

Cluster node install guide

F. Bruckner,N. Mc Guire,M. Poetl

Opentech EDV-Research GmbH

June 7, 2004

Contents

| | |
|--|-----------|
| 1. Introduction | 1 |
| 1.1. License | 1 |
| 2. Slackware 9.1 Install | 1 |
| 2.0.1. Partitioning | 2 |
| 2.1. Starting setup | 5 |
| 2.2. Final steps before reboot | 12 |
| 2.3. System Boot | 13 |
| 2.4. Cluster Software CD | 13 |
| 3. Cluster Node setup | 14 |
| 3.0.1. User setup | 14 |
| 3.1. Non-interactive ssh | 16 |
| 4. Installing LAM | 17 |
| 4.1. Configuring LAM | 18 |
| 4.2. Compiling/Installing | 19 |
| 4.3. Post Install config | 19 |
| 4.3.1. Shell settings | 20 |
| 4.4. bhost | 21 |
| 5. hello world | 22 |
| 5.1. Checking the setup | 22 |
| 5.2. booting lam | 23 |
| 5.3. hello.c | 24 |
| 6. Some typical problems | 26 |
| 7. List of Acronyms | 28 |
| 8. Cluster Software Sources | 29 |

Contents

| Version | Author | Date | Comment |
|---------|----------------------|-------------|------------|
| 1.0 | Nicholas Mc Guire | 28 May 2004 | First shot |

1. Introduction

This manual is not a in depth introduction to installing and running Linux but, as the name says, a kick-start only. It should guide you step by step to get you up and running on a Linux based LAM/MPI cluster quickly. Although this document describes the steps for LAM/MPI on a Slackware Linux 9.1 system, steps will more or less be the same on a different distribution. Feedback, especially on trying this for other platforms would be appreciated.

You should have CD1 and CD2 lying on your desk now - if not then grab the Slackware 9.1 from the next download site:

<http://www.slackware.org/getslack/>

for a list of sites world wide.

1.1. License

This kick-start manual is Licensed under FDL V1.2 <http://www.gnu.org/copyleft/fdl.html>. For a full list of software packages that you need (referred to here as the Cluster Software CD) see the appendix.

2. Slackware 9.1 Install

- Boot from CD: < ENTER >

The boot prompt is only intended for passing additional kernel parameters - norm ally necessary if you have some non-standard hardware, also if the default `bare.i` kernel does not work, press [F2] at the boot prompt for a list of possible kernels to boot.

- boot: < ENTER >
- Enter 1 to select a keyboard:
- Keyboard map selection:

qwertz/de-latin1-nodeadkeys.map < OK >

- Keyboard test

```
1 < OK >
```

Not a very intuitive interface that requires to type in 1 to the text field before hitting < OK > - but thats Slackware...

You may now login as 'root'

Slackware login:

At this point Slackware is running a minimum system in a ramdisk so you actually are login into the Linux box as root at this point. So type in root and hit < ENTER >

2.0.1. Partitioning

As noted above Slackware boots into a minimum system loaded into a ramdisk - so you have the 'standard' GNU/Linux tools available for system setup. Slackware does not bother providing a 'User Friendly' wrapper to these functions, you simply use them on the command line and that ensures that you actually know what you are doing.

```
# fdisk /dev/hda
```

The number of cylinders for this disk is set to 15017.

There is nothing wrong with that, but this is larger than 1024, and could in certain setups cause problems with:

- 1) software that runs at boot time (e.g., old versions of LILO)
- 2) booting and partitioning software from other OSs (e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):

To check the existing partition table use the 'p' command

Command (m for help): p

Disk /dev/hda: 123.5 GB, 123522416640 bytes

2. Slackware 9.1 Install

255 heads, 63 sectors/track, 15017 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

| Device | Boot | Start | End | Blocks | Id | System |
|-----------|------|-------|-------|------------|----|------------|
| /dev/hda1 | | 1 | 103 | 827316 | 82 | Linux swap |
| /dev/hda2 | * | 104 | 15016 | 119788672+ | 83 | Linux |

First we delete all partitions as this is going to be a pure Linux box. If there are partitions defined make sure you delete them in reverse order - so start with the highest numbered partition and delete one by one (in my case this was 2).

```
Command (m for help): d
Partition number (1-4): 2
```

```
Command (m for help): d
Selected partition 1
```

```
Command (m for help): 1
```

Next we create two new partitions one as a Linux filesystem (we will simply put it all in one big chunk for now) and one swap partition.

```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
```

We respond with 'p' for a primary partition and then get the prompt for the partition number.

```
p
Partition number (1-4): 1
First cylinder (1-15017, default 1): <ENTER>
```

As our first partition should start at the first cylinder we simply hit enter.

```
Using default value 1
Last cylinder or +size or +sizeM or +sizeK (1-15017, default 15017): +512M
```

2. Slackware 9.1 Install

On the first partition we are going to put the swap partition, so we request 512MB for the first partition, the '+' tells fdisk to increment 512MB starting at the current cylinder position, which is 1 in our case. We could give it a cylinder number too but then you must calculate the size yourself...

```
Command (m for help): n
Command action
  e   extended
  p   primary partition (1-4)
p
Partition number (1-4): 2
First cylinder (64-15017, default 64):
Using default value 64
Last cylinder or +size or +sizeM or +sizeK (64-15017, default 15017):
Using default value 15017
```

The second partition is again a primary partition and will simply be the full remaining disk, which is offered by default.

