IBM Systems and Technology Group

Under the Hood: Red Hat / SUSE Linux Configuration Cross-Reference

Session 9211 - August 17,2006

Mark Ver

Test and Integration Center for Linux - markver@us.ibm.com

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

ECKD
ESCON*
FICON*
Hipersockets
IBM*
IBM eServer
System z
z/OS*
z/VM*
zSeries*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Intel ® is a registered trademark of the Intel Corporation in the United States, other countries or both.
Linux ® is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Penguin (Tux) compliments of Larry Ewing (lewing@isc.tamu.edu) and The GIMP.
Red Hat ® is a registered trademark of Red Hat, Inc. in the United States, other countries or both
SUSE ® is a registered trademark of Novell, Inc. in the United States, other countries or both
UNIX ® is a registered trademark of The Open Group in the United States and other countries

* All other products may be trademarks or registered trademarks of their respective companies.

Agenda

Networking
- Setting up routes
- Defining an additional (qeth) interface
- Configuring the system as a router
- qeth driver options

Disk Configuration
- Setting up additional DASD devices

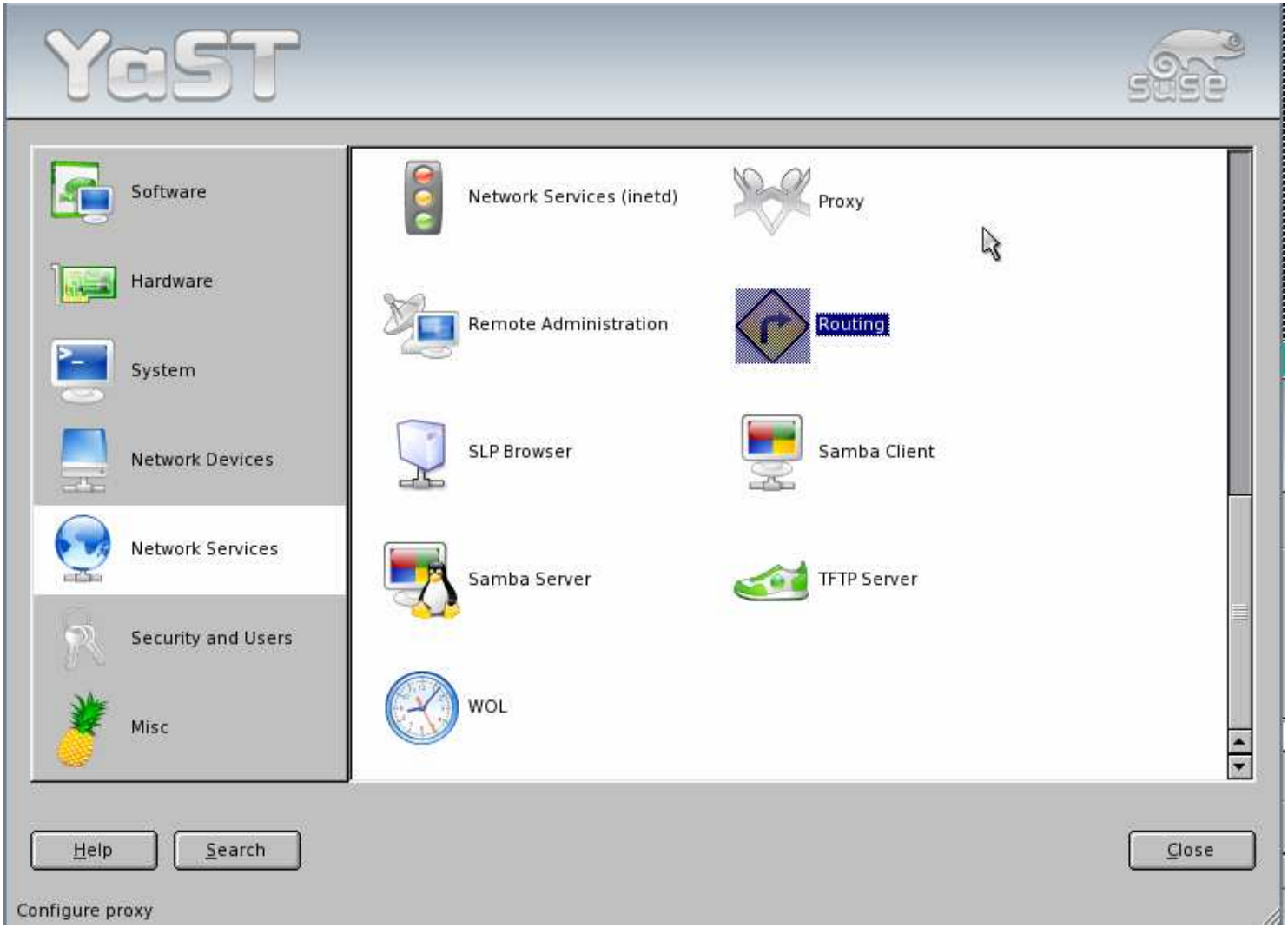
Boot setup
- Boot menus and zfcp.conf
- The initrd
- Initialization scripts

Brief look into SUSE LINUX Enterprise Server 10 (SLES-10)

Appendix
- Links and other Sources
- Slides for Additional Information

Note: Configuration information presented here applies to Red Hat Enterprise Linux 4 (RHEL4) and SUSE Linux Enterprise Server 9 (SLES-9).

