The openMosix HOWTO

Kris Buytaert

buytaert@be.stone−it.com

Revision History

Revision v0.70	22 August 2002
Stripped out empty parts, replaced Mosixview with openMosixView	
Revision v0.50	6 July 2002
First openMosix HOWTO	
Revision v0.20	5 July 2002
Latest Mosix HOWTO (for now)	
Revision v0.17	28 June 2002
Revision v0.15	13 March 2002
Revision v0.13	18 Feb 2002
Revision ALPHA 0.03	09 October 2001

Table of Contents

Table of Contents

Table of Contents

Chapter 1. Introduction

1.1. openMosix HOWTO

In the beginning there was Mosix, then came openMosix, in my opinion a more interresting project. Not only from a technical viewpoint but also given the more correct license. I made the decision to focus this HOWTO on openMosix rather than on Mosix, mainly based on the fact that openMosix has a bigger userbase. (Moshe Bar states that about 97% of the old Mosix community has been converted to openMosix.) (20020705) Although lots of information might be valuable to both users of Mosix and openMosix. I decided to mainly split the HOWTO. The latest release of the Mosix HOWTO, containing info about both Mosix and OpenMosix will be 0.20 My intention is to focus on the openMosix HOWTO, however not neglecting the Mosix users. More info on *http://howto.ipng.be/Mosix−HOWTO/*

1.2. Introduction

This document gives a brief description to openMosix, a software package that turns a network of GNU/Linux computers into a computer cluster. Along the way, some background to parallel processing is given, as well as a brief introduction to programs that make special use of openMosix's capabilities. The HOWTO expands on the documentation as it provides more background information and discusses the quirks of various distributions.

Since the creation of this HOWTO some people of the Mosix team created openMosix (more info later) both openMosix and Mosix are being discussed in this HOWTO

Kris Buytaert got involved in this piece of work when Scot Stevenson was looking for somebody to take over the Job, this was during February 2002 While initially we discussed both Mosix and openMosix this version of the HOWTO now mainly focusses on openMosix. Please note that the document often still mentions Mosix where it should read openMosix.

("FEHLT", in case you are wondering, is German for "missing"). You will notice that some of the headings are not as serious as they could be. Scot had planned to write the HOWTO in a slightly lighter style, as the world (and even the part of the world with a burping penguin as a mascot) is full of technical literature that is deadly. Therefore some parts still have these comments

1.3. Disclaimer

Use the information in this document at your own risk. I disavow potential liability for the contents of this document. Use of th concepts, examples, and/or other content of this document is ent at your own risk.

All copyrights are owned by their owners, unless specifically no otherwise. Use of a term in this document should not be regarde affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

You are strongly recommended to take a backup of your system before major installation and backups at regular intervals.

1.4. Distribution policy

1.5. New versions of this document

Official New versions of this document can be found on the web pages of *the Linux Documentation Project* Drafts and Beta version will be available on *howto.ipng.be* in the appropriate subfolder. Changes to this document will usually be discussed on the Mosix and openMosix Mailing Lists. See the *openMosix* and *Mosix Hompage* for details.

1.6. Feedback

Currently this HOWTO is being maintained by Kris Buytaert, please do send questions about Mosix to the mailing lists.

Please send comments, questions, bugfixes, suggestions, and of course praise about this document to the author.

If you have a technical question about Mosix itself, please post them on the (Open)Mosix mailing list. Do not repeat not send them to the Scot who doesn't know squat about the internals, finds anything written in C++ terribly confusing, and learned Python mainly because the rat on the book cover was so cute.

Chapter 2. So what is openMosix Anyway ?

2.1. A very, very brief introduction to clustering

Most of the time, your computer is bored. Start a program like xload or top that monitors your system use, and you will probably find that your processor load is not even hitting the 1.0 mark. If you have two or more computers, chances are that at any given time, at least one of them is doing nothing. Unfortunately, when you really do need CPU power − during a C++ compile, or coding Ogg Vorbis music files − you need a lot of it at once. The idea behind clustering is to spread these loads among all available computers, using the resources that are free on other machines.

The basic unit of a cluster is a single computer, also called a "node". Clusters can grow in size − they "scale" − by adding more machines. A cluster as a whole will be more powerful the faster the individual computers and the faster their connection speeds are. In addition, the operating system of the cluster must make the best use of the available hardware in response to changing conditions. This becomes more of a challenge if the cluster is composed of different hardware types (a "heterogeneous" cluster), if the configuration of the cluster changes unpredictably (machines joining and leaving the cluster), and the loads cannot be predicted ahead of time.

2.1.1. A very, very brief introduction to clustering

2.1.1.1. HPC vs Fail−over vs Load−balancing

Basically there are 3 types of clusters, the most deployed ones are probably the Fail−over Cluster and the Load−balancing Cluster, HIGH Performance Computing.

Fail−over Clusters consist of 2 or more network connected computers with a separate heartbeat connection between the 2 hosts. The Heartbeat connection between the 2 machines is being used to monitor whether all the services are still in use, as soon as a service on one machine breaks down the other machine tries to take over.

With load−balancing clusters the concept is that when a request for say a web−server comes in, the cluster checks which machine is the lease busy and then sends the request to that machine. Actually most of the times a Load−balancing cluster is also Fail−over cluster but with the extra load balancing functionality and often with more nodes.

The last variation of clustering is the High Performance Computing Cluster, this machine is being configured specially to give data centers that require extreme performance the performance they need. Beowulf's have been developed especially to give research facilities the computing speed they need. These kind of clusters also have some load−balancing features, they try to spread different processes to more machines in order to gain performance. But what it mainly comes down to in this situation is that a process is being parallelized and that routines that can be ran separately will be spread on different machines in stead of having to wait till they get done one after another.

2.1.1.2. Mainframes and supercomputers vs. clusters

Traditionally Mainframes and Supercomputers have only been built by a selected number of vendors, a company or organization that required the performance of such a machine had to have a huge budget available for it`s Supercomputer. Lot`s of universities could not afford them the costs of a Supercomputer, therefore other alternatives were being researched by them. The concept of a cluster was born when people first tried to spread different jobs over more computers and then gather back the data those jobs produced. With cheaper and more common hardware available to everybody, results similar to real Supercomputers were only to be dreamed of during the first years, but as the PC platform developed further, the performance gap between a Supercomputer and a cluster of multiple personal computers became smaller.

2.1.1.3. Cluster models [(N)UMA, PVM/MPI]

There are different ways of doing parallel processing, (N)UMA, DSM , PVM, MPI are all different kinds of Parallel processing schemes.

(N)UMA , (Non−)Uniform Memory Access machines for example have shared access to the memory where they can execute their code. In the Linux kernel there is a NUMA implementation that varies the memory access times for different regions of memory. It then is the kernel's task to use the memory that is the closest to the CPU it is using.

PVM / MPI are the tools that are most commonly being used when people talk about GNU/Linux based Beowulf's. MPI stands for Message Passing Interface it is the open standard specification for message passing libraries. MPICH is one of the most used implementations of MPI, next to MPICH you also can use LAM , another implementation of MPI based on the free reference implementation of the libraries.

PVM or Parallel Virtual Machine is another cousin of MPI that is also quite often being used as a tool to create a Beowulf. PVM lives in user space so no special kernel modifications are required, basically each user with enough rights can run PVM.

2.1.1.4. Mosix's role

The Mosix software packages turns networked computers running GNU/Linux into a cluster. It automatically balances the load between different nodes of the cluster, and nodes can join or leave the running cluster without disruption. The load is spread out among nodes according to their connection and CPU speeds.

Since Mosix is part of the kernel and maintains full compatibility with normal Linux, a user's programs, files, and other resources will all work as before with no changes necessary. The casual user will not notice the difference between Linux and Mosix. To him, the whole cluster will function as one (fast) GNU/Linux system.

2.2. The story so far

2.2.1. Historical Development

(To Be Written) The name "Mosix" comes from FEHLT. The 6th incarnation of Mosix was developed for BSD/OS. GNU/Linux was chosen as a development platform for the 7th incarnation in DATE_FEHLT because of

2.2.2. Current state

(To Be Written) Like most active Open Source programs, Mosix's rate of change tends to outstrip the the follower's ability to keep the documentation up to date. See the Mosix Home Page for current news. The following relates to Mosix VERSION FEHLT for the Linux kernel FEHLT as of DATUM FEHLT:

2.2.3. openMosix

openMosix is in addition to whatever you find at mosix.org and in full appreciation and respect for Prof. Barak's leadership in the outstanding Mosix project .

Moshe Bar has been involved for a number of years with the Mosix project (www.mosix.com) and was co−project manager of the Mosix project and general manager of the commercial Mosix company.

After a difference of opinions on the commercial future of Mosix, he has started a new clustering company − Qlusters, Inc. − and Prof. Barak has decided not to participate for the moment in this venture (although he did seriously consider joining) and held long running negotiations with investors. It appears that Mosix is not any longer supported openly as a GPL project. Because there is a significant user base out there (about 1000 installations world−wide), Moshe Bar has decided to continue the development and support of the Mosix project under a new name, openMosix under the full GPL2 license. Whatever code in openMosix comes from the old Mosix project is Copyright 2002 by Amnon Bark. All the new code is copyright 2002 by Moshe Bar.

openMosix is a Linux−kernel patch which provides full compatibility with standard Linux for IA32−compatible platforms. The internal load−balancing algorithm transparently migrates processes to other cluster members. The advantage is a better load−sharing between the nodes. The cluster itself tries to optimize utilization at any time (of course the sysadmin can affect these automatic load−balancing by manual configuration during runtime).

This transparent process−migration feature make the whole cluster look like a BIG SMP−system with as many processors as available cluster−nodes (of course multiplied with 2 for dual−processor systems). openMosix also provides a powerful mized for HPC−applications, which unlike NFS provides cache consistency, time stamp consistency and link consistency.

There could (and will) be significant changes in the architecture of the future openMosix versions. New concepts about auto−configuration, node−discovery and new user−land tools are discussed in the openMosix−mailing−list.

To approach standardization and future compatibility the proc−interface changes from /proc/mosix to /proc/hpc and the /etc/mosix.map was exchanged to /etc/hpc.map. Adapted command−line user−space tools for openMosix are already available on the web−page of the project and from the current version (1.1) Mosixview supports openMosix as well.

The hpc.map will be replaced in the future with a node−auto−discovery system.

openMosix is supported by various competent people (see www.openMosix.org) working together around the world. The gain of the project is to create a standardize clustering−environment for all kinds of HPC−applications.

openMosix has also a project web−page at *http://openMosix.sourceforge.net* with a CVS tree and mailing−list for the developer and user.

2.3. Mosix in action: An example

Mosix clusters can take various forms. To demonstrate, let's assume you are a student and share a dorm room with a rich computer science guy, with whom you have linked computers to form a Mosix cluster. Let's also assume you are currently converting music files from your Cd's to Ogg Vorbis for your private use, which is legal in your country. Your roommate is working on a project in C++ that he says will bring World Peace. However, at just this moment he is in the bathroom doing unspeakable things, and his computer is idle.

So when you start a program called FEHLT to convert Bach's .... from .wav to .ogg format, the Mosix routines on your machine compare the load on both nodes and decide that things will go faster if that process is sent from your Pentium−233 to his Athlon XP. This happens automatically − you just type or click your commands as you would if you were on a standalone machine. All you notice is that when you start two more coding runs, things go a lot faster, and the response time doesn't go down.

Now while you're still typing ...., your roommate comes back, mumbling something about red chile peppers in cafeteria food. He resumes his tests, using a program called 'pmake', a version of 'make' optimized for parallel execution. Whatever he's doing, it uses up so much CPU time that Mosix even starts to send subprocesses to your machine to balance the load.

This setup is called *single−pool*: All computers are used as a single cluster. The advantage/disadvantage of this is that you computer is part of the pool: Your stuff will run on other computers, but their stuff will run on your's, too.

2.4. Components

2.4.1. Process migration

With mosix you can start a process on one machine and find out it actually runs on another machine in the cluster. Each process has it own unique home node (UHN) where it is created.

Migration means that a process is splitted in 2 parts, a user part and a system part The user part will be moved to a remote node where the system part will stay on the UHN and. This system−part is sometimes called the deputy process, this process takes cares of resolving most of the system calls. Mosix takes care of the communication between those 2 processes.

2.4.2. The Mosix File System (MFS)

MFS is a feature of openmosix which allows you to access remote filesystems in a cluster as if they were locally mounted. The filesystem of your other nodes can be mounted on /mfs and you will e.g. find the files in /home on node 3 on each machine in /mfs/3/home

2.4.3. Direct File System Access (DFSA)

Both Mosix and openMosix provide a cluster−wide file−system (MFS) with the DFSA−option (direct file−system access). It provides access to all local and remote file−systems of the nodes in an Mosix or openMosix cluster.

Chapter 3. Features of Mosix

3.1. Pros of Mosix

No extra packages required No Code changes required

3.2. Cons of Mosix

Kernel Dependent Not Everything works this way Shared memory issues

Issues with Multiple Threads not gaining performance.

You won't gain performance when running 1 single process such as your Browser on a Mosix Cluster , the process won't spread itselve over the cluster. Except of course your process will migrate to a more performant machine.

3.3. Extra Features in openMosix

As I write this in august 2002 , the first extra feature in openMosix is a fact, Louis Zechtzer implemented the first version of the auto−discovery daemon

Version 0.0.1 supports:

Auto−discovery of nodes via multicast messaging.
Aliases for hosts with multiple interfaces.
Basic routing (in the case where true multicast routing is undesirable).

The code can be downloaded from: *http://www.coxbit.ch/openMosix/autodisc−0.0.1.tar.gz*

Chapter 4. Requirements and Planning

4.1. Hardware requirements

Installing a basic clusters requires at least 2 machines with network connected. Either using a cross−cable between the two network cards or a switch or hub. Off course the faster your network−cards the easier you will get better performance for your cluster.

These days Fast Ethernet is standard, putting multiple ports in a machine isn`t that difficult, but make sure to connect them through other physical networks in order to gain the speed you want. Gigabit Ethernet is getting cheaper any day now but I suggest that you don`t rush to the shop spending your money before you have actually tested your setup with multiple 100Mbit cards and noticed that you really do need the extra network capacity. Next to putting a gigabit card you might also want to try bonding different 100Mbit cards together.

4.2. Hardware Setup Guidelines

Setting up a big cluster requires some thinking to be done, where are you going to put the machines, not under a table somewhere or in the middle of your office. It`s ok if you just want to do some small tests , but if you are planning to deploy a N node cluster you will have to make sure that the environment that will hold this machine is capable of doing so.

