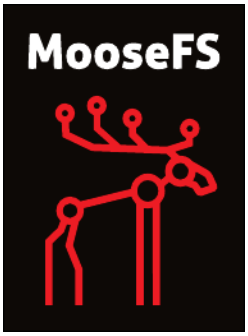


Installing MooseFS 2.0 Step by Step Tutorial

Core Technology Development & Support Team

October 17, 2014

© 2013-2014 v. 1.4.1
Piotr Robert Konopelko, Core Technology Development & Support Team.
All rights reserved.

Proofread by Agata Kruszona-Zawadzka
Coordination & layout by Piotr Robert Konopelko.

Please send corrections to peter@mfs.io.

Contents

Chapter 1

Introduction

Notice: there is one dependency to resolve: users' computers need FUSE package to mount MooseFS. It can be downloaded and installed from repositories.

1.1 Key differences between versions 1.6.2x and 2.0.x

1. Master host(s) configuration is done solely via DNS – it is no longer possible to list master(s) IP address(es) in clients' and chunkservers' configuration; default name for master domain is `mfsmaster`, it can be changed in configuration files;
2. In Pro version metaloggers become optional, they can be replaced by additional master servers; in CE version it is still recommended to set up metaloggers.
3. `Mfsmetarestore` tool is no longer present in the system; instead, it is enough to start the master process with `-a` switch;
4. Configuration files now sit in `mfs` subdirectory inside the `etc` directory (this change was introduced in 1.6.27).

1.2 Many Master Servers – how does it work?

In previous MooseFS versions you had only one master process and any number of metaloggers. In the event of master failure, system administrator was able to retrieve "metadata" information from the metalogger and start a new master (on a new machine, if necessary), so the file system was up and running again. But this was always causing the system to be unavailable to clients for a period of time and required manual work to bring it back up.

New Pro version introduces many master servers working together in multiple roles. One role is "leader". The leader master is acting as it used to for the chunkservers and clients. There is never more than one leader in any working system.

The other role is "follower". The follower master is doing what metaloggers used to do – it downloads metadata from the leader master and keeps it. But unlike a metalogger, if a leader

master stops working, a follower master is immediately ready to take on the role of leader. If the leader master fails, a new candidate for leader is chosen from the followers. The candidate assumes a role of "elect", that automatically converts to "leader" as soon as more than half of the chunkservers connect to elect. There can be more than one follower in the system.

The whole switching operation is almost invisible to the system users, as it usually takes between a couple to a dozen or so seconds. When/if the former leader master starts working again, it assumes the role of follower. If a follower master fails, it has no effect on the whole system. If such a master starts working again, it again assumes the role of follower.

Chapter 2

Things to do before installation

For the sake of this document, it's assumed that your machines have following IP addresses:

- Master servers: 192.168.1.1, 192.168.1.2
- Chunk servers: 192.168.1.101, 192.168.1.102 and 192.168.1.103
- Users' computers (clients): 192.168.2.x

2.1 Configuring Domain Name Service

Before you start installing MooseFS, you need to have working DNS. It's needed for MooseFS to work properly with several master servers, because DNS can resolve one host name as more than one IP address.
All IPs of machines, which will be master servers, must be included in DNS configuration file and resolved as "`mfsmaster`" (or any other selected name), e.g.:

Listing 2.1: DNS entries

```
mfsmaster   IN  A   192.168.1.1     ; address of first master server
mfsmaster   IN  A   192.168.1.2     ; address of second master server
```

More information about configuring DNS server is included in supplement.

2.2 Adding repository

To install MooseFS 2.0 Pro or CE you need to add MooseFS Official Supported Repositories to your system. This process is described at `http://get.moosefs.com` (please select your distribution in menu on the left) or in paragraph 2.2 in document named *Installing MooseFS 2.0 Step by Step Tutorial*.

At this time there are repositories available for Ubuntu / Debian, RHEL / CentOS / Fedora, FreeBSD and MacOS X.

