SS03 Build Your Own Cluster

Installation Instructions for Red Hat 8.0 and Clustermatic 3

You should have the following parts:

- 4 single-processor Athlon PCs
- 4 Intel network cards (already installed)
- 4 network cables
- 1 fast ethernet switch and its power adapter
- 1 3Com network card (already installed)
- 1 keyboard
- 1 mouse
- 1 monitor
- 1 power strip
- 5 power cables
- 2 Red Hat 8.0 CD-ROMs (discs 1 & 2, from www.redhat.com)
- 1 Clustermatic 3 CD-ROM (from www.clustermatic.org)
- 1 TCB Summer School CD-ROM (NAMD examples and binaries)
- 4 floppy disks

Part 1: Install Red Hat Linux 8.0 on the Master Node

If you've installed Red Hat before, the following may be quite tedious. If you've never installed Red Hat before, the following may be quite mysterious. It's a necessary evil in either case.

1. Plug the monitor into the power strip and turn it on.
2. Find the machine with two network cards; this is the master node.
3. Plug the master node into the power strip and connect the monitor, keyboard, and mouse.
4. Power on the master node, open the CD-ROM drive, insert Red Hat 8.0 disc 1, and press the reset button. The machine should boot from the CD-ROM.
   If you wait too long and your machine starts booting off of the hard drive just press the reset button to make it boot from the CD-ROM. If your machine still insists on booting from the hard drive you may need to modify its BIOS settings.
5. When the Red Hat welcome screen comes up, type **linux text** before the timeout expires.
   You could do a graphical install if you really wanted to, but these instructions are for text mode, which is useful if you don't have a mouse. If you miss the timeout, press the reset button.
6. Skip testing the CD media.
   This takes far too long and has no real benefit for fresh installs.
7. Say OK to the "Welcome to Red Hat Linux!" message.
8. Select English as your installation language.
9. Select a US model keyboard.
10. If your mouse was not automatically detected, select a generic 3-button PS/2 mouse.
11. Select a Workstation install.
    We typically use Custom, but Workstation is good enough for now.
12. Select Autopartition.
    We don't store files on our cluster machines, even on the master nodes, so it doesn't matter how

the disk is set up. We use dedicated fileservers for storage.

13. Select "Remove all partitions on this system".
    Again, we don't keep data on cluster machines.
14. Yes, you really want to delete all existing data.
    Of course, at home you might not want to do this.
15. Use the GRUB boot loader.
    Clustermatic has problems with LILO, the only other option.
16. There are no special kernel options for the boot loader configuration.
17. Do not set a GRUB password.
18. There are no other operating systems to boot.
19. Install the boot loader on the master boot record.
    Unless you've got another operating system living there that you'd care to run again.
20. For device eth0, use bootp/dhcp and do not activate on boot.
    This will be the interface to the private network. It is configured by the beowulf startup scripts, so you just need to get through installation and keep the normal network startup scripts from messing with it.
21. For device eth1, use bootp/dhcp and do not activate on boot.
    In a production cluster this would be the interface to the outside world, activated on boot with a fixed IP address assigned by your network administrator. Since there is no network for this hands-on session, you can just leave it unconfigured.
22. Customize the high security firewall option it to make eth0 a trusted device and incoming SSH the only allowed protocol.
    Be very careful doing this step, since your slave nodes will not boot if eth0 is not a trusted device!
23. No additional languages.
24. The hardware clock should be set to GMT. Pick your time zone.
25. Pick a root password that you will remember. Write it down.
26. Pick a username and password so you don't have to be root all the time.
    You should only log in as root when you absolutely need to.
27. You don't need to create any more accounts.
    If this will be a multi-user production system you will want to make everyone their own account at some point, but you don't need to do it during initial installation.
28. You don't need to customize the software selection or pick individual packages.
    However, you may want to do this for a production system. This is by far the easiest time to add packages to your cluster. On the other hand, the default install has 2 GB of software, so you could save some time in the next step if you pared the list down.
29. Wait about 15 minutes for the installation to finish, inserting disc 2 when asked.
30. Make a boot floppy.
    Having a Linux boot floppy can be invaluable. A floppy made now will be unable to load kernel modules once you install Clustermatic, but it will still allow you to boot your machine and fix any misconfigurations. You probably won't need it today, though.
31. Accept the suggested video card configuration.
    You would normally only use the console of a production cluster during initial configuration or adding nodes, and you don't need a GUI for either of those, so there is little reason to configure X-Windows. Having multiple terminals available will be useful for this exercise, so we'll go ahead and configure X-Windows anyway.
32. Accept the default monitor configuration, even if it is "Unprobed Monitor".
33. Accept the default color depth and resolution, and the graphical login.
    For a production system you would probably want a text login by default, since you will probably

remove the mouse at some point.

