the drivers that are loaded and used by the system after installation has taken place (post-install).

The pre-install environment drivers used to perform an installation are contained in the initial ramdisk (initrd) image used to initiate the installation. RDM typically uses the initrd image located in the ‘images/pxeboot’ directory on the Red Hat installation media when performing a Linux Native Install (LNI). The file is usually named ‘initrd.img’ or ‘initrd-everything.img’ and contains a compressed ext2 file-system.

The post-install environment drivers are typically provided manually by the customer via a Red Hat driver floppy diskette or by placing the files (module-info, modules.dep, pcitable, etc.) in the root directory of a partition on the hard drive. RHEL AS installation program, anaconda, has a facility[1] that allows drivers to be located on a network-accessible volume at the time of installation, greatly enhancing unattended installations.

[1] `driverdisk` (optional)

Driver diskettes can be used during kickstart installations. You need to copy the driver diskette’s contents to the root directory of a partition on the system's hard drive. Then you need to use the `driverdisk` command to tell the installation program where to look for the driver disk.

**NOTE:** The driver diskette image (referred to as dd.img and 26k4970.img in subsequent examples) must be of the VFAT or FAT format for RHEL AS. If your diskette image is of the EXT2 format then it must be converted or copied to a new diskette.

**NOTE:** When performing an installation from the network, only the required network device drivers should be added to the Red Hat installation initial ramdisk image (initrd).

### *Augmentation Process*

In the following example, we are going to add the new ServeRAID adapter drivers to the RHEL AS distribution as part of an RDM installation. The ServeRAID adapter drivers are typically distributed with a Red Hat driver disk image by default, so the process is greatly simplified. If you have obtained a driver that is not shipped with a Red Hat driver diskette or as a diskette image, then you will have to compile the module manually for the BOOT and post-install environments and analyze the source code to determine the PCI device and vendor IDs for the devices that are supported and recognized by the device driver; the manual process is beyond the scope of this document.

I. Modify Pre-install/Installation Boot Diskette Image

1. Identify and obtain the Red Hat initial ramdisk image that will be used to perform the installation.

   For this example, we will be using the initial ramdisk image located in the ‘images/pxeboot’ directory on the Red Hat installation media. This file is named ‘initrd.img’ on the RHEL AS source media.

2. Mount the boot image via the loopback device from a Linux terminal. The initrd.img stored in the pxeboot directory is a compressed image, so it must be decompressed prior to mounting.

   ```
   # mkdir /mnt/image1

   # gunzip -c < initrd.img > initrd-new.img

   # mount -o loop initrd-new.img /mnt/image1
   ```

3. Locate the driver files and copy them to a staging area. The driver files are stored in the ‘modules’ directory in the initrd.

   ```
   # mkdir –p /tmp/bootimg/modules
   ```

Alternatively, a network location can be specified for the driver diskette:

driverdisk --source=ftp://path/to/dd.img
driverdisk --source=http://path/to/dd.img
driverdisk --source=nfs:host:/path/to/im

```
# cp /mnt/image1/modules/* /tmp/bootimg

# ls -l /tmp/bootimg

total 1760
-rw-r--r--    1 root     root         2482 Feb 23 15:49 module-info
-rw-r--r--    1 root     root      1747032 Feb 23 15:49 modules.cgz
-rw-r--r--    1 root     root         2993 Feb 23 15:49 modules.dep
-rw-r--r--    1 root     root         5488 Feb 23 15:49 modules.pcimap
-rw-r--r--    1 root     root        32553 Feb 23 15:49 pcitable
```

4. Decompress and extract the modules.cgz archive to the ‘modules’ directory created in the staging area.

```
# cd /tmp/bootimg

# ls –l

total 1764
-rw-r--r--    1 root     root         2482 Feb 23 15:49 module-info
drwxr-xr-x    2 root     root         4096 Feb 23 15:57 modules
-rw-r--r--    1 root     root      1747032 Feb 23 15:49 modules.cgz
-rw-r--r--    1 root     root         2993 Feb 23 15:49 modules.dep
-rw-r--r--    1 root     root         5488 Feb 23 15:49 modules.pcimap
-rw-r--r--    1 root     root        32553 Feb 23 15:49 pcitable

# cd modules

# gunzip -c < ../modules.cgz | cpio –id
7734 blocks

# ls –l

total 4
drwx------    3 root     root         4096 Feb 23 15:57 2.4.21-4.ELBOOT
```

