Installation of Marabou

Nigel Warr

July 2016

Contents

1 Introduction

The Marabou system has two parts to it:

- the readout which runs on the power PC under Lynx OS
- the analysis which runs on the DAQ computer under Linux

The power PC has no hard drive of its own, so it must load its operating system via tftp and mount its directories via *nfs*[1]. The DAQ computer acts as server. The two are connected on a private network, so the DAQ computer needs a second ethernet card.

2 The boot procedure of the power PC

First of all the BIOS of the power PC puts out a *bootp*[2] request. This is replied to by the dhcp[3] server on the main DAQ computer. It is on a private network, so there is only the DAQ computer which can reply, so it needs a dhcp server. For our purposes bootp and dhcp are synonymous.

Once the DAQ computer has replied and assigned an IP address to the power PC, it has an IP address but still no kernel and no file system. So its BIOS puts out a *tftp*[4] request. Again, it is the DAQ computer which must reply and provide it with the appropriate image, which contains the Lynx OS kernel for the particular type of power PC. Miniball uses the Lynx OS v2.5 version on a RIO-2.

The power PC does not use a nameserver but has the correspondence of IP address and name hardcoded. Note that the names used by the power PC have no relation to the ones on the network. In particular the DAQ computer is called minidaq, irrespective of its name on the normal network.
At this stage, the power PC can boot but it still has no files. During the boot procedure, it makes an nfs mount request to access /usr/diskless/lynx/RIO2_2.5 and /mbdata on the DAQ computer. The power PC mounts the /mbdata of the DAQ computer on its own /mbdata, but it mounts minidaq:/usr/diskless/lynx/RIO2_2.5/ on /nfs of its system.

3 IP addresses on the private network

The IP addresses on the private network are fixed.

[1] nfs = network file system.
[2] bootp = bootstrap protocol.
[3] dhcp = dynamic host configuration protocol, which is very similar to bootp.
[4] tftp = trivial file transfer protocol.

IP address	Name	Description
192.168.1.1	minidaq	main DAQ computer
192.168.1.2	miniaf	autofill computer
192.168.1.3	minihv	CAEN SY2527 HV mainframe
192.168.1.10	ppc-0	Power PC
192.168.1.11	ppc-1	reserved for another Power PC
192.168.1.12	ppc-2	reserved for another Power PC
192.168.1.13	ppc-3	reserved for another Power PC
192.168.1.14	ppc-4	reserved for another Power PC
192.168.1.15	ppc-5	reserved for another Power PC
192.168.1.16	ppc-6	reserved for another Power PC
192.168.1.17	ppc-7	reserved for another Power PC
192.168.1.18	ppc-8	reserved for another Power PC
192.168.1.19	ppc-9	reserved for another Power PC
192.168.1.20	ppc-10	reserved for another Power PC
192.168.1.21	ppc-11	reserved for another Power PC

Normally, HV mainframe and the autofill computer are on a different private network to the main DAQ computer and the power PC, but reserving these IP addresses ensures that a single computer can perform either role, merely by plugging and unplugging a few things. The additional ppc-1 to ppc-11 are reserved in case we ever want more VME crates.

On the DAQ computer these should be entered into the */etc/hosts* file by adding lines as appropriate. The syntax is the IP address followed by the name separated by spaces; one entry per line.

4 Minicom

If the power PC fails to boot for whatever reason, it is very hard to tell what is happening. For this reason, it is a good idea to connect the serial connector and use *minicom*. The power PC has a special type of connector for its RS232 serial connection, so you need the special cable. It is best to leave it connecting the power PC to the RS232 port of the DAQ computer all the time, so the cable doesn't get lost! The software *minicom* needs to be installed (this is included in the standard distributions like CentOS 6 etc.)

Minicom needs to be run as root, so log into the main DAQ computer as root and start it with

```
minicom
```

The first time you run minicom, you should run with

```
minicom -s
```

in order to set the device, otherwise it won't start at all. You can also set the baud rate to 9600, with no parity, 8 data bits and 1 stop bit.

Press CTRL-A to get a status line at the bottom. This should indicate something like:

```
CTRL-A Z for help | 9600 8N1 | NOR | Minicom 2.3 | VT102 | Online 00:02
```

The important part is the "9600 8N1" which gives the baud rate, bits, parity and stop bits. If this is something else, you need to change it. Pressing CTRL-A again will remove the status line.

