Linux MENDEL Installation How-to

Last updated on June 23, 2010

MENDEL Software

The Linux source distribution for Mendel's Accountant can be downloaded from: http://sourceforge.net/projects/mendelsaccount . To install the software, as root user run the following commands:

```
# tar xvfz mendel_v1.4.7.tar.gz
# cd mendel_v1.4.7
# ./configure
# make
# make install
```

Note, in the above statements, replace mendel_v1.4.7.tar.gz and mendel_v1.4.7 with the specific name of the Linux version you downloaded. The configure script will check to make sure all the needed software requirements are installed on your system and are compatible with Mendel's Accountant. It will also prompt the user for where the files should be installed, and how much memory should be allocated for Mendel. By default, the configure script will configure Mendel's Accountant to run using (a) the C engine (which is slower than the Fortran version), (b) without multi-tribe support, and (c) without a queuing system (which will allow a Linux user or users to submit as many jobs as they like, and will queue the jobs and schedule them when resources are available). The only third-party software package that is required for the default install of Mendel's Account is Gnuplot. To be more specific, the only third-party software that does not come pre-installed on typical Linux systems – e.g. both Apache web server and Perl is also required but typically is provided with most Linux distributions. In case the user wants to run with multi-tribe support, or use it in conjuction with a queuing system, the following instructions are provided as a reference manual.

Installing and configuring third-party software

MENDEL requires the following software three software packages to be already installed and properly working on the Linux system: Gnuplot, Apache webserver, and Perl. Apache and Perl are typically already installed on most Linux systems. There are a couple additional packages that are needed if running with multiple tribes, or if running on a large multi-user Linux cluster. This document explains in detail how to install these auxiliary packages.

Software package	**Supported version**	**Packaged with Fedora**	**Website**
Gnuplot	>= 4.2.3	Yes	http://gnuplot.info
Apache	>= 2.0	Yes	http://apache.org
Perl	>= 5.8.8	Yes	http://perl.org
[*]MPICH2	>= 1.0.5	No	www.mcs.anl.gov/research/projects/mpich2
[†]TORQUE	1.2	No	http://ClusterResources.com
Intel Fortran Compiler	>= 8.0	No	http://software.intel.com/en-us/intel-compilers/

In addition, if using Linux as a MENDEL client (i.e. run the software on a Linux system), the Konqueror web browser must be used (Firefox will not work correctly). Install the Konqueror web browser, by installing kdebase package.

Gnuplot

Gnuplot 4.2 must be installed with support for PNG. The easiest way to install Gnpulot is by using yum as root:

```
        # yum install gnuplot
```

If yum is not installed, the source code can be download from http://gnuplot.info. Then compile and install. After installation, please test that it is indeed 4.2 by using the following command:

```
        # gnuplot --version
```

Pre 4.2 versions will encounter some errors when plotting MENDEL results. Also to be able to create the needed PNG files for the plots, Gnuplot 4.2 and later uses a library libgd (www.libgd.org), which in turn uses a library called libpng (http://sf.net/projects/libpng). Therefore, if gnuplot is compiled manually (instead of using rpm or yum), both libgd and libpng must first be installed. The proper order of installing these packages should be as follows:

1. first install libpng from http://sourceforge.net/projects/libpng (click Download, libpng-stable, then libpng-1.2.37.tar.gz) or try the direct link:

[*] This package is optional. It is only needed if running with multiple tribes.

[†] This package is optional. It is only needed if running on a Linux cluster, as opposed to a single Linux workstation or laptop.

http://internap.dl.sourceforge.net/sourceforge/libpng/libpng-1.2.37.tar.gz

2. Then, install libgd (www.libgd.org) -- click Downloads, then click .tar.gz under "Download the latest" or try the direct link: http://www.libgd.org/releases/gd-2.0.35.tar.gz

3. After libgd is installed, make gnuplot in the following manner:

```
a. # cd gnuplot-4.2.3
b. # ./configure --with-png=/usr/local/lib
c. # make
d. # make install
```

Apache 2.2.3 Server

Once the Apache server is installed, make sure that it is running. One way is by opening a browser, and opening the URL: http://127.0.0.1. If the file /var/www/html/index.html (and is set with proper permissions 644), you should see the contents there. If no index.html file exists, you should see a file titled "Apache test page". Another way to test if the sytem is running is by typing the command:

- `ps –aux | grep httpd`

The configuration for the Apache web server is `/etc/httpd/conf/httpd.conf`. By default, this should be setup to allow programs located in the default cgi-bin path: `/var/www/cgi-bin` to execute.

