Building Linux Systems under IBM VM

Simon Williams

Table of Contents

About the author

Simon Williams is a Senior I/T Specialist working in the Enterprise Server Group of IBM Australia. His responsbility is technical support of "new technologies" on the S/390 and zSeries platforms. These technologies include Linux/390, WebSphere, Domino and Unix System Services. Before joining IBM in 1997 he was a Senior MVS Systems programmer with the Australia Federal Government.

Simon can be contacted at: `simonwil@au1.ibm.com`

1. Building Linux Systems under IBM VM

1.1. The goal

In this guide we demonstrate just one of many approaches to building multiple Linux systems as Virtual Machine Guests running under the IBM VM/ESA and z/VM operating systems.

1.2. Background

VM/ESA and z/VM (hereafter refered to as VM) are IBM operating systems that enable other operating systems to run as guests. Depending on the resources of the physical machine it is possible to run hundreds or thousands of Linux servers under VM.

1.3. Assumptions

* It is assumed that you are going to install Linux/390 on a supported level of hardware and software. Please review all documentation that comes with your Linux/390 distribution.

* It is assumed that you are the site's VM systems programmer or equivalent and have MAINT access to the VM system. Alternatively, you could be a Linux administrator working in conjunction with the site's VM systems programmer.

* It is assumed that all necessary devices, including the appropriate console, DASD, and communications devices, have already been defined to the VM system.

* This paper currently uses SuSE 7.0 Linux/390 for all examples. TurboLinux and RedHat will be added in future versions of this document.

* In this paper we use Virtual Channel to Channel devices to communicate with the Linux TCP/IP stack. There are several other methods for communicating with Linux. Other communications methods will be added in future versions of this document.

* It is assumed that all prerequisite maintenance for VM and the processor hardware has been installed.

2. Steps to build Linux systems as VM Guests

The following steps are required to successfully build Linux systems as VM guests:

- Format and label DASD

- Create user definitions

- Set up TCP/IP definitions (see Appendix B for complete details)

- Build the first Linux instance

- Create additional Linux instances

- Tailor Linux instances

A detailed discussion of these steps follows.

2.1. Format and label DASD

Note: It is assumed that the DASD has been previously initialized using ICKDSF INSTALL.

We need to allocate disk drives (known as DASD or Direct Access Storage Devices) for use by the Linux VM guests.

VM employs the concept of minidisks, whereby physical DASD can be split into smaller sizes, called minidisks, and assigned to individual VM guest systems.

In this guide we take the approach of dedicating whole DASD volumes for Linux systems. However, these devices are split up into minidisks owned by individual Linux guests.

To be on the safe side, we recommend initializing and labelling all the Linux DASD with a volume serial that makes it obvious they are Linux disks. If one of these disks is inadvertently varied online to another LPAR on the machine it should be obvious that the disk belongs to the Linux environment.

One example of a volume serial naming convention is: LInnnn

LI identifies this disk as a Linux disk.
nnnn is the device number of the volume.

2.1.1. Steps to format and label a DASD volume

```
ATT 63B TO * AS 163B                                          (1)
CPFMTXA                                                       (2)


ENTER FORMAT, ALLOCATE, LABEL, OR QUIT:
FORMAT                                                        (3)
ENTER THE VDEV TO BE PROCESSED OR QUIT:
163B                                                          (4)


ENTER THE CYLINDER RANGE TO BE FORMATTED ON DISK 163B OR QUIT:
0000 0000                                                    (5)
ENTER THE VOLUME LABEL FOR DISK 163B:
LI0001                                                       (6)
ICK00002I ICKDSF PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
ENTER ALLOCATION DATA
TYPE CYLINDERS
.................
PERM 0000-END                                                (7)


END
```

Description of the process:

1. Attach physical device 63B to ourselves as virtual device 163B.

2. Invoke the CP format utility.

3. We want to format the volume.

4. The virtual device address.

5. We want to label the volume. Formatting cylinder 0000 is a quick way of accomplishing this.

6. The volume label we want to give this device.

7. PERM indicates that the corresponding cylinders are to be used to allocate user minidisks.

Once the physical packs have been initialized, you should define the VM guest users. As part of the guest definition you will define minidisks, these being subareas of the physical disks you've just initialized.

2.2. Create user definitions

The VM user directory describes the configuration and operating characteristics of each VM guest. Note: This paper does not cover the Directory Maintenance program.

Edit the User Directory file with xedit as follows:

X   USER DIRECT

Note: *You must be logged on as MAINT to do this.*

If certain directory control statements are repeated for several users, you can make use of directory profiles to save space in the directory. Since we can potentially create many hundreds of Linux guest systems on a single VM, we create a profile for Linux guests to define the things they have in common:

```
PROFILE LINDFLT                    (1)
  SPOOL 000C 2540 READER           (2)
  SPOOL 000D 2540 PUNCH A
  SPOOL 000E 1403 A
  CONSOLE 009 3215 T               (3)
  LINK MAINT 0190 0190 RR          (4)
  LINK MAINT 019D 019D RR
  LINK MAINT 019E 019E RR
```

This profile comprises the following specifications:

1. The name of the Profile is LINDFLT.

2. The SPOOL statement defines virtual unit record devices. In this profile we have defined virtual reader, punch, and printer devices

3. The CONSOLE statement defines the virtual machine console.

4. We define links to the MAINT user's minidisks. These minidisks contain common commands and utilities that most guests require.