If we now print the current partition table we see the two desired partitions, but they are both marked as Linux, we need one to be a swap partition.

```
Command (m for help): p

Disk /dev/hda: 123.5 GB, 123522416640 bytes
255 heads, 63 sectors/track, 15017 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes
```

| Device | Boot | Start | End | Blocks | Id | System |
|-----------|------|-------|-------|-----------|----|--------|
| /dev/hda1 | | 1 | 63 | 506016 | 83 | Linux |
| /dev/hda2 | | 64 | 15017 | 120118005 | 83 | Linux |

So the next step is to change our first partition to Linux swap with the 't' command.

```
Command (m for help): t
Partition number (1-4): 1
Hex code (type L to list codes): L

0  Empty                1c  Hidden W95 FAT3 70  DiskSecure Mult bb  Boot Wizard hid
1  FAT12                 1e  Hidden W95 FAT1 75  PC/IX                be  Solaris boot
```

2. Slackware 9.1 Install

```
2 XENIX root          24 NEC DOS             80 Old Minix          c1 DRDOS/sec (FAT-
3 XENIX usr           39 Plan 9              81 Minix / old Lin   c4 DRDOS/sec (FAT-
4 FAT16 <32M         3c PartitionMagic    82 Linux swap        c6 DRDOS/sec (FAT-
5 Extended           40 Venix 80286        83 Linux             c7 Syrix
6 FAT16              41 PPC PReP Boot      84 OS/2 hidden C:   da Non-FS data
7 HPFS/NTFS          42 SFS                85 Linux extended   db CP/M / CTOS / .
8 AIX                4d QNX4.x              86 NTFS volume set  de Dell Utility
9 AIX bootable       4e QNX4.x 2nd part    87 NTFS volume set  df BootIt
a OS/2 Boot Manag   4f QNX4.x 3rd part    8e Linux LVM         e1 DOS access
b W95 FAT32          50 OnTrack DM          93 Amoeba            e3 DOS R/O
c W95 FAT32 (LBA)   51 OnTrack DM6 Aux    94 Amoeba BBT        e4 SpeedStor
e W95 FAT16 (LBA)   52 CP/M              9f BSD/OS            eb BeOS fs
f W95 Ext'd (LBA)   53 OnTrack DM6 Aux    a0 IBM Thinkpad hi  ee EFI GPT
10 OPUS              54 OnTrackDM6         a5 FreeBSD          ef EFI (FAT-12/16/
11 Hidden FAT12      55 EZ-Drive           a6 OpenBSD           f0 Linux/PA-RISC b
12 Compaq diagnost  56 Golden Bow         a7 NeXTSTEP          f1 SpeedStor
14 Hidden FAT16 <3   5c Priam Edisk        a8 Darwin UFS        f4 SpeedStor
16 Hidden FAT16      61 SpeedStor          a9 NetBSD            f2 DOS secondary
17 Hidden HPFS/NTF  63 GNU HURD or Sys   ab Darwin boot      fd Linux raid auto
18 AST SmartSleep    64 Novell Netware     b7 BSDI fs           fe LANstep
1b Hidden W95 FAT3   65 Novell Netware     b8 BSDI swap         ff BBT
Hex code (type L to list codes): 82
Changed system type of partition 1 to 82 (Linux swap)
```

You should then check that the partition table is correct and then call the write command to actually write the new partition table to disk. Until you type 'w' for write nothing on the disk was changed - so you can quit any time by pressing < CNTRL > <C> or 'q' at the menu.

2.1. Starting setup

Slackware has a setup program on the ramdisk that you invoke by simply typing in

```
# setup
```

we then select the key-map again - see above - and proceed on to setting up our swap disk, the partition is all ready created and the setup script will find it, so we just need to activate it.

- SWAP SPACE DETECTED

/dev/hda1

- FORMATTING SWAP PARTITION...
- SWAP SPACE CONFIGURED

/dev/hda1

The partition we set up for the Linux filesystem needs to be formatted next, first we select the partition from the presented possibilities, which is only /dev/hda2 in our case, and hit < ENTER >, next we are asked for the method of formatting, Format is OK for almost all systems, if the hard-disk is some old disk (and not too large...) you might want to select 'Check' which will actually check each block and update the bad-blocks list if necessary.

- Select Linux installation partition:

/dev/hda2

- FORMAT PARTITION /dev/hda2

```
NOTE: This will erase all data on it.  
Format < OK >  
Check  
No
```

We suggest using ext3 filesystem, it will not save your data if you crash the kernel with a buggy kernel module, but it is a good protection against power-failure or reset button induced problems... The inode density can be left at the suggested default value.