Perl

Perl is usually installed at the time when Linux is installed. You can test if it is already installed by typing:

```
# perl --version
```

If it is not installed, you can install one of three ways, listed from easiest to most difficult:

1. yum install perl
2. rpm –i perl-5.8.8-10.i386.rpm (using from Linux installation CD)
3. downloading from perl.org and compiling

Also, for the ability to import case parameters from other servers, perl-libwww must be installed. This can be installed using the following command:

```
# yum install perl-libwww-perl
```

MPICH

MPICH stands for Message Passing Interface Chameleon. The latest version, MPICH2, can be downloaded from http://www.mcs.anl.gov/research/projects/mpich2/. Download the platform labeled “Source Code (1.0.8p1) mpich2-1.0.8p1.tar.gz”. To install type the following as root:

```
# tar xvfz mpich2-1.0.8p1.tar.gz
# cd mpich2-1.0.8p1
# ./configure
# make
# make install
```

There are more instructions below for configuring MPICH. The configure script will automatically check to see if MPICH2 has already been installed, and will check to make sure that it is setup correctly.

TORQUE Resource Manager

The Torque resource manager (TRM) is a scheduler which manages and schedules MENDEL processes on available processors. It is an updated version of the OpenPBS (PBS – portable batch system) software and is also freely available for download via http://www.clusterresources.com. To configure to Torque to work with Mendel’s Accountant, first download and install the Torque package according to the instructions which follow, then run “./configure --with-torque”, then “make”, then “make install”.

Installing Torque. After downloading torque, unpack it as root user:

```
# cd /usr/local/src
# tar xvfz /mydir/…/torque.tgz
# cd /usr/local/src/torque
# ./configure --with-default-server=localhost
Note: If just a standalone machine (not cluster), localhost
should work as hostname. However, if it is a cluster machine,
replace localhost the appropriate hostname of the cluster head
node.
# make
# make install
# ./torque.setup root
```

This will compile Torque for your specific system, as well as install some necessary programs for schedule management such as `qmgr`, `qsub`, `qstat`, `pbsnodes`, `pbs_mom`, which are all called via MENDEL’s perl modules. The torque.setup script will create a default queue called batch. These Torque programs are briefly described here:

- qsub - program used for submitting new MENDEL jobs to the queue
- qmgr – pbs batch system manager

- qstat – show status of pbs batch jobs
- pbsnodes – used to control status of compute nodes. Pbsnodes will mark the nodes as either free, down, or offline.

On some systems, the user apache as executed through the web server will not be able to find the default web server in `/var/spool/torque/server_name`. To get around this problem, the following line should be added to the Apache configuration file `/etc/httpd/conf/httpd.conf`:

```
       SetEnv PBS_DEFAULT my.server.edu
```

After modifying httpd.conf, the Apache web server must be restarted by either of the following commands: sytem-config-services or chkconfig httpd restart.

After setting up, you should also add the file /var/spool/torque/mom_priv/config to contain the following contents. For example, it should look like the following for the PBS server named *hs84* with a compute node named *localhost*:

```
$pbsserver         hs84                  # note: hostname running
pbs_server
$logevent      255             # bitmap of which events to log
$clienthost    localhost
```

Adding nodes to the system.