2.2.1. Create first Linux user

We create the first Linux user at this point. This user is slightly different from subsequent ones since it will have a minidisk definition for the */usr* filesystem that will be shared with all other Linux VM Guests.

```
USER LI2000 LI2000 128M 2047M G                          (1)
INCLUDE LINDFLT                                          (2)
IPL CMS PARM AUTOCR                                      (3)
MACHINE ESA                                              (4)
SPECIAL 2000 CTCA VMIP                                   (5)
SPECIAL 2001 CTCA VMIP
MDISK 191 3390 0001 0050 LI0313 MR ALL SOME FEW
MDISK 200 3390 0001 0200 LI0319 MR ALL SOME FEW
MDISK 201 3390 0202 2800 LI0319 MR ALL SOME FEW
MDISK 202 3390 0001 3300 LI0339 MR ALL SOME FEW
```

The entries that make up the VM Guest definition in USER DIRECT have the following meanings:

1. The user ID of this virtual machine is LI2000. The password is LI2000. The virtual machine has 128MB defined as the central storage size at logon. You can increase the storage size up to 2047MB using the DEFINE STORAGE command (this is a disruptive change).
    Before increasing the storage size above 128MB, it is important to analyze your VM paging environment and the characteristics of your Linux system's memory use. This analysis is outside the scope of this document.

2. We include the LINDFLT profile (see above).

3. CMS will boot automatically at logon (required for installation of Linux).

4. Sets the virtual machine to run in ESA mode.

5. Virtual CTCs are defined as this Linux system will communicate using them.

Each MDISK statement creates a minidisk for a VM guest user.

We have defined the following minidisks for the first Linux user:

Minidisk	DASD type	Size in cyls	Real DASD	Usage
191	3390	50	LI0313	Guest "home" disk. Used for PROFILE EXEC
200	3390	200	LI0319	Linux swap
201	3390	2800	LI0319	/ (root filesystem, one per Linux instance)
202	3390	3300	LI0339	/usr (shared among all guests)

Note: The 202 minidisk is initially defined as mounted READ/WRITE. Once the first Linux instance has been installed we will change this to READ-ONLY so that this MINIDISK can be shared among all Linux instances.
It is important to note that if the /usr filesystem is shared you cannot re-compile the kernel using the existing directory structure (/usr needs to be mounted read-write).

The maximum space on a single 3390 Model 3 is 3339 cylinders. This equates to roughly 2.8 GB. However, when the DASD is formatted to use the *ext2* filesystem under Linux you get around 2.3 GB of usable space.

It is recommended that you map out how much disk storage you wish to use for each Linux instance.

The rule of thumb for the SuSE 7.0 installation is as follows:

SuSE Almost everything (822 M- 2.73 G)

SuSE Development system (315 M- 1.16 G)

SuSE DMZ base system (92 - 303.7 M)

SuSE Minimum system (72 - 154.2 M)

SuSE Network oriented system (335 - 923.1 M)

SuSE Default system with office (242 - 785.6 M)

Note: *The amount of storage required varies depending on the type of filesystem you choose to install the code. To determine exactly what packages are installed in each of the configurations, please review the *.sel files, which reside in /suse/setup.descr on the first SuSE CD.*

2.2.2.   Create additional Linux user IDs

You should then create as many additional Linux VM Guest users as required.

```
USER LI2010 LI2010 128M 2047M G
 INCLUDE LINDFLT
 IUCV ANY
 IUCV ALLOW
 IPL CMS PARM AUTOCR
 MACHINE ESA
 SPECIAL 2000 CTCA VMIP
 SPECIAL 2001 CTCA VMIP
 LINK LI2000 0202 0202 RR
 MDISK 191 3390 0052 0050 LI0313 MR ALL SOME FEW
 MDISK 200 3390 0001 0200 LI0620 MR ALL SOME FEW
 MDISK 201 3390 0202 2800 LI0620 MR ALL SOME FEW
```

For the additional Linux user we have created the following minidisks:

Minidisk	DASD type	Size	Real DASD	Usage
191	3390	50	LI0313	Guest "home" disk
200	3390	200	LI0620	Linux swap
201	3390	2800	LI0620	/ (root filesystem, one per Linux instance).

Note: *We link LI2000's 202 minidisk read-only and use it as our 202 minidisk. The 202 minidisk as previously described is used by the shared /usr filesystem.*

Add additional users as needed, remembering to calculate the disk space use for each system.

Once you've added all the Linux systems you require, save the USER DIRECT file and run the following command from the CMS command line:

DISKMAP USER

DISKMAP summarizes the MDISK statements in the user directory. The output produced by DISKMAP shows gaps and overlaps between minidisk assignments.

Once this has run, you should verify that the disk allocations do not conflict. Do this by reviewing the newly created USER DISKMAP file.

Check for any overlaps by editing USER DISKMAP and do a search on the string "overlap", as follows:

X   USER DISKMAP   (open USER DISKMAP with the editor)

/overlap   (perform a search on the string overlap)

If you find any overlaps, correct them and issue these commands again. When the allocations are correct, run the command:

DIRECTXA USER (EDIT

If necessary, correct any syntax errors and issue this command again. Otherwise, if all is well, run:

DIRECTXA USER

This will load the updated User Directory into VM, so you can now log on to the newly created users.

Note: *It is vital that you review Appendix B. "TCP/IP definitions" at this point, understand the definitions you have already completed, and perform any additional ones before continuing, in order to ensure that you have TCP/IP connectivity between Linux guests and the VM TCP/IP stack.*

2.3. Build the first Linux instance

We now build the initial Linux system, which we will then use to clone additional Linux instances.