2.2.1 Repository branches

Our repository contains two branches: `stable` and `current`. Version from `stable` branch has been tested both in the production and in our test environment. Version from `current` branch – only in our test environment. MooseFS versions in these branches are upgraded automatically after finishing the tests.

At the time of writing this guide, `stable` branch contains version 2.0.39-1, and `current` branch contains version 2.0.40-1.

`Stable` branch is default and you don't need to make any changes in default URL: `http://ppa.moosefs.com/stable/`.

If you want to use `current` branch, you just need to replace `stable` with `current` after `http://ppa.moosefs.com/` and before `apt`, `yum`, `freebsd` or `osx`, so URL will look like:

```
http://ppa.moosefs.com/current/[rest of url]
```

It is also possible to use version number instead of "branch" if you want to upgrade to a specific version of MooseFS (e.g. 2.0.40-1):

```
http://ppa.moosefs.com/2.0.40/[rest of url]
```

If you want to use this option, please remember you need to manually change version number on each server to the selected one before doing an upgrade.

2.2.2 Ubuntu / Debian

First, add the key:

Listing 2.2: Adding the repo key

```
# wget -O - http://ppa.moosefs.com/stable/apt/moosefs.key | apt-key add -
```

Then add the appropriate entry in `/etc/apt/sources.list.d/moosefs.list`:

- For Ubuntu 14.04 Trusty:
  `deb http://ppa.moosefs.com/stable/apt/ubuntu/trusty trusty main`
- For Ubuntu 12.04 Precise:
  `deb http://ppa.moosefs.com/stable/apt/ubuntu/precise precise main`
- For Ubuntu 10.10 Maverick:
  `deb http://ppa.moosefs.com/stable/apt/ubuntu/maverick maverick main`
- For Debian 7.0 Wheezy:
  `deb http://ppa.moosefs.com/stable/apt/debian/wheezy wheezy main`
- For Debian 6.0 Squeeze:
  `deb http://ppa.moosefs.com/stable/apt/debian/squeeze squeeze main`
- For Debian 5.0 Lenny:
  `deb http://ppa.moosefs.com/stable/apt/debian/lenny lenny main`

After that do:
```
# apt-get update
```

2.2.3 CentOS / Fedora / RHEL

Add the appropriate key to package manager:

Listing 2.3: Adding the repo key

```
# curl "http://ppa.moosefs.com/stable/yum/RPM-GPG-KEY-MooseFS" > /etc/pki/rpm-
    gpg/RPM-GPG-KEY-MooseFS
```

Next you need to add the repository entry and update yum repo:

Listing 2.4: Adding the repo and updating yum repo

```
# curl "http://ppa.moosefs.com/stable/yum/MooseFS.repo" > /etc/yum.repos.d/
    MooseFS.repo
# sudo yum update
```

2.2.4 MacOS X

It's possible to run all components of the system on Mac OS X systems, but most common scenario would be to run the client (`mfsmount`) that enables Mac OS X users to access resources available in MooseFS infrastructure.

In case of Mac OS X – since there's no default package manager – we release `.pkg` files containing only binaries, without any startup scripts, that are normally available in Linux packages.

To install MooseFS CE on Mac please follow the steps:

- download and install FUSE for Mac OS X package from `http://osxfuse.github.io`
- download and install MooseFS packages from `http://ppa.moosefs.com/stable/osx/moosefs-ce-current.pkg`

You should be able to mount MooseFS filesystem in `/mnt/mfs` issuing the following command:

```
$ sudo mfsmount /mnt/mfs
```

If you've exported filesystem with additional options like password protection, you should include those options in `mfsmount` invocation, as described in documentation.

2.3 Differences in package names between Pro and CE version

The packages in MooseFS 2.0 Pro version are named according to the following pattern:

- `moosefs-pro-master`
- `moosefs-pro-cli`
- `moosefs-pro-chunkserver`
- `moosefs-pro-metalogger`

- `moosefs-pro-client`

etc.