34. Remove the CD-ROM and floppy and reboot the machine.
35. Use the Quit button to skip the additional configuration screens.

Part 2: Install Clustermatic 3 on the Master Node

The following will be new to everyone. You'll need to know how to use a unix text editor. The examples below use the mouse-driven editor "gedit" rather than the more common "vi".

1. Log in as root and open a terminal (in the System Tools menu).
2. **gedit /etc/modules.conf** and swap eth0 and eth1 if necessary so that the network alias lines read:

   ```
   alias eth0 eepro100
   alias eth1 3c59x
   ```

   The hands-on master nodes have two network cards, one from Intel and one from 3Com. Editing /etc/modules.conf ensures that the Intel network card is the private network (eth0) and the 3Com network card is the outside world (eth1). If you had two identical network cards you would need to determine which was which by trial and error. The hands-on slave nodes only have the Intel cards.
3. Remove the conflicting oprofile package with **rpm -e oprofile**
   The message "error: package oprofile is not installed" is OK.
4. Insert the Clustermatic 3 CD and wait for it to mount.
   If you're not running X-Windows you need to mount it with **mount /mnt/cdrom**
5. Install Clustermatic with **rpm --nodeps -Uvh /mnt/cdrom/RPMS/athlon/***
   Substitute the i686 directory if any of your machines have Intel processors. The --nodeps option allows us to ignore some mpich dependencies that we don't care about.
6. Unmount the Clustermatic CD with **eject** and remove it.
7. **gedit /boot/grub/grub.conf** and edit the default line (a zero-based index into the list of kernels that follows) to indicate the 2.4.19-lanl.22 kernel.
   If you had dual-processor or hyperthreaded nodes you would use the 2.4.19-lanl.22smp kernel.
8. **gedit /etc/beowulf/config** and edit the following lines:
   - On the interface line, change "eth1:0" to "eth0":

     ```
     interface eth0 10.0.4.1 255.255.255.0
     ```

   - Change the nodes line to the number of slave nodes you will have:

     ```
     nodes 3
     ```

   - Change the iprange line to provide the corresponding number of addresses:

     ```
     iprange 0 10.0.4.10 10.0.4.12
     ```

   - Change the kernelimage to point at the proper kernel:

     ```
     kernelimage /boot/vmlinuz-2.4.19-lanl.22
     ```

     You would use vmlinuz-2.4.19-lanl.22smp if you had dual-processor or hyperthreaded nodes.
9. **gedit /etc/beowulf/config.boot** to eliminate everything but eepro100 (the module for our private

network card) from the bootmodule line. Just comment out all the old bootmodule lines and replace with this:

```
bootmodule eepro100
```

This makes the boot image much smaller, for faster boots. It is particularly useful if you want to ignore a network port that is built into the motherboard in favor of a separate gigabit card.