Log on to the LI2000 user ID. Minidisk A (device 191) won't be formatted yet, so:

FORMAT 191 A

The output from this command is:

```
DMSFOR603R FORMAT will erase all files on disk B(191). Do you wish to continue?
Enter 1 (YES) or 0 (NO).
1
DMSFOR605R Enter disk label:
LIN191
DMSFOR733I Formatting disk A
DMSFOR732I 50 cylinders formatted on A(191)
```

You should then create a PROFILE EXEC file, (think of it as an autoexec.bat for your VM guest user). Here's an example:

```
/* VM LINUX Profile exec */
/* trace ira  */
'VMFCLEAR'
'CP SET MSG ON'
'CP SET EMSG ON'
'SET IMPCP ON'
'SET RUN ON'
'CP TERM CHARDEL OFF'
'VMLINK TCPMAINT 592 ( NONAMES'
'CP SET PF10 RETR'
'CP SET RETR MAX'
'CP TERM HOLD OFF'
/*  'CP TERM MORE 1 1'  */
'CP COUPLE 2000 VMIP ' '20' || substr(userid(),5,1) || '1'
'CP COUPLE 2001 VMIP ' '20' || substr(userid(),5,1) || '0'
```

Note: *As you can see from the above PROFILE, we have commented out the line 'CP TERM MORE 1 1'. Remove the comments after you are comfortable with booting Linux. It will allow you to boot Linux without having to <CLEAR> the screen each time it fills with boot messages.*

Once you've created this file, run the PROFILE EXEC by typing the command:

PROFILE

In order to install the first Linux instance you must go through a number of steps.

• Upload a copy of the Linux kernel, parmfile and initial ramdisk to the VM system.

• Boot Linux using the VM Reader.

• Build the first Linux instance using YaST.

• Reboot using the newly installed Linux system (i.e. IPL from a minidisk instead of the VM Reader).

2.3.1. Upload Linux IPL material to VM

Locate the LINUX for S/390 boot files (on the SuSE 7.0 for S/390 CD number 1) and transfer them to minidisk A (device 191). The following commands show you how to do this:

```
FTP 9.185.246.25                                                    (1)
VM TCP/IP FTP Level 320
Connecting to 9.185.246.25, port 21
220 linux390 FTP server (Version wu-2.4.2-VR17(1) Thu May 18 03:18:13 EDT 2000)
ready.
USER (identify yourself to the host):
simonw                                                             (2)
>>>USER simonw
331 Password required for simonw.
Password:                                                          (3)
>>>PASS ********
230 User simonw logged in.
Command:
bin                                                                (4)
>>>TYPE i
200 Type set to I.
locsite fix 80                                                     (5)
Command:
get /suse/images/vmrdr.ikr   suse.image      (this is the Linux kernel image)    (6)
>>>PORT 9,185,246,26,4,5
200 PORT command successful.
>>>RETR /sues/images/vmrdr.ikr
150 Opening BINARY mode data connection for vmrdr.ikr (1511884 bytes).
226 Transfer complete.
1511920 bytes transferred in 1.972 seconds. Transfer rate 766.69 Kbytes/sec.
```

```
Command:
get /suse/images/initrd   suse.initrd        (this is an initial RAMdisk file, it is          (7)
                                             used as a root file system on initial boot)

>>>PORT 9,185,246,26,4,7
200 PORT command successful.
>>>RETR /suse/images/initrd
150 Opening BINARY mode data connection for initrd (9862505 bytes).
8299680 bytes transferred.
226 Transfer complete.
9862560 bytes transferred in 12.366 seconds. Transfer rate 797.55 Kbytes/sec.
Command:
asc                                                                                  (8)
>>>TYPE a
200 Type set to A.
Command:
get /suse/images/parmfile   suse.parm        (this is the boot parameter file)            (9)
>>>PORT 9,185,246,26,4,8
200 PORT command successful.
>>>RETR parmfile
150 Opening ASCII mode data connection for parmfile (38 bytes).
226 Transfer complete.
40 bytes transferred in 0.081 seconds. Transfer rate 0.49 Kbytes/sec.
Command:
quit
```

An explanation of the process:

1. Use FTP to connect to the server where the files are located.

2. Logon with your user ID.

3. Enter your password.

4. Set binary file transfer.

5. Set the record length of the transferred files to 80 bytes and the record format to fixed.

6. Copy the kernel file (/suse/images/vmrdr.ikr) to SUSE IMAGE A.

7. Copy the initial ramdisk file (/suse/images/initrd) to SUSE INITRD A.

8. Set ASCII file transfer.

9. Copy the parm line file (/suse/images/parmfile) to SUSE PARM A.

Note: *You might need to modify the values in the parameter line file to suit your environment.*

2.3.2.   Creating LIN EXEC

In order to easily boot the Linux system using the VM Reader we now create a REXX program.

Create file LIN EXEC on minidisk A.

> **X LIN EXEC**

This REXX program loads the kernel into the VM reader and also boots the kernel from the reader.

```
/**/
'close rdr'
'purge rdr all'
'spool punch * rdr'
'PUNCH SUSE IMAGE A (NOH'
'PUNCH SUSE PARM A (NOH'
'PUNCH SUSE INITRD A (NOH'
'change rdr all keep nohold'
'ipl 00c clear'
```

These commands have the following meanings:

/**/	Informs the system that the file is a REXX executable.
'close rdr'	Closes all open files in the reader so that they can be purged.
'purge rdr all'	Empties the VM reader. You should ensure that any important reader files have been moved to another location before issuing this command!
'spool punch * rdr'	Directs the output of the punch device to the reader.
'PUNCH SUSE IMAGE A (NOH'	Moves the LINUX boot file to the reader.
'PUNCH SUSE PARM A (NOH'	Moves the LINUX parameters file to the reader.
'PUNCH SUSE INITRD A (NOH'	Moves the initial RAMdisk file (initial root file system) to the reader.
'change rdr all keep nohold'	Makes sure the content of the reader is not changed or deleted after the process is finished.
'ipl 00c clear'	Sends the reader an Initial Program Load (IPL) command — this boots LINUX for S/390.

2.3.3.   Boot Linux using the VM READER

You are now ready to boot the initial Linux system using the command:

    LIN

You'll see a whole lot of messages fly past on the console (remember to hit the <CLEAR> key to clear the screen).