In MooseFS 2.0 Community Edition (CE) the packages are named according to the following pattern:

- `moosefs-ce-master`
- `moosefs-ce-cli`
- `moosefs-ce-chunkserver`
- `moosefs-ce-metalogger`
- `moosefs-ce-client`

etc.

Chapter 3

MooseFS installation process on dedicated machines

In this tutorial it is assumed, that you have MooseFS 2.0 Pro version. If you use Community Edition, please type '`ce`' instead of '`pro`' in package names.

In this tutorial it is also assumed, that you have Ubuntu/Debian installed on your machines. If you have another distribution, please use appropriate package manager instead of `apt`.

3.1 Master server(s) installation

Warning: Configuration files on all Master Servers must be consistent!

In MooseFS 2.0 master server (and also other modules) installation can be accomplished by running the command listed below:

Listing 3.1: Installing master server

```
# apt-get install moosefs-pro-master
```

Sample configuration files will be created in `/etc/mfs` with the extension *.dist. You'll use these files as your target configuration files:

Listing 3.2: Copying default config files as target configuration files

```
# cd /etc/mfs
# cp mfsmaster.cfg.dist mfsmaster.cfg
# cp mfsexports.cfg.dist mfsexports.cfg
```

If you would like to change any of the settings you should uncomment the appropriate line and set a different value. For the lines which are commented the system will use built-in default values, i.e. those listed in commented lines.

File `mfsmaster.cfg` contains master server settings. You can find out more information about this file in the man pages (`man mfsmaster.cfg`).

File `mfsexports.cfg` specifies which users' computers can mount the file system and with what privileges. For example, to specify that only machines addressed as `192.168.2.x` can use the whole structure of MooseFS resources (`/`) in read/write mode, in the first line which is not commented out change the asterisk (`*`) to `192.168.2.0/24`, so that you'll have:

Listing 3.3: Changes to mfsexports.cfg

```
192.168.2.0/24       /       rw,alldirs,maproot=0
```

If you are setting up Pro version, you should now place proper `mfslicence.bin` file into `/etc/mfs` directory:

Listing 3.4: Instaling mfslicence.bin file

```
# cp /path/to/mfslicence.bin /etc/mfs
```

The `mfslicence.bin` file must be installed on all master servers.
`mfslicence.bin` file is not necessary on CE master servers.

After configuring `mfsmaster` it is recommended to configure `mfsmaster` autostart. It can be done in appropriate configuration file: `/etc/default/moosefs-pro-master` by setting `MFSMASTER_ENABLE` variable to `true`.

Listing 3.5: Configuring mfsmaster autostart

```
MFSMASTER_ENABLE=true
```

At this point it is possible to run the master server (using the standard way to run services):