Setting up routes on SUSE Linux



Setting up routes on SUSE Linux

Routing configuration is stored in the following file:

- /etc/sysconfig/network/routes

Sample contents of the routes file:

- # network gateway netmask device
- default 192.168.70.1 - -
- 10.10.230.0 192.168.70.98 255.255.255.0 qeth-bus-ccw-0.0.0600

Note:

- You can also put similar statements into an ifroute-* file in /etc/sysconfig/network (ex. ifroute-qeth-bus-ccw-0.0.0600) and get the same result.

Setting up routes on Red Hat Enterprise Linux

Routing info is stored in the following two files

- /etc/sysconfig/network
- /etc/sysconfig/network-scripts/route-xxx (xxx=eth0,eth1,hsi0,etc ...)

Sample contents of the /etc/sysconfig/network file:

- NETWORKING=yes
- HOSTNAME=LTIC0031.pdl.pok.ibm.com
- GATEWAY=192.168.70.1
- ARP=no

Sample contents of an /etc/sysconfig/network-scripts/route-eth0 file:

- ADDRESS0=10.10.230.0
- NETMASK0=255.255.255.0
- GATEWAY0=192.168.70.98
- ADDRESS1=9.12.20.0
- NETMASK1=255.255.255.0
- GATEWAY1=192.168.70.32

Alternate contents (old format):

- 10.10.230.0/24 via 192.168.70.98

Adding a qeth Interface on SUSE Linux

The main two files involved in network setup

- /etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.xxxx
- /etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.xxxx

The basic steps:

- Copy the hwcfg file (renaming it to refer to the new target device)
  - Ex. cp hwcfg-qeth-bus-ccw-0.0.0900 hwcfg-qeth-bus-ccw-0.0.0600
- Modify the contents of the new hwcfg file with the new device numbers and portname
- Copy the ifcfg file (renaming it to refer to the new device)
  - Ex. cp ifcfg-qeth-bus-ccw-0.0.0900 ifcfg-qeth-bus-ccw-0.0.0600
- Modify the contents of the new ifcfg file with the desired IP info
- Trigger device setup
  - Ex.

```
cd /sys/bus/ccwgroup/drivers/qeth
echo 0.0.0600,0.0.0601,0.0.0602 > group
```

Adding a qeth Interface on SUSE Linux

- Contents of sample file: hwcfg-qeth-bus-ccw-0.0.0600

```
#!/bin/sh
#
# hwcfg-qeth-bus-ccw-0.0.0600
#
# Hardware configuration for a qeth device at 0.0.0600
# Automatically generated by netsetup
#

STARTMODE="auto"
MODULE="qeth_mod"
MODULE_OPTIONS=""
MODULE_UNLOAD="yes"

# Scripts to be called for the various events.
SCRIPTUP="hwup-ccw"
SCRIPTUP_ccw="hwup-ccw"
SCRIPTUP_ccwgroup="hwup-qeth"
SCRIPTDOWN="hwdown-ccw"

# CCW_CHAN_IDS sets the channel IDs for this device
# The first ID will be used as the group ID
CCW_CHAN_IDS="0.0.0600 0.0.0601 0.0.0602"

# CCW_CHAN_NUM set the number of channels for this device
# Always 3 for an qeth device
CCW_CHAN_NUM=3

# CCW_CHAN_MODE sets the port name for an OSA-Express device
CCW_CHAN_MODE="dt70“
```

Adding a qeth Interface on SUSE Linux

- Contents of sample file: ifcfg-qeth-bus-ccw-0.0.0600

```
BOOTPROTO="static"
STARTMODE="onboot“
UNIQUE=""
IPADDR="192.168.70.178"
MTU="1492"
NETMASK="255.255.255.0"
NETWORK="192.168.70.0"
BROADCAST="192.168.70.255"
```

Adding a qeth Interface on SUSE Linux

- Interface bring up is based on device number – Introduction or removal of a numerically lower ccwgroup id could reorder the device name assignment after reboot (ex. in original setup 0900 was eth0, but with new setup 0600 becomes eth0 and 0900 becomes eth1).