I`m talking about preparing one or more 19" racks to host the machines, configure the appropriate network topology, either straight, single connected or even a 1 to 1 cross connected network between all your nodes. You will also need to make sure that there is enough power to support such a range of machines. That your air−conditioning system supports the load and that in case of power−failure your UPS can cleanly shut down all the required systems. You might want to invest in a KVM Switch in order to facility access to the machines consoles.

But even if you don`t have the number of nodes that justify these investments, make sure that you can always easily access the different nodes, you never know when you have to replace the fan or a hard−disk of a machine in trouble. If that means that you have to unload a stack of machines to reach the bottom one hence shutting down your cluster you are in trouble.

4.3. Software requirements

The systems we plan to use will need a basic Linux installation of your choice, Red Hat , Suse , Debian or another distribution, it doesn't really matter which one. What does matter is that the kernel is at least on 2.4 level, and that your network−cards are configured correctly, next to that you`ll need a healthy space of swap.

4.4. Planning your cluster

How to configure openMOSIX clusters with a pool of servers and a set of (personal) workstations, you have different options that have ll their advantages and disadvantages.

- In a *Single−pool* all the servers and workstations are used as a single cluster: each machine is a part of the cluster and can migrate processes to each other existing node. This off course makes your workstation a part of the pool.

- In an environment that is called a *Server−pool* servers are a part of the cluster while workstations are no part of it. If you want to run applications on the cluster you will need to specifically log on to it. However your workstation will also stay clean and no remote processes will migrate to it.
- A third alternative is called an *Adaptive−pool*, here servers are shared while workstations join or leave the cluster, Imagine your workstation being used during daytime by yourselve, but as soon as you log out in the evening a script tells the workstation to join the cluster and start crunching numbers this way your machine is being used while you don't need if , you need the resources of the machine again just run the openmosix stop script and processes will stay away from your cluster.

Chapter 5. Distribution specific installations

5.1. Installing Mosix

This chapter deals with installing Mosix and openMosix on different distributions. It won't be an exhaustive list of all the possible combinations. However throughout the chapter you should find enough information on installing Mosix in your environment.

Techniques for installing multiple machines with Mosix will be discussed in the next chapter.

5.2. Getting openMosix

You can download the latest versions of openMosix from http://sourceforge.net/project/showfiles.php?group_id=46729 You can either choose the binary (even in rpm) compiled for UP or SMP or download the source code. You will need both the kernel patch or binaries and the userland tools. Alternatively you can use the CVS version

```
cvs −d:pserver:anonymous@cvs.openmosix.sourceforge.net:/cvsroot/openmosix login
cvs −z3 −d:pserver:anonymous@cvs.openmosix.sourceforge.net:/cvsroot/openmosix co linux−openmosix
cvs −z3 −d:pserver:anonymous@cvs.openmosix.sourceforge.net:/cvsroot/openmosix co userspace−tools
```

please take care that CVS trees DO BREAK now and then and that it might not be the easiest way to install Mosix ;−)

5.3. Getting Mosix (obsolete)

You can download Mosix from the www.mosix.org *http://www.mosix.org/txt_distribution.html* for the Kernel Patches, and after you accepted the License agreement *http://www.mosix.org/txt_download.html* for the UserLand tools.

5.4. openMosix General Instructions

5.4.1. Kernel Compilation

Always use pure vanilla kernel−sources from e.g. www.kernel.org to compile an openMosix kernel! Be sure to use the right openMosix version depending on the kernel−version. Do not use the kernel that comes with any Linux−distribution; it won't work.

Download the actual version of openMosix and untar it in your kernel−source directory (e.g. /usr/src/linux−2.4.16). If your kernel−source directory is other than "/usr/src/linux−[version_number]" at least the creation of a symbolic link to "/usr/src/linux−[version_number]" is required. Now apply the patch using the patch utility:

```
patch −Np1 < openMosix1.5.2moshe
```

This command displays now a list of patched files from the kernel−sources. Enable the openMosix−options in the kernel−configuration e.g.

```
...
CONFIG_MOSIX=y
# CONFIG_MOSIX_TOPOLOGY is not set
CONFIG_MOSIX_UDB=y
# CONFIG_MOSIX_DEBUG is not set
# CONFIG_MOSIX_CHEAT_MIGSELF is not set
CONFIG_MOSIX_WEEEEEEEEE=y
CONFIG_MOSIX_DIAG=y
CONFIG_MOSIX_SECUREPORTS=y
CONFIG_MOSIX_DISCLOSURE=3
CONFIG_QKERNEL_EXT=y
CONFIG_MOSIX_DFSA=y
CONFIG_MOSIX_FS=y
CONFIG_MOSIX_PIPE_EXCEPTIONS=y
CONFIG_QOS_JID=y
...
```

and compile it with:

```
make dep bzImage modules modules_install
```

After compilation install the new kernel with the openMosix options within you boot−loader e.g. insert an entry for the new kernel in /etc/lilo.conf and run lilo after that.

Reboot and your openMosix−cluster(node) is up!

5.4.2. hpc.map

Syntax of the /etc/hpc.map file Before starting openMosix there has to be a /etc/hpc.map configuration file (on each node) which must be equal on each node. The hpc.map contains three space separated fields:

```
openMosix−Node_ID               IP−Address(or hostname)          Range−size
```

An example hpc.map could look like this:

```
1       node1   1
2       node2   1
3       node3   1
4       node4   1
```

or

```
1       192.168.1.1     1
2       192.168.1.2     1
3       192.168.1.3     1
4       192.168.1.4     1
```

or with the help of the range−size these both examples are equal with:

```
1       192.168.1.1     4
```

openMosix "counts−up" the last byte of the ip−address of the node according to its openMosix−ID. (if you use a range−size greater than 1 you have to use ip−addresses instead of hostnames)

If a node has more than one network−interfaces it can be configured with the ALIAS option in the range−size field (which is equal to set the range−size to 0) e.g.

```
1       192.168.1.1     1
2       192.168.1.2     1
3       192.168.1.3     1
4       192.168.1.4     1
4       192.168.10.10   ALIAS
```

Here the node with the openMosix−ID 4 has two network−interfaces (192.168.1.4 + 192.168.10.10) which are both visible to openMosix.

Always be sure to run the same openMosix version AND configuration on each of your Cluster nodes!

Start openMosix with the "setpe" utility on each node : setpe −w −f /etc/hpc.map Execute this command (which will be described later on in this HOWTO) on every node in your openMosix cluster. Installation finished now, the cluster is up and running :)

5.4.3. MFS

At first the CONFIG_MOSIX_FS option in the kernel configuration has to be enabled. If the current kernel was compiled without this option recompilation with this option enabled is required. Also the UIDs and GUIDs in the cluster must be equivalent. The CONFIG_MOSIX_DFSA option in the kernel is optional but of course required if DFSA should be used. To mount MFS on the cluster there has to be an additional fstab−entry on each nodes /etc/fstab.

for DFSA enabled:

```
mfs_mnt         /mfs            mfs     dfsa=1          0 0
```

for DFSA disabled:

```
mfs_mnt          /mfs           mfs     dfsa=0          0 0
```

the syntax of this fstab−entry is:

```
[device_name]           [mount_point]   mfs     defaults        0 0
```

After mounting the /mfs mount−point on each node, each nodes file−system is accessible through the /mfs/[openMosix_ID]/ directories.

With the help of some symbolic links all cluster−nodes can access the same data e.g. /work on node1

```
on node2 :      ln −s /mfs/1/work /work
on node3 :      ln −s /mfs/1/work /work
on node3 :      ln −s /mfs/1/work /work
...
```

Now every node can read+write from and to /work !

The following special files are excluded from the MFS:

- the /proc directory
- special files which are not regular−files, directories or symbolic links e.g. /dev/hda1

Creating links like:

```
ln −s /mfs/1/mfs/1/usr
```

or

```
ln −s /mfs/1/mfs/3/usr
```

is invalid.

The following system calls are supported without sending the migrated process (which executes this call on its home (remote) node) going back to its home node:

read, readv, write, writev, readahead, lseek, llseek, open, creat, close, dup, dup2, fcntl/fcntl64, getdents, getdents64, old_readdir, fsync, fdatasync, chdir, fchdir, getcwd, stat, stat64, newstat, lstat, lstat64, newlstat, fstat, fstat64, newfstat, access, truncate, truncate64, ftruncate, ftruncate64, chmod, chown, chown16, lchown, lchown16, fchmod, fchown, fchown16, utime, utimes, symlink, readlink, mkdir, rmdir, link, unlink, rename

Here are situations when system calls on DFSA mounted filesystems may not work:

different mfs/dfsa configuration on the cluster−nodes
dup2 if the second file−pointer is non−DFSA
chdir/fchdir if the parent dir is non−DFSA
pathnames that leave the DFSA−filesystem
when the process which executes the system−call is being traced
if there are pending requests for the process which executes the system−call

Next to the /mfs/1/ /mfs/2/ and so on files you will find some other directories as well.

Table 5−1. Other Directories

/mfs/here	The current node where your process runs
/mfs/home	Your home node
/mfs/magic	The current node when used by the "creat" system call (or an "open" with the "O_CREAT" option) − otherwise, the last node on which an MFS magical file was successfully created (this is very useful for creating temporary−files, then immediately unlinking them)
/mfs/lastexec	The node on which the process last issued a successful "execve" system−call.
/mfs/selected	The node you selected by either your process itself or one of its ancesstors (before forking this process), writing a number into "/proc/self/selected".

Note that these magic files are all 'per proccess'. That is their contents is dependent upon which proccess opens them.

5.5. Red Hat and openMosix

If you are running a RedHat 7.2 or 7.3 version, this is probalby the easiest *Mosix install you have ever done. Choose the appropriate openMosix RPM's from sourceforge. They have precompiled kernels (as I write this 2.4.17) that work seamlessly , I have tested them on several machines including Latptops with PCMCIA cards and Servers with SCSI disks. If you are a grub user the kernel rpm even modifies your grub.conf. So all you have to do is install 2 rpm's

```
rpm −vih openmosix−kernel−2.4.17−openmosix1.i686.rpm openmosix−tools−0.2.0−1.i386.rpm
```

And edit your /etc/mosix.map Since this seems to be a problem for lot's of people let's go with another example. Say you have 3 machines. 192.168.10.220, 192.168.10.78 and 192.168.10.84. Your mosix.map wil look like this.

```
[root@oscar0 root]# more /etc/mosix.map
# MOSIX CONFIGURATION
# ===================
#
# Each line should contain 3 fields, mapping IP addresses to MOSIX node−numbers:
# 1) first MOSIX node−number in range.
# 2) IP address of the above node (or node−name from /etc/hosts).
# 3) number of nodes in this range.
#
# Example: 10 machines with IP 192.168.1.50 − 192.168.1.59
# 1        192.168.1.50     10
#
# MOSIX−#  IP  number−of−nodes
# ============================
1 192.168.10.220 1
2 192.168.10.78  1
3 192.168.10.84  1
```

Now by rebooting the different machines with the newly installed kernel you will get 1 step closer to having a working cluster.

Most RedHat installations have 1 extra thing to fix. You often get the following error.

```
[root@inspon root]# /etc/init.d/openmosix start
Initializing openMosix...
setpe: the supplied table is well−formatted,
but my IP address (127.0.0.1) is not there!
```

This means that your hostname is not listed in /etc/hosts with the same ip as in your mosix.map You might have a machine called omosix1.localhost.org in your hostfile listed as

```
127.0.0.1       omosix1.localhost.org localhost
```

If you modify your /etc/hosts to look like below. openMosix will have less troubles starting up.

```
192.168.10.78   omosix1.localhost.org
127.0.0.1       localhost
```

```
[root@inspon root]# /etc/init.d/openmosix start
Initializing openMosix...
[root@inspon root]# /etc/init.d/openmosix status
This is MOSIX node #2
Network protocol: 2 (AF_INET)
MOSIX range     1−1     begins at 192.168.10.220
MOSIX range     2−2     begins at inspon.localhost.be
MOSIX range     3−3     begins at 192.168.10.84
Total configured: 3
```

5.6. Suse 7.1 and Mosix (obsolete)

5.6.1. Versions Required

The following is based on using SuSE 7.1 (German Version), Linux Kernel 2.2.19, and Mosix 0.98.0.

The Linux Kernel 2.2.18 sources are part of the SuSE distribution. Do not use the default SuSE 2.2.18 kernel, as it is heavily patched with SuSE stuff. Get the patch for 2.2.19 from your favorite mirror such as . If there are further patches for the 2.2.* kernel RROR URL HERE by the time you read this text, get those, too.

If one of your machines is a laptop with a network connection via PCMCIA, you will need the PCMCIA sources, too. They are included in the SuSE distribution as MISSING: RPM HERE.

Mosix 0.98.0 for the 2.2.19 kernel can be found on *http://www.mosix.org/* as MOSIX−0.98.0.tar.gz . While you are there, you might want to get some of the contributed software like qps or mtop. Again, if there is a version more current than 0.98.0 by the time you read this, get it instead.

SuSE 7.1 ships with a Mosix−package as a rpm MISSING: RPM HERE Ignore this package. It is based on Kernel 2.2.18 and seems to have been modified by SuSE (see /usr/share/doc/packages/mosix/README.SUSE). You are better off installing the Mosix sources and installing from scratch.

5.6.2. Installation

We're assuming your hardware and basic Linux system are all set up correctly and that you can at least telnet (or ssh) between the different machines. The procedure is described for one machine. Log in as root. Install the sources for the 2.2.18 Kernel in /usr/src. SuSE will place them there automatically as /usr/src/linux−2.2.18 if you install the RPM RPM NAME. Rename the directory to /usr/src/linux−2.2.19. Remove the existing link /usr/src/linux and create a new one to this directory with

```
        ln −s /usr/src/linux−2.2.19 linux
```

(assuming you are in /usr/src). Patch the kernel to 2.2.19 (or whatever the current version is). If you do not know to do this, check the Linux Kernel HOWTO. Make a directory /usr/src/linux−2.2.19−mosix and copy the contents of the vanilla kernel /usr/src/linux−2.2.19 there with the command

```
        cp −rp linux−2.2.19/* linux−2.2.19−mosix/
```

This gives you a clean backup kernel to fall back on if something goes wrong. Remove the /usr/src/linux link (again). Create a link /usr/src/linux to /usr/src/linux−2.2.19−mosix with

```
        ln −s /usr/src/linux−2.2.19−mosix linux
```

to make life easier. Change to /tmp, copy the Mosix sources there and unpack them with the command

```
        tar xfz MOSIX−0.98.0.tar.gz
```

Do not unpack the resulting tar archives such as /tmp/user.tar that appear.

5.6.3. Setup

- Run the install script /tmp/mosix.install and follow instructions.