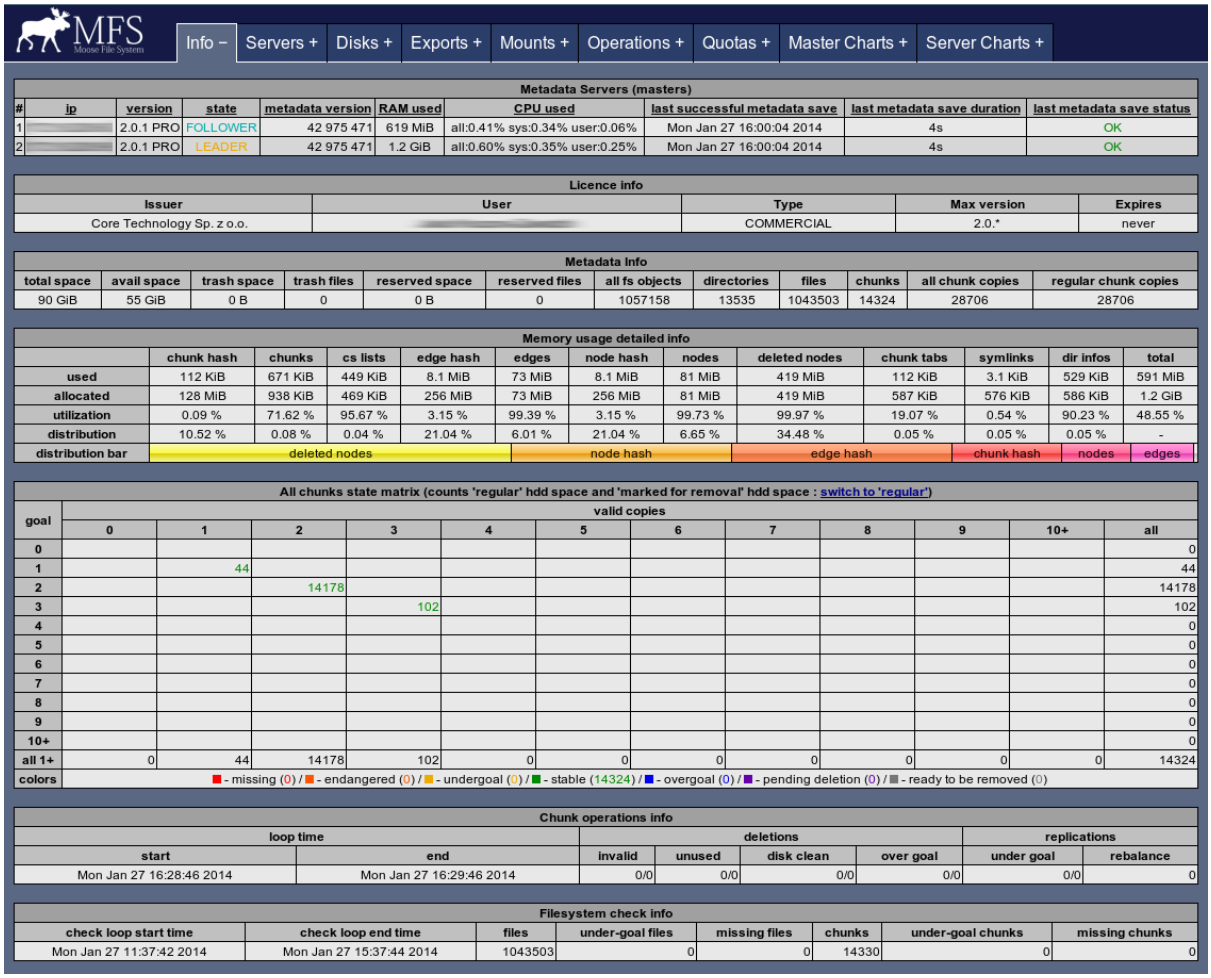
Listing 3.6: Running mfsmaster

```
# service moosefs-pro-master start
```

To install second (third, etc.) master server just repeat steps listed above on another machine.

3.2 MooseFS CGI and CGI server installation

MooseFS CGI monitor interface is used to let user observe and analyze current MooseFS status (as you can see on the screenshots presented below):



MFS Moose File System — Info − | Servers + | Disks + | Exports + | Mounts + | Operations + | Quotas + | Master Charts + | Server Charts +

Metadata Servers (masters)

#	ip	version	state	metadata version	RAM used	CPU used	last successful metadata save	last metadata save duration	last metadata save status
1	[redacted]	2.0.1 PRO	FOLLOWER	42 975 471	619 MiB	all:0.41% sys:0.34% user:0.06%	Mon Jan 27 16:00:04 2014	4s	OK
2	[redacted]	2.0.1 PRO	LEADER	42 975 471	1.2 GiB	all:0.60% sys:0.35% user:0.25%	Mon Jan 27 16:00:04 2014	4s	OK

Licence info

Issuer	User	Type	Max version	Expires
Core Technology Sp. z o.o.	[redacted]	COMMERCIAL	2.0.*	never

Metadata Info

total space	avail space	trash space	trash files	reserved space	reserved files	all fs objects	directories	files	chunks	all chunk copies	regular chunk copies
90 GiB	55 GiB	0 B	0	0 B	0	1057158	13535	1043503	14324	28706	28706