- Persistent Names:
  - Allows user to map network device to a specific interface name and keep it that way even when some configured devices disappear.
  - Ex. (Add in /etc/sysconfig/network/ifcfg-qeth-bus-ccw-0.0.xxxx)

    PERSISTENT_NAME="eth0“

Adding a qeth Interface on Red Hat Enterprise Linux

The main files used for configuring network devices on Red Hat Enterprise Linux

- /etc/modprobe.conf
- /etc/sysconfig/network-scripts/ifcfg-xxx

Basic steps for configuration on Red Hat Enterprise Linux

- Add an alias statement in /etc/modprobe.conf for the new device:
  - Ex. alias eth1 qeth
- Copy an existing ifcfg (renaming it to new device name)
  - Ex. cp ifcfg-eth0 ifcfg-eth1
- Modify the new ifcfg file with the desired Device and IP info
- Trigger network setup
  - Ex. service network restart

Adding a qeth Interface on Red Hat Enterprise Linux

- Sample file: ifcfg-eth0

```
# IBM QETH
DEVICE=eth0
BOOTPROTO=static
BROADCAST=192.168.70.255
IPADDR=192.168.70.231
NETMASK=255.255.255.0
NETTYPE=qeth
NETWORK=192.168.70.0
ONBOOT=yes
PORTNAME=DT70
SUBCHANNELS=0.0.0600,0.0.0601,0.0.0602
TYPE=Ethernet
```

Adding a qeth Interface on Red Hat Enterprise Linux

- Red Hat network setup tool does not handle zSeries devices.
- Bring up is based on interface name. When a device becomes unavailable, the interface names get remapped. Unfortunately this causes a mismatch with the ifcfg file names, so the interfaces get the wrong IP info.
  - Ex. Original setup:
    - ifcfg-eth0 -> 900,901,902 -> eth0 -> 10.10.10.1
    - Ifcfg-eth1 -> 600,601,602 -> eth1 -> 192.168.70.36

      Setup after OSA ccwgroup 900-902 becomes unavailable:
    - Ifcfg-eth0 -> NA
    - Ifcfg-eth1 -> 600,601,602 -> eth0 -> 10.10.10.1
- There is no persistent names parameter in Red Hat Enterprise Linux.

Configuring system as a router on SUSE Linux

Manual (Temporary) Configuration

- Set router options for the qeth driver ccwgroup
  - Ex.

    echo primary_router > /sys/bus/ccwgroup/drivers/qeth/0.0.0600/route4
- Set linux ip forwarding on
  - Ex.

    echo 1 > /proc/sys/net/ipv4/ip_forward

For Configuration on boot up

- Setup qeth router options in the hwcfg-qeth-bus-ccw-0.0.xxxx:
  - Ex.

    QETH_OPTIONS="route4=primary_router“
- Modify IP_FORWARD parameter in /etc/sysconfig/sysctl to turn IP forwarding on:
  - Ex.

    IP_FORWARD=“yes”

Configuring system as a router: Red Hat Enterprise Linux

Manual (Temporary) Configuration

- Set router options for the qeth driver ccwgroup
  - Ex.

    echo primary_router > /sys/bus/ccwgroup/drivers/qeth/0.0.0600/route4
- Set linux ip forwarding on
  - Ex.

    echo 1 > /proc/sys/net/ipv4/ip_forward

Configuration for boot up

- Setup qeth router options in /etc/sysconfig/network-scripts/ifcfg-ethX:
  - Ex.

    OPTIONS="route4=primary_router“
- Use /etc/sysctl.conf to enable IP forwarding
  - Ex.

    net.ipv4.ip_forward = 1

QETH Parameters

SUSE

- /etc/sysconfig/hardware/hwcfg-qeth-bus-ccw-0.0.xxxx
- QETH_OPTIONS="option=value"
  - echo value > /sys/bus/ccwgroup/drivers/qeth/0.0.xxxx/option
  - Ex.

    QETH_OPTIONS="route4=primary_router fake_ll=1"
- QETH_IPA_TAKEOVER=<anything>
  - echo 1 > /sys/bus/ccwgroup/drivers/qeth/0.0.xxxx/ipa_takeover/enable
- QETH_LAYER2_SUPPORT=<anything>
  - echo 1 > /sys/bus/ccwgroup/drivers/qeth/0.0.xxxx/layer2

Red Hat

- /etc/sysconfig/network-scripts/ifcfg-ethX
- OPTIONS="option=value "
  - Ex.

    OPTIONS="route4=primary_router ipa_takeover/enable=1 layer2=1"

QETH Parameters and Network Scripts

SUSE

- The main script that handles the qeth driver options is /etc/sysconfig/hardware/scripts/hwup-qeth
- In General:
  - Device bring up is handled by the scripts in
    /etc/sysconfig/hardware/scripts/*
  - Interface bring up and IP setup is handled by the scripts in
    /etc/sysconfig/network/scripts/*

Red Hat

- The script handling the qeth driver options is /etc/sysconfig/network-scripts/network-functions
- That file basically provides functions/procedures used for device and ip initialization by ifup and the other scripts in the /etc/sysconfig/network-scripts/ directory.