- SELECT FILESYSTEM FOR /dev/hda2

```
ext2  
ext3 < OK >  
reiserfs
```

- SELECT INODE DENSITY FOR /dev/hda2

2. Slackware 9.1 Install

```
4096 1 inode per 4096 bytes. < OK >
2048 1 inode per 2048 bytes.
1024 1 inode per 1024 bytes.
```

- FORMATTING...
- DONE ADDING LINUX PARTITIONS TO /etc/fstab

So now the systems partitions are set up and formatted - we are ready to fill the disk up with content. But before that we have to select an installation media, which is the CD we booted from in our case, this menu question makes sense because Slackware can also be installed starting with a floppy disk and if you have a really fast university network (does something like this actually exist ?) then NFS install may be an option.

In case you booted from the CD the auto-detection will work fine, so we select [auto] and hit < OK >.

- SOURCE MEDIA SELECTION

```
1 Install from a Slackware CD or DVD < OK >
2 Install from a hard drive partition
3 Install from NFS (Network File System)
4 Install from a pre-mounted directory
```

- SCANNING FOR CD or DVD DRIVE

```
auto < OK >
manual
```

- SCANNING...

If this does not kick your CD-ROM drive up then you should try manual selection.

Once you have your source media set we go to the package selection. Slackware allows you to select each package individually, which can take a very long time, so unless you really want a minimum system using the prepackaged selections is fine and will result in a system with all tools we need for real-time. The only thing we deselect here is KDE and GNOME, simply to reduce install time and because we are not concerned with X-setup for this session. We want to show you the power of the command-line, you can learn how to play with X-Windows later :) .

After de-selecting KDE and GNOME we can simply install everything and hit < OK >.

- PACKAGE SERIES SELECTION

NOTE: If you install without KDE and GNOME, you will only need disc 1.

- SELECT PROMPTING MODE

full < OK >

- Installing...

If you did not de-select KDE and GNOME then you will be prompted for the second disk, in our case this does not happen, so we would go right to the kernel selection.

- INSERT NEXT DISC

Continue < OK >

- Installing...

You should not necessarily select the hottest and most optimized kernel here, you should select the safest kernel for the system, for IDE based systems the bare.i from the cdrom is what you want.

- INSTALL LINUX KERNEL

```
bootdisk
cdrom < OK >
floppy
skip
```

- CHOOSE LINUX KERNEL

```
/cdrom/kernels/bare.i/bzImage < OK >
```

It is a wise thing to create a boot-disk for a development system, sooner or later you might damage the system with your first (buggy) kernel modules, and as we never make backups... a boot-disk is helpful. In this kickstart session we will skip this step though.

We are not going to bother with the modem, and for desk-top systems you probably will not need the hot-plug subsystem, but it does not hurt to enable it.

- MAKE BOOTDISK

```
Create  
Skip < SKIP >
```

- MODEM CONFIGURATION

```
no modem < OK >
```

- ENABLE HOTPLUG SUBSYSTEMS AT BOOT?

```
< Yes >
```

We need a boot-loader to actually start the system on power-on, so next we configure the Linux LOader - LILO. If you know lilo and want some special options set, select 'expert' if you are a new-bee, take the 'simple' option, it will work in more or less all cases where you have an IDE based system.

Selecting frame-buffer console is important or you will not get the penguin logo in the top left hand corner of your screen...

- INSTALL LILO

```
simple < OK >  
expert  
skip
```

- CONFIGURE LILO TO USE FRAME BUFFER CONSOLE?

```
1024x768x256 < OK >
```

If you have a CD-burner in your system, which is quite common, then you want to set up that CD-ROM as a SCSI device via the ide-scsi emulation, so we pass the device specific module to LILO telling it to use scsi emulation for /dev/hdc in this case.

LILO is put on the Master Boot Record (MBR). After this step the Linux loader LILO is installed and the system could boot. Note that you only should put Lilo on the MBR if it is a stand-alone installation, if you want to share the hardware among different Linux and non-Linux installs refer to `Multiboot-with-LILO` in the Linux HOWTO collection.

- OPTIONAL LILO append="<kernel parameters>" LINE

```
hdc=ide-scsi < OK >
```

- SELECT LILO DESTINATION

```
Root
Floppy
MBR < OK >
```

The rest of the configuration is not that general and may be different in your case. First we set up the mouse and configure General Purpose Mouse-support - GPM which allows using the mouse in text mode.

- MOUSE CONFIGURATION

```
ps2 < OK >
bare 2 button serial mouse
ms 3 button serial mouse
```

- GPM CONFIGURATION < Yes >

The network configuration is site specific, so you need to get the infos from your network admin. The infos you will need are:

- Host name
- Domain name
- IP address
- Netmask
- Default gateway
- Domain Name Server (DNS)

With these informations ready we can proceed with the network configuration.