Memory usage detailed info

	chunk hash	chunks	cs lists	edge hash	edges	node hash	nodes	deleted nodes	chunk tabs	symlinks	dir infos	total
used	112 KiB	671 KiB	449 KiB	8.1 MiB	73 MiB	8.1 MiB	81 MiB	419 MiB	112 KiB	3.1 KiB	529 KiB	591 MiB
allocated	128 MiB	938 KiB	469 KiB	256 MiB	73 MiB	256 MiB	81 MiB	419 MiB	587 KiB	576 KiB	586 KiB	1.2 GiB
utilization	0.09 %	71.62 %	95.67 %	3.15 %	99.39 %	3.15 %	99.73 %	99.97 %	19.07 %	0.54 %	90.23 %	48.55 %
distribution	10.52 %	0.08 %	0.04 %	21.04 %	6.01 %	21.04 %	6.65 %	34.48 %	0.05 %	0.05 %	0.05 %	-
distribution bar	deleted nodes			node hash		edge hash			chunk hash	nodes	edges	

All chunks state matrix (counts 'regular' hdd space and 'marked for removal' hdd space : switch to 'regular')

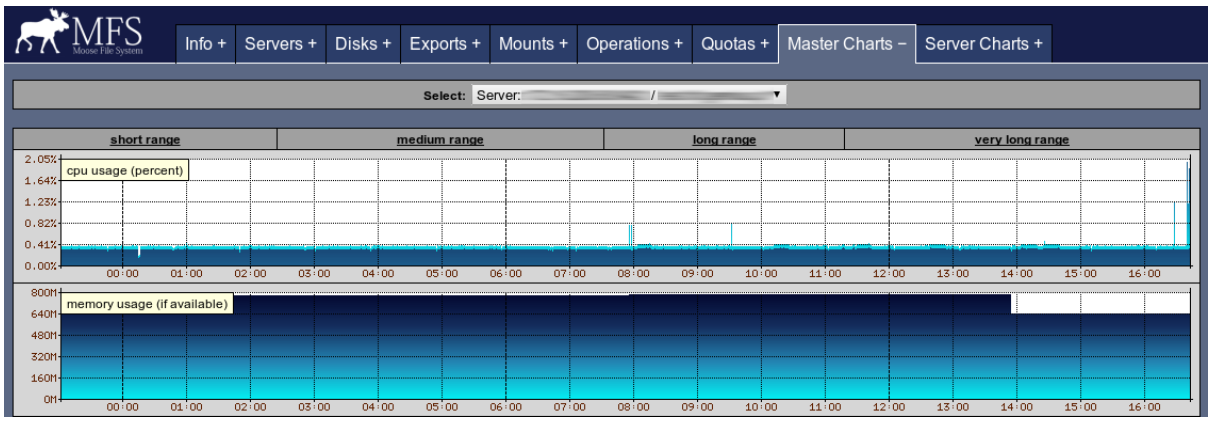
goal	valid copies 0	1	2	3	4	5	6	7	8	9	10+	all
0												0
1		44										44
2			14178									14178
3				102								102
4												0
5												0
6												0
7												0
8												0
9												0
10+												0
all 1+	0	44	14178	102	0	0	0	0	0	0	0	14324
colors	■ - missing (0) / ■ - endangered (0) / ■ - undergoal (0) / ■ - stable (14324) / ■ - overgoal (0) / ■ - pending deletion (0) / ■ - ready to be removed (0)											