5. Identify and obtain the Red Hat driver diskette images.

   Red Hat driver diskette images are usually distributed as a single binary file. These images may be written to a floppy diskette using a program like rawrite or ‘dd’ or mounted via the loopback device under Linux. In this example, we will use ServeRAID - Version 6.11 Drivers for Red Hat Linux diskette #2. The ServeRAID driver offering consists of two diskette images; the first diskette contains drivers for the 2.2 series of kernels but, as we saw in step 4, we are interested in the driver for the 2.4.21-4.EL series of kernels which is contained on diskette #2.

   The image file obtained from the IBM PC Support Website will be referred to as ‘26k4970.img’ in subsequent steps.

6. Mount the driver diskette image via the loopback device from a Linux terminal. Typically, driver diskette images do not have to be decompressed.

```
# mkdir /mnt/image2

# mount -o loop 26k4970.img /mnt/image2
```

7. Locate the driver files and copy them to a second staging area. The driver files are stored in the root directory of the driver disk image.

```
# mkdir -p /tmp/drvimg/modules

# cp /mnt/image2/* /tmp/drvimg

# ls -l /tmp/drvimg

total 1196
-rwxr-xr-x    1 root     root       169380 Feb 23 16:59 ipssend
-rwxr-xr-x    1 root     root           51 Feb 23 16:59 modinfo
drwxr-xr-x    2 root     root         4096 Feb 23 16:58 modules
-rwxr-xr-x    1 root     root      1015046 Feb 23 16:59 modules.cgz
-rwxr-xr-x    1 root     root            0 Feb 23 16:59 modules.dep
-rwxr-xr-x    1 root     root          141 Feb 23 16:59 pcitable
-rwxr-xr-x    1 root     root           17 Feb 23 16:59 rhdd-6.1
-rwxr-xr-x    1 root     root            1 Feb 23 16:59 rhdd-6.2
-rwxr-xr-x    1 root     root         5511 Feb 23 16:59 updrh.sh
```

8. Decompress and extract the modules.cgz archive to the ‘modules’ directory created in the staging area.

```
# cd /tmp/drvimg

# ls –l

total 1764
-rw-r--r--    1 root     root         2482 Feb 23 15:49 module-info
drwxr-xr-x    2 root     root         4096 Feb 23 15:57 modules
-rw-r--r--    1 root     root      1747032 Feb 23 15:49 modules.cgz
-rw-r--r--    1 root     root         2993 Feb 23 15:49 modules.dep
-rw-r--r--    1 root     root         5488 Feb 23 15:49 modules.pcimap
-rw-r--r--    1 root     root        32553 Feb 23 15:49 pcitable

# cd modules

# gunzip -c < ../modules.cgz | cpio –id
7734 blocks

# ls
2.4.18-14            2.4.20-8BOOT        2.4.7-10
2.4.18-14bigmem      2.4.20-8smp         2.4.7-10BOOT
2.4.18-14BOOT        2.4.20-9            2.4.7-10enterprise
2.4.18-14smp         2.4.20-9bigmem      2.4.7-10smp
2.4.18-27.7.x        2.4.20-9smp         2.4.9-e.24
2.4.18-27.7.xbigmem  2.4.21-4.EL         2.4.9-e.24BOOT
2.4.18-27.7.xsmp     2.4.21-4.ELBOOT     2.4.9-e.24enterprise
2.4.18-27.8.0        2.4.21-4.ELhugemem  2.4.9-e.24smp
2.4.18-27.8.0bigmem  2.4.21-4.ELsmp      2.4.9-e.24summit
2.4.18-27.8.0smp     2.4.2-2             2.4.9-e.3
2.4.18-3             2.4.2-2BOOT         2.4.9-e.3BOOT
2.4.18-3bigmem       2.4.2-2enterprise   2.4.9-e.3enterprise
2.4.18-3BOOT         2.4.2-2smp          2.4.9-e.3smp
```

```
2.4.18-3smp          2.4.3-12              2.4.9-e.3summit
2.4.20-8             2.4.3-12enterprise  2.4.20-8bigmem
2.4.20-9             2.4.3-12smp
```