  Mosix should be enabled for run level 3 (full multiuser with network, no xdm) and 5 (full multiuser with network and xdm). There is no run level 4 in SuSE 7.1.

  The Mosix install script does not give you the option of creating a boot floppy instead of an image. If you want a boot floppy, you will have to run "make bzdisk" after the install script is through.

  Do not repeat /not/ reboot.

- The install script in Mosix 0.98.0 is made for Red Hat distributions and therefore fails to set up some SuSE files correctly. It tries to put stuff in /sbin/init.d/, which in fact is /etc/init.d/ (or /etc/rc.d/) with SuSE. Also, there is no /etc/rc.d/init.d/ in SuSE. So:

    - ♦ Copy /tmp/mosix.init to /etc/init.d/mosix and make it executable with the command

      ```
                       chmod 754 /etc/init.d/mosix
      ```
    - ♦ MISSING − MODIFY ATD stuff "/etc/rc.d/init.d/ATD" BY HAND
    - ♦ MISSING − MODIFY THE "/etc/cron.daily/slocate.cron" FILE
    - ♦ The other files − /etc/inittab, /etc/inetd.conf, /etc/lilo.conf − are modified correctly.
- Edit the file /etc/inittab to prevent some processes from migrating to other nodes by inserting the command "/bin/mosrun −h" in the following lines:

  Run levels:

  ```
          l0:0:wait:/bin/mosrun −h /etc/init.d/rc 0
          l1:1:wait:/bin/mosrun −h /etc/init.d/rc 1
          l2:2:wait:/bin/mosrun −h /etc/init.d/rc 2
          l3:3:wait:/bin/mosrun −h /etc/init.d/rc 3
          l5:5:wait:/bin/mosrun −h /etc/init.d/rc 5
          l6:6:wait:/bin/mosrun −h /etc/init.d/rc 6
  ```

  (Remember, there is no run level 4 in SuSE 7.1)

  Shutdown and sulogin:

  ```
      ~~:S:respawn:/bin/mosrun −h /sbin/sulogin
          ca::ctrlaltdel:/bin/mosrun −h /sbin/shutdown −r −t 4 now
          sh:12345:powerfail:/bin/mosrun −h /sbin/shutdown −h now THE \
             POWER IS FAILING
  ```

  It is not necessary to prevent the /sbin/mingetty processes from migrating − in fact, if you do, all of the child processes started from your login shell will be locked, too [Note to German readers: This is mistake in the article "Zwischen Multiprocessing und Cluster−Computing" on Mosix in "Linux−Magazin" 6/2000].
- To enable the processes started by your window manager to migrate, edit the files ~/.xinitrc and ~/.xsession by going to the end of the file and changing the line "exec $WINDOWMANAGER" to

  ```
         exec /bin/mosrun −l $WINDOWMANAGER
  ```

  You should be able to enable migration for all users' window mangers by modifying the equivalent line in /etc/X11/xdm/Xsession MISSING: NOT TESTED YET. However, see section 8 "Notes" for reasons why you might not want to do this by default.
- The command to start and stop Mosix (do not repeat /not/ do this now) is

  ```
         /etc/init.d/mosix {start|stop|status|restart|reload}
  ```

  To have Mosix start automatically at boot time, go to /etc/init.d/ . In the subdirectories ./rc3.d and ./rc5.d, create the following links:

  ```
          ln −s ../mosix S30mosix
          ln −s ../mosix K01mosix
  ```

  The first line causes Mosix to be called as the last part of the install procedure for the given run level, the second line closes it down as one of the first services.
- Create a file /etc/mosix.map following the instructions in the Mosix documentation. In the most simple case, you will have n computers which have their IP−addresses in sequence so that the map file will simply look like

  ```
          1   IP−address of first node  n
  ```

This is where a lot of errors occur, let me clarify this with an example. Suppose you have 5 machines, 10.0.0.1, 10.0.0.2 , 10.0.0.100 , 10.0.0.101 and 10.0.0.150 your mosix.map would look like

```
        1  10.0.0.1     2
        3  10.0.0.100   2
        5  10.0.0.150   1
```

PLEASE VERIFY THIS !!!!!!!

- Run "/etc/versionate", which will most probably tell you that the Mosix module already has a version. Do it anyway.
- Now, finally, reboot. The computer should come up running Mosix.

5.7. Debian and Mosix (to be replaced with openMosix version)

Installing mosix on a Debian based machine can be done as described below First step is downloading the packages from the net. Since we are using a Debian setup we needed
http://packages.debian.org/unstable/net/mosix.html
http://packages.debian.org/unstable/net/kernel−patch−mosix.html
http://packages.debian.org/unstable/net/mps.html You can also apt−get install them ;) Next part is making the kernel mosix capable. Copy the patch.$kernel version in to your /usr/src/linux−$version directory run

```
patch −p0 < patches.2.4.10
```

Check your kernel config and run

```
make dep ; make clean ; make bzImage ; make modules ; make modules_install
```

You now will need to edit your /etc/mosix/mosix.map This file has a bit a strange layout. We have 2 machines 192.168.10.65 and 192.168.10.94 This gives us a mosix.map that looks like

```
1 192.168.10.65 1
2 192.168.10.94 1
```

After rebooting with this kernel (lilo etc you know the drill), you then should have a cluster of mosix machines that talk to each−other and that do migration of processes. You can test that by running the following small script ..

```
awk 'BEGIN {for(i=0;i<10000;i++)for(j=0;j<10000;j++);}'
```

a couple of times, and monitor it`s behaviour with mon where you will see that it spreads the load between 2 different nodes. If you have enabled Process−arrival messages in your kernel you will notice that each time a remote (guest) process arrives on your node a Weeeeeee will be printed and each time a local proces returns you will see a Woooooo on your console. So basically If you don`t see any of those messages during the running of a program and if you have this option enabled in your kernel you might conclude that no processes migrate. We also setup Mosixview (0.8) on the debian machine

```
apt-get install mosixview
```

In order to be able to actually use Mosixview you will need to run it from a user who can log in to the different nodes as root. We suggest you set this up using ssh. Please note that there is a difference between the ssh and ssh2 implemtations .. if you have a identity.pub ssh wil check authorized_keys, if you have id_rsa.pub you will need authorized_keys2 !! Mosixview gives you a nice interface that shows the load of different machines and gives you the possibility to migrate processes manually. A detailed discussion of Mosixview can be found elsewhere in this document.

5.8. Other distributions

Based on the explanations above you should be able to install Mosix on most other Linux platforms.

Chapter 6. Cluster Installation

6.1. Cluster Installations

This chapter does not deal with installing Mosix as such, it does however deal with installing multiple machines with mosix. Automated or semi automated mass installs.

6.2. DSH, Distributed Shell

At the time of this writing (July 2002) DSH's most current release is available from http://www.netfort.gr.jp/~dancer/software/downloads/ More info on the package can be found on http://www.netfort.gr.jp/~dancer/software/dsh.html The latest version available for download is 0.0.19.7 You will need both libdshconfig−0.0.20.1.tar.gz and dsh−0.0.19.7.tar.gz Start with installing libdshconfig

```
./configure
make
make install
```

Repeat the process for the dsh package.

Say we have a small cluster with a couple of nodes. To make life easier we want type each command once but have it excecuted on each node. You then have to create a file in $HOME/.dsh/group/clustername that lists the ip's of your cluster. eg.

```
[root@inspon root]# cat .dsh/group/mosix
192.168.10.220
192.168.10.84
```

As an example we run ls on each of these machines We use −g to use the mosix group (this way you can create subsets of a group with different configurations)

```
[root@inspon root]# dsh −r ssh −g mosix ls
192.168.10.84: anaconda−ks.cfg
192.168.10.84: id_rsa.pub
192.168.10.84: install.log
192.168.10.84: install.log.syslog
192.168.10.84: openmosix−kernel−2.4.17−openmosix1.i686.rpm
192.168.10.84: openmosix−tools−0.2.0−1.i386.rpm
192.168.10.220: anaconda−ks.cfg
192.168.10.220: id_dsa.pub
192.168.10.220: id_rsa.pub
192.168.10.220: openmosix−kernel−2.4.17−openmosix1.i686.rpm
192.168.10.220: openmosix−tools−0.2.0−1.i386.rpm
192.168.10.220: oscar−1.2.1rh72
192.168.10.220: oscar−1.2.1rh72.tar.gz
```

Note that neither of the machines ask for a password. This is because we have set up rsa authentication between the different accounts. If you want to run commands with multiple parameters you will have either have to put the command between quotes.

```
[root@inspon root]# dsh −r ssh −g mosix "uname −a"
192.168.10.84: Linux omosix2.office.be.stone−it.com 2.4.17−openmosix1 #1
Wed May 29 14:32:28 CEST 2002 i686 unknown
192.168.10.220: Linux oscar0 2.4.17−openmosix1 #1 Wed May 29 14:32:28 CEST
```

```
2002 i686 unknown
```

or use the −c −− option. Both give basically the same output.

```
[root@inspon root]# dsh −r ssh −g mosix −c −− uname −a
192.168.10.220: Linux oscar0 2.4.17−openmosix1 #1 Wed May 29 14:32:28 CEST
2002 i686 unknown
192.168.10.84: Linux omosix2.office.be.stone−it.com 2.4.17−openmosix1 #1
Wed May 29 14:32:28 CEST 2002 i686 unknown
```

Chapter 7. ClumpOS

7.1. What is Clump/OS

There is currently no openMosix version available of Clump/OS. However according to the mailing list one is planned.

```
From:   Jean−David Marrow
Cc:     clumpos−list
Subject:        Re: [clumpos−list] clump/os and OpenMosix
Date:   29 Apr 2002 14:54:26 −0400

the OpenMosix−based release is on hold, for internal reasons, until
OpenMosix for 2.4.17 (or greater) kernel become available... and since i
don't have a release schedule for OpenMosix itself, i'm afraid that i
can't tell you exactly when that'll be... ;)

jdm
```

clump/os is a CD−based Linux /MOSIX mini−distribution designed to allow users to quickly, or temporarily, add nodes to a MOSIX cluster; As I write this in march 2002 the version (release 5.x) is a 5.3M ISO download.

This chapter has been contributed by Jean−David Marrow who is the main author of Clump/OS.

7.2. How does it work

At boot−time, clump/os will auto−probe for network cards, and, if any are detected, try to configure them via DHCP. If successful, it will create a mosix.map file based on the assumption that all nodes are on local CLASS C networks, and configure MOSIX using this information.

clump/os Release 4 best supports machines with a single connected network adapter. The MOSIX map created in such cases will consist of a single entry for the CLASS−C network detected, with the node number assigned reflecting the IP address received from DHCP. (On the 192.168.1 network, node #1 will be 192.168.1.1, etc.) If you use multiple network adapters Expert mode is recommended as the assignment of node numbers is sensitive to the order in which network adapters are detected. (Future releases will support complex topologies and feature more intelligent MOSIX map creation.)

clump/os will then display a simple SVGA monitor (clumpview) indicating whether the node is configured, and, if it is, showing the load on all active nodes on the network. When you've finished using this node, simply press [ESC] to exit the interface and shutdown.

Alternatively, or if auto−configuration doesn't work for you, then you can use clump/os in Expert mode. Please note that clump/os is not a complete distribution or a rescue disk; the functionality present is the bare minimum required for a working MOSIX server node.

It works for us, but may not work for you; if you experience difficulties, please email us with as much information about your system as possible −− after you have investigated the problem. (See Problems? and Expert mode. You might also consider subscribing to the clump/os mailing list.)

7.3. Requirements

As the purpose of clump/os is to add nodes to a cluster, it is assumed that you already have a running MOSIX cluster −− or perhaps only a single MOSIX node −− from which you will be initiating jobs. All machines in the cluster must conform to the following requirements:

- clump/os Machine(s) 586+ CPU,
- bootable CD−ROM
- NIC
- 64M+ RAM (the system is loaded entirely into a ramdisk; this means that you should have at least 64M of RAM (and likely more) to accommodate the approx. 16M ramdisk, space needed for Linux itself, and space for your work. This approach was chosen so that the same CD−ROM can be used to configure multiple systems.)
- Master Machine(s) Linux 2.4.17, MOSIX 1.5.7 (manually configured)
- Network Environment Running DHCP server (f you don't, or won't, run DHCP, you can still manually configure your system; see Problems? and Expert Mode. Using DHCP is highly recommended, however, and will greatly simplify your life in the long run.)

The following network modules are present, although not all support auto−probing; if you don't see support for your card in this list, then clump/os will not work for you even in Expert Mode.

3c501.o 3c503.o 3c505.o 3c507.o 3c509.o 3c515.o 3c59x.o 8139cp.o 8139too.o 82596.o 8390.o ac3200.o acenic.o at1700.o cs89x0.o de4x5.o depca.o dgrs.o dl2k.o dmfe.o dummy.o e2100.o eepro.o eepro100.o eexpress.o epic100.o eth16i.o ewrk3.o fealnx.o hamachi.o hp−plus.o hp.o hp100.o lance.o lp486e.o natsemi.o ne.o ne2k−pci.o ni5010.o ni52.o ni65.o ns83820.o pcnet32.o sis900.o sk98lin.o smc−ultra.o smc9194.o starfire.o sundance.o sungem.o sunhme.o tlan.o tulip.o via−rhine.o wd.o winbond−840.o yellowfin.o

Please also note that clump/os may not work on a laptop, definitely doesn't support PCMCIA cards, and will probably not configure MOSIX properly if your machine contains multiple connected Ethernet adapters; see Note 1. This is a temporary limitation of the configuration scripts, and the Release 3/4 kernels which are compiled without CONFIG_MOSIX_TOPOLOGY

7.4. Getting Started

You can download the latest clump/os ISO under the terms of the GPL, without warranty of any kind, from the clump os website. Afterwards you have to burn the image to CD−ROM, insert the CD into your drive, and reboot. (More detailed instructions are in the works, but all the information you need is somewhere on this page −− please read the notes in the margin!)

7.5. Problems ?

- *The CD−ROM doesn't boot*

  Check your BIOS settings to make sure that your machine is configured to boot from the CD−ROM drive; also make sure that the CD−ROM is the first boot device.
- *The SVGA interface doesn't work, or the display is incorrect*

  Boot into Expert mode, and send us mail describing your video hardware so that we can correct this in future versions. (You won't be able to use clumpview for now.) If at all possible, please send us a working libsvga configuration file for the machine in question.

- *The network adapter isn't detected/autoconfigured (or no DHCP)*

  If you see a message (in clumpview) stating that no Ethernet devices were configured, or that this node isn't configured yet, then either your Ethernet card was not detected or the system was not able to configure the card via DHCP.