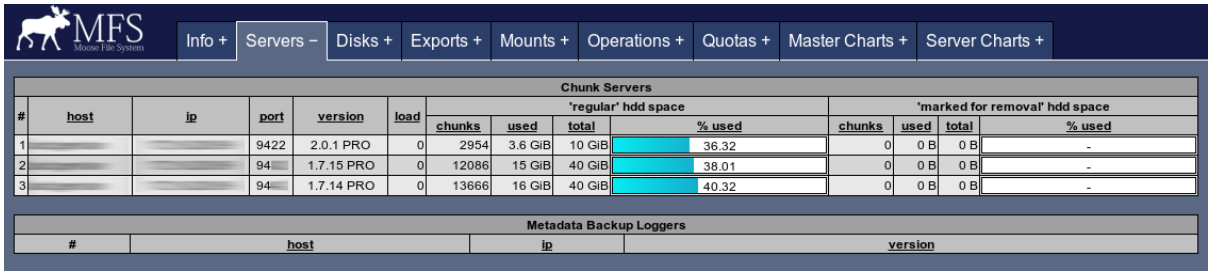
Chunk operations info

loop time		deletions				replications	
start	end	invalid	unused	disk clean	over goal	under goal	rebalance
Mon Jan 27 16:28:46 2014	Mon Jan 27 16:29:46 2014	0/0	0/0	0/0	0/0	0/0	0

Filesystem check info

check loop start time	check loop end time	files	under-goal files	missing files	chunks	under-goal chunks	missing chunks
Mon Jan 27 11:37:42 2014	Mon Jan 27 15:37:44 2014	1043503	0	0	14330	0	0





We recommend installing MooseFS CGI Monitor on first master server.

Listing 3.7: MooseFS CGI and CGI server installation

```
# apt-get install moosefs-pro-cgiserv
# apt-get install moosefs-pro-cgi
```

Now you shoud set autostart of `mfscgiserv` in file `/etc/default/moosefs-pro-cgiserv` by setting `MFSCGISERV_ENABLE` variable to `true`.

Listing 3.8: Configuring mfscgiserv autostart

```
MFSCGISERV_ENABLE=true
```

You can now run CGI Monitor Server:

Listing 3.9: Running CGI monitor

```
# service moosefs-pro-cgiserv start
```

Information should now be available under `http://192.168.1.1:9425/` (for the moment there will be no data about chunk servers).

3.3 MooseFS CLI installation

MooseFS Command Line Interface (CLI) tool allows you to see various information about MooseFS' status. This tool has many options that basically allow you to check all the information that you would otherwise see in your CGI (e.g. if you encounter any problems with it). You can list all the options by invoking the tool with `-h` or `--help` switch:

Listing 3.10: `mfscli --help`

```
# mfscli --help
usage:
        /usr/bin/mfscli [-hpn28] [-H master_host] [-P master_port] [-f 0..3] -
            S(IN|LI|IG|MU|IC|IL|CS|ML|HD|EX|MS|MO|QU) [-o order_id [-r]] [-m
            mode_id]
        /usr/bin/mfscli [-hpn28] [-H master_host] [-P master_port] [-f 0..3] -
            C(RC/ip/port|BW/ip/port)

common:

        -h : print this message
        -p : force plain text format on tty devices
        -s separator : field separator to use in plain text format on tty
            devices (forces -p)
        -2 : force 256-color terminal color codes
        -8 : force 8-color terminal color codes
        -H master_host : master address (default: mfsmaster)
```

```
        -P master_port : master client port (default: 9421)
        -n : do not resolve ip adresses (default when output device is not tty
            )
        -f frame charset number : set frame charset to be displayed as table
            frames in ttymode
                -f0 : use simple ascii frames '+','-','|' (default for non utf
                    -8 encodings)
                -f1 : use utf-8 frames:

                -f2 : use utf-8 frames:

                -f3 : use utf-8 frames:                                    (
                    default for utf-8 encodings)

monitoring:

        -S data set : defines data set to be displayed
                -SIN : show full master info
                -SIM : show only masters states
                -SIG : show only general master (leader) info
                -SLI : show only licence info
                -SIC : show only chunks info (goal/copies matrices)
                -SIL : show only loop info (with messages)
                -SCS : show connected chunk servers
                -SMB : show connected metadata backup servers
                -SHD : show hdd data
                -SEX : show exports
                -SMS : show active mounts
                -SMO : show operation counters
                -SQU : show quota info
        -o order_id : sort data by column specified by 'order id' (depends on
            data set)
        -r : reverse order
        -m mode_id : show data specified by 'mode id' (depends on data set)

commands:

        -C command : perform particular command
                -CRC/ip/port : remove given chunkserver from list of active
                    chunkservers
                -CBW/ip/port : send given chunkserver back to work (from grace
                     state)
                -CRS/sessionid : remove given session
```

The `mfscli` with `-SIN` option will display basic info similar to the "Info" tab in CGI.