9. Copy the drivers from the driver image staging area to the boot image staging area. The destination directory depends on distribution and is typically ${KERNEL_VERSION}${TARGET} where $KERNEL_VERSION is equal to 2.4.21-4.EL and $TARGET is equal to BOOT in this case. A new directory has been added in RHEL AS, however, that denotes the architecture for which the device driver modules have been compiled. In this example, the driver modules are located in a subdirectory named for the target architecture, which is i386.

```
# cd /tmp/drvimg/modules.2.4.21-4.ELBOOT

# cp ips.o ../bootimg/modules/2.4.21-4.ELBOOT/i386
```

10. Merge entries (PCI IDs, module name and driver description) in the driver image ‘pcitable’ file to the boot image ‘pcitable’ file.

```
# cd /tmp

# cat drvimg/pcitable
0x1014  0x002e  "ips"   "IBM|ServeRAID Controller"
0x1014  0x01bd  "ips"   "IBM|ServeRAID Controller"
0x9005  0x0250  "ips"   "IBM|ServeRAID Controller"

# cat bootimg/pcitable | grep \"ips\"
0x1014  0x002e  "ips"   "IBM|SCSI RAID Adapter [ServeRAID]"
0x1014  0x01bd  "ips"   "IBM|ServeRAID Controller"
0x9005  0x0250  "ips"   "Adaptec|ServeRAID Controller"
```

In this case, no changes have to be made because the boot image pcitable already contains the necessary entries for the ServeRAID (ips) driver. If there had been any differences, only lines that contained unique PCI Vendor and Device IDs would have been added to the boot image pcitable file.

11. Merge entries in the driver image ‘module-info’ file to the boot image ‘module-info’ file.

```
# cd /tmp

# grep -A 2 ips bootimg/module-info
ips
        scsi
        "IBM ServeRAID"

# cat drvimg/modinfo
Version 0
ips
  scsi_hostadapter
  "IBM ServeRAID"
```

In this case, there is already an entry for the ServeRAID driver in the boot image module-info file, obviating the need for a merge. Had there been any differences, however, then the text present in the driver image module-info file would have replaced the text in the boot image module-info file.

12. Merge entries in the driver image ‘module.deps’ file to the boot image ‘module.deps’ file.

    The ServeRAID driver doesn’t rely on any drivers that need to be explicitly defined in the modules.dep file, but the process is essentially the same – add all entries (lines) referring to the new device driver.

13. Recreate the ‘modules.cgz’ archive in the boot image.

```
# cd /tmp/bootimg/modules

# ls
2.4.21-4.ELBOOT

# find . -type f | grep –v modules.cpio | cpio -o -H crc > modules.cpio
7828 blocks

# gzip -9 -c < modules.cpio > modules.cgz

# cp modules.cgz ..

# cd ..

# ls
module-info  modules  modules.cgz  modules.dep  modules.pcimap  pcitable

# rm -rf modules
```

14. Copy the modified files to the boot image.

    We mounted the boot image at mount point ‘/mnt/image1’ in this example.

```
# cd /tmp/bootimg

# ls
module-info  modules.cgz  modules.dep  modules.pcimap  pcitable

# cp module-info modules.cgz modules.dep pcitable /mnt/image1/modules
```

15. Umount and compress the modified boot image.

```
# umount /mnt/image1

# gzip -9 -c < initrd-new.img > initrd.img
```

16. Place the image in the appropriate directory.

II. Modify or Create the Post-install Driver Diskette Image

1. Create the supplemental driver disk containing drivers that will be used in the post-install environment.

Up to this point, we have only been concerned with the drivers that will be used during installation (the BOOT environment) to access storage and network devices. Since we are only updating the ServeRAID drivers in this example, this step is easy – we simply make the driver diskette image file (26k4970.img) available to the Red Hat installer by placing it on an accessible storage device (local or remote) and specify the location in the kickstart file (see the Kickstart File Modification section below).

If we were adding the ServeRAID adapter to an existing driver diskette image, then we would have to follow a process similar to that described in Section I, but instead of adding the device driver and diskette image files to the boot image, we would add them to an already existing driver diskette image file.