- CONFIGURE NETWORK? < Yes >

- ENTER HOSTNAME

```
rt120 < OK >
```

- ENTER DOMAINNAME for 'rt120'

```
hofr.at < OK >
```

- SETUP IP ADDRESS FOR 'rt120.hofr.at'

```
static IP < OK >  
DHCP  
loopback
```

- Fill in the
 - IP address
 - Netmask
 - Default gateway
 - Domain Name Server (DNS)
- CONFIRM SETUP COMPLETE < Yes >
- CONFIRM STARTUP SERVICES TO RUN < OK >
- CONSOLE FONT CONFIGURATION < No >

Setting up the clock: assume the clock is UTC and select your time-zone from the list.

- HARDWARE CLOCK SET TO UTC?

```
No  
Yes < OK >
```

- TIMEZONE CONFIGURATION

```
Europe/Vienna < OK >
```

If you did not select the KDE and GNOME packages during installation, you should not select KDE or gnome here... but there are a number of interesting and light weight window managers around that are worth giving a look.

- SELECT DEFAULT WINDOW MANAGER FOR X

```
xinitrc.kde < OK >  
xinitrc.gnome
```

2.2. Final steps before reboot

Last thing the system needs before we can reboot is a root password, for this session you should set it to 'npasswd', but in your company network or at home, make sure you have a reasonable root-password that will not be guessed easily.

- WARNING: NO ROOT PASSWORD DETECTED Would you like to set a root password? < Yes >

```
New password: npasswd  
Re-enter new password: npasswd
```

- SETUP COMPLETE < OK >
- EXIT < OK >

This terminates the setup program of Slackware, and we can reboot the system. Just to make sure the filesystems are cleared properly we do:

```
# umount -a  
# <CTRL>--<ALT>--<DELETE>
```

...and don't forget to remove the CD-ROM or you will fall into an endless loop.

2.3. System Boot

At the LILO prompt you can add boot-command-line parameters for the kernel. This is helpful for instance, if you set up X (which is in init 4) and your screen just flickers, then you simply type in 'linux init 3', and boot the system to text-mode only. for more info on available settings check the BootPrompt-HOWTO locate in /usr/doc/Linux-HOWTOs/BootPrompt-HOWTO on your Slackware 9.1 distribution.

```
boot: linux < ENTER >
```

After the boot messages scrolled by you get the login prompt:

```
rt115 login: root
Password: nopasswd
```

We hope that the messages produced by Slackware after login - the so called fortunes - are politically correct, but we take no responsibility for these messages....

2.4. Cluster Software CD

As the default location to attache the cdrom is /mnt/cdrom (see /etc/fstab for the default on your system), you normally can use the command

```
root@rt115:~ # mount /mnt/cdrom
```

and all necessary informations would be extracted from /etc/fstab - as we want to show you as much of what is happening beneath the covers - we will use the explicit mount command and create a mount point first.

```
root@rt115:~ # mkdir /cdrom
root@rt115:~ # mount -t iso9660 /dev/hdb /cdrom
```

In the system used for this HOWTO the cdrom was the secondary slave device on the IDE subsystem so /dev/hdb in this case - you must replace any references to /dev/hdb by what is given by your system configuration to find the device quickly - type in the following:

```
root@rt115:~# dmesg | grep hd
```

3. Cluster Node setup

```
hda: IC35L120AVV207-0, ATA DISK drive
hdb: LITE-ON COMBO LTC-48161H, ATAPI CD/DVD-ROM drive
hda: attached ide-disk driver.
hda: host protected area => 1
hda: 241254720 sectors (123522 MB) w/1821KiB Cache, CHS=15017/255/63
  hda: hda1 hda2
...
```

This tells us that the CDRom is attached as hdb.

If you don't have it mounted yet (see the previous steps) mount it now.

We will leave it mounted for now so that we can access all the necessary cluster software in the following steps. If you don't have the CD then you can download all packages from the internet - all software on the CD is free software (with varying licenses though - so please check the individual licenses !).

3. Cluster Node setup

We assume here that you have Slackware 9.1 default installation set up on the system. If this is not the case then refer to the Slackware install section above and then return here.

3.0.1. User setup

Adding a user must be done as root. As we will not run the cluster as the root user for a number of serious security issues, and the lam team designed lam to be run as non-root user ONLY, we add a user on all nodes, note that the setup of the user should be consistent (UID,GID,home,shell) or it can be quite painful to get lam to operate properly (although it is possible to do so, and on heterogenous network cluster it might even be necessary).

```
root@rtl21:~# adduser
```

```
Login name for new user []: hofrat
```

```
User ID ('UID') [ defaults to next available ]: 1834
```

```
Initial group [ users ]:
```

```
Additional groups (comma separated) []:
```

3. Cluster Node setup

Home directory [/home/hofrat]

Shell [/bin/bash]

Expiry date (YYYY-MM-DD) []:

New account will be created as follows:

```
-----  
Login name.....:  hofrat  
UID.....:  1834  
Initial group....:  users  
Additional groups:  [ None ]  
Home directory...:  /home/hofrat  
Shell.....:  /bin/bash  
Expiry date.....:  [ Never ]
```

This is it... if you want to bail out, hit Control-C. Otherwise, press ENTER to go ahead and make the account.