To get to the "Comm Parameters" menu, press "p" when you have this status line. Press the appropriate letters (as indicated) to select 9600 speed, no parity 8 bits of data and 1 stop bit and then press enter. This is a sequence like "c" then "q" and then enter.

Then go to the "configuration" menu by pressing CTRL-A and "o" and select "Save setup as dfl", then "exit".

You should get to the boot menu of the power PC at this stage.

Use CTRL-A and "x" and then "yes" to exit.

5 Configuring the second ethernet card

The old DAQ computer (pcepuis16) has, for some reason, three ethernet ports, two on the mainboard (eth0 and eth1) and one on an additional card (eth2). We use eth1 for the normal ethernet connection to the outside world and eth2 for the private network to the power PC. For some reason we don't use eth0. You can use the tool *system-config-network* to set up the network. The settings for eth2 should be:

Name	eth2
Device	eth2
Use DHCP	not set
Static IP	192.168.1.1
Netmask	255.255.255.0
Default gateway IP	192.168.1.254
Primary DNS Server	not set
Secondary DNS Server	not set

Obviously, if you use eth1 the name and device are different. Make sure the network is restarted after you change it.

The new DAQ computer (hpdaqpc) has one onboard Intel ethernet port (eno1) and we added an additional Realtek PCIexpress card which provides an additional one (ens5). We use the onboard card for the CERN network and the additional card for the power PC. Note the odd names!

6 Setting up the firewall

The best solution is to turn on the firewall, allowing just ssh for the CERN network and allow everything on the private network to the power PC. So we have something like the following in /etc/sysconfig/iptables

```
# Firewall configuration written by system-config-firewall
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -i eth2 -j ACCEPT
-A INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A INPUT -m state --state NEW -m udp -p udp --dport 7001 -j ACCEPT
-A FORWARD -m state --state ESTABLISHED,RELATED -j ACCEPT
-A FORWARD -p icmp -j ACCEPT
-A FORWARD -i lo -j ACCEPT
-A FORWARD -i eth2 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

which was produced by running system-config-firewall with the following in */etc/sysconfig/system-config-firewall*:

```
# Configuration file for system-config-firewall

--enabled
--trust=ens5
--port=7001:udp
--service=ssh
--service=nfs
```

Note that since we trust ens5, where the power PC is, it doesn't need nfs open, but this allows us to mount the partition from another machine on the CERN network. We also need ssh to let us log in remotely.

7 Setting up dhcp

First install the *dhcp* package. This ships with the Linux distributions, but is not usually installed by default. Then you need to set up the */etc/dhcp/dhcpd.conf* file which configures it. Note that on some systems this is */etc/dhcpd.conf*. The file should look as follows:

```
#
# DHCP Server Configuration file.
# Miniball config
# R. Lutter Apr-2007
#

use-host-decl-names on;
ddns-update-style none;
deny client-updates;

authoritative;

subnet 192.168.1.0 netmask 255.255.255.0 {
    option subnet-mask 255.255.255.0;
    option routers 192.168.1.1;
    option broadcast-address 192.168.1.255;
    option domain-name-servers 192.168.1.1;
    option ip-forwarding off;
    next-server 192.168.1.1;
    range 192.168.1.128 192.168.1.200;
}

group rio2 {
  next-server 192.168.1.1;
  server-name "minidaq";

  host ppc-0-amd {
        hardware ethernet 00:80:A2:00:8D:5E;
        fixed-address ppc-0;
        filename "lynx/boot.rio2-amd0";
  }
  host ppc-0-feth {
        hardware ethernet 00:80:A2:00:18:7E;
        fixed-address ppc-0;
        filename "lynx/boot.rio2-feth0";
  }
}

group rio3 {
```

```
  next-server 192.168.1.1;
  server-name "minidaq";

  host ppc-0 {
        hardware ethernet 00:80:a2:01:03:02;
#       hardware ethernet 00:80:a2:01:03:20;
        fixed-address ppc-0;
        option root-path "minidaq:/usr/diskless/lynx/RIO3_3.1";
        filename "lynx/boot.rio3";
  }
}
```

In other words, it is completely rewritten for our purposes! The IP address of the DAQ computer on the private network (192.168.1.1) is given as router and name server. Also the ethernet addresses are given for the specific power PCs, as well as the file they need to boot. This is the file which will be loaded in the tftp step.

Once you have the correct configuration for dhcpd, you need to enable it and start it with:

```
systemctl enable dhcpd.service
systemctl restart dhcpd.service
```

If you get this set up correctly, and press the reset of the power PC, you should get a message like:

```
Sep 25 18:28:20 pcepuis16 dhcpd: BOOTREQUEST from 00:80:a2:00:18:7e via eth2
Sep 25 18:28:20 pcepuis16 dhcpd: BOOTREPLY for 192.168.1.10 to ppc-0-feth (00:80:a2:00:18:7e
```

in the system log of the DAQ computer. Via minicom you should be able to see that it starts to try and access its boot image via tftp. Of course, this won't work because we haven't set up tftp yet.

8 Setting up tftp

First of all you need to install the *tftp-server* package. The tftp server is started from xinetd, so this should pull in the *xinetd* package as well.

When you install tftp-server, by default the service is disabled, so edit */etc/xinetd.d/tftp* and change the line "disable = yes" to "disable = no".

Then you need to set up the directory structure under */var/lib/tftpboot*. In particular you need the files: /var/lib/tftpboot/lynx/boot.* which are the boot images. If you are using the one with the hardware address 00:80:a2:00:18:7e as in the example above, it needs the file */var/lib/tftpboot/lynx/boot.rio2-feth0*.

If you no longer have these files, there is a backup on the castor system at:

```
/castor/cern.ch/isolde/miniball/tftpboot_for_power_pc.tar
```

To read it you need to log in as isolcdr on the main DAQ computer (you don't need a password when logging in from the miniball account via ssh). Then make sure it is staged:

```
stager_qry -M /castor/cern.ch/isolde/miniball/tftpboot_for_power_pc.tar
```

If this gives a "No such file or directory" message, then you need to sage it with:

```
stager_get -M /castor/cern.ch/isolde/miniball/tftpboot_for_power_pc.tar
```

Staging means that the file was on tape and the robot has to find the right tape and mount it, then transfer the file to disk, so it can take quite some time (up to a few hours). Keep querying the status with stager_qry until it is "STAGED".

If it is something like "STAGED" or "CANBEMIGR", you can just copy it with:

```
rfcp /castor/cern.ch/isolde/miniball/tftpboot_for_power_pc.tar .
```

and then unpack the tar file so the files are in the right place.

Finally you need to enable and start the xinetd service.

```
systemctl enable xinetd.service
systemctl restart xinetd.service
```

Now, if you try to boot the power PC, it should succeed in downloading its boot image and starting its kernel. However, it will still be unable to mount the file systems it needs via nfs.

9 Setting up nfs

Make sure the package *nfs-utils* is installed. As the power PC does not understand nfsv4, we have to disable this, by setting the RPCNFSDARGS variable in */etc/sysconfig/nfs*:

```
RPCNFSDARGS="-N 4"
```

hen enable and start the service.

```
systemctl enable nfs.service
systemctl restart nfs.service
```

On the DAQ computer, add the following to */etc/exports*

```
/usr/diskless/lynx 192.168.1.10(rw,no_root_squash,sync)
/home              192.168.1.10(rw,no_root_squash,sync)
/mbdata            192.168.1.10(rw,no_root_squash,sync)
/usr/daq           192.168.1.10(rw,no_root_squash,sync)
```

Then run the command following command as root:

```
exportfs -ar
```

After this, if you reboot the power PC again, it should go all the way through to the login prompt.

10 Logging into the power PC

Before you log in, you need to install the *rsh*[5] package.

Then create a file */mbdata/miniball/.rhosts* on the main DAQ computer with the following contents:

```
ppc-0 miniball
minidaq miniball
localhost.localdomain miniball
localhost miniball
```

Note, that we don't need to use the names like pcepuis16 or hpdaqpc in this file, because the power PC only knows the DAQ computer as "minidaq" irrespective of what its name on the CERN network is.

Then you can do the following as user miniball:

```
rsh ppc-0
```

You should not need a password. At this point you have the power PC up and running. Note that since, marabou itself has to be able to log in, logins without passwords are essential! Note also, that the names as seen from the power PC are different from those seen from the DAQ computer.

11 Obtaining marabou

Install the *subversion* package and then run:

```
svn co https://svn.physik.uni-muenchen.de/repos/marabou/trunk/marabou marabou
```

This should create a directory *marabou* with all the source code under it.

[5] rsh = remote shell.

12 Patching marabou

In the marabou directory containing the source code, create a file called *MyConfig.mk* containing:

```
EXTRA_CFLAGS += $(shell root-config --cflags)
EXTRA_CXXFLAGS += $(shell root-config --cflags)
```

In *expconf/inc/TMrbEvent.h* add the line

```
#include "TMrbConfig.h"
```

just before it includes TMrbLofNamedX.h.

13 Compiling marabou

In the marabou directory, run:

```
./configure
make clean
mkdir bin obj lib include
make
```

14 Installing marabou

To install under /usr/daq/marabou-svn_20160704 you can then do the following:

```
make BINDIR=/usr/daq/marabou/svn_20160704/bin \
     LIBDIR=/usr/daq/marabou/svn_20160704/lib \
     INCDIR=/usr/daq/marabou/svn_20160704/include \
     DATADIR=/usr/daq/marabou/svn_20160704/data \
     MACRODIR=/usr/daq/marabou/svn_20160704/macros \
     ICONPATH=/usr/daq/marabou/svn_20160704/icons \
     SOUNDPATH=/usr/daq/marabou/svn_20160704/sounds \
     TEMPLDIR=/usr/daq/marabou/svn_20160704/templates \
     OBJDIR=/usr/daq/marabou/svn_20160704/obj \
     DOCHPRDIR=/usr/daq/marabou/svn_20160704/doc install
```

Note that we install under /usr/daq because that directory is mounted on the power PC.

15 Using marabou

In order to use marabou, you need to define some environment variables as follows:

```
export MARABOU=/usr/daq/marabou/svn_20160704
export PATH=$PATH:$MARABOU/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$MARABOU/lib
export CPLUS_INCLUDE_PATH=$CPLUS_INCLUDE_PATH:$MARABOU/include
```

These lines can be added to the *.bash_profile* in your home directory.

You also need a .rootrc file in your home directory. This needs at least to have:

```
Unix.*.Root.MacroPath:      .:macros:config:~/rootmacros:~/scripts:$MARABOU/macros
```

16 Testing the power PC side

We can test the low-level MBS side of things by logging onto the power PC and then starting mbs, then running the startup.scom script and then starting the acquisition. To do this we do:

```
mbs
@startup
sta acq
```

Note that it is important that no other acquisition processes are running. If it starts up, we can issue commands like "show rates" to see that it is really acquiring something. Then "sto acq" to stop.

We can use "resl" from the shell (i.e. not from mbs) to reset and kill all the mbs processes.

17 Troubleshooting

17.1 Segmentation fault on Clear MBS in C_analyze

If the "Connect to" field doesn't have "ppc-0", it crashes. So set it correctly.

17.2 Only gar-ex-ppcXX available in C_analyze

If the "Connect to" field has only names like "gar-ex-ppc01" etc. add the line

```
TMrbAnalyze.UseShortPPCNames: TRUE
```

to the .rootrc file.

17.3 Segmentation fault in DGFControl

If C_analyze crashes with "*** Break *** segmentation violation", add:

```
Gui.IconPath: $MARABOU/icons:$ROOTSYS/icons
```

to the .rootrc file.

17.4 Segmentation fault in C_analyze

If C_analyze crashes with

```
Error in <TGRadioButton::TGRadioButton>: rbutton_*.xpm not found
```

add:

```
Gui.IconPath: $MARABOU/icons:$ROOTSYS/icons
```

to the .rootrc file.

17.5 Need dynamic mapping

If starting the mbs test procedure on the power PC gives messages like

```
-ppc-0  :read_meb   :[mapVME] adc1: Static mapping 0xf10000 -> 0x80121000, addrMod=0x9
```

and then at some point an "Out of free memory" this is because we should be using dynamic mapping. To do this add:

```
TMrbConfig.VMEMapping: 4
```

to the .rootrc file, then redo the "root Config.C" step and recompile the code on the power PC.