### *Kickstart File Modification*

You must specify the location of the supplemental driver diskette image created in step II by placing the following keyword and parameters in the Red Hat kickstart file:

If we are installing via FTP --

```
        driverdisk --source=ftp://path/to/26k4970.img
```

If we are installing via HTTP --

```
        driverdisk --source=http://path/to/26k4970.img
```

If we are installing via NFS --

```
        driverdisk --source=nfs:host:/path/to/26k4970.img
```

When adding a network device driver (such as bcm5700 and e1000) to the Red Hat installation initial ramdisk image you must perform the following steps:

Create the following script (update_modules.sh) and place it in the redhat directory of the NFS share. Please note that, by default, it is assumed that the driver diskette image you created in section I has the filename ‘dd.img’ and is located in the root directory of the NFS share.

```
#!/bin/sh

# constants
IMAGE_FILE=/dd.img
DEPMOD=/sbin/depmod

# save current directory
CWD=`pwd`
# create temporary directories
mkdir /tmp/mnt-$$
mkdir /tmp/modules-$$
mount -o loop $IMAGE_FILE /tmp/mnt-$$
cd /tmp/modules-$$
if [ ! -e /tmp/mnt-$$/modules.cgz ]; then
    echo "malformed driver diskette image: $IMAGE_FILE"
    exit 1
```

```
fi
gunzip -c < /tmp/mnt-$$/modules.cgz | cpio -id

for moduleDir in *; do
    for moduleFile in `find $moduleDir -name "*.o"`; do
        if [ -d /lib/modules/$moduleDir ]; then
            targetModuleFiles=`find /lib/modules/$moduleDir -name
"$moduleFile"`

            if [ ."$targetModuleFiles" = ."" ]; then
                cp $moduleFile /lib/modules/$moduleDir/kernel
            else
                for targetModuleFile in $targetModuleFiles; do
                    cp $moduleFile $targetModuleFile
                done
            fi

            # update module dependencies
            $DEPMOD -a $moduleDir
        fi
    done
done

cd $CWD
umount /tmp/mnt-$$

# clean up
rm -rf /tmp/modules-$$
rm -rf /tmp/mnt-$$
```

1. Run “chmod +x update_modules.sh” to mark it executable.
2. Add the following text (with appropriate changes to reflect the names of the files you created) to the %post section of the kickstart file in order to facilitate execution of the script created in step 1.

```
# copy driver diskette image from the root directory of
# the NFS share to the root directory on the target machine

cp /mnt/source/dd.img /mnt/sysimage

# copy module update script from the root directory of
# the NFS share to the root directory on the target machine

cp /mnt/source/update_modules.sh /mnt/sysimage

# Execute the script in the chrooted environment
chroot /mnt/sysimage ./update_modules.sh
```

# Appendix 2: SUSE Driver Image Augmentation

### *SLES Driver Disk Overview*

SLES driver diskette packages are typically comprised of the following files:

- update.pre

  The update.pre file is a shell script that is executed by the SuSE installer (YaST) immediately after installation has commenced.

- update.post

  The update.post files is a shell script that is executed by the YaST installer after the installation has finished and the update.tar.gz archive has been extracted onto the installed system.

- update.tar.gz

  The file update.tar.gz is a gzipped tar archive that includes the updated driver modules. The archive should contain the driver modules compiled for all kernel versions available on the distribution in both SMP and uniprocessor configurations. The archive may include other files that should be installed into the system (e.g. additional documentation, new device nodes in /dev etc.)

  **NOTE:** This archive will be expanded in the root directory of the target system, so it should have a structure that includes paths relative to the root directory. For example, some of the drivers in the ServeRAID driver package will ultimately be located in the /lib/modules/2.4.19-64GB/kernel/drivers/scsi directory, so this file will be stored in the update.tar.gz file with the path:

  /lib/modules/2.4.19-64GB/kernel/drivers/scsi/ips.o

SLES boot images are typically comprised of the following files and directories:

- linuxrc
- linuxrc.config
- /modules

There are two sets of drivers that are used during a SLES installation -- the drivers required for transmission and retrieval of the driver diskette image and control file (pre-install) and the drivers that are loaded and used by the system after installation has taken place (post-install).

The pre-install environment drivers used to perform an installation are contained in the initial ramdisk (initrd) image used to initiate the installation. The file is usually named ‘initrd’ and contains a compressed ext2 or minix file-system.

The post-install environment drivers are typically provided manually by the customer via a SLES driver floppy diskette or by placing the files (update scripts, update.tar.gz) in the root directory of a partition on the hard drive.