  If you don't have a DHCP server configured and running on your local network, clump/os will never autoconfigure; if you have multiple connected network adapters, then clump/os may not configure MOSIX properly. If auto−probing for your network adapter doesn't work, or if you aren't using DHCP, then you'll have to configure your card manually in Expert mode −− using insmod, ifconfig, and route −− and then configure MOSIX via setpe.

  If you do need to manually configure your network adapter, please advise us. We'd like to solve this problem, if possible, or at least document which network cards auto−probe correctly.
- *Migrating processes generate errors ("Network Unreachable")*

  This rare problem can be caused by conflicts resulting from differently configured kernels −− even if you are using the correct MOSIX and Linux kernel versions. If clump/os correctly detects all your nodes, but migrating processes generate errors, then please compare your master node's kernel configuration file with the R4.x kernel .config.
- *Migrating processes generate errors ("Process migration failed: incompatible topology")*

  You are likely using master nodes with CONFIG_MOSIX_TOPOLOGY defined, which is not supported by clump/os at this time. See Requirements, and compare your kernel configuration as per the previous FAQ; you will need to recompile your master node kernel(s).

If you don't find your issue here, please consider posting to the clump/os mailing list. (Please note that only subscribers are permitted to post; click on the link for instructions.) You should also make certain that you are using the latest versions of MOSIX and clump/os, and that the versions −− clump/os R4.x and MOSIX 1.5.2 at the time of this writing −− are in sync.

7.6. Expert Mode

If you hold down shift during the boot process, you have the option of booting into Expert mode; this will cause clump/os to boot to a shell rather than to the graphical interface. From this shell you can attempt to insert the appropriate module for your network adapter (if autoprobing failed), and/or configure your network and MOSIX manually. Type "halt" to shut down the system. (Note that since the system resides in RAM you can't hurt yourself too badly by rebooting the hard way if you have to −− unless you have manually mounted any partitions rw, that is, and we don't recommend doing so at this point.)

If you want to run clumpview, execute:

```
     open −s −w −− clumpview −−drone −−svgalib
```

This will force the node into 'drone' mode (local processes will not migrate), and will force clumpview to use SVGALIB; the open command will ensure that a separate vt is used.

Please be advised that the environment provided was initially intentionally minimalistic; if you require additional files, or wish to copy files from the system to another machine, your only option is nc (netcat −− a great little utility, btw), or mfs if MOSIX is configured. From version R5.4 on size is no longer a primary consideration.

Expert mode (and clump/os for that matter) is 'single−user'; this is one of the reasons that utilities such as ssh are not included. These and other similar decisions were made in order to keep clump/os relatively small, and do not affect cluster operation.

From version R5.4, if you experience problems in Expert Mode, you can boot into Safe Mode; in Safe Mode no attempt is made at autoconfiguration.

Chapter 8. Administrating openMosix

8.1. Basic Administration

openMosix provides the advantage of process migration to HPC−applications. The administrator can configure and tune the openMosix−cluster by using the openMosix−user−space−tools or the /proc/hpc interface which will be now described in detail.

8.2. Configuration

The values in the flat files in the /proc/hpc/admin directory presenting the current configuration of the cluster. Also the administrator can write its own values into these files to change the configuration during runtime, e.g.

Table 8−1. Changing /proc/hpc parameters

echo 1 > /proc/hpc/admin/block	−blocks the arrival of remote processes
echo 1 > /proc/hpc/admin/bring	−bring all migrated processes home

...

Table 8−2. /proc/hpc/admin/

(binary files)	config	the main configuration file (written by the setpe util)
(flat files)	block	allow/forbid arrival of remote processes
	bring	bring home all migrated processes
	dfsalinks	list of current symbolic dfsa−links
	expel	sending guest processes home
	gateways	maximum number of gateways
	lstay	local processes shoud stay
	mospe	contains the openMosix node id
	nomfs	disables/enables MFS
	overheads	for tuning
	quiet	stop collecting load−balacing informations
	decayinterval	interval for collecting informations about load−balancing
	slowdecay	default 975
	fastdecay	default 926
	speed	speed relative to PIII/1GHz)
	stay	enables/disables automatic process migration

Table 8−3. Writing a 1 to the following files /proc/hpc/decay/

clear	clears the decay statistics
cpujob	tells openMosix that the process is cpu−bound
iojob	tells openMosix that the process is io−bound
slow	tells openMosix to decay its statistics slow
fast	tells openMosix to decay its statistics fast

Table 8−4. Informations about the other nodes

/proc/hpc/nodes/[openMosix_ID]/cpus	how many cpu's the node has
/proc/hpc/nodes/[openMosix_ID]/load	the openMosix load of this node
/proc/hpc/nodes/[openMosix_ID]/mem	available memory as openMosix believes
/proc/hpc/nodes/[openMosix_ID]/rmem	available memory as Linux believes
/proc/hpc/nodes/[openMosix_ID]/speed	speed of the node relative to PIII/1GHz
/proc/hpc/nodes/[openMosix_ID]/status	status of the node
/proc/hpc/nodes/[openMosix_ID]/tmem	available memory
/proc/hpc/nodes/[openMosix_ID]/util	utilization of the node

Table 8−5. Additional Informations about local processes

/proc/[PID]/cantmove	reason why a process cannot be migrated
/proc/[PID]/goto	to which node the process should migrate
/proc/[PID]/lock	if a process is locked to its home node
/proc/[PID]/nmigs	how many times the process migrated
/proc/[PID]/where	where the process is currently being computed
/proc/[PID]/migrate	same as goto remote processes
/proc/hpc/remote/from	the home node of the process
/proc/hpc/remote/identity	additional informations about the process
/proc/hpc/remote/statm	memory statistic of the process
/proc/hpc/remote/stats	cpu statistics of the process

8.3. the userspace−tools

These following tools are providing easy administration to openMosix clusters.

```
migrate −send a migrate request to a process
                syntax:
                        migrate [PID] [openMosix_ID]
```

```
mon             −is a ncurses−based terminal monitor
                 several informations about the current status are displayed in bar−charts
```

```
mosctl          −is the openMosix main configuration utility
                syntax:
                        mosctl  [stay|nostay]
```

```
                                [stay|nolstay]
                                [block|noblock]
                                [quiet|noquiet]
                                [nomfs|mfs]
                                [expel|bring]
                                [gettune|getyard|getdecay]

                        mosctl  whois   [openMosix_ID|IP−address|hostname]

                        mosctl  [getload|getspeed|status|isup|getmem|getfree|getutil]   [openMosi

                        mosctl  setyard [Processor−Type|openMosix_ID||this]

                        mosctl  setspeed        interger−value

                        mosctl  setdecay interval       [slow fast]
```

Table 8−6. more detailed

stay	no automatic process migration
nostay	automatic process migration (default)
lstay	local processes should stay
nolstay	local processes could migrate
block	block arriving of guest processes
noblock	allow arriving of guest processes
quiet	disable gathering of load−balancing informations
noquiet	enable gathering of load−balancing informations
nomfs	disables MFS
mfs	enables MFS
expel	send away guest processes
bring	bring all migrated processes home
gettune	shows the current overhead parameter
getyard	shows the current used Yardstick
getdecay	shows the current decay parameter
whois	resolves openMosix−ID, ip−addresses and hostnames of the cluster
getload	display the (openMosix−) load
getspeed	shows the (openMosix−) speed
status	displays the current status and configuration
isup	is a node up or down (openMosix kind of ping)
getmem	shows logical free memory
getfree	shows physical free mem
getutil	display utilization
setyard	sets a new Yardstick−value
setspeed	sets a new (openMosix−) speed value
setdecay	sets a new decay−interval

```
mosrun          −run a special configured command on a chooosen node
```

```
                syntax:
                        mosrun  [-h|openMosix_ID| list_of_openMosix_IDs] command [arguments]
```

The mosrun command can be executed with several more commandline options. To ease this up there are several preconfigured run−scripts for executing jobs with a special (openMosix) configuration.

Table 8−7. extra options for mosrun

nomig	runs a command which process(es) won't migrate
runhome	executes a command locked to its home node
runon	runs a command which will be directly migrated and locked to a node
cpujob	tells the openMosix cluster that this is a cpu−bound process
iojob	tells the openMosix cluster that this is a io−bound process
nodecay	executes a command and tells the cluster not to refresh the load−balancing statistics
slowdecay	executes a command with a slow decay interval for collecting load−balancing statistics
fastdecay	executes a command with a fast decay interval for collecting load−balancing statistics

```
setpe           −manual node configuration utility
                syntax:
                        setpe   −w −f   [hpc_map]
                        setpe   −r [−f  [hpc_map]]
                        setpe   −off

−w reads the openMosix configuration from a file (typically /etc/hpc.map)
−r writes the current openMosix configuration to a file (typically /etc/hpc.map)
−off turns the current openMosix configuration off
```

```
tune            openMosix calibration and optimizations utility.
                (for further informations review the tune−man page)
```

Additional to the /proc interface and the commandline−openMosix utilities (which are using the /proc interface) there is a patched "ps" and "top" available (they are called "mps" and "mtop") which displays also the openMosix−node ID on a column. This is useful for finding out where a specific process is currently being computed.

The administrator can have a overview about the current status of the cluster and its nodes with the "Mosix Cluster Information Tool PHP" which can be found at http://wijnkist.warande.uu.nl/mosix/ . (the path to the NODESDIR has to be adjusted to $NODESDIR="/proc/hpc/nodes/")

For smaller cluster it might also be useful to use Mosixview which is a GUI for the most common administration tasks.

Chapter 9. Tuning Mosix

9.1. Optimizing Mosix

Editorial Comment: To be checked with openMosix versions

Login a normal terminal as root. Type

```
       setpe −r
```

which, if everything went right, will give you a listing of your /etc/mosix.map. If things did not go right, try

```
        setpe −w −f /etc/mosix.map
```

to set up your node. Then, type

```
       cat /proc/$$/lock
```

to see if your child processes are locked in your mode (1) or can migrate (0). If for some reason you find your processes are locked, you can change this with

```
        echo 0 > /proc/$$/lock
```

until you fix the problem. Repeat the whole configuration scheme for a second computer. The programs tune_kernel and prep_tune that Mosix uses to calibrate the individual nodes do not work with the SuSE distribution. However, you can fake it. First, bring the computer you want to tune and another computer with Mosix installed down to single user mode by typing

```
        init 1
```

as root. All other computers on the network should be shutdown if possible. On both machines, run the following commands:

```
        /etc/init.d/network start
        /etc/init.d/mosix start
        echo 1 > /proc/mosix/admin/quiet
```

This fakes prep_tune and the first parts of tune_kernel. Note that if you have a laptop with a pcmcia network card, you will have to run

```
        /etc/init.d/pcmcia start
```

instead of "/etc/init.d/network start". On the computer you want to tune, run tune_kernel and follow instructions. Depending on your machines, this can take a while − if you have a dog, this might be the time to go on that long, long walk you've always promised him. tune_kernel will create a program called "pg" in /root for testing reasons. Ignore it. After tuning is over, copy the contents of /tmp/overheads to the file /etc/overheads (and/or recompile the kernel). Repeat the tuning procedure for each computer. Reboot, enjoy Mosix, and don't forget to brag to your friends about your new cluster.

9.2. Channel Bonding Made Easy

Contributed by Evan Hisey

Channel bonding is actually horrible easy. This may explain the lack of documentaion on this subject A bonded network appears as a normal network to the applications. All machines on a subnet must be either bonded teh same way. Bonded and non−bonded machine really don't talk well to each other.

Channel bonding needs at least to physcial sub−nets but can have more(Currently I have a tri−bonded cluster). To enable bonding you need to either compile in to the kernel or as a module (bonding.o) the Channel Bonding kernel code, as of 2.4.x is it a standardoption of the kernel. The nics are setup as normal with except that you only us 'ifconfig' to initialize the first card of the bond. 'ifenslave' is used to intialize the remaining cards in the bonded connection. 'ifenslave' can be locate in the linux/Documentation/network/ directory. It will need to be compiled as it is a .c file. The basic format for use is

```
ifenslave <master> <slave1> <slave2>
```

...'. Channel bonded networks can connect to standard networks via a router or bridge that supports channel bonding(I just use an extra nic and portforwarding in the head node).

Chapter 10. Special Cases

10.1. Laptops and PCMCIA Cards

If you are installing Mosix on a Laptop, you will have to recompile the PCMCIA sources, because they are distributed as a separate package and not as kernel modules. On a Suse 7.1 machine, in theory, this should work by installing the packages and then running

```
        rpm −ba /usr/src/packages/SPECS/pcmcia.spec
```

as described in the SuSE manual [on page 358 of the German edition]. However, the script tends to get confused by the location of the libraries of the vanilla version and the Mosix version, so after running the above line, you will have to go to the sources in /usr/src/kernel−modules/pcmcia and run

```
        make config
```

When prompted for the "Module install directory", change the default setting of "/lib/modules/2.2.19" to

```
        /lib/modules/2.2.19−mosix
```

Then run "make" and "make install", which should put the pcmcia modules in /lib/modules/2.2.19−mosix/pcmcia . Note that you must be running the Mosix kernel before you recompile the pcmcia sources.