Red Hat and SUSE Linux Networking

Red Hat

- GUI tools do not handle zSeries OSA device
- Device configuration and IP configuration is done all in the same file
- Network config files and device initialization order is based on interface number

SUSE

- The tool “yast” can be used to help with adding and editing device and IP configuration
- Device configuration and IP configuration is split into a hwcfg file and an ifcfg file
- Network config files and device initialization order is based on ccw bus_id

Setting up Additional DASD on SUSE Linux

Default Procedure:

- Bring desired DASD online
  - Use yast, dasd_configure, or chccwdev
  - Ex.

    dasd_configure 0.0.0200 1 0

- Run mkinitrd

- Run zipl to pick up the new initrd
  - Ex.

    zipl –V

Setting up Additional DASD on SUSE Linux

DASD in more depth

- Configuration locations that affect the DASD list:
  - Kernel parameters (dasd=…) as configured in /etc/zipl.conf
  - The initrd (taken from the current dasd configuration when you run mkinitrd)
  - The hwcfg-dasd-* files located in /etc/sysconfig/hardware – these files are created when you use yast or dasd_configure to bring the disks online.
- The initrd setup is what SUSE Linux creates by default when you install SLES-9 from scratch.
- The dasd= parameter has priority, overrides any configuration in the initrd, and is what you normally see if you updated your system from SLES-8.
  - Ex. parameters = “dasd=200,201,202 root=/dev/dasda1 selinux=0 TERM=dumb”
- Note: Currently on SLES-9 if you use yast or dasd_configure to bring the DASD online,,but fail to run mkinitrd/zipl, the disks do come online during boot up. But they cannot be mounted via /etc/fstab.

Setting up Additional DASD on SUSE Linux

DASD and device files

- By default SUSE Linux only has devices files for the first 26 dasd (that is dasda-dasdz).
- So if you add more than 26 disks, you sometimes need to manually create the device files:
  - Ex.

    mknod /dev/dasdaa b 94 104

    mknod /dev/dasdaa1 b 94 105
- When bringing dasd online, SUSE Linux automatically creates by-id and by-path entries which can be used to identify disks more easily:
  - Ex.

    lrwxrwxrwx 1 root root 11 Jul 19 13:06 /dev/disk/by-id/LX572A -> ../../dasdb

    lrwxrwxrwx 1 root root 12 Jul 19 13:06 /dev/disk/by-id/LX572A1 -> ../../dasdb1

    lrwxrwxrwx 1 root root 11 Jul 19 13:06 /dev/disk/by-path/ccw-0.0.0201 -> ../../dasdb

    lrwxrwxrwx 1 root root 12 Jul 19 13:06 /dev/disk/by-path/ccw-0.0.0201p1 -> ../../dasdb1
- Turning on boot.udev allows clean up of entries in /dev/disk/by-label/ and /dev/disk/by-path/ directories.
  - Ex.

    chkconfig boot.udev on

    /etc/init.d/boot.udev start

Setting up Additional DASD on Red Hat Enterprise Linux

Default Procedure:

- Modify dasd module options in /etc/modprobe.conf to include your new dasd
  - Ex.

    options dasd_mod dasd=200,201,202

- Run mkinitrd to create an initrd that includes the modified module options
  - Ex.

    cd /boot

    mkinitrd -v initrd.new `uname –r`

- Rename initrd.new as needed (make sure it matches ramdisk= entry for the current kernel in /etc/zipl.conf).
- Run zipl to make the changes effective on boot up:
  - Ex.

    zipl -V

Setting up Additional DASD on Red Hat Enterprise Linux

Device files on Red Hat Enterprise Linux

- Nothing is permanent in /dev/ - udev controls it all.
- All device files are created automatically by udev.
- Devices are removed automatically when the device goes offline.
- There is a delay between the time the device is brought online and when the device file gets created.
- Similar to the SUSE Linux /dev/disk/by-path entries, Red Hat Enterprise Linux creates links to allow identification of the dasd by device number:
  - Ex.

    lrwxrwxrwx 1 root root 11 Jul 19 02:34 /dev/dasd/0.0.0201/disc -> ../../dasdb

    lrwxrwxrwx 1 root root 12 Jul 19 02:34 /dev/dasd/0.0.0201/part1 -> ../../dasdb1

Red Hat and SUSE Linux Disk Setup

Red Hat

- No graphical configuration tools
- Single location for DASD configuration
- Single location for zFCP SCSI configuration
- Dynamic device file creation/removal (sometimes requires waiting logic in scripts – to adjust for the delay between when the device is brought online and when the device file is available)

SUSE

- yast can be used to partially configure DASD and zFCP SCSI (will need to issue manual commands to add disk configuration to initrd and make devices available for use with fstab)
- Multiple locations for DASD configuration
- Multiple config files for zFCP SCSI (depending on the number of FCP ccw devices used).
- Predefined set of available device files. Partial dynamic device file creation. No automatic device file removal (except possibly some cleanup of /dev/disk/ entries when boot.udev is active).