Nodes can be added to the scheduler using the Queue manager: `qmgr`. As root type:

```
       # qmgr
       qmgr> create node localhost np=1
       qmgr> create node c01 np=2
       qmgr> create node c02 np=2
              :      :
```

This will modify the file `/var/spool/torque/server_priv/nodes` on the head node. If the nodes each have more than one processor, this should be specified when the node is created. For more help, type help create node at the qmgr prompt. The name of these nodes (e.g. here head c01 c02) must be consistent with the names provided in `/etc/hosts` file. A simple test you should do is make sure you can ping each of your compute nodes. For example,

```
       # ping localhost
       # ping c01
       # ping c02
          :
```

If they do not respond to the ping command, they will not work with PBS. If this is just a single computer, you can define a single compute node named *localhost* (assuming it is defined as such in

/etc/hosts file).

Starting the scheduler

In order for the scheduler to be up and running, two programs must be started by the super/root user on the head node: pbs_server and pbs_sched. These programs can simply be started by the root user typing in each of these commands each time the head node is booted. In order to have the system automatically startup the PBS/Torque server, you can include three additional scripts in /etc/rc.d/init.d named pbs_mom, pbs_sched, and pbs_server (see Appendix for example script files). Once the files are setup, you should run chkconfig add for each script (e.g. chkconfig --add pbs_server).

In a similar manner, *pbs_mom must be started on and running each compute node*. Note: the head node can also run a *pbs_mom*, but is not recommended if the head node is only a single processor. If the head node has multiple processors, it is recommended that one process be not allocated and free to server the web server, and other requests. The number of processors allocated for the head node can be changed using the qmgr.

Shutting down the PBS system

To shutdown or restart the PBS system, for the pbs_server, type:

```
# qterm -t quick
```

For pbs_mom or pbs_sched, you can use the following commands:

```
# killall pbs_mom
# killall pbs_sched
```

Intel Fortran Compiler

By default, Mendel installs and uses the C-version of the program. However, Fortran is more efficient at numerical computations, and thus runs considerably faster than the C version. A free Fortran compile can be downloaded at:
http://software.intel.com/en-us/articles/non-commercial-software-development/
After installing the Intel Fortran compiler, to install Mendel's Accountant with Fortran support, use the following command:

```
# ./configure –with-fortran
```

The configure script will prompt the user for the path to the Fortran compiler executable (e.g. /opt/intel/fc/10.1.013/bin/ifort), so it is best to find that information before running the configure script.

Additional System Administration

The following information is essential for the system to be able to work correctly.

NFS Mounting

If you are running on a cluster computer with separate nodes networked together, all the nodes must share a common directory for storing the MENDEL data files. Since apache only has write permissions under `/var/www/html`, it is logical to export this directory via NFS. In order to export this file, the superuser (or root user) must modify the `/etc/exports` file to include the following line:

```
/var/www          10.0.0.0/255.255.255.0(rw,sync,no_root_squash)
/home             10.0.0.0/255.255.255.0(rw,sync,no_root_squash)
/usr/local        10.0.0.0/255.255.255.0(sync)
```

The 10.0.0.0 must be changed to the match the local subnetwork you are using, as defined in the file /etc/hosts. For some systems, this number may be 192.168.*.0 where * represents a number anywhere between 0 and 255. For more information about exporting file systems via NFS, type `man exportfs`. Once this file has been modified, the root user must run the command `exportfs –a` to update the NFS system. Also, one must make sure the NFS service is running. This can be started several ways. The easiest way, if running Fedora Linux, is by executing the following program as root user: `system-config-services`. Select NFS and click start, and make sure to click Save, so that NFS will automatically be started everytime the system is booted. Alternatively, this NFS can be started via the following command:

```
    # chkconfig nfs --levels 345 on
```

Exporting the directories from the head node is only half the battle. The next step is to include instructions for mounting the exported directory on each of the compute nodes. This is accomplished by adding the following line to `/etc/fstab` for each of the compute nodes:

```
    192.168.1.1:/var/www (ro,sync)
```

Before this directory can be mounted, `/var/www` must be created and completely empty on every compute node. If `/var/www` already exists on the compute nodes, it can be renamed to `/var/www.local` and a new `/var/www` created via `mkdir /var/www`.