For example:

```
0000001 FILE  PURGED
RDR FILE 0002 SENT FROM LI2010  PUN WAS 0002 RECS 019K CPY  001 A NOHOLD NOKEEP
RDR FILE 0003 SENT FROM LI2010  PUN WAS 0003 RECS 0001 CPY  001 A NOHOLD NOKEEP
RDR FILE 0004 SENT FROM LI2010  PUN WAS 0004 RECS 123K CPY  001 A NOHOLD NOKEEP
0000003 FILES CHANGED
0000003 FILES CHANGED


Linux version 2.2.16 (root@ikr_rdr.suse.de) (gcc version 2.95.2 19991024 (release)) #1 SMP Fri
Nov 3 09:38:59 GMT 2000
Command line is: ramdisk_size=32768 root=/dev/ram0 ro
We are running under VM
This machine has no IEEE fpu
Initial ramdisk at: 0x02000000 (9862560 bytes)
Detected device 2000 on subchannel 0000 - PIM = 80, PAM = 80, POM = FF
Detected device 2001 on subchannel 0001 - PIM = 80, PAM = 80, POM = FF
Detected device 0202 on subchannel 0002 - PIM = C0, PAM = C0, POM = FF
Detected device 0191 on subchannel 0003 - PIM = C0, PAM = C0, POM = FF
Detected device 0200 on subchannel 0004 - PIM = C0, PAM = C0, POM = FF
Detected device 0201 on subchannel 0005 - PIM = C0, PAM = C0, POM = FF
Detected device 000C on subchannel 0006 - PIM = 80, PAM = 80, POM = FF
Detected device 000D on subchannel 0007 - PIM = 80, PAM = 80, POM = FF
Detected device 000E on subchannel 0008 - PIM = 80, PAM = 80, POM = FF
```

Eventually you will see the welcome text for SuSE Linux:

```
=                                                               =
==-                Welcome to SuSE Linux S/390 7.0                 -==
=                                                               =


First, select the type of your network device:
0) no network
1) OSA Token Ring
2) OSA Ethernet
3) OSA-Express Gigabit Ethernet (experimental)
4) Channel To Channel
5) Escon
6) IUCV (experimental)
Enter your choice (1-6):
4
```

You should then select the relevant networking device that you're going to use.

Network setup

Select your type of network (*in our example we use CTC devices*)

The following messages will be displayed (note: user responses to the prompts are in **bold**):

```
Please enter the options for the CTC device driver now.
Specify which channel addresses Linux should use (recommended)
or enter 'auto' to autosense all available channel devices,
but be careful if you share channel devices!

ctc options:
Device address for read channel (e.g. 40A) or 'auto': auto
Unloading ctc module (if already loaded)
rmmod: module ctc is not loaded
Trying to start the ctc module now...
insmod -v ctc  :
Using /lib/modules/2.2.16/misc/ctc.o
Symbol version prefix 'smp_'
CTC driver Version: 1.25.2.5  initialized
channel: 2 Parallel channel found - 0 ESCON channel found
ctc0: read dev: 2000 irq: 0000 - write dev: 2001 irq: 0001

lsmod now shows all loaded modules:
ctc                21200   0 (unused)

Ensure that the device you want to use is detected as ctc0
Was the loading of "ctc" successful (Yes/No) ? yes

Please enter your full host name (e.g. s390.suse.com): li2000.system.au.ibm.com
Please enter your IP address: 9.185.246.231
Please enter the IP address of your peer: 9.185.246.28
Please enter the IP address of the DNS server (leave blank for none): 9.185.246.30
Please enter the DNS search domain (e.g. suse.com): system.au.ibm.com
Please enter the MTU (Maximum Transfer Unit, leave blank for default: 2000):
1492
Configuration for ctc0 will be:
Full host name   : li2000.system.au.ibm.com
IP address       : 9.185.246.231
Peer IP address  : 9.185.246.28
DNS IP address   : 9.185.246.30
DNS search domain: system.au.ibm.com
MTU size         : 1492
Is this correct (Yes/No) ? yes
```

For security reasons you have to set an temporary installation
system password for the user "root".
You'll be asked for it only when you telnet into this installation
system to limit the access to it and it will be cleared as soon
as you shut down or reset the installation system

Please enter the temporary installation password: **guiness**
Temporary installation password set.
restarting syslogd:
Jan 5 06:58:46 li2000 syslogd 1.3-3: restart.
ifconfig ctc0 9.185.246.231 pointopoint 9.185.246.28 mtu 1500

ctc0: connected with remote side
/sbin/ifconfig ctc0 :
ctc0   Link encap:Serial Line IP
         inet addr:9.185.246.231 P-t-P:9.185.246.28 Mask:255.255.255.255
         UP POINTOPOINT RUNNING NOARP MTU:1500 Metric:1
         RX packets:0 errors:0 dropped:0 overruns:0 frame:0
         TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
         colli0ions:0 txqueuelen:100

Trying to ping my IP address:
PING 9.185.246.231 (9.185.246.231): 56 data bytes
64 bytes from 9.185.246.231: icmp_seq=0 ttl=255 time=0.583 ms
64 bytes from 9.185.246.231: icmp_seq=1 ttl=255 time=0.308 ms
64 bytes from 9.185.246.231: icmp_seq=2 ttl=255 time=0.443 ms
 --- 9.185.242.235 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.308/0.444/0.583 ms

Trying to ping the IP address of the peer:
PING 9.185.246.28 (9.185.246.28): 56 data bytes
64 bytes from 9.185.246.28: icmp_seq=0 ttl=60 time=1.857 ms
64 bytes from 9.185.246.28: icmp_seq=1 ttl=60 time=1.548 ms
64 bytes from 9.185.246.28: icmp_seq=2 ttl=60 time=1.597 ms
 --- 9.185.246.28 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 1.548/1.667/1.857 ms
Peer seems to respond to our pings, continuing.

Trying to ping the IP address of the DNS Server:
PING 9.185.246.30 (9.185.246.30): 56 data bytes
64 bytes from 9.185.246.30: icmp_seq=0 ttl=254 time=11.999 ms
64 bytes from 9.185.246.30: icmp_seq=1 ttl=254 time=7.911 ms
64 bytes from 9.185.246.30: icmp_seq=2 ttl=254 time=8.114 ms

```
 --- 9.185.246.30 ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 7.911/9.341/11.999 ms
Network Setup finished, running inetd...

SuSE Instsys li2000:/ #
```

You can now telnet to 9.185.246.231 and start the installation program YaST.