10. Reboot the machine to use the new kernel and settings with **reboot**

Part 3: Attach and Boot the Slave Nodes

1. Plug the private network switch into an outlet on the power strip.
2. Connect the master node's Intel network card (this is the same type of card that the slave nodes have, while the other card has a barely legible "3Com" ingraved on it) to the switch.
   In a production cluster, the 3Com card would be connected to the building network. You don't actually have access to the building network today, so you'll need to use your imagination.
3. Log in as root and open a terminal.
4. Create the level 2 boot image with **beoboot -2 -n**
   This builds the second stage boot image, which the slaves will download from the master over the network. You only need to run this command when you change the boot options in /etc/beowulf/config or /etc/beowulf/config.boot.
5. Open a second terminal and run **/usr/lib/beoboot/bin/nodeadd -a eth0** there. The nodeadd program will run until you kill it with Ctrl-C. Leave it running!
   This process is only needed when adding new nodes to the cluster. The nodeadd program captures the hardware ethernet address of any machine trying to boot on the private network (eth0), adds it to node list in /etc/beowulf/config, and makes the beoboot daemon read the new list (-a). When a new node is detected nodeadd will print the hardware address followed by a message about sending SIGHUP to beoserv.
6. For each slave node:
   - Insert an unused floppy in the master node.
   - Write a level 1 beoboot image to the floppy (/dev/fd0) with **beoboot -1 -k /boot/vmlinuz-2.4.19-lanl.22beoboot -f -o /dev/fd0**
   - Move the floppy to the slave node.
   - Connect the slave node network card to the switch.
   - Connect the monitor to the slave node.
   - Power on the slave node and watch it boot from the floppy.
     You will see two sequences of Linux kernel boot messages, one from the first stage beoboot kernel on the floppy and another from second stage kernel downloaded from the master node. The slave is done booting when you see the message "Node setup completed successfully."
   - Connect the monitor to the master node again.
   - Leave the floppy in the slave node, do not remove it.
7. Switch to the second terminal and kill nodeadd with **Ctrl-C**
8. Run **tail /etc/beowulf/config** to see the new (uncommented) node addresses.
9. Check the status of the cluster with **bpstat**
   Make sure as many nodes are up as the number of slaves you have. If you had not modified the nodes and iprange lines in /etc/beowulf/config to match the size of your cluster, you would see the extra nodes harmlessly listed as down.

10. Examine the log file from node 0 with **less /var/log/beowulf/node.0**
    Each node has its own log file in /var/log/beowulf. These log files only contain output from the final stages of slave startup, after the second stage kernel has contacted the master node.
11. View the kernel messages from node 0 with **bpsh 0 dmesg | less**
    The bpsh command allows any binary installed on the master node to execute on one or more slave nodes (see options in the appendix). Interpreted scripts or programs requiring files found only on the master node cannot be run via bpsh.
12. Reboot all the slaves with **bpctl --slave all --reboot**
    If you see any "Node is down" messages, these indicate that fewer than the number of nodes given in /dev/beowulf/config were up when you issued the command.
13. Log out.

Part 4: Running NAMD on the Cluster

1. Log in as the normal user you created during installation.
2. Insert the TCB Summer School CD and wait for it to mount.
   If you're not running X-Windows you need to mount it with **mount /mnt/cdrom**
3. Copy (cp) the following files from /mnt/cdrom/programs/namd to your home directory:
   - NAMD_2.5b2ss03_Linux-i686-Clustermatic.tar.gz
   - apoa1.tar.gz
   - bpti_imd.tar.gz
4. Unmount the CD with **eject** and remove it.
5. In your home directory, uncompress each file with **tar xzf *filename***
6. Run **ln -s NAMD_2.5b2ss03_Linux-i686-Clustermatic NAMD** to make the path to the NAMD binaries easier to type.
7. Check how many slave nodes are up with **bpstat -t allup**, then add one (for the master node) to calculate the number of currently available processors in your cluster.
   If you had dual-processor nodes you would need to multiply this number by two.
8. Run NAMD with **~/NAMD/charmrun +p*N* ~/NAMD/namd2 apoa1/apoa1.namd** where *N* is the number of available processors you found above.
   *While NAMD is running, you can run **top** in a second terminal to see the running processes.*
9. Locate the performance information printed to standard output, i.e., the "TIMING:" lines printed every 20 steps in the apoa1 example. The most relevant value is the "Wall" time per step.
   For a larger cluster, you would want to look for the "Benchmark time:" lines, which are printed after load balancing. Since load balancing is less important for four-node clusters we'll save some time by taking results from the first couple of "TIMING:" lines.
10. Experiment with varying *N* to measure NAMD's speedup.
    The amount of memory needed by the apoa1 simulation is very close to 256 MB, the total memory installed on each node. This may result in very slow performance when running on one node, and very impressive superlinear speedups when running on two nodes.
11. Measure the speedup of NAMD when running bpti_imd/bpti.namd, which is a much smaller simulation.
    Ignore the initial 200 steps of minimization. It may be convenient to pipe (|) the output of NAMD through **grep TIMING** *since the output will scroll by very quickly. The BPTI simulation is much smaller, so you will measure a lower speedup.*
12. Experiment by measuring performance when you ask for more processors than are available.
    Charmrun will create extra namd2 processes across the nodes to reach the specified number.