Security

One thing that will cause much frustration in getting things working on the server is Linux's internal security firewall. If the firewall is enabled (using iptables and ipchains), the following ports must be opened: SSH, WWW (HTTP), NFS. It is recommended during the setup process, that the firewall be disabled until once everything is setup, then re-enabled and tested to make sure everything still works with the firewall. The easiest way to manage open these ports and enable/disable the firewall is by using the program: system-config-securitylevel in Fedora. Once the changes are made and saved, the new settings are effectively immediately without even having to reboot the computer.

Setting up rsh-server

MPI uses remote shell (rsh) to communicate between nodes. Since rsh in the past has been used by many hackers in the past to break into Unix/Linux systems, Linux systems such has Fedora have made it not trivial to setup an rsh-server. Included here are the basic instructions for setting up the rsh-server. The rsh-server is handled by xinetd (if xinetd has not been installed it must be installed) which only launches the programs when they are needed (rather than having them running constantly in the background). Note, this must be setup and running on each compute node.

```
# yum install xinetd
# yum install rsh-server
# cd /etc/xinetd.d
```

The following files must be modified: rexec, rsh, rlogin, kexec, klogin, ksh. For each of these files one line must be changed in the following manner:

```
# disabled = yes
```

must be changed to:

```
# disabled = no
```

Once these changes are made and saved, xinetd must be restarted using the following command:

```
# /etc/init.d/xinetd restart
```

Note, this rsh-server and the corresponding instructions above must be executed on every compute server in the cluster on the system.

Getting MPICH working

Starting the MPD Subsystem

The Apache web server needs to be installed and also running. All MENDEL runs, since they are launched through the web interface, are run as user apache. Thus, several items must be setup for user apache. If using MPICH2, a ring of MPD (multi-purpose daemons) must be created for the …

A file called `.mpd.conf` must be created in user apache’s home directory (/var/www). This will have to be done by the root user, since the permissions in /var/www do not allow write permission by user apache. Thus, the following instructions are given for root user:

```
# cd /var/www
# echo “MPD_SECRETWORD=nr76-k2x” > .mpd.conf
# chown apache.apache .mpd.conf
# chmod 600 .mpd.conf
```

The MPD_SECRETWORD should be a unique password. This identical file (either by NFS mounting /var/www or copying the file to each node) should exist in /var/www on all nodes of the cluster.

For MPICH2, a ring of MPD (multi-purpose daemons) must be started for user apache. Doing this requires allowing user apache to login with shell access.

On Ubuntu/Debian systems, Apache uses the user “www-data”, which should already be setup on your system. You may check /etc/passwd to verify that the following line should exists:

```
www-data:x:33:33:www-data:/var/www:/bin/sh
```

On Fedora/CentOS/Redhat systems,the file /etc/passwd must be modified by root user on *every* node and the following line should be changed from:

```
apache:x:48:48:Apache:/var/www:/sbin/nologin
```

to:

```
apache:x:48:48:Apache:/var/www:/bin/bash
```

Then, for security reasons, a password should be given to apache. This is accomplished by typing the following command as root user:

```
# passwd apache
```

Then enter the new password.

Modifying mpdlib.py

A modification must be made to the file mpdlib.py (by default this is installed in /usr/local/bin) to allow user apache to start use MPD. This file is modified by copying the following two lines:

```
elif hasattr(os,'getuid')  and  os.getuid() == 0:    # if ROOT
       parmsRCFilename = os.path.abspath('/etc/mpd.conf')
```

and pasting them just after the previous line, and making the following modifications:

```
        elif  hasattr(os,'getuid')    and    os.getuid()  ==  48:    #  if
apache
                parmsRCFilename = os.path.abspath('/var/www/.mpd.conf')
```

If you are using a Debian/Ubuntu system, instead of using uid 48 above, you will need to use 33, the uid of the user "www-data". You may want to check /etc/passwd, and double check that www-data is using a uid of 33. Try running "id www-data" to get the uid of user www-data.