10.2. Disk−less nodes

At first you have to setup a DHCP−server which answers the DHCP−request for an ip−address when a disk−less client boots. This DHCP−Server (i call it master in this HOWTO) acts additional as an NFS−server which exports the whole client−file−systems so the disk−less− cluster−nodes (i call them slaves in this HOWTO) can grab this FS (file−system) for booting as soon as it has its ip. Just run a "normal"−MOSIX setup on the master−node. Be sure you included NFS−server−support in your kernel−configuration. There are two kinds (or maybe a lot more) types of NFS:

```
kernel−nfs
or
nfs−daemon
```

It does not matter which one you use but my experiences shows to use kernel−nfs in "older" kernels (like 2.2.18) and daemon−nfs in "newer" ones. The NFS in newer kernels sometimes does not work properly. If your master−node is running with the new MOSIX−kernel start with one file−system as slave−node. Here the steps to create it: Calculate at least 300−500 MB for each slave. Create an extra directory for the whole cluster−file−system and make a symbolic−link to /tftpboot. (The /tftpboot−directory or link is required because the slaves searches for a directory named /tftpboot/ip−address−of−slave for booting. You can change this only by editing the kernel−sources) Then create a directory named like the ip of the first slave you want to configure, e.g. mkdir /tftpboot/192.168.45.45 Depending on the space you have on the cluster−filesystem now copy the whole filesystem from the master−node to the directory of the first slave. If you have less space just copy:

```
/bin
/usr/bin
/usr/sbin
/etc
/var
```

You can configure that the slave gets the rest per NFS later. Be sure to create empty directories for the mount−points. The filesystem−structure in /tftpboot/192.168.45.45/ has to be similar to / on the master.

```
/tftpboot/192.168.45.45/etc/HOSTNAME                    //insert the hostname of the slave
/tftpboot/192.168.45.45/etc/hosts                       //insert the hostname+ip of the slave
```

Depending on your distribution you have to change the ip−configuration of the slave :

```
/tftpboot/192.168.45.45/etc/rc.config
/tftpboot/192.168.45.45/etc/sysconfig/network
/tftpboot/192.168.45.45/etc/sysconfig/network−scripts/ifcfg−eth0
```

Change the ip−configuration for the slave as you like. Edit the file

```
/tftpboot/192.168.45.45/etc/fstab               //the FS the slave will get per NFScoresponding t
/etc/exports                                    //the FS the master will export to the slaves
```

e.g. for a slave fstab:

```
master:/tftpboot/192.168.88.222  /       nfs     hard,intr,rw    0 1
none    /proc   nfs     defaults        0 0
master:/root     /root   nfs     soft,intr,rw    0 2
master:/opt      /opt    nfs     soft,intr,ro    0 2
master:/usr/local        /usr/local      nfs     soft,intr,ro    0 2
master:/data/      /data nfs     soft,intr,rw    0 2
master:/usr/X11R6        /usr/X11R6      nfs     soft,intr,ro    0 2
master:/usr/share        /usr/share      nfs     soft,intr,ro    0 2
master:/usr/lib        /usr/lib      nfs     soft,intr,ro    0 2
master:/usr/include        /usr/include      nfs     soft,intr,ro    0 2
master:/cdrom        /cdrom      nfs     soft,intr,ro    0 2
master:/var/log  /var/log        nfs     soft,intr,rw    0 2
```

e.g. for a master exports:

```
/tftpboot/192.168.45.45         *(rw,no_all_squash,no_root_squash)
/usr/local                      *(rw,no_all_squash,no_root_squash)
/root                           *(rw,no_all_squash,no_root_squash)
/opt                            *(ro)
/data                           *(rw,no_all_squash,no_root_squash)
/usr/X11R6                      *(ro)
/usr/share                      *(ro)
/usr/lib                        *(ro)
/usr/include                    *(ro)
/var/log                        *(rw,no_all_squash,no_root_squash)
/usr/src                        *(rw,no_all_squash,no_root_squash)
```

If you mount /var/log (rw) from the NFS−server you have on central log−file! (it worked very well for me. just "tail −f /var/log/messages" on the master and you always know what is going on)

The cluster−filesystem for your first slave will be ready now. Configure the slave−kernel now. If you have the same hardware on your cluster you can reuse the configuration of the master−node. Change the configuration for the slave like the following:

```
CONFIG_IP_PNP_DHCP=y
and
CONFIG_ROOT_NFS=y
```

Use as less modules as possible (maybe no modules at all) because the configuration is a bit tricky. Now (it is well described in the Beowulf−HOWTO) you have to create a nfsroot−device. It is only used for patching the slave−kernel to boot from NFS.

```
mknod /dev/nfsroot b 0 255
rdev bzImage /dev/nfsroot
```

Here "bzImage" has to be your diskless−slave−kernel you find it in /usr/src/linux−version/arch/i386/boot after successful compilation. Then you have to change the root−device for that kernel

```
rdev −o 498 −R bzImage 0
```

and copy the kernel to a floppy−disk

```
dd if=bzImage of=/dev/fd0
```

Now you are nearly ready! You just have to configure DHCP on the master. You need the MAC−address (hardware address) of the network card of your first slave. The easiest way to get this address is to boot the client with the already created boot−floppy (it will fail but it will tell you its MAC−address). If the kernel was configured alright for the slave the system should come up from the floppy, booting the diskless−kernel, detecting its network−card and sending an DHCP− and ARP request. It will tell you its hardware address during that moment! It looks like : 68:00:10:37:09:83. Edit the file /etc/dhcp.conf like the following sample:

```
option subnet−mask 255.255.255.0;
default−lease−time 6000;
max−lease−time 72000;
subnet 192.168.45.0 netmask 255.255.255.0 {
     range 192.168.45.253 192.168.45.254;
     option broadcast−address 192.168.45.255;
     option routers 192.168.45.1;
}
host firstslave
{
     hardware ethernet 68:00:10:37:09:83;
     fixed−address firstslave;
     server−name "master";
}
```

Now you can start DHCP and NFS with their init scripts:

```
/etc/init.d/nfsserver start
/etc/init.d/dhcp start
```

You got it!! It is (nearly) ready now!

Boot your first−slave with the boot−floppy (again). It should work now. Shortly after recognizing its network−card the slave gets its ip−address from the DHCP−server and its root−filesystem (and the rest) per NFS.

You should notice that modules included in the slave−kernel−config must exist on the master too, because the slaves are mounting the /lib−directory from the master. So they use the same modules (if any).

It will be easier to update or install additional libraries or applications if you mount as much as possible from the master. On the other hand if all slaves have their own complete filesystem in /tftpboot your cluster may be a bit faster because of not so many read/write hits on the NFS−server.

You have to add a .rhost file in /root (for user root) on each cluster−member which should look like this:

```
node1   root
node2   root
node3   root
....
```

You also have to enable remote−login per rsh in the /etc/inetd.conf. You should have these two lines in it

if your linux−distribution uses inetd:

```
shell   stream  tcp     nowait  root    /bin/mosrun mosrun −l −z /usr/sbin/tcpd in.rshd −L
login   stream  tcp     nowait  root    /bin/mosrun mosrun −l −z /usr/sbin/tcpd in.rlogind
```

And for xinetd:

```
service shell
{
socket_type     = stream
protocol        = tcp
wait            = no
user            = root
server          = /usr/sbin/in.rshd
server_args     = −L
}
service login
{
socket_type     = stream
protocol        = tcp
wait            = no
user            = root
server          = /usr/sbin/in.rlogind
server_args     = −n
}
```

You have to restart inetd afterwards so that it reads the new configuration.

```
/etc/init.d/inetd restart
```

or There may be another switch in your distribution−configuration−utility where you can configure the security of the system. Change it to "enable remote root login". Do not use this in insecure environments!!! Use SSH instead of RSH! You can use MOSIXVIEW with RSH or SSH. Configuring SSH for remote login without password is a bit tricky. Take a look at the "HOWTO use MOSIX/MOSIXVIEW with SSH?" at this website. If you want to copy files to a node in this diskless−cluster you have now two possibilities. You can use rcp or scp for copying remote or you can use just cp and copy files on the master to the cluster−filesystem of one node. The following two commands are equal:

```
rcp /etc/hosts 192.168.45.45./etc
cp /etc/hosts /tftpboot/192.168.45.45/etc/
```

Chapter 11. Common Problems

11.1. My processes won't migrate

Help process XYZ doesn't migrate. Moshe Bar explains below why some processes migrate and why some don't. But before that you can always look in /proc/$pid/ there often is a file cantmove which will tell you why a certain process can't migrate.

Processes can also be locked You can check if a process is locked with: cat /proc/$PID/lock where $PID is the processid of the process in question. Now let's Moshe do his explanation :

Ok, this simple program should always migrate if launched more times than number of local CPUs. So for a 2−way SMP system, starting this program 3 times will start migration if the other nodes in the cluster have at least the same speed like the local ones:

```
int main() {
    unsigned int i;
    while (1) {
        i++;
    }
    return 0;
}
```

On a Pentium 800Mhz CPU it takes quite a while to overflow.

This sample program with content like this will never migrate:

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>

...

key_t key; /* key to be passed to shmget() */
int shmflg; /* shmflg to be passed to shmget() */
int shmid; /* return value from shmget() */
int size; /* size to be passed to shmget() */

...

key = ...
size = ...
shmflg) = ...

if ((shmid = shmget (key, size, shmflg)) == −1) {
   perror("shmget: shmget failed"); exit(1); } else {
   (void) fprintf(stderr, "shmget: shmget returned %d\n", shmid);
   exit(0);
  }
...
```

Program using piples like this do migrate nicely:

```
int pdes[2];

```

```
pipe(pdes);
if ( fork() == 0 )
  { /* child */
                                 close(pdes[1]); /* not required */
                                 read( pdes[0]); /* read from parent */
                                 .....
                 }
else
                 { close(pdes[0]); /* not required */
                                 write( pdes[1]); /* write to child */
                                 .....
                 }
```

Programs using pthreads since version 2.4.17 do migrate:

```
//
// Very simple program demonstrating the use of threads.
//
// Command-line argument is P (number of threads).
//
// Each thread writes "hello" message to standard output, with
//   no attempt to synchronize.  Output will likely be garbled.
//
#include <iostream>
#include <cstdlib>              // has exit(), etc.
#include <unistd.h>             // has usleep()
#include <pthread.h>            // has pthread_ routines

// declaration for function to be executed by each thread
void * printHello(void * threadArg);

// ---- Main program -------------------------------------------------

int main(int argc, char* argv[]) {

  if (argc < 2) {
    cerr << "Usage:  " << argv[0] << " numThreads\n";
    exit(EXIT_FAILURE);
  }
  int P = atoi(argv[1]);

  // Set up IDs for threads (need a separate variable for each
  //   since they're shared among threads).
  int * threadIDs = new int[P];
  for (int i = 0; i < P; ++i)
    threadIDs[i] = i;

  // Start P new threads, each with different ID.
  pthread_t * threads = new pthread_t[P];
  for (int i = 0; i < P; ++i)
    pthread_create(&threads[i], NULL, printHello,
                   (void *) &threadIDs[i]);

  // Wait for all threads to complete.
  for (int i = 0; i < P; ++i)
    pthread_join(threads[i], NULL);

  // Clean up and exit.
  delete [] threadIDs;
  delete [] threads;
  cout << "That's all, folks!\n";
  return EXIT_SUCCESS;
}

// ---- Code to be executed by each thread ---------------------------

// pre:  *threadArg is an integer "thread ID".
// post:  "hello" message printed to standard output.
//        return value is null pointer.
void * printHello(void * threadArg) {
  int * myID = (int *) threadArg;
  cout << "hello, world, ";
  // pointless pause, included to make the effects of
  //   synchronization (or lack thereof) more obvious
  usleep(10);
  cout << "from thread " << *myID << endl;
  pthread_exit((void* ) NULL);
}
```

Programs using all kinds of file descriptors, including sockets do migrate (sockets are not migrated with the process however, files are migrated if using oMFS/DFSA)

(all above code is by Moshe as Moshe Bar or by Moshe as CTO of Qlusters, Inc.)

Please also refer to the man pages of mosix , they also give an adequate explanation why processes don't migrate.

If for some reason your processes stay locked while they shouldn't To allow locked processes tp migrate simply put

```
# tell shells to allow subprocs to migrate to other nodes
echo 0 > /proc/self/lock
```

into /etc/profile Warning : This fix will allow *all* process to migrate not just the ones you want. To only allow specific process to migrate use 'mosrun −l' to unlock only the desired process.

11.2. I don`t see all my nodes

First of all , are you using the same kernel version on each machine ? The 'same−kernel' refers to the version. You can build different kernel images of the same source version to meet the hardware/software needs of a given node. However you wil need toe make sure that when you install openMosix on your cluster, all your machines should have the openmosix−x.x.x−y kernel installed, in contrast to having one machine running openmosix−x.x.z−x, another running openmosix−x.x.x−y, another running openmosix x.x.x−z, and so on and so forth

When you run mosmon, press t to see the total of machines running. Does it warn you that mosix is not running?

If yes, then make sure your machine's ip is included in /etc/mosix.map (don't use 127.0.0.1 − if your machine's ip is such, then you probably have problems with your dhcp server/nameserver). If it does not tell you that mosix is not running, see what machines show up. Do you see only your machine?

If yes, then your machine is most likely running a firewall and is not letting openmosix through.

If not, then the problem is most likely with the machine that doesn't show up. Also: Do you have two nic cards on a node? then you have to edit the /etc/hosts file to have a line that has the following format

```
non−cluster_ip  cluster−hostname.cluster−domain cluster−hostname
```

You might also need to set up a routing table, which is a whole different subject.

Maybe you used different kernel−parameters on each machine? Especially if you use the 'Support clusters with a complex network topology' option you should take care that you use the same value for the also appearing option 'Maximum network−topology complexity support' on each machine.

11.3. I often get errors: No such process

I often get the error

```
bash: child setpgid (4061 to 4061): No such process
```

what does this mean ?

The above line meas that the shell you were using has acutallly migrated to another node ? This printout from bash is caused by a bug in old version of openmosix, but a fix has been commited. (Muli Ben−Yehuda mulix@actcom.co.il)

Chapter 12. openMosixview

12.1. Introduction

openMosixview is the next version and a complete rewrite of Mosixview. It is a cluster−management GUI for openMosix−cluster and everybody is invited to download and use it (at your own risk and responsibility). The openMosixview−suite contains 5 usefull applications for monitoring and administrating openMosix−cluster.

openMosixview the main monitoring+admistration application

openMosixprocs a process−box for managing processes

openMosixcollector collecting daemon which logs cluster+node informations

openMosixanalyzer for analyzing the data collected by the openMosixcollector

openMosixhistory a process−history for your cluster

All parts are accessable from the main application window. The most common openMosix−commands are executable by a few mouse−clicks. An advanced execution dialog helps to start applications on the cluster. "Priority−sliders" for each node simplifying the manual and automatic load−balancing. openMosixview is now adapted to the openMosix−autodiscovery and gets all configuration−values from the openMosix /proc−interface.

12.2. openMosixview vs Mosixview

openMosixview is fully designed for openMosix cluster only. The Mosixview−website (and all mirrors) will stay as they are but all further developing will continue with openMosixview located at the new domain *www.openmosixview.com*

If you have: questions, features wanted, problems during installation, comments, exchange of experiences etc. feel free to mail me, Matt Rechenburg or subscribe to the openMosix/Mosixview−mailinglist and mail to the openMosix/Mosixview−mailinglist

changes: (to Mosixview 1.1) openMosixview is a complete rewrite "from the scratch" of Mosixview! It has the same functionalities but there are fundamental changes in ALL parts of the openMosixview source−code. It is tested with a constantly changing cluster topography (required for the openMosix auto−discovery) All "buggy" parts are removed or rewritten and it (should ;) run much more stable now.

adapted to the openMosix−autodiscovery

not using /etc/mosix.map or any cluster−map file anymore

removed the (buggy) map−file parser

rewrote all parts/functions/methods to a cleaner c++ interface

fixed some smaller bugs in the display

replaced MosixMem+Load with the openMosixanalyzer

.. many more changes

12.3. Installation

Requirements

QT >= 2.3.0

root rights !

rlogin and rsh (or ssh) to all cluster−nodes without password the openMosix userland−tools mosctl, migrate, runon, iojob, cpujob ... (download them from the www.openmosix.org website)

Documentation about openMosixview There is a full HTML−documentation about openMosixview included in every package. You find the startpage of the docu in your openMosixview installation directory: openmosixview/openmosixview/docs/en/index.html

The RPM−packages have their installation directories in: /usr/local/openmosixview

12.3.1. Installation of the RPM−distribution

Download the latest version of openMosixview rpm−package. Then just execute e.g.:

```
rpm −i openmosixview−1.2.rpm
```

This will install all binaries in /usr/bin To uninstall:

```
rpm −e openmosixview
```

12.3.2. Installation of the source−distribution

Download the latest version of openMosixview and unzip+untar the sources and copy the tarball to e.g. /usr/local/.