### *Augmentation Process*

In the following example, we are going to add the new ServeRAID adapter drivers to the SLES 8 distribution as part of a sample RDM installation. The ServeRAID adapter drivers are typically distributed with a SLES driver disk image by default, so the process is greatly simplified. If you have obtained a driver that is not shipped with a SLES driver diskette or as a diskette image, then you will have to compile the module manually for the BOOT and post-install environments and analyze the source code to determine the PCI device and vendor IDs for the devices that are supported and recognized by the device driver; the manual process is beyond the scope of this document.

I. Modify Pre-install/Installation Boot Diskette Image

1. Identify and obtain the SLES initial ramdisk image that will be used to perform the installation.

   For this example, we will be using the initial ramdisk image located in the ‘boot/loader’ directory on the SLES installation media. This file is named ‘initrd’ on the SLES 8 source media.

2. Mount the boot image via the loopback device from a Linux terminal. The initrd stored in the loader directory is a compressed image, so it must be decompressed prior to mounting.

   ```
   # mkdir /mnt/image1

   # gunzip -c < initrd > initrd-new.img

   # mount -o loop initrd-new.img /mnt/image1
   ```

3. Identify and obtain the SLES driver diskette images.

   SLES driver diskette images are usually distributed as a single binary file. These images may be written to a floppy diskette using a program like rawrite or ‘dd’ or mounted via the loopback device under Linux. In this example, we will use IBM ServeRAID driver version 6.11 for SLES.

The image file will be referred to as ‘driver_dd.img’ in subsequent steps.

4. Mount the driver diskette image via the loopback device from a Linux terminal. Typically, driver diskette images do not have to be decompressed.

```
# mkdir /mnt/image2

# mount -o loop driver_dd.img /mnt/image2
```

5. Locate the driver modules and copy them to the boot image. The driver files are stored in the ‘linux/suse/${ARCH}-${VERSION}/modules’ directory in the initrd, where ${ARCH} is equal to the machine architecture and ${VERSION} is the SuSE version number. In this example, the full path is ‘linux/suse/i386-8.0/modules’.

   Some driver packages already contain the update.tar.gz file. If this is the case, then you will have to extract the module from this archive prior to copying it to the initrd.img.

```
# cp /mnt/image2/linux/suse/i386-8.0/modules/* /mnt/image1/modules
```

6. Add directives to linuxrc.config indicating that the new module should be loaded during the pre-installation phase.

   In this example, a previous version of the ServeRAID driver already exists in the default SLES 8 boot image. If the driver was not present then the following line would have to be added to the linuxrc.config file:

```
Insmod:      /modules/ips.o
```

7. Umount and compress the modified boot image.

```
# umount /mnt/image1

# gzip -9 -c < initrd-new.img > initrd.img
```

8. Place the image in the appropriate directory.

## II. Modify or Create the Post-install Driver Diskette Image

1. Create the supplemental driver disk containing drivers that will be used in the post-install environment.

   Up to this point, we have only been concerned with the drivers that will be used during installation for access to storage and network devices. Since we are only updating the ServeRAID drivers in this example, the files present in the update image placed in the installation source file path and comply with a specific directory structure.

You may encounter drivers for SLES in one of three states: binary drivers compiled for each supported kernel, binary drivers compiled for each supported kernel and stored in the requisite update.tar.gz, drivers in source code form.

If the update.tar.gz file already exists already then you simply need to copy this file to the root of the NFS share (to support the process that will be outlined below). If you only have the binary drivers, then you need to lay them out in the directory structure that will be present on the installed machine. As exemplified above, ServeRAID driver package will ultimately be located in the /lib/modules/2.4.19-64GB/kernel/drivers/scsi directory, so this file will be stored in the update.tar.gz file with the path:

```
/lib/modules/2.4.19-64GB/kernel/drivers/scsi/ips.o
```

For example, you would perform the following steps:

```
# mkdir –p lib/modules/2.4.19-64GB/kernel/drivers/ips.o

# cp 2.4.19-64GB/ips.o lib/modules/2.4.19-64GB/kernel/drivers/ips.o

# tar cvzf update.tar.gz lib
```

and them place the update.tar.gz file in the root of the NFS share. If you only have driver source code at your disposal, then you need to compile the driver for each supported kernel and repeat the steps outlined above.

