```
                                ======   Setting up icecream ======
```

Software

I would recommend a fresh install of -current, and exclude E,KDE,X,XAP,& XFCE as these won't be needed for the client machines.

Post-install, set your machine hostname to whatever your fancy is; *icecream* ignores this and uses the domain name (next prompt), so make sure that all of the machines you intend to use for *icecream* are in the same domain! Next, select static IP and assign it an IP of the form 192.168.x.x, where the first x is usually 0, and the second x can be any number between 1 and 254. (0 and 255 are used for special purposes).

On first boot, the client machines should have the *rc.iceccd* startup script made executable. Select the machine you intend to use as a scheduler machine and make the *rc.icecc-scheduler* script executable. Only one machine can have this script executable! This is usually the main machine. I usually don't set up an additional user account, as these machines are on a private network and totally inaccessible from the Internet; if you feel uncomfortable using root, feel free to add a user. No additional setup for *icecream* is necessary.

There's no need to define CC to point to *icecc* because if /etc/profile.d/icecream.sh sees either iceccd or icecc-scheduler running on the machine, it will add /usr/libexec/icecc/bin to the beginning of the $PATH, and this directory contains symlinks from all the usual compiler names to /usr/bin/icecc.

All there is to do is make */etc/rc.d/rc.icecc-scheduler* executable on one machine, and */etc/rc.d/rc.iceccd* executable on all the machines that will be part of the compile cluster. And then set -j high enough.

Hardware

Obviously, several computers are involved. Just how many is actually immaterial to *icecream*, but can accommodate what spare computers you have lying around.

Connect all computers to a switch (I have all mine connected to a 5-port switch; you may want to use a bigger switch if you have more). Make sure the computer that will be running the scheduler can “see” the client computers using ping.

I have all my machines hooked up to a keyboard and monitor for local control, but if you like to ssh into your machines “remotely”, feel free to do so.

That's it! You can use any number of packages, but I used a kernel compile to demonstrate *icecc*'s power. The number of jobs used will obviously be dependent on the total number of cores in your cluster. I have a total of 20 (1 octa-core machine, 2 quad-core machines and 2 dual-core). I add one extra job per machine, so I use a command like this:

make -j 25

The load on my main machine dropped significantly, from ~115 degrees F to ~80 degrees, and compile time dropped appreciably as well.