Boot menus and zipl.conf

- RHEL-4 and SLES-9 both support boot menus. But RHEL-4 uses boot menus automatically and SLES-9 does not.
  - Ex.

    zIPL v1.3.2 interactive boot menu

    0. default (linux)

    1. linux
    2. bkup

    Note: VM users please use '#cp vi vmsg <input>'

    Please choose (default will boot in 15 seconds):
    #cp vi vmsg 2

Boot menus and zipl.conf

- Doing a menu setup on SUSE Linux: sample zfcp.conf

```
[defaultboot]
#    default = ipl
defaultmenu = menu1

[ipl]
    target = /boot/zipl
    image = /boot/image
    ramdisk = /boot/initrd
    parameters = "root=/dev/dasdb1 selinux=0 TERM=dumb elevator=cfq"

[bkup]
    target = /boot/zipl
    image = /boot/image.bkup
    ramdisk = /boot/initrd.bkup
    parameters = "root=/dev/dasdb1 selinux=0 TERM=dumb elevator=cfq"

:menu1
target = /boot/zipl
timeout = 10
prompt = 1
1 = ipl
2 = bkup
default = 1
```

Boot menus and zipl.conf

```
[root@LTIC0031 boot]# zipl -V
[Using config file '/etc/zipl.conf'
Target device information
  Device.....: 5e:04
  Partition.....: 5e:05
  Device name.....: dasdb
  DASD device number.....: 0201
  Type.....: disk partition
  Disk layout.....: ECKD/compatible disk layout
  Geometry - heads.....: 15
  Geometry - sectors.....: 12
  Geometry - cylinders.....: 3339
  Geometry - start.....: 24
  File system block size.....: 4096
  Physical block size.....: 4096
  Device size in physical blocks..: 600996
Building bootmap '/boot//bootmap'
Building menu 'rh-automatic-menu'
Adding #1: IPL section 'linux' (default)
  kernel image.....: /boot/vmlinuz-2.6.9-34.EL at 0x10000
  kernel parmline...: 'root=LABEL=/' at 0x1000
  initial ramdisk...: /boot/initrd-2.6.9-34.EL.img at 0x800000
Adding #2: IPL section 'bkup'
  kernel image.....: /boot/image.bkup at 0x10000
  kernel parmline...: 'root=LABEL=/' at 0x1000
  initial ramdisk...: /boot/initrd.bkup at 0x800000
Preparing boot device: dasdb (0201).
Preparing boot menu
  Interactive prompt.....: enabled
  Menu timeout.....: 15 seconds
  Default configuration...: 'linux'
Syncing disks...
Done.
```


The initrd

- The initrd - “initial ramdisk” – gives the boot process access to drivers and other configuration information for bringing up needed devices (ex. DASD).
- The tool for creating an initrd is mkinitrd – usually when this is executed it checks for device drivers currently in use and automatically includes them in the initrd creation.
  - Ex. SUSE: `mkinitrd`

    Red Hat: `mkinitrd -v -f initrd \`uname -r\``
- Sometimes the driver you want is not included automatically so you’ll have to manually let mkinitrd know what drivers to include.
- SuSE: modify /etc/sysconfig/kernel and rerun mkinitrd
  - Ex.

    INITRD_MODULES=“dasd_eckd_mod zfcp sd_mod reiserfs“
- Red Hat: use “—with=” parameter on the mkinitrd call
  - Ex.

```
mkinitrd -v --with=zfcp --with=sd_mod --with=dasd_eckd_mod -f initrd `uname -r`
```

The initrd

- It is sometimes useful to take a look at the contents of a particular initrd
- On SLES-9 the initrd is a gzipped ext2 filesystem – to access its contents, you expand the initrd and mount it as a loop device
  - Ex.

    LTIC0008:~ # mkdir testdir

    LTIC0008:~ # cp /boot/initrd /root/initrd.gz

    LTIC0008:~ # gunzip initrd.gz

    LTIC0008:~ # mount -o loop initrd testdir

    LTIC0008:~ # ls testdir

    . .. bin dev etc lib lib64 linuxrc mnt proc sbin sys tmp
- Things you can find on the SLES-9 initrd:
  - linuxrc – the file that calls insmod/modprobe for modules in the initrd (including the DASD driver).
  - You can also check for device drivers in the initrd, ex.

    LTIC0008:~ # find testdir/lib/modules | grep .ko

    testdir/lib/modules/2.6.5-7.244-s390x/kernel/fs/reiserfs/reiserfs.ko

    testdir/lib/modules/2.6.5-7.244-s390x/kernel/drivers/s390/block/dasd_mod.ko

    testdir/lib/modules/2.6.5-7.244-s390x/kernel/drivers/s390/block/dasd_fba_mod.ko

    testdir/lib/modules/2.6.5-7.244-s390x/kernel/drivers/s390/block/dasd_eckd_mod.ko

The initrd

- On RHEL-4 the initrd is a gzippped cpio archive – to access its contents, you expand the initrd and then extract the archive contents into a directory.
  - Ex.