Creating new account...

Changing the user information for hofrat

Enter the new value, or press ENTER for the default

```
Full Name []: Der Herr Hofrat  
Room Number []:  
Work Phone []:  
Home Phone []:  
Other []:
```

Changing password for hofrat

Enter the new password (minimum of 5, maximum of 127 characters)

Please use a combination of upper and lower case letters and numbers.

New password:

Re-enter new password:

Password changed.

Account setup complete.

root@rtl21:~# exit

logout

Connection to rtl21 closed.

3.1. Non-interactive ssh

Whats now left to do is to configure access to the nodes. We need to allow non-interactive authentication. LAM by default assumens rsh which is not really a good idea as it is very unsafe, and clusters have become a prepered target of script-kiddies lately, so we will set up ssh to allow non-interactive logins for the user-hofrat.

Note that there are boot rpi's that allow to operate a cluster without using ssh or non-interactive logins (see the Bproc documentation). but for now we take a "classic" LAM/MPI setup.

```
root@rtl20:~/.ssh$ ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter file in which to save the key (/home/hofrat/.ssh/id_dsa):
Enter passphrase (empty for no passphrase): <ENTER>
Enter same passphrase again: <ENTER>
Your identification has been saved in /home/hofrat/.ssh/id_dsa.
Your public key has been saved in /home/hofrat/.ssh/id_dsa.pub.
The key fingerprint is:
21:e6:4e:ee:61:fe:70:a8:37:2d:9b:ad:02:c9:49:28 root@rtl20
root@rtl20:~/.ssh$ ls
id_dsa id_dsa.pub known_hosts
```

The id_dsa now contains the new private key and id_dsa.pub the public key, this now needs to be distributed to all nodes that are supposed to have access via ssh without requireing to type in the password.

```
root@rtl20:~/.ssh$ cat id_dsa.pub | ssh -l hofrat
rtl21 "cat - > ~/.ssh/authorized_keys"
root@rtl21's password:
root@rtl20:~/.ssh$ ssh -l hofrat rtl21
Last login: Fri Jun  4 13:42:26 2004 from rtl20.hofr.at
Linux 2.4.22.
root@rtl21:~$
```

The first ssh used to distribute the public key asks for the password , from then on there is no password prompt - we get logged in directly. For multiple keys you simply append it to the authorized_keys file

What is left to do is to turn off the beautiful fortune messages in Slackware 9.1 - that is done by editing /etc/profile.d/bsd-games-login-fortune.sh or you might simply remove

it. A further file that produces output on login that you most likely don't need is `/etc/motd` you can also safely remove this file. The last message we get on logins is the `lastlog` - you can turn off the `lastlog` in `/etc/login.defs` (set `LASTLOG_ENAB no`). Further you need to remove any existing log file in `/var/log/lastlog` - from then on, logins should simply be a switch in the prompt.

```
root@rtl20:~$ ssh -l hofrat rtl21
root@rtl21:~$
```

4. Installing LAM

First we download the latest LAM version from www.lam-mpi.org, at time of writing this is `lam-7.0.6.tar.bz2`. We unpack it in the users home directory as the configuration and compilation steps don't require any privileged access. In fact in the setup we choose here, the installation does not require any root privileges either as we set it up to run from within the users home directory. In case of a default installation, which drops it all into `/usr/local/bin` and the appropriate man-page locations, the install step would need root-privileges.

Before we `untar` anything - we check where it would end up with the `-t` flag to `tar`.

```
hofrat@rtl21:~$ tar -tjf lam-7.0.6.tar.bz2
lam-7.0.6/
lam-7.0.6/config/
lam-7.0.6/config/cxx_find_exception_flags.m4
...
<CNTRL>--<C>
```

If we don't yet have a `lam-7.0.6` directory, this location is fine, we can stop the `tar -tjf` with a `<CNTRL>--<C>`.

Rerun the `tar` command with the `-x` flag to extract the lam source tree.

```
hofrat@rtl21:~$ tar -xjf lam-7.0.6.tar.bz2
```

Get used to this procedure - it will save you some badly messed up directories. Once in a while a `tar` archive ends up on a ftp server that might unpack in the local directory and not in a subdirectory - or the subdirectory might exist and you will overwrite things.

4.1. Configuring LAM

As we want to build it in a per-user environment, we need to create a few directories to hold the documentation and binaries. For this purpose we create the directories that lam needs to put the binaries and libraries and a directory for the manual pages of lam and mpi.

```
hofrat@rtl21:~$ mkdir {bin,lib,man}
hofrat@rtl21:~$ ls
bin/ lam-7.0.6/ lam-7.0.6.tar.bz2 lib/ man/
```

With this setup we can now configure and compile lam as a per-user setup

```
hofrat@rtl21:~/lam-7.0.6$ ./configure --prefix=/home/hofrat
```

We also could use `--prefix=$HOME` instead and For fine tuning of the installation directories you might want to use:

```
--bindir=DIR          user executables [EPREFIX/bin]
--libdir=DIR          object code libraries [EPREFIX/lib]
--mandir=DIR          man documentation [PREFIX/man]
```

For the full set of possible configuration options use `./configure --help`, but with respect to directory settings you don't need more than the three above.

debugging related options:

- `-without-profiling`
enable profiling (this refers to lams MPI profiling interface not gprof or the like)
- `-with-purify`
zero memory before using it - this is really only needed when debugging/memory checking lam applications - it will slow down lam.
- `-with-debug`
turn on debugging - this will seriously slow down things .