```
gunzip openmosixview−1.2.tar.gz
tar −xvf openmosixview−1.2.tar
```

12.3.3. Automatic setup−script

Just cd to the openmosixview−directory and execute

```
./setup [your_qt_2.3.x_installation_directory]
```

12.3.4. Manual compiling

Set the QTDIR−Variable to your actual QT−Distribution, e.g.

```
export QTDIR=/usr/lib/qt−2.3.0 (for bash)
or
setenv QTDIR /usr/lib/qt−2.3.0 (for csh)
```

12.3.5. Hints

(from the testers of openMosixview/Mosixview who compiled it on diffrent linux−distributions, thanks again) Create the link /usr/lib/qt pointing to your QT−2.3.x installation e.g. if QT−2.3.x is installed in /usr/local/qt−2.3.0

```
ln -s /usr/local/qt-2.3.0 /usr/lib/qt
```

Then you have to set the QTDIR environment variable to

```
export QTDIR=/usr/lib/qt (for bash)
or
setenv QTDIR /usr/lib/qt (for csh)
```

After that the rest should work fine:

```
./configure
make
```

then do the same in the subdirectory openmosixcollector, openmosixanalyzer, openmosixhistory and openmosixviewprocs. Copy all binaries to /usr/bin

```
cp openmosixview/openmosixview /usr/bin
cp openmosixviewproc/openmosixviewprocs/mosixviewprocs /usr/bin
cp openmosixcollector/openmosixcollector/openmosixcollector /usr/bin
cp openmosixanalyzer/openmosixanalyzer/openmosixanalyzer /usr/bin
cp openmosixhistory/openmosixhistory/openmosixhistory /usr/bin
```

And the openmosixcollector init−script to your init−directory e.g.

```
cp openmosixcollector/openmosixcollector.init /etc/init.d/openmosixcollector
or
cp openmosixcollector/openmosixcollector.init /etc/rc.d/init.d/openmosixcollector
```

Now copy the openmosixprocs binary on each of your cluster−nodes to /usr/bin/openmosixprocs

```
rcp openmosixprocs/openmosixprocs your_node:/usr/bin/openmosixprocs
```

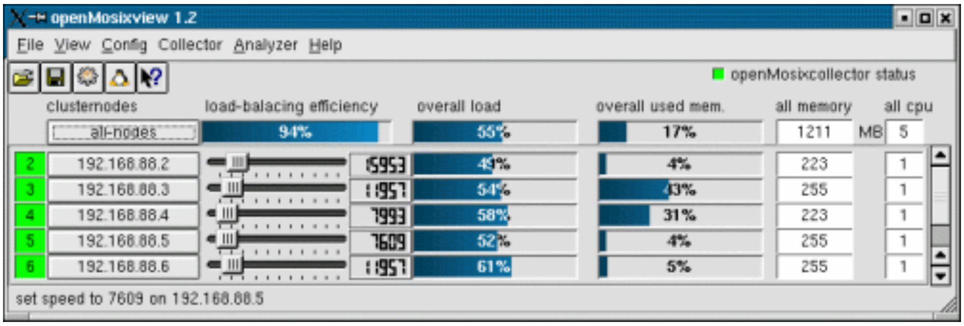
You can now execute mosixview

```
openmosixview
```

12.4. using openMosixview

12.4.1. main application

Here is a picture of the main application−window. The functionality is explained in the following.



	clusternodes	load-balacing efficiency		overall load	overall used mem.	all memory		all cpu
	all-nodes	94%		55%	17%	1211	MB	5
2	192.168.88.2		15953	49%	4%	223		1
3	192.168.88.3		11957	54%	43%	255		1
4	192.168.88.4		7993	58%	31%	223		1
5	192.168.88.5		7609	52%	4%	255		1
6	192.168.88.6		11957	61%	5%	255		1

set speed to 7609 on 192.168.88.5

openMosixview displays a row with a lamp, a button, a slider, a lcd−number, two progressbars and some labels for each cluster−member. The lights at the left are displaying the openMosix−Id and the status of the cluster−node. Red if down, green for avaiable.

If you click on a button displaying the ip−address of one node a configuration−dialog will pop up. It shows buttons to execute the most common used "mosctl"−commands. (described later in this HowTo) With the "speed−sliders" you can set the openMosix−speed for each host. The current speed is displayed by the lcd−number.

You can influence the load−balancing of the whole cluster by changing these values. Processes in a openMosix−Cluster are migrating easier to a node with more openMosix−speed than to nodes with less speed. Sure it is not the physically speed you can set but it is the speed openMosix "thinks" a node has. e.g. a cpu−intensive job on a cluster−node which speed is set to the lowest value of the whole cluster will search for a better processor for running on and migrate away easily.

The progressbars in the middle gives an overview of the load on each cluster−member. It displays in percent so it does not represent exactly the load written to the file /proc/hpc/nodes/x/load (by openMosix), but it should give an overview.

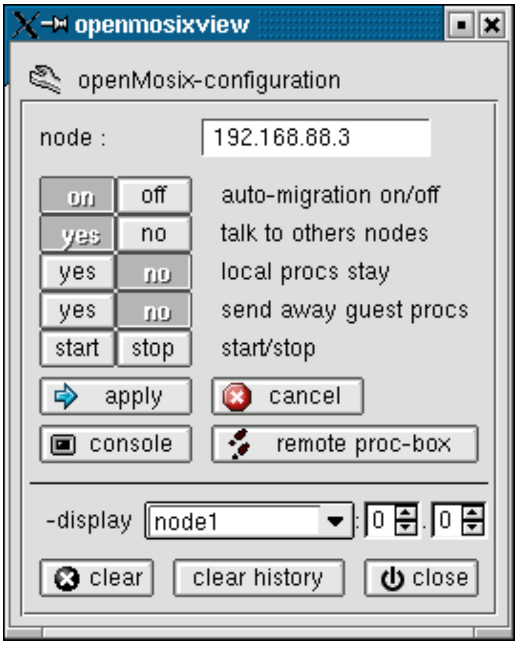
The next progressbar is for the used memory the nodes. It shows the currently used memory in percent from the avaiable memory on the hosts (the label to the right displays the avaiable mem). How many CPUs your cluster have is written in the box to the right. The first line of the main windows contains a configuration button for "all−nodes". You can configure all nodes in your cluster similar by this option.

How good the load−balancing works is displayed by the progressbar in the top left. 100% is very good and means that all nodes nearly have the same load.

Use the collector− and analyzer−menu to manage the openMosixcollector and open the openMosixanalyzer. This two parts of the openMosixview−application suite are usefull for getting an overview of your cluster during a longer period.

12.4.2. the configuration−window

This dialog will popup if an "cluster−node"−button is clicked.



openMosix-configuration

node : 192.168.88.3

[✓ on] off auto-migration on/off

[✓ yes] no talk to others nodes

yes [✓ no] local procs stay

yes [✓ no] send away guest procs

start stop start/stop

-display node1 : 0 . 0

The openMosix−configuration of each host can be changed easily now. All commands will be executed per "rsh" or "ssh" on the remote hosts (even on the local node) so "root" has to "rsh" (or "ssh") to each host in the cluster without prompting for a password (it is well described in a beowulf documentation or on the HowTo's on this page how to configure it).

The commands are:

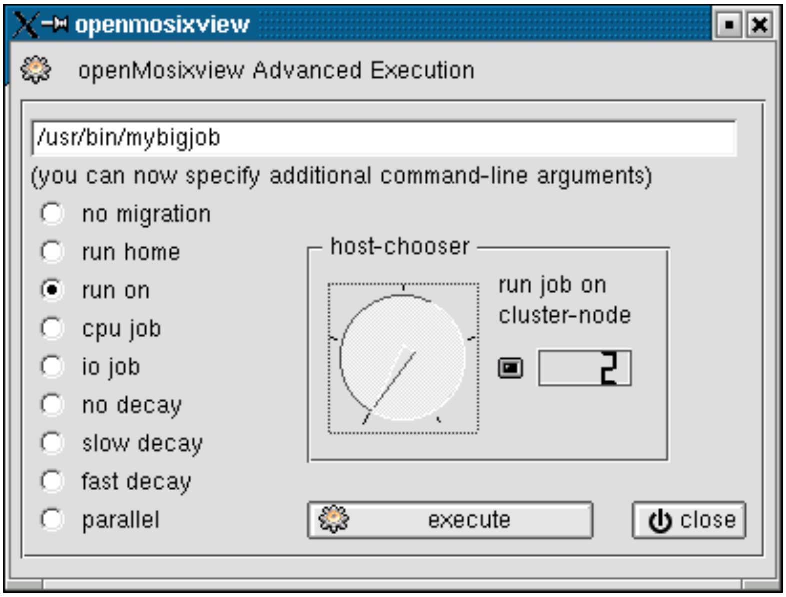
```
automigration on/off
quiet yes/no
bring/lstay yes/no
exspel yes/no
openMosix start/stop
```

If openMosixprocs is properly installed on the remote cluster−nodes click the "remote proc−box"−button to open openMosixprocs (proc−box) from remote. xhost +hostname will be set and the display will point to your localhost. The client is executed on the remote also per "rsh" or "ssh". (the binary openmosixprocs must be copied to e.g. /usr/bin on each host of the cluster) openMosixprocs is a process−box for managing your programs. It is usefull to manage programs started and running local on the remote nodes and is described later in this HowTo.

If you are logged on your cluster from a remote workstation insert your local hostname in the edit−box below the "remote proc−box". Then openMosixprocs will be displayed on your workstation and not on the cluster−member you are logged on. (maybe you have to set "xhost +clusternode" on your workstation). There is a history in the combo−box so you have to write the hostname only once.

12.4.3. advanced−execution

If you want to start jobs on your cluster the "advanced execution"−dialog may help you.



openMosixview Advanced Execution

/usr/bin/mybigjob

(you can now specify additional command-line arguments)

○ no migration
○ run home
● run on
○ cpu job
○ io job
○ no decay
○ slow decay
○ fast decay
○ parallel

host-chooser

run job on cluster-node 2

Choose a program to start with the "run−prog" button (fileopen−icon) and you can specify how and where the job is started by this execution−dialog. There are several options to explain.

12.4.4. the command−line

You can specify additional commandline−arguments in the lineedit−widget on top of the window.

Table 12−1. how to start

−no migration	start a local job which won't migrate
−run home	start a local job
−run on	start a job on the node you can choose with the "host−chooser"
−cpu job	start a computation intensive job on a node (host−chooser)
−io job	start a io intensive job on a node (host−chooser)
−no decay	start a job with no decay (host−chooser)
−slow decay	start a job with slow decay (host−chooser)
−fast decay	start a job with fast decay (host−chooser)
−parallel	start a job parallel on some or all node (special host−chooser)

12.4.5. the host−chooser

For all jobs you start non−local simple choose a host with the dial−widget. The openMosix−id of the node is also displayed by a lcd−number. Then click execute to start the job.

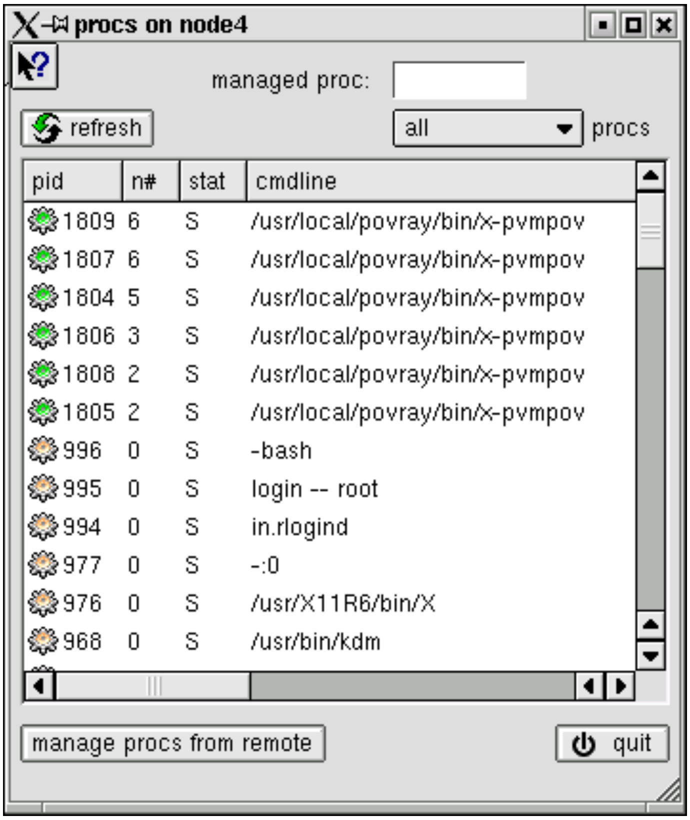
12.4.6. the parallel host−chooser

You can set the first and last node with 2 spinboxes. Then the command will be executed an all nodes from the first node to the last node. You can also inverse this option.

12.5. openMosixprocs

12.5.1. intro

This process−box is really usefull for managing the processes running on your cluster.



pid	n#	stat	cmdline
1809	6	S	/usr/local/povray/bin/x-pvmpov
1807	6	S	/usr/local/povray/bin/x-pvmpov
1804	5	S	/usr/local/povray/bin/x-pvmpov
1806	3	S	/usr/local/povray/bin/x-pvmpov
1808	2	S	/usr/local/povray/bin/x-pvmpov
1805	2	S	/usr/local/povray/bin/x-pvmpov
996	0	S	-bash
995	0	S	login -- root
994	0	S	in.rlogind
977	0	S	-:0
976	0	S	/usr/X11R6/bin/X
968	0	S	/usr/bin/kdm

You should install it on every cluster−node!

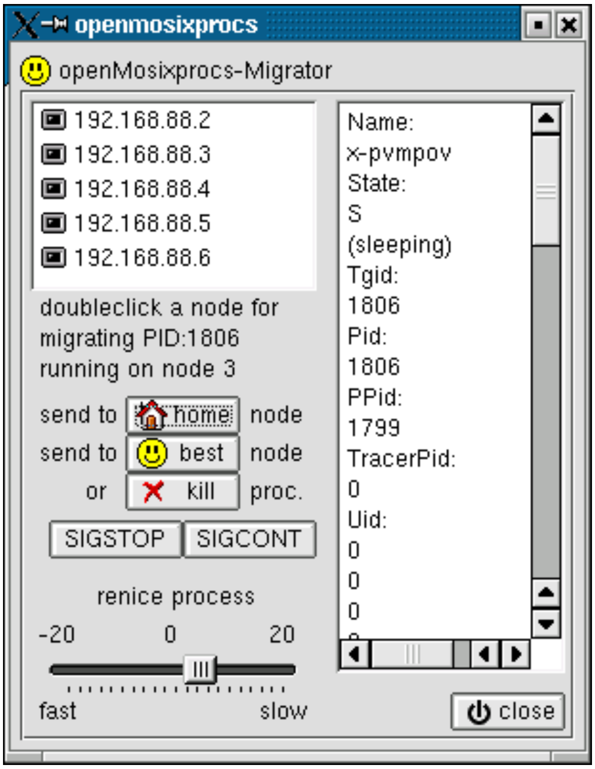
The processlist gives an overview what is running where. The second column displays the openMosix−node ID of each process. 0 means local, all other values are remote nodes. Migrated processes are marked with a green icon and nonmoveable processes have a lock.

By doubleclicking a process from the list the migrator−window will pop−up for managing e.g. migrating the process. There are also options to migrate the remote processes away, send SIGSTOP and SIGCONT to it or to "renice" it.

If you click on the "manage procs from remote" button a new window will come up (the remote−procs windows) displaying the process currently migrated to this host.

12.5.2. the migrator−window

This dialog will popup if process from the processbox is clicked.



192.168.88.2
192.168.88.3
192.168.88.4
192.168.88.5
192.168.88.6

doubleclick a node for migrating PID:1806 running on node 3