    [root@LTIC0031 ~]# mkdir testdir

    [root@LTIC0031 ~]# cp /boot/initrd-2.6.9-34.EL.img /root/initrd.gz

    [root@LTIC0031 ~]# gunzip initrd.gz

    [root@LTIC0031 ~]# cd testdir

    [root@LTIC0031 testdir]# cat ../initrd | cpio –ivd

    [root@LTIC0031 testdir]# ls

    bin dev etc init lib loopfs proc sbin sys sysroot

- What to check on the RHEL-4 initrd
  - The ./init file controls driver loading (including DASD)
  - The device drivers are usually in the ./lib directory ex.

    [root@LTIC0031 testdir]# ls lib

    dasd_eckd_mod.ko dasd_fba_mod.ko dasd_mod.ko ext3.ko jbd.ko

The Initialization Scripts

- On both SUSE and Red Hat Enterprise Linux, the daemons for services (ex. vsftpd, xinetd, syslogd) are started and stopped using initialization scripts
- They are executed according to the presence of start and stop scripts in the rcX.d directory for the run level the system is trying to enter
  - Ex.

    rc3.d/K10sshd ← stop script

    rc3.d/S12sshd ← start script
- The run level that the system enters during boot up is determined by the id –initdefault entry in /etc/inittab:
  - Ex. id:3:initdefault:
- Services can be configured on/off for boot by using the chkconfig tool
  - Ex. chkconfig sshd on

    chkconfig sshd off

    chkconfig –list

The Initialization Scripts

- Locating scripts on SUSE Linux
  - All the major initialization scripts are located in /etc/init.d/
  - /etc/init.d/boot is the system initialization script. It also executes scripts in /etc/init.d/boot.d/ as well as /etc/init.d/boot.local (start and stop scripts in boot.d are symlinks to boot.* files in /etc/init.d)
  - The runlevel directories are in /etc/init.d/ (ex. /etc/init.d/rc3.d). Start and stop scripts in the runlevel directory are symlinks to files in /etc/init.d
    - Ex.

      LTIC0008:/etc/init.d/rc3.d # pwd

      /etc/init.d/rc3.d

      LTIC0008:/etc/init.d/rc3.d # ls -l *xinetd

      lrwxrwxrwx 1 root root 9 Jul 19 18:47 K08xinetd -> ../xinetd

      lrwxrwxrwx 1 root root 9 Jul 19 18:47 S14xinetd -> ../xinetd

- For starting/stopping services (for the running system), SUSE Linux provides symlinks that are in the default command path.
  - Ex.

    lrwxrwxrwx 1 root root 18 Jul 5 15:22 /sbin/rcsyslog -> /etc/init.d/syslog

    rcsyslog start

    rcsyslog stop

The Initialization Scripts

- Locating scripts on Red Hat Enterprise Linux
  - Most of the initialization scripts are in /etc/rc.d/init.d/. The system initialization script rc.sysinit, as well as rc and rc.local, are in /etc/rc.d/
  - /etc/init.d is a symlink to /etc/rc.d/init.d/
  - The runlevel directories are in /etc/rc.d/ (ex. /etc/rc.d/rc3.d) and scripts within the runlevel directories are just symlinks to files in /etc/rc.d/init.d/
    - Ex.

      [root@LTIC0031 rc3.d]# pwd

      /etc/rc.d/rc3.d

      [root@LTIC0031 rc3.d]# ls -l *xinetd

      lrwxrwxrwx 1 root root 16 Mar 22 14:30 S56xinetd -> ../init.d/xinetd

- Red Hat Enterprise Linux provides a command “service” for stopping and starting services (for the running system):
  - Ex.

    service xinetd start

    service xinetd stop

Red Hat and SUSE Linux Boot Setup

Red Hat

- Automatic use of Boot menus
- mkinitrd requires initrd and kernel version parameters and manual insertion of modules into initrd is done from additional arguments to mkinitrd
- initrd (RHEL-4) is a gzipped cpio archive
- System initialization file is /etc/rc.d/rc.sysinit
- Services can be started/stopped with the "service" command, ex:
  service sshd start
  service sshd stop

SUSE

- Need to manually define menu configuration in /etc/zipl.conf
- mkinitrd requires no arguments and it can pick up intended initrd modules from a parameter in /etc/sysconfig/kernel
- initrd (SLES-9) is a gzipped ext2 file system
- System initialization file is /etc/init.d/boot
- Services can usually be started/stopped by calling a symlink name, ex:
  rcsshd start
  rcsshd stop