13. Experiment by measuring performance when you run more than one NAMD job at once.

Appendix: Usage Options for Common Bproc Utilities

bpstat: monitor status of slave nodes

```
Usage: bpstat [options] [nodes ...]
  -h,--help     Display this message and exit.
  -v,--version  Display version information and exit.

 The nodes argument is a comma delimited list of the following:
   Single node numbers - "4"    means node number 4
   Node ranges         - "5-8"  means node numbers 5,6,7,8
   Node classes        - "allX" means all slave nodes with status X
                         "all"  means all slave nodes
 More than one nodes argument can be given.
 Valid node states are:
        down    boot    error   unavailable     up

 Node list display flags:
  -c,--compact      Print compacted listing of nodes. (default)
  -l,--long         Print long listing of nodes.
  -a,--address      Print node addresses.
  -s,--status       Print node status.
  -n,--number       Print node numbers.
  -t,--total        Print total number of nodes.

 Node list sorting flags:
  -R,--sort-reverse Reverse sort order.
  -N,--sort-number  Sort by node number.
  -S,--sort-status  Sort by node status.
  -O,--keep-order   Don't sort node list.

 Misc options:
  -U,--update   Continuously update status
  -L,--lock     "locked" mode for running on an unattended terminal
  -A hostname   Print the node number that corresponds to a
                host name or IP address.
  -p            Display process state.
  -P            Eat "ps" output and augment. (doesnt work well.)
```

bpctl: alter state of slave nodes

```
Usage: bpctl [options]
  -h,--help              Print this message and exit
  -v,--version           Print version information and exit
  -M,--master            Send a command to the master node
  -S num,--slave num     Send a command to slave node num

  -s state,--state state Set the state of the node to state
  -r dir,--chroot dir    Cause slave daemon to chroot to dir
  -R,--reboot            Reboot the slave node
  -H,--halt              Halt the slave node
  -P,--pwroff            Power off the slave node
  --cache-purge-fail     Purge library cache fail list
  --cache-purge          Purge library cache
  --reconnect master[:port[,local[:port]]]
                         Reconnect to front end.
```

```
  -m mode,--mode mode    Set the permission bits of a node
  -u user,--user user    Set the user ID of a node
  -g group,--group group Set the group ID of a node

  -f                     Fast - do not wait for acknowledgement from
                         remote nodes when possible.

The valid node states are:
        down    boot    error   unavailable     up
```

bpsh: run programs on slave nodes

```
Usage: bpsh [options] nodenumber command
       bpsh -a [options] command
       bpsh -A [options] command
       -h     Display this message and exit
       -v     Display version information and exit
  Node selection options:
       -a     Run the command on all nodes which are up.
       -A     Run the command on all nodes which are not down.
  IO forwarding options:
       -n     Redirect stdin from /dev/null
       -N     No IO forwarding
       -L     Line buffer output from remote nodes.
       -p     Prefix each line of output with the node number
              it is from. (implies -L)
       -s     Show the output from each node sequentially.
       -d     Print a divider between the output from each
              node. (implies -s)
       -b ##  Set IO buffer size to ## bytes.  This affects the
              maximum line length for line buffered IO. (default=4096)
       -I file
       --stdin file
              Redirect standard in from file on the remote node.
       -O file
       --stdout file
              Redirect standard out to file on the remote node.
       -E file
       --stderr file
              Redirect standard error to file on the remote node.
```

bpcp: copy files to slave nodes

```
Usage: bpcp [-p] f1 f2
       bpcp [-r] [-p] f1 ... fn directory

       -h     Display this message and exit.
       -v     Display version information and exit.
       -p     Preserve file timestamps.
       -r     Copy recursively.

  Paths on slave nodes are prefixed by nodenumber:, e.g., 0:/tmp/
```

See Also

Clustermatic web site (www.clustermatic.org) and Clustermatic 3 README

BProc: Beowulf Distributed Process Space web site (bproc.sourceforge.net)