2. Generate the following script (named update_modules.sh) and place it in the root of the NFS share (installation source file share) ensuring that it is marked executable.

```
#!/bin/sh

MKINITRD=/sbin/mk_initrd
TAR=/bin/tar
BOOTLOADER=/sbin/lilo

# expand module archive (copies modules to postinstall environment)
$TAR -xvzf /update.tar.gz

# recreate initrd
$MKINITRD

# re-run bootloader (only necessary if the bootloader is LILO)
$BOOTLOADER
```

### Control File Modification

You must add the following directives to the SuSE control file to initiate (execute) the script generated in step II. 2. The following example assumes that the post-installation root partition will be mounted at mount point /mnt. The highlighted lines were added to the example control file below:

```
<source><![CDATA[#!/bin/sh
for cmd in $(cat /proc/cmdline) ; do
  if echo $cmd | grep ^install ; then
    NFS_SERVER=$(echo $cmd | cut -f 3 -d /)
    NFS_DIR=/$(echo $cmd | cut -f 4- -d /)
  fi
done
SOURCE_DIR=/RDM_AGENT
RDM_CHROOT=true
export SOURCE_DIR NFS_SERVER NFS_DIR RDM_CHROOT
mkdir $SOURCE_DIR
mount ${NFS_SERVER}:${NFS_DIR} ${SOURCE_DIR}
/${SOURCE_DIR}/postscript

# post-installation process
cp ${SOURCE_DIR}/update.tar.gz /mnt
cp ${SOURCE_DIR}/update_modules.sh /mnt

chmod +x /mnt/update_modules.sh
chroot /mnt /update_modules.sh

umount /RDM_AGENT

]]></source>
```

## Appendix 3: Complete Kickstart File

This is the complete kickstart file used while producing this document.

```
#System  language
lang en_US
#Language modules to install
langsupport ja_JP.eucJP --default=en_US
#System keyboard
keyboard us
#System mouse
mouse generic3ps/2
#Sytem timezone
timezone America/New_York
#Root password
rootpw --iscrypted $1$61TUZDRZ$X0lYwzbCpGGoR3eJzU7lI0
#Reboot after installation
#reboot
#Use text mode install
text
#Install Red Hat Linux instead of upgrade
install
#Use CDROM installation media
#cdrom
nfs --server 10.1.6.230 --dir /share/redhat/
#System bootloader configuration
bootloader --location=mbr
#Clear the Master Boot Record
zerombr yes
#Partition clearing information
clearpart --all --initlabel
part /      --grow --size=1000
#part /home  --grow --size=1000
part /usr   --grow --size=1000
part swap   --grow --size=1000
#System authorization infomation
auth  --useshadow  --enablemd5
#Network information
network --bootproto=dhcp --device=eth0
#Firewall configuration
firewall --disabled
#Do not configure XWindows
skipx
#Package install information
%packages --resolvedeps
nc
%post --nochroot
echo starting post-script
/mnt/source/postscript
```