```
Name:
x-pvmpov
State:
S
(sleeping)
Tgid:
1806
Pid:
1806
PPid:
1799
TracerPid:
0
Uid:
0
0
0
```

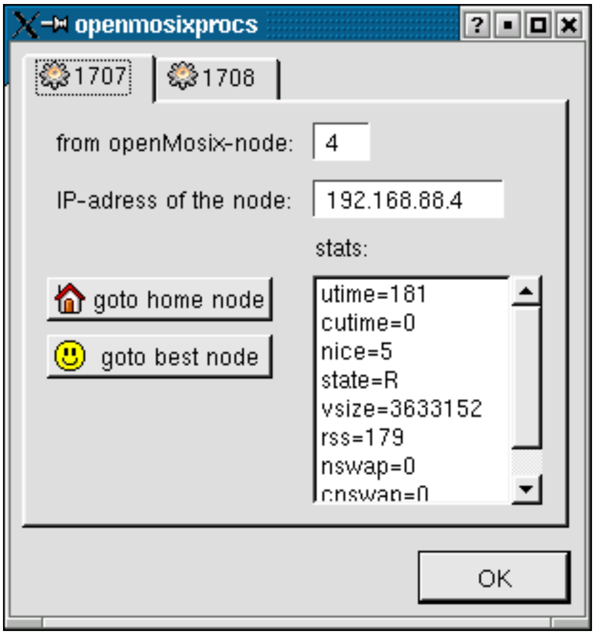
The openMosixview−migrator window displays all nodes in your openMosix−cluster. This window is for managing one process (with additional status−information). By doubleclicking on an host from the list the process will migrate to this host. After a short moment the process−icon for the managed process will be green, which means it is running remote.

The "home"−button sends the process to its home node. With the "best"−button the process is send to the best avaiable node in your cluster. This migration is influenced by the load, speed, cpu's and what openMosix "thinks" of each node. It maybe will migrate to the host with the most cpu's and/or the best speed. With the "kill"−button you can kill the process immediatly.

To pause a program just click the "SIGSTOP"−button and to continue the "SIGCONT"−button. With the renice−slider below you can renice the current managed process (−20 means very fast, 0 normal and 20 very slow)

12.5.3. managing processes from remote

This dialog will popup if the "manage procs from remote"−button beneath the process−box is clicked



from openMosix-node: 4

IP-adress of the node: 192.168.88.4

stats:

```
utime=181
cutime=0
nice=5
state=R
vsize=3633152
rss=179
nswap=0
cnswap=0
```

The TabView displays processes that are migrated to the local host. The procs are coming from other nodes in your cluster and currently computed on the host openMosixview is started on. Similar to the two buttons in the migrator−window the process is send home by the "goto home node"−button and send to the best avaiable node by the "goto best node"−button.

12.6. openMosixcollector

The openMosixcollector is a daemon which should/could be started on one cluster−member. It logs the openMosix−load of each node to the directory /tmp/openmosixcollector/* These history log−files analyzed by the openMosixanalyzer (as described later) gives an nonstop overview of the load, memory and processes in your cluster. There is one main log−file called /tmp/openmosixcollector/cluster Additional to this there are additional files in this directory to which the data is written.

At startup the openMosixcollector writes its PID (process id) to /var/run/openMosixcollector.pid

The openMosixcollector−daemon restarts every 12 hours and saves the current history to /tmp/openmosixcollector[date]/* These backups are done automatically but you can also trigger this manual.

There is an option to write a checkpoint to the history. These checkpoints are graphically marked as a blue vertical line if you analyze the history log−files with the openMosixanalyzer. For example you can set a checkpoint when you start a job on your cluster and another one at the end..

Here is the explanation of the possible commandline−arguments:

```
openmosixcollector −d      //starts the collector as a daemon
openmosixcollector −k      //stops the collector
openmosixcollector −n      //writes a checkpoint to the history
openmosixcollector −r      //saves the current history and starts a new one
openmosixcollector         //print out a short help
```

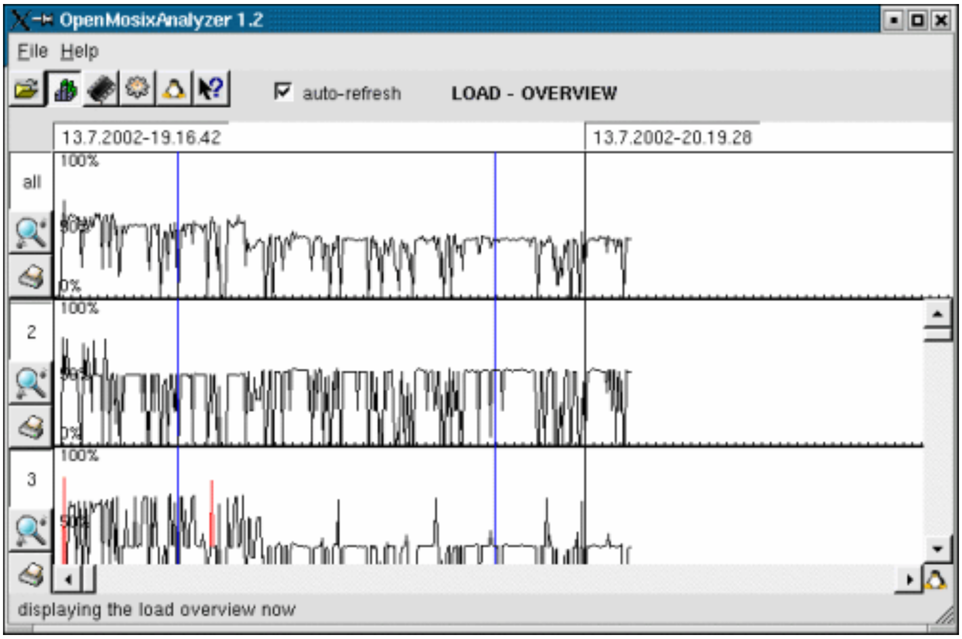
You can start this daemon whith its init−script in /etc/init.d or /etc/rc.d/init.d. You just have to create a symbolic link to one of the runlevels for automatic startup.

How to analyze the created logfiles is described in the openMosixanalyzer−section.

12.7. openMosixanalyzer

12.7.1. the load−overview

This picture shows the graphical Load−overview in the openMosixanalyzer (Click to enlarge)



displaying the load overview now

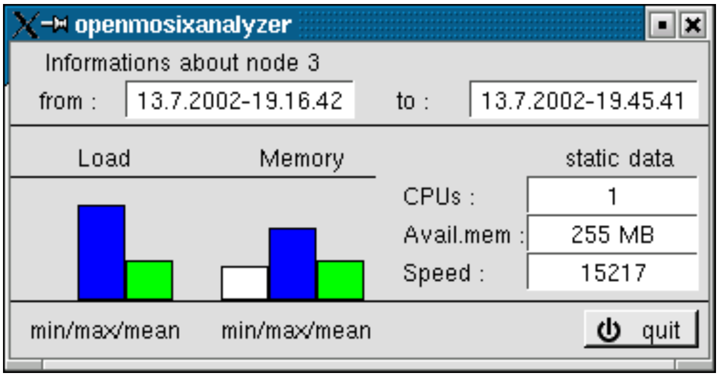
With the openMosixanalyzer you can have a non−stop openMosix−history of your cluster. The history log−files created by openMosixcollector are displayed in a graphically way so that you have a long−time overview what happened and happens on your cluster. The openMosixanalyzer can analyze the current "online" logfiles but you can also open older backups of your openMosixcollector history logs by the filemenu. The logfiles are placed in /tmp/openmosixcollector/* (the backups in /tmp/openmosixcollector[date]/*) and you have to open only the main history file "cluster" to take a look at older load−informations. (the [date] in the backup directories for the log−files is the date the history is saved) The start time is displayed on the top and you have a full−day view in the openMosixanalyzer (12 h).

If you are using the openMosixanalyzer for looking at "online"−logfiles (current history) you can enable the "refresh"−checkbox and the view will auto−refresh.

The load−lines are normally black. If the load increases to >75 the lines are drawn red. These values are openMosix−−informations. The openMosixanalyzer gets these informations from the files /proc/hpc/nodes/[openMosix ID]/*

The Find−out−button of each nodes calculates several usefull statistic values. Clicking it will open a small new window in which you get the avarage load− and mem values and some more statically and dynamic informations about the specific node or the whole cluster.

12.7.2. statistical informations about a cluster−node



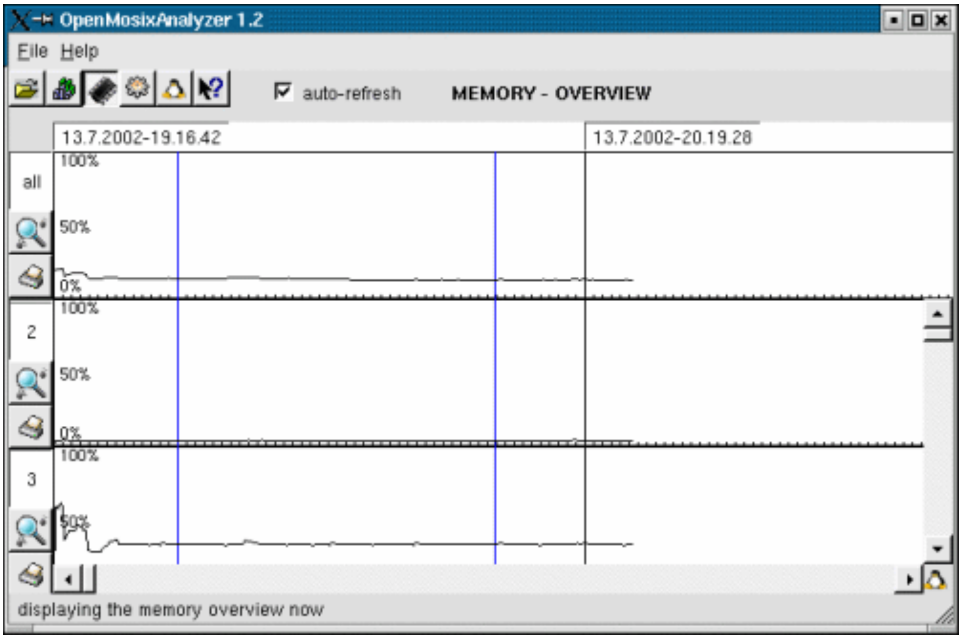
Informations about node 3

from : 13.7.2002-19.16.42 to : 13.7.2002-19.45.41

CPUs : 1
Avail.mem : 255 MB
Speed : 15217

If there are checkpoints written to the load−history by the openMosixcollector they are displayed as a vertical blue line. You now can compare the load values at a certain moment much easier.

12.7.3. the memory−overview



13.7.2002-19.16.42 13.7.2002-20.19.28

displaying the memory overview now

This picture shows the graphical Memory−overview in the openMosixanalyzer

With Memory−overview in the openMosixanalyzer you can have a non−stop memory history similar to the Load−overview. The history log−files created by openMosixcollector are displayed in a graphically way so that you have a long−time overview what happened and happens on your cluster. It analyze the current "online" logfiles but you can also open older backups of your openMosixcollector history logs by the filemenu.

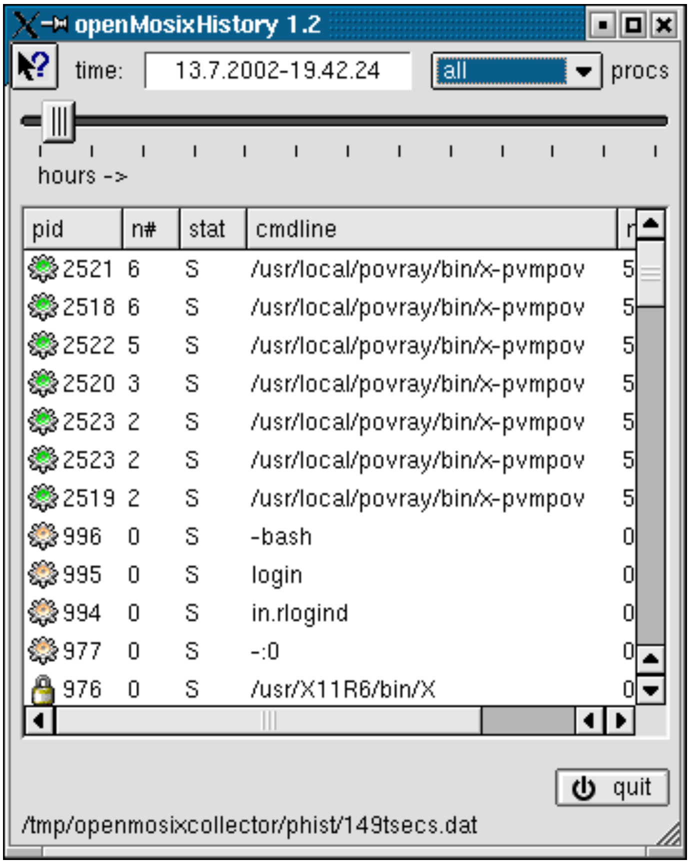
The displayed values are openMosix−informations. The openMosixanalyzer gets these informations from the files