2.3.4. Build the first Linux instance using YaST

Note: *A telnet session requirement -- The telnet session must be at least 80 x 25 lines.
An OS/2 telnet session, for example, opens by default with only 24 lines. To navigate
on the telnet screen, use the arrow keys to move up and down in selection lists, and the
tab key to select actions.*

*The function keys do not always map correctly in YaST. For example, F6 might be
interpreted as F5. Use the numerical and punctuation keys (1–0) instead, for example
instead of F1 use 1 and instead of F10 use 0. (The cursor must be on a non-input field
for this to work.)*

When you telnet to the LINUX system you must login as **root**; the password is the
temporary password you entered earlier.

For example:

    li2000 login: **root**

    Password: **guiness**

The following screen is displayed:

```
>>> >>> >>> >>> >>> >>>  SuSE Linux S/390 7.0  <<< <<< <<< <<< <<< <<<


  1. If you want to check which devices the dasd driver can see,
     run 'insmod dasd probeonly' (may take long) and check
     the output of 'cat /proc/dasd/devices'.
     Remove the dasd driver with 'rmmod dasd' afterwards.

  2. Choose the device numbers you want to use for SuSE Linux S/390
       !!!   BE CAREFUL WHEN SELECTING DASDs -   !!!
       !!! YOU MAY DESTROY DATA ON SHARED DEVICES !!!

  3. Enter 'insmod dasd dasd=<li0t of devices>'
     Remember to separate devices by commas (<dev_no>,<dev_no>),
     syntax for ranges is <from_dev_no>-<to_dev_no>
     like
         'insmod dasd dasd=FD00-FD0F,FD40,FD42,FD80-FD86'

     Note: When updating, you have to load the dasd driver with the
     same DASD order as in the installed system  - see documentation
     for further information.

  4. Start installation or update with  'YaST'.
```

Before installing Linux, you need to identify which disk devices you will be using (i.e. the minidisk devices, not the physical 3390 devices).

You should run: **insmod dasd probeonly**

Follow this with: **rmmod dasd**

This will generate a list of all online DASD devices. To review the list check /proc/dasd/devices.

For example:

```
SuSE Instsys li2000:/root # cat /proc/dasd/devices
0400(none) at (94:0) is   dasda:unknown
0401(none) at (94:4) is   dasdb:unknown
0191(ECKD) at (94:8) is   dasdc:active  at blocksize: 4096, 9000 blocks, 35 MB
0200(ECKD) at (94:12) is   dasdd:active  at blocksize: 4096, 36000 blocks, 140 MB
0201(ECKD) at (94:16) is   dasde:active  at blocksize: 4096, 504000 blocks, 1968 MB
0202(ECKD) at (94:20) is   dasdf:active  at blocksize: 4096, 594000 blocks, 2320 MB
000C(none) at (94:24) is   dasdg:unknown
000D(none) at (94:28) is   dasdh:unknown
000E(none) at (94:32) is   dasdi:unknown
0009(none) at (94:36) is   dasdj:unknown
0190(ECKD) at (94:40) is   dasdk:active  at blocksize: 4096, 19260 blocks, 75 MB
019D(ECKD) at (94:44) is   dasdl:active  at blocksize: 1024, 50490 blocks, 49 MB
019E(ECKD) at (94:48) is   dasdm:active  at blocksize: 4096, 12060 blocks, 47 MB
0120(ECKD) at (94:52) is   dasdn:active  at blocksize: 4096, 12060 blocks, 47 MB
```

You can see devices 0200, 0201, and 0202 - we defined these minidisks earlier when we edited the VM user directory.

You should now select which devices you wish to use for the Linux install.

To incorporate these devices into the running Linux system you should enter the command:

insmod dasd dasd=from_Device_number-to_Device_number

In our example we use:

insmod dasd dasd=0200-0202

Remember, in our example we are using minidisk:

> 0200 for the swap device
>
> 0201 for the root filesystem (/)
>
> 0202 for the /usr filesystem

- Start the Linux installation program by entering:

    yast

- On the language selection panel, use the arrow keys to choose a language, for example English.

- On the "SELECTION OF THE INSTALLATION MEDIUM" panel, select "Installation from an FTP site".

Note: In our example we install SuSE from a distribution stored on an FTP server. Other valid installation methods include using NFS and SMB.

- On the "TYPE OF INSTALLATION" panel, select "Install Linux from scratch".

- On the "SELECT SWAP PARTITION" panel, if you have a DASD definition like the one used in this guide you should see something like this:

```
    Device name  Blocks

    dasd(0200)  143988
    dasd(0201)  2015988
    dasd(0202)  2375988
    <no swap-partition>
```

- Select 0200 as the SWAP device.

- On the "PARTITION HARDDRIVES" panel, select "Do Not Partition."

- You'll then see a "CREATING FILESYSTEMS" panel.

- We want to set up device 201 as the root filesystem and device 202 as the /usr filesystem.

- Select dasd(0201), then press "4" to change the mount point to   /   (the root filesystem).

- Then select "6" (option "Normal Format") to flag that filesystem as requiring a format.

- Select dasd(0202) then press "4" to change the mount point to /usr.

- Then select "6" (option "Normal Format") to flag that filesystem as requiring a format.

- Select Continue to begin formatting. You'll then see the "CREATING FILESYSTEMS" panel. This will take a little while.