# Appendix 4: Complete YaST control File

This is the complete YaST control file (“file.xml”) used while producing this document

```
<?xml version="1.0"?>
<!DOCTYPE profile SYSTEM "/usr/lib/YaST2/profile.dtd">
<profile xmlns="http://www.suse.com/1.0/yast2ns"
xmlns:config="http://www.suse.com/1.0/configns" >
 <configure>
<scripts>
 <chroot-scripts config:type="list" >
   <script>
    <filename>chroot.sh</filename>
    <interpreter>shell</interpreter>
    <source><![CDATA[#!/bin/sh
for cmd in $(cat /proc/cmdline) ; do
  if echo $cmd | grep ^install ; then
    NFS_SERVER=$(echo $cmd | cut –f 3 –d /)
    NFS_DIR=/$(echo $cmd | cut –f 4- -d /)
  fi
done
SOURCE_DIR=/RDM_AGENT
RDM_CHROOT=true
export SOURCE_DIR NFS_SERVER NFS_DIR RDM_CHROOT
mkdir $SOURCE_DIR
mount ${NFS_SERVER}:${NFS_DIR} ${SOURCE_DIR}
/${SOURCE_DIR}/postscript
umount /RDM_AGENT

]]></source>
   </script>
  </chroot-scripts>
  </scripts>
  <networking>
   <dns>
    <dhcp_hostname config:type="boolean" >
true</dhcp_hostname>
    <domain>test.net</domain>
    <hostname>elfman</hostname>
    <nameservers config:type="list" />
    <searchlist config:type="list" />
   </dns>

   <interfaces config:type="list" >
    <interface>
     <BOOTPROTO>dhcp</BOOTPROTO>
     <STARTMODE>onboot</STARTMODE>
     <UNIQUE></UNIQUE>
     <device>eth0</device>
     <module>bcm5700</module>
     <options>options=0</options>
    </interface>
   </interfaces>
   <routing config:type="list" />
  </networking>
  <runlevels>
   <default>5</default>
   <services config:type="list" />
  </runlevels>
  <users config:type="list" >
   <user>
    <encrypted config:type="boolean" >false</encrypted>
    <user_password>CzKt94SXmJDmL</user_password>
    <username>root</username>
   </user>
   <user>
    <encrypted config:type="boolean" >true</encrypted>
    <user_password>s1hqUhkjhkjhk</user_password>
    <username>dsmouse</username>
   </user>
  </users>
  <x11>
   <color_depth config:type="integer" >16</color_depth>
   <configure_x11 config:type="boolean" > true</configure_x11>
   <display_manager>kdm</display_manager>
   <enable_3d config:type="boolean" >false</enable_3d>
   <monitor>
    <display>
     <bandwidth config:type="integer" >0</bandwidth>
     <height config:type="integer" >768</height>
     <max_hsync config:type="integer" >48</max_hsync>
     <max_vsync config:type="integer" >90</max_vsync>
     <min_hsync config:type="integer" >30</min_hsync>
     <min_vsync config:type="integer" >50</min_vsync>
     <width config:type="integer" >1024</width>
    </display>
    <monitor_device>DT-1536A</monitor_device>
    <monitor_vendor>DAYTEK</monitor_vendor>
   </monitor>
   <resolution>1024x768</resolution>
  </x11>
 </configure>
 <install>
   <general>
   <keyboard>english-us</keyboard>
   <language>en_US</language>
   <mouse>probe</mouse>
   <mode>
      <confirm config:type="boolean">false</confirm>
      <forceboot config:type="boolean">false</forceboot>
      <reboot config:type="boolean">false</reboot>
   </mode>
   <reboot config:type="boolean" >true</reboot>
   <timezone>Canada/Eastern</timezone>
  </general>
  <partitioning config:type="list" >
   <drive>
    <device>/dev/sda</device>
    <partitions config:type="list" >
     <partition>
      <crypt_fs config:type="boolean" >false</crypt_fs>
      <crypt_key></crypt_key>
      <filesystem config:type="symbol" >ext2</filesystem>
      <filesystem_id config:type="integer" > 31</filesystem_id>
      <format config:type="boolean" >true</format>
      <mount>/boot</mount>
      <size>400mb</size>
      <type>Linux native</type>
     </partition>
     <partition>
      <crypt_fs config:type="boolean" >false</crypt_fs>
      <crypt_key></crypt_key>
      <filesystem config:type="symbol" >xfs</filesystem>
      <filesystem_id config:type="integer" > 131</filesystem_id>
      <format config:type="boolean" >true</format>
      <mount>/</mount>
      <size>4gb</size>
      <type>Linux native</type>
     </partition>
     <partition>
      <crypt_fs config:type="boolean" >false</crypt_fs>
      <crypt_key></crypt_key>
      <filesystem config:type="symbol" >swap</filesystem>
      <filesystem_id config:type="integer" > 131</filesystem_id>
      <format config:type="boolean" >true</format>
      <mount>swap</mount>
      <size>auto</size>
      <type>Linux native</type>
     </partition>
     <partition>
      <crypt_fs config:type="boolean" >false</crypt_fs>
      <crypt_key>anas12</crypt_key>
      <filesystem config:type="symbol" > reiser</filesystem>
      <filesystem_id config:type="integer" > 131</filesystem_id>
      <format config:type="boolean" >true</format>
      <mount>/local</mount>
      <size>max</size>
      <type>Linux native</type>
     </partition>
    </partitions>
    <use>all</use>
   </drive>
  </partitioning>
  <report>
   <confirm config:type="boolean" >false</confirm>
   <errors>
    <log config:type="boolean" >true</log>
    <show config:type="boolean" >true</show>
    <timeout config:type="integer" >30</timeout>
   </errors>
   <messages>
    <log config:type="boolean" >true</log>
    <show config:type="boolean" >true</show>
    <timeout config:type="integer" >10</timeout>
   </messages>
   <warnings>
    <log config:type="boolean" >true</log>
    <show config:type="boolean" >true</show>
    <timeout config:type="integer" >10</timeout>
   </warnings>
  </report>
  <software>
    <base>Minimal</base>
  </software>
  <bootloader>
     <loader_type>grub</loader_type>
     <loader_device>/dev/sda</loader_device>
     <write_bootloader
config:type="boolean">true</write_bootloader>
     <location>mbr</location>
     <linear  config:type="boolean">false</linear>
     <lba_support  config:type="boolean">false</lba_support>
     <activate  config:type="boolean">true</activate>
     <kernel_parameters></kernel_parameters>
     <initrd_modules  config:type="list" >
     </initrd_modules>
   </bootloader>
 </install>
</profile>
```