Especially in larger clusters with some network bottlenecks between faster sub-units timing issues can be nasty to track down - so some timing values can help eliminate these errors. Generally if you need to increase these values to get the system stable your applications must be taking these timing issues into account. For most users the defaults are ok though.

- `-with-lamd-boot=SEC`
timeout (default 60 seconds) for booted node to respond before the boot is considered failed, on large systems this might need to be raised, and when using topologies with slow (WAN) links between sub-clusters this also may be needed.
- `-with-lamd-ack=MICROSEC`
microseconds until a packet is resent if no ack is received, the default of half a second should be sufficient. If increasing this helps you probably have a network problem. In any case increasing this value will not hurt.
- `-with-lamd-hb=SEC`
Time between heartbeat packets when using the lamd RPI, if you set the other times to large values to get the system stable you probably want the heartbeats also to run with a low frequency - making this value large should not hurt.

4.2. Compiling/Installing

```
hofrat@rtl21:~/lam-7.0.6$ make
...hours later...
hofrat@rtl21:~/lam-7.0.6$ make install
```

Note that in our setup we can do the install as user because all of the files being installed will go below our home directory. After doing the make install step, the directories we created above should contain:

- the compiler wrappers `mpif77,mpicc,mpic++` in `bin`
- the lam management tools `lamboot,laminfo,etc` in `bin`
- the lam and mpi libraries along with the romio lib in `lib` (note that romio is a lam extension and can be disabled at configuration time - default is on though)
- the lam/mpi manual pages in `man`. If you only ask questions after reading the man pages from lam you will not have many questions to ask - so its a good idea to read them !

4.3. Post Install config

This per-user setup requires a few per-user setting for lam to operate properly, in the following section we discribe the setup we used - this is a bit Slackware 9.1 specific, but it should help on non-slackware systems as well.

4.3.1. Shell settings

To actually be able to access the binaries, libs and manual pages installed in the last step, your shell needs a few settings added/changed.

setting up your environment for the per-user lam installation:

- executables:

By default the directory `/home/hofrat/bin` is not searched for executables - so we need to add it to our `PATH` environment variable.

```
hofrat@rtl21:~$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr...
hofrat@rtl21:~$ export PATH=~/.bin:$PATH
```

Now you should be able to type in `mpicc` and get an error like:

```
/usr/lib/gcc-lib/i486-slackware-linux/3.2.3/../../../../
crt1.o(.text+0x18): In function '_start':
: undefined reference to 'main'
collect2: ld returned 1 exit status
mpicc: No such file or directory
```

Which is ok - we did not provide any source file so the compiler will fuss, we just want to make sure the compiler is actually found. If you get:

```
hofrat@rtl21:~$ mpicc
-bash: mpicc: command not found
```

then your `PATH` variable is not yet set properly or lam installation failed !

- man pages:

The man pages are by default searched only in the paths set in the shell specific startup files (i.e. `/etc/profile` for `bash`). To let man search the lam/mpi man pages in `/home/hofrat/man` you would do the following:

```
hofrat@rtl21:~/$ echo $MANPATH
/home/hofrat/man:/usr/local/man:/usr/man:/usr/X11R6/man:/usr/lib/java/man:/usr/share
hofrat@rtl21:~/$ export MANPATH=/home/hofrat/man:$MANPATH
```

4. Installing LAM

now `man lamboot` should pop up the man page for you, note that you don't need to bother about the subdirectories in `/man`, the `man` command only needs the top level man directory in its path.

- libraries:

Once things work, you can now add the above export commands to your `.profile` and `.bashrc`. The `.profile` is read for any login shells, so that is what will be read later when lam uses `rsh` or `ssh` to start the mpi task on the node, the `.bashrc` is read only for interactive shells, so it is sufficient to put the `MANPATH` in the `.bashrc`. One more thing to add is the agent that lam should use for remote access, in our case this is `ssh` so we export `LAMRSH=ssh` along with the other environment variables `PATH` and `MANPATH`.

For our setup the `.profile` would contain:

```
MANPATH=/home/hofrat/man/:$MANPATH
PATH=~ /bin:$PATH
CVSRSH=ssh
```

```
export MANPATH PATH CVSRSH
```

With these settings you can log out and log in again and all lam/mpi resources should be accessible.