2.3.5.   Set up the link to the FTP server and get the packages

Fill in the data for your FTP server and press **F5** to check the connection. Watch for the words **Settings OK** at the end of the messages.

```
FTP Settings YaST Version 1.07.1 --(c)1994-2000 SuSE GmbH


FTP Settings


FTP Server [Name|IP]  :        9.185.246.25
Server directory     :             /root/suse7
[ ] Use Proxy?
Proxy [Name|IP]      :           (no proxy)
[X] Default FTP Port?
Port [Number]        :                   21
[ ] Anonymous FTP?
Login            :            root
Password        :            ********
Timeout [Seconds]    :        60
```

If the connection is OK, press **F10** to start loading files from the FTP Server.

2.3.6.   Installing the packages from the FTP server

On this screen, select **Load configuration**:

A number of configurations are available to be loaded:

SuSE Almost everything	(822 - 2.73 G)
SuSE Development system	(315 - 1.16 G)
SuSE DMZ base system	(92 - 303.7 M)
SuSE Minimum system	(72 - 154.2 M)
SuSE Network oriented system	(335 - 923.1 M)
SuSE Default system with office	(242 - 785.6 M)

We shall select "SuSE Network oriented system."

• Start the installation from the FTP server by selecting **Start installation**.

• Preselected packages will be copied over. The status line at the top of the screen tells you how many packages remain to be copied

The **INSTALLATION COMPLETE** message tells you that the installation of the packages is now finished (press the Escape key to continue).

• The "SELECT KERNEL" panel is now displayed.

*Note: The destination of the .config file defaults to **/usr/src/linux**. Since we will be sharing the /usr filesystem across all Linux instances but need to customize the .config file for each one, we should change its destination to **/boot**.*

Select F3 to change the destination of the .config file to /boot.

• Once the kernel has been installed you will be prompted to enter the appropriate timezone.

• Finally you will be prompted for networking information - you should be able to use the same information that you entered during the initial install.

• Finally, edit the /etc/fstab file. Change the /usr filesystem so it is mounted read-only.

For example:

Before:	/dev/dasdc1	/usr	ext2	**defaults**	1	2
After:	/dev/dasdc1	/usr	ext2	**ro**	1	2

- Shut down the Linux system using the command:

  halt

- Once the Linux system has been halted, re-ipl CMS using the command:

  CP   IPL   CMS

- Before booting Linux directly from minidisk, it would be a good idea to edit the USER DIRECT file again, using the MAINT logonid. Change LI2000's 202 disk to be READ-ONLY (from READ/WRITE).

Before:

MDISK 202 3390 0001 3300 LIS339 **MR** ALL SOME FEW

After:

MDISK 202 3390 0001 3300 LIS339 **RR** ALL SOME FEW

- Log off and log on LI2000 to pick up the changes.
- You can now boot your new Linux system directly from the minidisk (that is, you no longer need to use the VM READER) as follows:

  IPL   201 CLEAR

- Once you have booted Linux, log on using the root user.
- We will shortly begin cloning this Linux system to make additional Linuxes. As part of the cloning process we need to change the identity of the target Linux instance from that of the source instance.

  We will install a publicly available package called **rpl** and use it to perform global search/replace operations on several system files.

  You should now download the **rpl** package available from:

  http://www.laffeycomputer.com/rpl.html

- Untar the package and follow the installation instructions.

2.4.   Creation of additional Linux instances

Once the first Linux system has been installed, additional Linux instances can now be created. Instead of using the approach we took for the first installation (booting from the VM Reader, running YaST, etc.) we can simply clone the first Linux instance.

The approach is as follows:

- Copy the PROFILE EXEC from LI2000 to LI2010.

- Copy the root filesystem from LI2000 to LI2010.

- Boot the new Linux instance.

- Tailor the new instance so that it has a unique identity (TCP/IP networking, etc).

- Create a swap file for the new system.

- Copy the PROFILE EXEC from LI2000 to LI2010. The PROFILE EXEC can be thought of as equivalent to an autoexec.bat under DOS. Among a number of other setup definitions, it connects this VM Guest's TCP/IP Virtual CTCs to those of the VM TCP/IP stack. From the CMS command line enter the following commands, one at a time.

```
LINK LI2000 191 192 RR

AC 192 B

FILEL * * B

COPY PROFILE EXEC B PROFILE EXEC A
```

Execute the PROFILE EXEC by typing "PROFILE" from the command line

- Copy the root filesystem from LI2000 to LI2010. We shall use the DDR utility to do this. DDR is often available as a "public" utility and as such the next step can be run from the LI2000 user. If you find that DDR is not available you can run it using the MAINT logonid.

Before running DDR (from LI2000) make sure that the LI2010 user is not logged on. Also ensure that the LI2000 Linux system was halted cleanly - the DDR utility copies byte for byte and as such any inconsistencies in an improperly shutdown system would be propagated during a copy.

```
LINK LI2010 201 502 W          link LI2010's 201 minidisk as 502


You'll be prompted for the password (which is "FEW" in our example):


ENTER WRITE PASSWORD:
FEW


Ready; T=0.01/0.01 05:25:25


DDR
VM/ENTERPRISE SYSTEMS ARCHITECTURE DASD DUMP/RESTORE PROGRAM
ENTER:
SYSPRINT CONS
ENTER:
INPUT 201 DASD
ENTER:
OUTPUT 502 DASD
ENTER:
COPY ALL
HCPDDR711D VOLID READ IS
DO YOU WISH TO CONTINUE?  RESPOND YES, NO OR REREAD:
YES
HCPDDR716D NO VOL1 LABEL FOUND
DO YOU WISH TO CONTINUE?  RESPOND YES, NO OR REREAD:
YES
COPYING
COPYING DATA  01/15/01 AT 10.28.05  GMT FROM
INPUT CYLINDER EXTENTS     OUTPUT CYLINDER EXTENTS
 START    STOP        START      STOP
00000000  00002799   00000000  00002799
END OF COPY
```

- Copying will take some time to complete, depending on the size of the disk being copied and the speed of the DASD.

- Log on to LI2010 and IPL the newly copied Linux system to check that the copy operation worked.