Managing the MPD daemons

Once user apache is setup for shell access, login as user apache ("su – apache"). Then you will need to start a ring of MPD daemons. This is done by the following command (in this case, for example, we will start 4 daemons by on nodes listed in the file /etc/mpd.hosts):

```
# mpdboot --rsh=rsh –n 4 –f /etc/mpd.hosts
```

If the machine is not a cluster, but a single node (standalone server), one may simply type:

```
# mpdboot
```

where mpd.hosts (which needs to be created by the root user) is very similar to /var/spool/torque/server_priv/nodes, containing a list of nodes you want to start (but without np= statements). For example:

```
localhost
c01
c02
c03
```

The process of starting this ring of daemons is usually not trivial and may be the source of much frustration. mpdboot assumes all the pbs_mom's are running on all the compute nodes, no nodes are down, and communication is working properly between all nodes. If it works correctly, it should not give any output or error messages when executing the mpdboot command. Moreover, this MPD ring can be verified by typing: mpdtrace which should output a list of nodes in the ring. This is the easiest way to start an MPD ring. Alternatively, an MPD ring can be started by executing an "mpd" command on each node separately. See "mpd --help" or "mpdhelp" for more instructions.

Setting up user accounts (optional)

One may want to setup username and passwords to control use of computer. This can be done by creating a file named .htaccess in /var/www/html/mendel1.0.3. This file should contain the following contents:

```
AuthUserFile /var/www/html/mendel/.htpasswd
AuthGroupFile /dev/null
AuthName Password-Protection
AuthType Basic

<Limit GET POST>
Require valid-user
</Limit>
```

Then, there must be a corresponding file in /var/www/html/mendel/.htpasswd which contains the usernames and encrypted passwords. Encrypted passwords can be generated by using the command htpasswd. One way to use this command to create a password for bob is as follows:

```
httpasswd –n bob
New password:
Re-type new password:
bob:27BAKy3Efkrbk
```

This last line should then be copied into the file /var/www/html/mendel/.htpasswd, which only contains several of these lines with usernames, and encrypted passwords for different users.

In order for this file to work, the `AllowOverride` option for the /var/www/html directory (in the file /etc/httpd/conf/httpd.conf) must be set to `All` is shown below:

```
<Directory "/var/www/html">
        Options Indexes Includes FollowSymLinks
        #AllowOverride None
        AllowOverride All
        Allow from all
        Order allow,deny
</Directory>
```

Once changes are made to httpd.conf, the Apache web server must be restarted. This can be accomplished by executing the following command as root user:

```
# httpd -k restart
```

When testing whether the new security is going to work in a browser, if the website has already been accessed via the browser, the web browser must be exited, and then restarted.

Troubleshooting

Note: Security-Enhanced Linux (SELinux) may prevent MENDEL operating correctly. There should be a way to adjust the fine-grained conrols to allow MENDEL to work properly. If you are using a Fedora sytem, you can run the command system-config-securitylevel. The easy fix is to change the state of SELinux from Enforcing to Permissive mode.

I get 'Unauthorized Request' to every qmgr command:
The $PBS_HOME/server_name needs to be the same name as that associated with your primary interface. Hint: type “ifconfig” to find IP address associate with primary network interface (e.g. eth0 or eth1). Then make sure /etc/hosts file has the entry you are specifying as PBS_SERVER after that IP address.

I get the error message “qsub: Bad UID for job execution”:
This message may have to do with the fact that pbs_mom is not correctly running or that the node is down. You can run pbsnodes command and see if the node is down. You can manually change the state of the node by starting qmgr and typing at the qmgr prompt “set node localhost state = free” (assuming the compute node is named localhost).

Cannot connect to default server host 'localhost' - check pbs_server daemon.
qmgr: cannot connect to server (errno=111) Connection refused

This message is given when running one of the Torque commands (`qmgr, qstat, etc)` message and it means it does not like the way the localhost has been specified in the /etc/hosts file. Make sure localhost is specified as follows in /etc/hosts:

```
127.0.0.1        localhost        localhost.localdomain
```

PBS nodes are marked as "down"

If compute nodes are marked down when running `pbsnodes` command, this is most likely directly related to pbs_mom (machine oriented mini-server) there are several things to check:

- make sure *pbs_server* and *pbs_sched* are running on the head node, and *pbs_mom* is running on each compute node
- make sure you have setup the /var/spool/torque/mom_priv/config file (see above)
- make sure compute nodes are configured correctly
- Also, check the run log `/var/log/messages` to see if there are any error messages why the node went offline.
- Also, if you are using a firewall, make sure that ports 15,000 – 15,004 are open.