## Appendix 5: Example RDAgent Script

The following RDAgent script must be used to circumvent Red Hat Bug ID 134054. Please see Red Hat BUG ID 134054: ("anaconda ip= and pxelinux IPAPPEND don't coexist peacefully") regarding this issue at: https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=134054.

**NOTE:** This script file must be used in conjunction with the pxelinux changes described in step 3c.

```
#!/bin/sh

echo Running RDM post-install script

########################################
# Setup variables
########################################

export DEV=eth0

if [ ."$RDM_CHROOT" = ."" ]; then
   DEST_DIR=/
else
   DEST_DIR=/mnt/sysimage
fi

if [ ."$SOURCE_DIR" = ."" ]; then
   SOURCE_DIR=/mnt/source
fi

########################################
# Parse kernel command line for network
# information
########################################


for CMD in $(cat /proc/cmdline); do
   if echo $CMD | grep ^ip 2>&1 > /dev/null; then
      IP_LINE=$(echo $CMD | cut -f 2 -d \! )

      CLIENT=$(echo $IP_LINE  | cut -f  1 -d :)
      SERVER=$(echo $IP_LINE  | cut -f  2 -d :)
      GATEWAY=$(echo $IP_LINE | cut -f  3 -d :)
      NETMASK=$(echo $IP_LINE | cut -f  4 -d :)
   fi
```

```
done

########################################
# Set environment variables for rdagent
########################################

TARGET=$(ifconfig  $DEV | grep HWaddr  | sed 's/^.*HWaddr //' | \
sed 's/://g')

export server_ip client_ip uuid bootmac ip
export server_ip=$SERVER
export client_ip=$CLIENT
export ip=${CLIENT}:${SERVER}:${GATEWAY}:${NETMASK}

bootmac=$(echo $TARGET | sed -e 's/://g')

export RDRASLEVEL=0
export RDSTATUS=RDAGEN000I
export RDRETPATH=/
export RDRETFILE=none

${SOURCE_DIR}/rdagent /e

if ( test -f ./_rdm.bat) ; then
   . ./_rdm.bat
fi

########################################
# Process RDM commands
########################################

while [ ! -f /tmp/rdm_finished ] ; do
   ${SOURCE_DIR}/rdagent
   if ( test -f ./_rdm.bat) ; then
      . ./_rdm.bat
   fi
   rm -rf ./_rdm.bat
done
```

## Appendix 6: Syslinux 3.11 Patch

```
diff -Nru syslinux-3.11.orig/pxelinux.asm syslinux-3.11/pxelinux.asm
--- syslinux-3.11.orig/pxelinux.asm     2005-08-28 02:00:54.000000000 -
0400
+++ syslinux-3.11/pxelinux.asm  2005-10-25 21:47:09.000000000 -0400
@@ -2274,7 +2274,7 @@
 genipopt:
                pushad
                mov di,IPOption
-               mov eax,'ip='
+               mov eax,'ip!'
                stosd
                dec di
                mov eax,[MyIP]
```

# Notices

This information was developed for products and services offered in the U.S.A.

IBM might not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service might be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right might be used instead. However, it is the user responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM might have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement might not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM might make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product, and use of those Web sites is at your own risk.

IBM might use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Some software might differ from its retail version (if available) and might not include all user manuals or all program functionality.

IBM makes no representations or warranties regarding third-party products or services.

Edition notice