    IPL 201 CLEAR

Note: At this stage TCP/IP networking will not work as you are using the IP address of the first Linux instance. You need to tailor your newly-built instance.

- Log on to the Linux instance (from the console you just IPLed Linux from).

  Run the **rpl** program that we built in an earlier step. We use this program to check for all instances of the LI2000 IP host name and LI2000's IP address. These will then be changed to the unique values for LI2010.

  The commands to use (in our example):

```
rpl    li2000    li2010 *

rpl    246.231    246.232 *
```

- Create a swap file for the new system.

  dasdfmt -b 4096 -n 200   (we use minidisk 200 for the swap disk)

  mkswap /dev/dasda1   (make /dev/dasda1 as a swap device)

  swapon -a   (make all devices marked as swap available)

  swapon -s   (display the swap devices)

- Your system is now ready to be rebooted using:

  halt

- Re-boot your Linux system (it will now have it's own identity and networking connectivity).

  IPL   201 CLEAR

Appendix A   Creating a new disk on Linux/390

From VM:

Add a minidisk definition to the VM guest system by adding an entry to the user directory file.   For example:

```
MDISK 205 3390 0001 3300 LI0600 MR ALL SOME FEW
```

From Linux:

Edit the /boot/parmfile adding the new dasd (the minidisk number) to the end of the list. For example:

```
dasd=0200,0201,0202,0205 root=/dev/dasdb1 noinitrd
```

Run the silo command to prepare the root DASD device to be IPLable (this will effectively incorporate the changes you've just made to the /boot/parmfile).

```
silo  -d  /dev/dasdb  -p  /boot/parmfile  -f  /boot/image  -b /boot/ipleckd.boot
```

Reboot Linux

You now need to format the newly created filesystem.

```
dasdfmt -b 4096 -n 205
```

Once that's done, make a filesystem on the newly formatted disk:

```
mke2fs -b 4096 /dev/dasde1
```

Then after you've done that you should make a mountpoint. A mountpoint is a point in the filesystem tree structure where you wish to mount a filesystem. For example:

```
mkdir  /home/zaphod
```

You can now mount the newly created filesystem, it is now available to store files:

```
mount   /dev/dasde1   /home/zaphod
```

Appendix B   TCP/IP definitions

In this document we use virtual CTC devices to communicate between a Linux instance's TCP/IP stack and the VM TCP/IP stack. Connection to external networks is made through a network interface card (an OSA2 card) defined to the VM TCP/IP stack.   There are several other means of connecting Linux to an IP network, this is but one of them.

Example Network
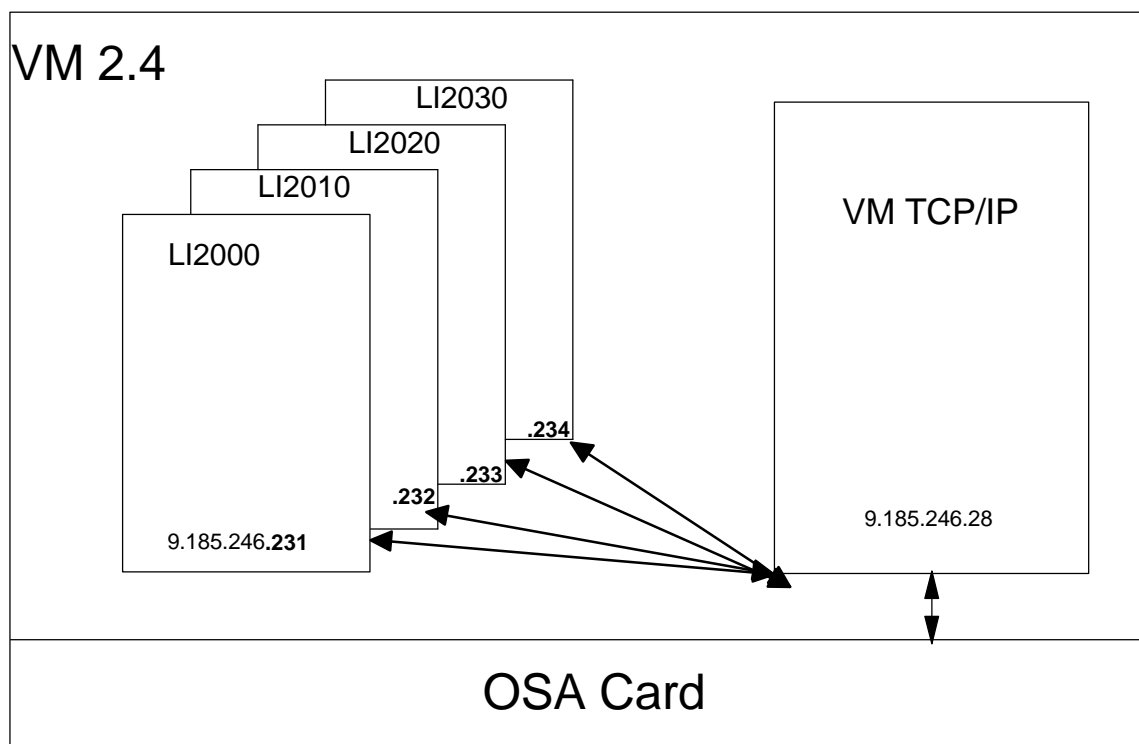
Networking Definitions have to be made in the following locations:

- The VM TCP/IP stack's PROFILE TCPIP
- User Directory - VM TCP/IP stack definition
- User Directory - LINUX user definitions
- PROFILE EXEC of each Linux Guest