Please visit http://www.clusterresources.com/pages/resources/documentation/common-issues/torque.php for the answer to this question.

APPENDIX

/etc/rc.d/init.d/pbs_server

```
#!/bin/sh
#
# pbsserv: This script will start and stop the PBS server
#
# description: PBS is a batch versitile batch system for SMPs and
clusters
#
### BEGIN INIT INFO
# Provides: pbsserv
# Required-Start: $network  $remote_fs
# Required-Stop: $network
# Default-Start: 3 5
# Default-Stop: 0 1 2 6
# Description: Start PBS server
### END INIT INFO

case "$1" in
  start)
        echo -n "Starting PBS Server"
        /usr/local/sbin/pbs_server
  ;;
  stop)
        echo -n "Stopping PBS Server"
        /usr/local/bin/qterm -t quick
  ;;
  status)
        echo -n "Checking for PBS server: "
        checkproc /usr/local/sbin/pbs_server
  ;;
  try-restart)
        $0 status >/dev/null &&  $0 restart
        ;;
  restart)
        echo "Restarting PBS server"
        $0 stop
        $0 start
  ;;
  *)
        echo "Usage: $0 {start|stop|status|restart|reload}"
        exit 1
esac
```

/etc/rc.d/init.d/pbs_sched

```
#!/bin/sh
#
# pbssched: This script will start and stop the PBS Scheduler
#
# description: PBS is a batch versitile batch system for SMPs and
clusters
#
### BEGIN INIT INFO
# Provides: pbssched
# Required-Start: $network pbsserv $remote_fs
# Required-Stop: $network
# Default-Start: 3 5
# Default-Stop: 0 1 2 6
# Description: Start PBS Sceduler
### END INIT INFO
```

```
case "$1" in
  start)
        echo -n "Starting the PBS Scheduler"
        /usr/local/sbin/pbs_sched
  ;;
  stop)
        echo -n "Stopping the PBS Scheduler"
        killall pbs_sched
  ;;
  status)
        echo -n "Checking for PBS Scheduler: "
        checkproc /usr/local/sbin/pbs_sched
  ;;
  try-restart)
        ## Stop the service and if this succeeds (i.e. the
        ## service was running before), start it again.
        $0 status >/dev/null &&  $0 restart
        # Remember status and be quiet
        ;;
  restart)
        echo "Restarting the PBS Scheduler"
        $0 stop
        $0 start
  ;;
  *)
        echo "Usage: $0 {start|stop|status|restart|try-restart}"
        exit 1
esac
```

/etc/rc.d/init.d/pbs_mom

```
#!/bin/sh
#
# pbsmom: This script will start and stop the PBS mom
#
# description: PBS is a batch versitile batch system for SMPs and
clusters
#
```

```
### BEGIN INIT INFO
# Provides: pbsmom
# Required-Start: $network  $remote_fs
# Required-Stop: $network
# Default-Start: 3 5
# Default-Stop: 0 1 2 6
# Description: Start PMB MOM
### END INIT INFO

# Source function library.
if [ -f /etc/init.d/functions ] ; then
  . /etc/init.d/functions
elif [ -f /etc/rc.d/init.d/functions ] ; then
  . /etc/rc.d/init.d/functions
else
  exit 0
fi

case "$1" in
  start)
        echo -n "Starting PBS Mom"
        /usr/local/sbin/pbs_mom
  ;;
  stop)
        echo -n "Stopping PBS Mom"
        killall pbs_mom
  ;;
  try-restart)
        $0 status >/dev/null &&  $0 restart

        # Remember status and be quiet
  ;;
  status)
        echo -n "Checking for PBS mom: "
        status pbs_mom
  ;;
  restart)
        echo "Restarting PBS Mom"
        $0 stop
        $0 start
        rc_status
  ;;
  *)
        echo "Usage: $0 {start|stop|status|restart|try-restart}"
        exit 1
esac
```