New on SLES-10

- Networking
  - yast network configuration panel has changed slightly
    - choice of network manager or traditional ifup
    - SLES-9 had buttons for "configure" (setup a new device) and "change" (edit a configured device). SLES-10 now uses “edit” button for both setup and modifying an existing device.
  - More general use of udev sets up automatic persistent naming:
    - The names are defined via /etc/udev/rules.d/30-net_persistent_names.rules
  - The ctc and iucv drivers are still available but being deemphasized.
    - This is noted in release-notes which are found in the /docu directory on the install CDs

- Disk Setup
  - udev now controls /dev (just like with RHEL nothing stays permanently on /dev)

```
metlnx03:~ # df | grep udev
dev                509868       208    509660   1% /dev
```

  - hwcfg-dasd-bus-ccw-* configuration files now work with /etc/fstab – no need to call mkinitrd every time you add a new dasd using dasd_configure or yast.
  - Filesystem labels are accessed via symlinks, ex: /dev/disk/by-label/labelname
  - Entries in disk/by-id/ and disk/by-path/ have changed, ex: ccw-0x0380 and ccw-0.0.0380-part1

New on SLES-10

- Installation and boot up
  - Installation changes
    - Network configuration dialog during Installation has changed in various ways (including addition of dialog for setting up Layer2 support), Ex.

      Enable OSI Layer 2 support?

      1) Yes
      2) No
    - Parameters for autoboot have changed (see docu/en/preparation.pdf in the installation CDs for details).
    - Setup script for configuration during first boot (with ssh display) has changed name. Was formerly /usr/lib/YaST2/bin/YaST2.sshinstall. Is now /usr/lib/YaST2/startup/YaST2.ssh
    - Installation via NFS / SMB now requires 512M of memory (see docu/en/preparation.pdf)
  - Initrd changes
    - The initrd is now a cpio archive. same as on RHEL-4
    - The linuxrc file is gone and processing, such as module loading, is now done in the “init” file
  - Boot Menu is used by default
    - Default menu offers 2 choices “ipl” (the default) and “failsafe”
  - The aaa_base package now provides a “service” command for executing an initialization script.

Summary

- We looked at various configuration tasks for Linux on a System z platform with the SUSE Linux Enterprise Server 9 (SLES-9) and Red Hat Enterprise Linux 4 (RHEL-4) distributions, including:
  - Setting up routes
  - Adding additional qeth interfaces
  - Setting up the system as a router
  - Configuring qeth driver parameters
  - Adding additional DASD
  - Using boot menus
  - Configuring the initrd with additional modules
  - Using initialization scripts for starting and stopping services
- In general both distros are quite similar with respect to performing the configuration tasks.
- Target files and parameter names may differ. It is useful to become familiar with the scripts and configuration locations specific to each distribution.
- The graphical tools available for the two distros were not necessarily designed with the mainframe in mind. They often have limited support for the System z platform and devices. However, the yast tool from SUSE has been successfully used within our test organization for various tasks, including: setting up routes, adding interfaces, and configuring qeth parameters. Often, the tool has also been used to set up additional DASD, though this involves running some additional commands afterwards to make the disks mountable from /etc/fstab
- SUSE Linux Enterprise Server 10 is the latest offering from SUSE and provides some configuration changes from SLES-9. Read the release-notes and docu/en/*.pdf material from the installation CDs to learn more about changes and restrictions that might affect installation and boot up.

Appendix

- Links and other Sources
  - Device Driver manual
    http://download.boulder.ibm.com/ibmdl/pub/software/dw/linux390/docu/l26cdd01.pdf
  - FC attached SCSI
    http://download.boulder.ibm.com/ibmdl/pub/software/dw/linux390/docu/l26cts00.pdf
  - Virtualization Cookbooks:
    http://www.redbooks.ibm.com/abstracts/sg246695.html (SLES9)
    http://www.redbooks.ibm.com/abstracts/sg247272.html (RHEL4)
  - Release Notes and Documentation pdf files for SUSE Linux (available on the Service Pack and installation CDs)
    - docu/RELEASE-NOTES.en.html - release notes
    - docu/en/preparation.pdf - Preparation Manual
  - Red Hat Enterprise Linux Documentation (The documentation is provided in a separate documentation CD/DVD):
    - RH-DOCS/*/ or RH-DOCS/pdf/
      - ig – installation guide
      - rg – reference guide
      - isa – introduction to system administration
      - Sg – security guide
    - RH Knowledge Base: http://kbase.redhat.com/faq/
      - Ex: http://kbase.redhat.com/faq/FAQ_85_6247.shtm - “IP already in use” issue with RHEL-4
    - Red Hat Enterprise Linux documentation online:
      - http://www.redhat.com/docs/manuals/enterprise/

- Slides with Additional Information
  - zFCP SCSI disk configuration
  - Updating your system

Configuring zFCP SCSI on SUSE Linux

- Each FCP device will have its own configuration file:
  /etc/sysconfig/hardware/hwcfg-zfcp-bus-ccw-0.0.xxxx
- The wwpn:lun mapping is defined with the ZFCP_LUNS parameter
  - Ex.