4.4. bhost

Lam needs to know what systems will be part of our cluster, as we set up two systems `rt120` and `rt121` we will put these two systems into the `bhost` file. This file is simply a list of systems that lam may use, and as we have single processor systems only we don't need to specify anything other than the host name. Note that lam can't only run SPMD applications, where the same application is running on all nodes, but also can run in MIMD setups, in which case you would need an application scheme to run. For now we will assume SPMD applications and thus only need a node list:

```
rt120
rt121
```

5. hello world

Now that access to the nodes is set up and lam is installed on all nodes we need an mpi application that we can run.

5.1. Checking the setup

But before we actually do that there are a few things we should check. Lam provides some status commands (`man laminfo` for details will not hurt) which will give us the list of available SSIs in our (default) configuration.

```
LAM/MPI: 7.0.6
  Prefix: /home/hofrat/
Architecture: i686-pc-linux-gnu
Configured by: hofrat
Configured on: Sun May 23 18:46:42 CEST 2004
Configure host: rtl20
  C bindings: yes
  C++ bindings: yes
Fortran bindings: yes
  C profiling: yes
  C++ profiling: yes
Fortran profiling: yes
  ROMIO support: yes
  IMPI support: no
  Debug support: no
  Purify clean: no
  SSI boot: globus (Module v0.5)
  SSI boot: rsh (Module v1.0)
  SSI coll: lam_basic (Module v7.0)
  SSI coll: smp (Module v1.0)
  SSI rpi: lamd (Module v7.0)
  SSI rpi: sysv (Module v7.0)
  SSI rpi: tcp (Module v7.0)
  SSI rpi: usysv (Module v7.0)
```

What this tells us is that we have the rsh boot ssi available and for remote communication we can use the lamd rpi or the tcp rpi (the other two rpi's are for shared memory systems, in our setup we are assuming networked workstations so we can't use those). The SSI modules for collective operations (coll SSIs) are not important for us (yet).

5.2. booting lam

At this point we are ready to boot the Local Area Multicomputer (LAM). Booting lam means to initiate the mpi management layer, no application yet. This layer is responsible for coordinating the nodes and managing stdout/stderr redirection etc. Once we launch lam we will see a daemon running on all nodes.

```
hofrat@rtl20:~$ lamboot -v -ssi boot rsh -ssi rsh_agent "ssh -x" bhost
```

```
LAM 7.0.6/MPI 2 C++/ROMIO - Indiana University
```

```
n-1<26780> ssi:boot:base:linear: booting n0 (192.168.1.40)
n-1<26780> ssi:boot:base:linear: booting n1 (192.168.1.41)
n-1<26780> ssi:boot:base:linear: finished
```

You might not need to pass the boot ssi in your setup if it is the default - but it never hurts to actually request exactly what you want and not rely on specific default ! The `-v` flag is not needed, but it helps a lot in case of errors and does not hurt in case everything goes well, so we will use it for lamboot in general.

Checking with ps we can see lamd running:

```
hofrat@rtl20:~$ ps auxw | grep lamd
hofrat  26783  0.0  0.5 2544 1328 ?        S    09:57
0:00 /home/hofrat/bin/lamd -H 192.168.1.40 -P 33505 -n 0 -o 0
hofrat@rtl20:~$ ssh -l hofrat rtl21 "ps auxw | grep lamd"
hofrat  32762  0.0  0.5 2536 1304 ?        S    08:47
0:00 /home/hofrat/bin/lamd -H 192.168.1.40 -P 33505 -n 1 -o 0
```

Note that you might get more lines of output than only the one as the `grep lamd` will also show us the `ssh` command and the `grep` command it self.

The `lamboot` command and arguments we passed on `rtl20` were translated to the `lamd` arguments used to actually invoke `lamd` on each node, the setting shown above mean:

- H the boot IP address
- n the node number (rank) this node has
- o the origin - where lam was started
- P the port number that lam will use for communication

The cluster is now running. So basically we are done with setting up a mini-super computer ;)

5.3. **hello.c**

We will start out with a simple mpi version of hello world, we will not explain details of the MPI calls here - but there are not only some good tutorials out there (see <http://www.lam-mpi.org> tutorials) but lam comes with a complete set of man pages so `man MPI_Init` will help.

```
/* hello.c
 *
 * GPL V2,
 * (C) 2004, Der Herr Hofrat <der.herr@hofr.at>
 * OpenTech EDV Research GmbH <http://www.opentech.at>
 */
/* message ping-pong:
 *     rank 0 send the message to rank 1
 *     rank 1 printf's it on stdout
 *     stdout is redirected to the root node via lamd
 *
 * compile (lam-mpi):
 *     mpicc -Wall -lmpi -llam hello.c -o hello
 * run (lam-mpi):
 *     mpirun n0-1 hello
 */
#include <stdio.h>    /* printf */
#include <unistd.h>   /* sleep */
#include <strings.h>  /* memset, strncpy */
#include <mpi.h>      /* MPI_* */

#define BUF_SIZE 64
#define PING_PONG 42
#define NODE0 0
#define NODE1 1

int main(int argc, char ** argv){
    int rank;
    int msg_size=0;
    char msg[BUF_SIZE];
    MPI_Status status;

    MPI_Init(&argc,&argv);
```