```
/proc/hpc/nodes/[openMosix−ID]/mem.
/proc/hpc/nodes/[openMosix−ID]/rmem.
```

```
/proc/hpc/nodes/[openMosix−ID]/tmem.
```

If there are checkpoints written to the memory−history by the openMosixcollector they are displayed as a vertical blue line.

12.7.4. openMosixhistory



time: 13.7.2002-19.42.24

all procs

pid	n#	stat	cmdline	r
2521	6	S	/usr/local/povray/bin/x-pvmpov	5
2518	6	S	/usr/local/povray/bin/x-pvmpov	5
2522	5	S	/usr/local/povray/bin/x-pvmpov	5
2520	3	S	/usr/local/povray/bin/x-pvmpov	5
2523	2	S	/usr/local/povray/bin/x-pvmpov	5
2523	2	S	/usr/local/povray/bin/x-pvmpov	5
2519	2	S	/usr/local/povray/bin/x-pvmpov	5
996	0	S	-bash	0
995	0	S	login	0
994	0	S	in.rlogind	0
977	0	S	-:0	0
976	0	S	/usr/X11R6/bin/X	0

/tmp/openmosixcollector/phist/149tsecs.dat

displays the processlist from the past

openMosixhistory gives a detailed overview which process was running on which node. The openMosixcollector saves the processlist from the host the collector was started on and you can browse this log−data with openMosixhistory. You can easy change the browsing time in openMosixhistory by the time−slider.

openMosixhistory can analyze the current "online" logfiles but you can also open older backups of your openMosixcollector history logs by the filemenu.

The logfiles are placed in /tmp/openmosixcollector/* (the backups in /tmp/openmosixcollector[date]/*) and you have to open only the main history file "cluster" to take a look at older load−informations. (the [date] in the backup directories for the log−files is the date the history is saved) The start time is displayed on the top/left and you have a 12 hour view in openMosixhistory.

12.8. openmosixview faq

12.8.1. I cannot compile openMosixview on my system?

At first QT >= 2.3.x is required. The QTDIR −environent variable has to be set to your QT−installation directories like it is well described in the INSTALL− file. In versions < 0.6 you can do a "make clean" and delete the two files: /openmosixview/Makefile /openmosixview/config.cache and try to compile again because i alway left the binary− and object−files in older versions. If you have any other problems post them to the openMosixview−mailinglist (or directly to me).

12.8.2. Can I use openMosixview with SSH?

Yes, until version 0.7 there is a built−in SSH−support. You have to be able to ssh to each node in your cluster without password (just like the same with using RSH this is required)

12.8.3. I started openMosixview but only the splash−screen appears. What is wrong?

Do not fork openMosixview in the background with & (e.g. openMosixview &). Maybe you cannot rsh/ssh (depends on what you want to use) as user root without password to each node? Try "rsh hostname" as root. You should not been promped for a password but soon get a login shell. (If you use SSH try "ssh hostname" as root.) You have to be root on the cluster because that is the only way the administrative commands executed by openMosixview requires root−privileges. openMosixview uses "rsh" as the default! If you only have "ssh" installed on your cluster edit (or create) the file /root/.openMosixview and put "1111" in it. This is the main−configuration file for openMosixview and the last "1" stands for "use ssh instead of rsh". This will cause openMosixview to use "ssh" even for the first start.

12.8.4. The openMosixviewprocs/mosixview_client is not working for me!

The openMosixview−client is executed per rsh (or ssh which you can configer whith a checkbox) on the remote host. It has to be installed in /usr/bin/ on each node. If you use RSH try: "xhost +hostname" "rsh hostname /usr/bin/openMosixview_client −display your_local_host_name:0.0" or if you use SSH try: "xhost +hostname" "ssh hostname /usr/bin/openMosixview_client −display your_local_host_name:0.0" If this works it will work in openMosixview too. openMosixview crashes with "segmantation fault"! Maybe you still use an old version of openMosixview/Mosixview ? in the mosix.map−parser (which is completly removed in openMosixview !!) (the versions openMosixview 1.2 and Mosixview > 1.0 are stable)

12.8.5. Why are the buttons in the openMosixview−configuration dialog not preselected?

(automigration on/off, blocking on/off......) I want them to be preselected too. The problem is to get the information of node. You have to login to each cluster−node because these information are not cluster−wide (to my mind). The status of each node is stored in the /proc/hpc/admin directory of each node. Everybody who knows a good way to get these information easy is invited to mail me.

12.9. openMosixview + ssh:

(this HowTo is for SSH2) You can read the reasons why you should use SSH instead of RSH everyday on the newspaper when another script−kiddy hacked into an insecure system/network. So SSH is a good decision at all.

```
freedom x security = constant    (from a security newsgroup)
```

That is why it is a bit tricky to configure SSH. SSH is secure even if you use it to login without being prompted for a password. Here is a (one) way to configure it.

At first a running secure−shell daemon on the remote site is required. If it is not already installed install it! (rpm −i [sshd_rpm_packeage_from_your_linux_distribution_cd]) If it is not already running start it with:

```
/etc/init.d/ssh start
```

Now you have to generate a keypair for SSH on your local computer whith ssh−keygen.

```
ssh−keygen
```

You will be prompt for a passphrase for that keypair. The passphrase normally is longer than a password and may be a whole sentence. The keypair is encrypted with that passphrase and saved in

```
/root/.ssh/identity    //your private key
and
/root/.ssh/identity.pub     //your public key
```

Do NOT give your private−key to anybody!!! Now copy the whole content of /root/.ssh/identity.pub (your public−key which should be one long line) into /root/.ssh/authorized_keys on the remote host. (also copy the content of /root/.ssh/identity.pub to your local /root/.ssh/authorized_keys like you did it with the remote−node because openMosixview needed password−less login to the local−node too!)

If you ssh to this remote host now you will be prompted for the passphrase of your public−key. Giving the right passphrase should give you a login.

What is the advantage right now??? The passphrase is normally a lot longer than a password!

The advantage you can get using the ssh−agent. It manages the passphrase during ssh login.

```
ssh−agent
```

The ssh−agent is started now and gives you two environment−variables you should set (if not set already). Type:

```
echo $SSH_AUTH_SOCK
and
echo $SSH_AGENT_PID
```

to see if they are exported to your shell right now. If not just cut and paste from your terminal. e.g. for the bash−shell:

```
SSH_AUTH_SOCK=/tmp/ssh−XXYqbMRe/agent.1065
export SSH_AUTH_SOCK
SSH_AGENT_PID=1066
export SSH_AGENT_PID
```

example for the csh−shell:

```
setenv SSH_AUTH_SOCK /tmp/ssh−XXYqbMRe/agent.1065
setenv SSH_AGENT_PID 1066
```

With these variables the remote−sshd−daemon can connect your local ssh−agent by using the socket−file in /tmp (in this example /tmp/ssh−XXYqbMRe/agent.1065). The ssh−agent can now give the passphrase to the remote host by using this socket (it is of course an encrypted transfer)!

You just have to add your public−key to the ssh−agent with the ssh−add command.

```
ssh−add
```

Now you should be able to login using ssh to the remote host without being prompted for a passwod!

You could (should) add the ssh−agent and ssh−add commands in your login−profile e.g.

```
eval `ssh−agent`
 ssh−add
```

Now it is started when you login on your local workstation. You have done it! I wish you secure logins now.

openMosixview There is a menu−entry which toggles using rsh/ssh with openMosixview. Just enable this and you can use openMosixview even in insecure network−environments. You should also save this configuration (the possibility for saveing the current config in openMosixview was added in the 0.7 version) because it gets initial data from the slave using rsh or ssh (just like you configured).

If you choose a service wich is not installed properly openMosixview will not work! (e.g. if you cannot rsh to a slave without being prompted for a password you cannot use openMosixview with RSH; if you cannot ssh to a slave without being prompted for a password you cannot use openMosixview with SSH)

Chapter 13. Hints and Tips

13.1. Locked Processes

If for some reason you find your processes are always locked in your home node and you can't find the reason, you can put the following lines into your ~/.profile as a stop−gap measure to automatically enable migration:

```
        if [ −x /proc/$$/lock ]; then
           echo 0 > /proc/$$/lock
        fi
```

However, you should make an effort to find out what the problems is − see the Mosix FAQ at *http://www.mosix.org/* for details.

13.2. Choosing your processes

You will probably want to test your setup before deciding which programs you want to enable migration for. For example, if you are running KDE2 on a slow machine and have a significantly faster machine has part of your Mosix cluster, you might find resource−hungry programs like kmail are migrated out. This is not a bad thing as such, however, it can lead to a brief moment when your writing is not displayed on the screen immediately.

13.3. Java and openMosix

Green Threads JVMs, allow for migration because each Java thread is a seperate process. Threads other than Java green thread JVMs cannot be migrated by Linux, so openMosix cannot migrate programs that use them. If you have the source so your Java application you might be able to compile the application nativ. In this case you might be able to migrate your applications to another node. However this still needs to be tested and documented.

Appendix A. More Info

A.1. Further Reading

A.2. Links

- *Mosix Debian HOWTO*
- *Mosix Mandrake Linux Terminal Server Project*
- *MOSIXVIEW, a GUI for managing openMosix−Cluster*
- *User Mode openMosix, a virtual openMosix cluster running in User−mode*
- *RxLinux, Web Interface for central configuration and management*
- *LTSP+OpenMosix: A Mini How−To*
- *FuBAR: An openMosix cluster at Texas AM − Corpus Christi*

A.3. Mailing List

- *openMosix mailing list*
- *http://lists.sourceforge.net/lists/listinfo/mosixview−user openmosix−view mailing list*
- *ClumpOS mailing list*

Appendix B. List of Working Software

This Appendix will contain a list of applications both working and non working as reported by openMosix users. Where possible a reason will be given why the application does not work.

Applications using shared memory continue to run as under standard Linux, but currently will not migrate. Threads other than Java green thread JVMs cannot be migrated by Linux, so openMosix cannot migrate programs that use them.

Table B−1. Working Software

Blast (patched)	*http://stl.bioinformatics.med.umich.edu/software/OM_BLAST_patch/openmosixpatch.html*
MJPEG tools	Because it uses both i/o intense and cpu intense pipes, it works very well on small clusters. The encoding and decoding are bound on the home node but the filters are migrated to the nodes increasing performace of very high quality file encoding.
bladeenc	easy rip your mp3 collection with: cdparanoia −B for n in \`ls *.wav\`; do bladeenc −quit −quiet $n −256 −copy −crc & done;
POVRAY	Spread your pic−frames to multiple proccesses by a shell script or use the parallel (PVM) version to do this automatically.
MPI	openMosix and MPI are like bread and peanut butter, they just love each other.
FLAC	A lossless audio encoder. httphttp://flac.sourceforge.net/

Appendix C. List of Non Working Software

Table C−1. Non Working Software

<table>
<tr><td>Java programs using native threads</td><td>do not migrate since they use shared memory. Green Threads JVMs, however, allow for migration because each Java thread is a seperate process.</td></tr>
<tr><td>Applications using pthreads</td><td>Moshe wrote this answer about threading and openMosix to the mailing list.
<pre>
From: Moshe Bar
To: openmosix−generallists.sourceforge.net
Subject: pthreads and The Matrix
Date: 07 Aug 2002 06:30:56 +0200
HI Larry

Not being able to migrate pthreads is not an openMosix limitation but a
Linux one. Contrary to platforms like Solaris where threads are
light−weight processes with their own address space, threads in Linux do
not have their own memory address space. You can see the threads with ps
because each thread is a "task" for the Linux scheduler. However, that
task cannot live on its own, it needs the address space where it was
born. If we migrate the pthread to another machine, which address space
would it be connected to? I am sure you saw Matrix, and at one poine
Neon asks what happends if somebody dies in the Matrix, if he would also
die in the real world. Trinity answered to him that a body (a pthread)
cannot live with a mind (ie memory, ie address space).

A great many things can be learned from Matrix.

But coming back to our subject here, once we have distributed shared
memory, we will be able to connect remote pthreads to their address
space back home. But that is still some time down the road. I guess
Matrix 2 will be out before we have distributed shared memory.

Moshe
</pre></td></tr>
<tr><td>mySQL</td><td>uses shared memory.</td></tr>
<tr><td>Apache</td><td>uses shared memory.</td></tr>
<tr><td>Mathematica</td><td>uses shared memory.</td></tr>
<tr><td>SAP</td><td>uses shared memory</td></tr>
<tr><td>Oracle</td><td>uses shared memory</td></tr>
<tr><td>Baan</td><td>uses shared memory</td></tr>
<tr><td>Postgres</td><td>uses shared memory</td></tr>
<tr><td>Python with threading enabled</td><td></td></tr>
</table>

A more recently updated list of both working and non working software can be found on *the wiki pages*

Appendix D. Credits

Scot W. Stevenson

I have to thank Scot W. Stevenson for all the work he did on this HOWTO before I took over. He made a great start for this document.

Assaf Spanier

worked together with Scott in drafting the layout and the chapters of this HOWTO. and now promised to help me out with this document.

Matthias Rechenburg

Matthias Rechenburg should be thanked for the work he did on Mosixview and the accompanying documentation , which we included in this HOWTO.

Jean−David Marrow

is the author of Clump/OS, he contributed the documentation on his distribution to the HOWTO.

Appendix E. GNU Free Documentation License

Version 1.1, March 2000

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other written document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or non−commercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you".

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front−matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The "Cover Texts" are certain short passages of text that are listed, as Front−Cover Texts or Back−Cover Texts, in the notice that says that the Document is released under this License.

A "Transparent" copy of the Document means a machine−readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard−conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine−generated HTML produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies of the Document numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front−Cover Texts on the front cover, and Back−Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine−readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly−accessible computer−network location containing a complete Transparent copy of the Document, free of added material, which the general network−using public has access to download anonymously at no charge using public−standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section entitled "History", and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. In any section entitled "Acknowledgements" or "Dedications", preserve the section's title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section as "Endorsements" or to conflict in title with any Invariant Section.

If the Modified Version includes new front−matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties−−for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front−Cover Text, and a passage of up to 25 words as a Back−Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front−Cover Text and one of Back−Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled "History" in the various original documents, forming one section entitled "History"; likewise combine any sections entitled "Acknowledgements", and any sections entitled "Dedications". You must delete all sections entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an "aggregate", and this License does not apply to the other self−contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document's Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front−Cover Texts being LIST, and with the Back−Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front−Cover Texts, write "no Front−Cover Texts" instead of "Front−Cover Texts being LIST"; likewise for Back−Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.