VM TCP/IP PROFILE

```
DEVICE LI2000 CTC 2000
LINK LI2000LNK CTC 0 LI2000

DEVICE LI2010 CTC 2010
LINK LI2010LNK CTC 0 LI2010

DEVICE LI2020 CTC 2020
LINK LI2020LNK CTC 0 LI2020

DEVICE LI2030 CTC 2030
LINK LI2030LNK CTC 0 LI2030

DEVICE OSA992 LCS 992
LINK TR992 IBMTR 1 OSA992
.......
HOME
9.185.246.28   LI2000LNK
9.185.246.28   LI2010LNK
9.185.246.28   LI2020LNK
9.185.246.28   LI2030LNK
9.185.246.28   TR992
........
GATEWAY
 9.0.0.0               =      TR992       1492     0.255.255.224  0.185.246.0
 9.185.246.231         =      LI2000LNK  1492  HOST
 9.185.246.232         =      LI2010LNK  1492  HOST
 9.185.246.233         =      LI2020LNK  1492  HOST
 9.185.246.234         =      LI2030LNK  1492  HOST
........
START LI2000
START LI2010
START LI2020
START LI2030
START OSA992
```

In this example we have added DEVICE definitions for 4 Linux instances. We would add as many definitions as we have Linux guest systems. We have used the device numbering scheme of 2000, 2010 etc. You should consider using a numbering scheme that is scalable. You can have up to around 100 CTC device pairs per TCP/IP stack.

CTC device numbering uses even/odd pairs, so for example device 2000 would be used as the CTC read channel, device 2001 would be used as the write channel.

USER DIRECTORY - TCP/IP Stack user definitions

```
 USER VMIP VMIP  32M 128M ABG
   INCLUDE TCPCMSU
   OPTION QUICKDSP SVMSTAT MAXCONN 1024 DIAG98 APPLMON
   SHARE RELATIVE 3000
   IUCV ALLOW
   IUCV ANY PRIORITY
   IUCV *CCS PRIORITY MSGLIMIT 255
   SPECIAL 2000 CTCA LI2000
   SPECIAL 2001 CTCA LI2000
   SPECIAL 2010 CTCA LI2010
   SPECIAL 2011 CTCA LI2010
   SPECIAL 2020 CTCA LI2020
   SPECIAL 2021 CTCA LI2020
   SPECIAL 2030 CTCA LI2030
   SPECIAL 2031 CTCA LI2030
   .......
```

Partial directory entry for the TCP/IP stack

In this user directory definition for the TCP/IP stack you can see that we have defined CTC adapter devices for 4 Linux systems.

The VM TCP/IP stack will communicate to the Linux systems using the appropriate device addresses, for example VM TCP/IP will talk to LI2000 using devices 2000 and 2001.

We would add as many CTCA pair definitions as there are Linux systems to connect to.

USER DIRECTORY - Linux user definitions

```
USER LINMNT4 LINMNT4 128M 2047M G
INCLUDE LINDFLT
IUCV ANY
IUCV ALLOW
IPL CMS PARM AUTOCR
MACHINE ESA
SPECIAL 2000 CTCA VMIP
SPECIAL 2001 CTCA VMIP
 MDISK 191 3390 0001 0050 LIA313 MR ALL SOME FEW
 MDISK 200 3390 0001 0200 LIS319 MR ALL SOME FEW
 MDISK 201 3390 0202 2800 LIS319 MR ALL SOME FEW
 MDISK 202 3390 0001 3300 LIS339 RR ALL SOME FEW
```

In this user directory definition for a Linux user you can see that we have defined CTC devices 2000 and 2001. Since each Linux VM Guest has it's own virtualized environment, all Linux guests are defined as using CTC devices 2000 and 2001 and will communicate with the VM TCP/IP stack using these devices.

PROFILE EXEC for each Linux Guest

```
/* VM LINUX Profile exec */
'VMFCLEAR'
'CP SET MSG ON'
'CP SET EMSG ON'
'SET IMPCP ON'
'SET RUN ON'
'CP TERM CHARDEL OFF'
'VMLINK TCPMAINT 592 ( NONAMES'
'CP SET PF10 RETR'
'CP SET RETR MAX'
'CP TERM HOLD OFF'
/*  'CP TERM MORE 1 1'  */
'CP COUPLE 2000 VMIP ' '20' || substr(userid(),5,1) || '1'
'CP COUPLE 2001 VMIP' '20' || substr(userid(),5,1) || '0'
```

The final step is to connect the CTC devices between the Linux guest and the TCP/IP stack. To do this we add the above "COUPLE" commands to the Linux guest's PROFILE EXEC. Remember the PROFILE EXEC is executed whenever a VM guest logs on (something you have to do before booting the Linux system).

We couple the Linux read channel to the VM TCP/IP write channel and the Linux write channel to the VM TCP/IP read channel.

Therefore, in our example Linux's CTC read channel 2000 is connected to the VM TCP/IP write channel 2001. Similarly, the Linux CTC write channel 2001 is connected to VM TCP/IP read channel 2000.

Once the above defintions have been built, you should now be able to log on to the Linux guest and see the following messages:

CTCA 2000 COUPLED TO VMIP 2001
CTCA 2001 COUPLED TO VMIP 2000

You now have IP connectivity from the Linux VM Guest to the VM TCP/IP stack.

Summary of networking defintions

Linux IP host name	Linux IP host address	Guest CTCs	TCPIP CTCs
LI2000	9.185.246.231	2000 / 2001	2000/2001
LI2010	9.185.246.232	2000 / 2001	2010/2011
LI2020	9.185.246.233	2000 / 2001	2020/2021
LI2030	9.185.246.234	2000 / 2001	2030/2031

Appendix C   Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this red paper.

IBM Redbooks

Linux for S/390   SG24-4987-00

IBM Manuals

Preparing for Installing SuSE LINUX for S/390   LNUX-1001-00

Installing SuSE LINUX for S/390   LNUX-1002-00

VM/ESA 2.4 Planning and Administration   SC24-5750-04

VM/ESA 2.4 CP Command and Utility Reference   SC24-5773-04

VM/ESA 2.4 TCP/IP Function Level 320 Planning and Customization
                                                                            SC24-5773-04

SuSE Manuals

SuSE Linux 7.0 Installation, Networking, Know How