5. *hello world*

```
memset(msg,0,BUF_SIZE);
msg_size=sizeof("hello mpi world");

MPI_Comm_rank(MPI_COMM_WORLD,&rank);

if(rank == 0){
    strncpy(msg,"hello mpi world\n",msg_size);
    MPI_Ssend(msg,
        msg_size,
        MPI_CHAR,
        NODE1,
        PING_PONG,
        MPI_COMM_WORLD);
}else if (rank==1){
    MPI_Recv(msg,
        msg_size,
        MPI_CHAR,
        NODE0,
        PING_PONG,
        MPI_COMM_WORLD,
        &status);
    printf("%s",msg);
}

MPI_Finalize();
return 0;
}
```

We now compile this with the lam wrapper compiler, which is called `mpicc` in lam (note that the MPI standard does not cover the build environment so this is implementation dependant!).

```
hofrat@rtl20:~/mpi$ mpicc -Wall -lmpi -llam hello.c -o hello
hofrat@rtl20:~/mpi$ scp hello hofrat@rtl21:/home/hofrat/mpi/
hello                               100% 4475KB   4.4MB/s   00:01
hofrat@rtl20:~/mpi$ mpirun n0-1 hello
hello mpi world
hofrat@rtl20:~/mpi$
```

If you get that output - then we are done with this kickstart - if not then there is one more section on some common problems.

6. Some typical problems

This list is not complete - we hope it will be of help any way - for more information on frequently ask questions see the FAQ section at <http://www.lam-mpi.org/>

```
hofrat@rtl20:~$ mpirun a.out
mpirun (locate_aschema): a.out: No such file or directory
```

missing n0-1 - lam does not know where to run it and the attempted interpretation of a.out as local application schema obviously failed.

```
hofrat@rtl20:~$ mpirun n0-1 a.out
mpirun: cannot start a.out on n1: No such file or directory
```

a.out can't be found on node n1, either its not there or the path settings are not ok.

But we can check if its a lam or lam-setting problem by running any arbitrary executable that we are sure is to be found in the system default path. Checking lam setup with an arbitrary program like /bin/ls will return some errors but we can see if lam is actually booted on all nodes and if the authentication mechanism is working.

```
hofrat@rtl20:~$ mpirun -v n0-1 ls
bhost          bproc-3.2.6.tar.gz  include          lib    mpi
bin            etc                 lam-7.0.6        linux  share
bproc-3.2.6    hofrat@rtl21       lam-7.0.6.tar.bz2  man
6832 ls running on n0 (o)
6833 ls running on n1
```

It seems that [at least] one of the processes that was started with mpirun did not invoke MPI_INIT before quitting (it is possible that more than one process did not invoke MPI_INIT -- mpirun was only notified of the first one, which was on node n0).

mpirun can *only* be used with MPI programs (i.e., programs that invoke MPI_INIT and MPI_FINALIZE). You can use the "lamexec" program to run non-MPI programs over the lambooted nodes.

```
bhost          bproc-3.2.6.tar.gz  lam-7.0.6        linux  sbin
bin            etc                 lam-7.0.6.tar.bz2  man    share
bproc-3.2.6    include            lib              mpi    testcluster
```

6. Some typical problems

The fussing lam does is ok - the important part is that it actually executed `ls` on all nodes, so we know `lamd` is basically up and running and access as well as `stdout` redirection is working as expected.

So what is left now is that `a.out` was compiled on the local node but not uploaded to the other node and thus can't be found there.

```
hofrat@rtl20:~$ ssh -l hofrat rtl21 which a.out
which: no a.out in (/home/hofrat/bin:/usr/local/sbin:/usr/sbin:/sbin:/usr/local/bin:/usr
```

obviously `a.out` is in none of the configured locations, as `which` prints the content of `$PATH` we can see if we need to add some path to the `.profile` (see above).

With such generic names like `a.out` one must be very careful though, the probability of one catching some other file than what you actually are intending to execute is quite large with such a default name - any way this would verify if `a.out` can be found in the users path or not.

After uploading the file to the other nodes, i.e. with:

```
hofrat@rtl20:~$ cd mpi
hofrat@rtl20:~/mpi$ scp a.out hofrat@rtl21:~/mpi/
```

And adding `/mpi` to our `PATH` variable on all nodes, we should be set. So now:

```
hofrat@rtl20:~/mpi$ mpirun -v n0-1 a.out
```

should run our first mpi application smoothly. If not - please send me your error that it can be added in here (<mailto:der.herr@hofr.at>) - thanks !

7. List of Acronyms

LAM - Local Area Multicomputer

MPI - Message Passing Interface

SSI - System Service Interface

ssh - Secure SHell

rpi - Request Progression Interface

8. Cluster Software Sources

lam-7.0.6.tar.bz2 - <http://www.lam-mpi.org/7.0/download.php>
Slackware 9.1 iso images - <http://www.slackware.org/getslack/>