    ZFCP_LUNS=“

    0x5005076300ccafc4:0x5735000000000000

    0x5005076300ceafc4:0x5735000000000000"
- Use zfcp_disk_configure to bring individual disks online
  - zfcp_disk_configure 0.0.0100 0x5005076300c1afc4 0x572a000000000000 1
- You can also use yast
- By default SUSE Linux has device files for /dev/sda through /dev/sdz

Configuring zFCP SCSI on Red Hat Enterprise Linux

- zFCP SCSI is set up in /etc/zfcp.conf
  - Ex.

    0.0.a210 0x01 0x5005076300c2afc4 0x00 0x572c000000000000

    0.0.a210 0x02 0x5005076300ceafc4 0x01 0x572d000000000000

- If you want the FCP devices to be available on boot up you need to run mkinitrd to get the configuration and needed modules into the initrd. Then you need to run zipl to pick up the new initrd and use it during boot up.
  - Ex.

    ```
    cd /boot
    mkinitrd -v --with=zfcp --with=sd_mod initrd.new `uname -r`
    ```

    - rename initrd.new as needed to match ramdisk entry in /etc/zipl.conf …

    ```
    zipl -V
    ```

- If you want to enable for current boot up, just run /sbin/zfcpconf.sh

Updating your system

- Determining your Service Pack level:
  - SUSE: use SPident tool:

    Ex.

    LTIC0008:~ # SPident

    CONCLUSION: System is up-to-date!
    found SLES-9-s390x-SP3

  - Red Hat: look at /etc/issue (this file is installed by the redhat-release rpm package)

    Ex.

    [root@LTIC0031 ~]# cat /etc/issue

    Red Hat Enterprise Linux AS release 4 (Nahant Update 3)

    Kernel \r on an \m

- Querying specific package levels using the rpm command
  - List all installed packages: `rpm -qa`
  - Show version-release for a particular package if installed, ex: `rpm -q glibc`
  - Query information on the package (including when the package was built): `rpm -qi glibc`
  - List packages according to when they were installed: `rpm -qa --last`

Updating your system

- Installing all the latest fixes from an online service (such as YaST Online Update (YOU) or Red Hat Network (RHN)).
  - SUSE:
    - use yast (“Online Update” module)
    - Available updates are listed according to patches (a patch being basically an instruction list and set of rpm packages addressed to fix a particular vulnerability or bug).
  - Red Hat:
    - use up2date
    - Available updates are listed by rpm package name.
- Installing an rpm package that was not on the system before
  - Either distro – copy the rpm package to the target system and use rpm command to install it.
  - SUSE specific
    - Configure the desired installation source on yast (“Change Source of Installation” module). Then use “Install and Remove Software” module to install the package
  - Red Hat specific
    - Run up2date to install the latest version of package from RHN
    - Or use system-config-packages to access an installation tree
      - Ex system-config-packages –tree=/rhel-4/U3/install
      - Requires an X display
      - Requires access to an install tree
      - Requires that the install tree have .discinfo copied to it.

Updating your system

- Updating to a specific service level using the latest Service Pack CDs
  - SUSE:
    - Service Pack CDs (ex. SP2, SP3) are not full installation CDs. They can be used to update installed packages on an existing system to the package levels available on the Service Pack CDs.
    - Use yast (“Patch CD Update” module)
    - “Patch CD Update” … sometimes requires latest version yast packages to recognize layout from the latest Service Pack install tree.
    - For example, to update from SLES-9 GA to SLES-9 SP3), you should update the following packages manually first (using packages from the target Service Pack), before running “Patch CD Update”:
      - yast2
      - yast2-core
      - yast2-online-update
      - yast2-packagemanager
      - yast2-packager
      - yast2-update
      - yast2-ncurses
      - yast2-qt
      - yast2-theme-SuSELinux (noarch package)
  - Red Hat:
    - Red Hat “Updates” (ex. U1, U2, U3) provide full installation CDs
    - Major upgrades (RHEL3 -> RHEL4) are supported by the installer. A minor upgrade from the previous update level (RHEL 4 U2 -> RHEL 4 U3) is supported via up2date using a dir-style repository (using the fully merged RedHat/RPMS/*.rpm from all Installation CDs).
      - Check out /etc/sysconfig/rhn/sources on how to specify the dir-style repository.
      - Call up2date with the channel name specified from the entry for your local directory repository:
        - `up2date --channel=my-favorite-rpms -u`

Updating System for Red Hat and SUSE Linux

Red Hat

- Online package update is done with up2date/RHN and available updates are listed by rpm package.

- RHEL Update CDs can be used:
  - for a full installation via anaconda
  - for dir-style repository upgrades using up2date on a fully merged binary RPM directory

SUSE

- Use yast/yast2 (“online update” module). Available fixes are displayed according to “patch” listings.

- SUSE Service Pack CDs contain only fixes. SUSE supports updating to specific service pack release with yast “Patch CD update”