To install `mfscli` just run the following command:

Listing 3.11: MooseFS CLI installation

```
# apt-get install moosefs-pro-cli
```

3.4 Backup servers (metaloggers) installation

In Pro version, when there are at least two master servers present metalogger is an optional tool, because when leader master fails, another one takes over its work.

In CE version we strongly recommend setting up at least one metalogger.

It is recommended, that the machine used to install the backup server is as strong as the master server (at least in regards to the amount of RAM). In case of the master server failure, after

importing changelogs to the metadata file, the metalogger server can be easily set up to take over functions of the managing server (more about this can be read at `http://moosefs.org`).

You need to issue the following commands to install and configure a MooseFS metalogger with default settings:

Listing 3.12: Installing and configuring metalogger

```
# apt-get install moosefs-pro-metalogger

# cd /etc/mfs
# cp mfsmetalogger.cfg.dist mfsmetalogger.cfg
```

For our test installation you'll leave mfsmetalogger.cfg unchanged. You can find out more information about this file in the man pages (`man mfsmetalogger.cfg`). In case you have changed the default name `mfsmaster` to another one, you need to uncomment and change the `MASTER_HOST` variable in `mfsmetalogger.cfg` file.

Now you are ready to start the backup server process:

Listing 3.13: Starting mfsmetalogger

```
# service moosefs-pro-metalogger start
```

Now you should set up automatic start of mfsmetalogger by changing `MFSMETALOGGER_ENABLE` variable to `true` in `/etc/default/moosefs-pro-metalogger`.

Listing 3.14: Configuring mfsmetalogger autostart

```
MFSMETALOGGER_ENABLE=true
```

To install second (third, etc.) metalogger just repeat steps listed above on another machine.

3.5 Chunk servers installation

Issue the following commands on the machines which are to be chunks servers:

Listing 3.15: Installing chunk server

```
# apt-get install moosefs-pro-chunkserver
```

Now prepare configuration files of the chunk servers:

Listing 3.16: Preparing configuration files

```
# cd /etc/mfs
# cp mfschunkserver.cfg.dist mfschunkserver.cfg
# cp mfshdd.cfg.dist mfshdd.cfg
```

For our test installation you'll leave mfschunkserver.cfg unchanged. You can find out more information about this file in the man pages (`man mfschunkserver.cfg`). In case you have changed the default name `mfsmaster` to another one, you also need to uncomment and change the `MASTER_HOST` variable in `mfschunkserver.cfg` file.

In the `mfshdd.cfg` file you'll give locations in which you have mounted hard drives/partitions designated for the chunks of the system. It is recommended that they are used exclusively for the MooseFS – this is necessary to manage the free space properly. For example, if you'll use `/mnt/mfschunks1` and `/mnt/mfschunks2` locations, add these two lines to `mfshdd.cfg` file:

Listing 3.17: Contents of mfshdd.cfg file

```
/mnt/mfschunks1
/mnt/mfschunks2
```

Before you start the chunk server, make sure that the user `mfs` has rights to write in the mounted partitions:

Listing 3.18: Changing ownership

```
# chown -R mfs:mfs /mnt/mfschunks1
# chown -R mfs:mfs /mnt/mfschunks2
```

At this moment the auto start of chunk server should be enabled. It can be done similarly to previous MooseFS' components (by editing `/etc/default/moosefs-pro-chunkserver` and setting `MFSCHUNKSERVER_ENBLE` variable to `true`).

Listing 3.19: Configuring autostart of mfschunkserver

```
MFSCHUNKSERVER_ENBLE=true
```

Now you are ready to start the chunk server:

Listing 3.20: Starting mfschunkserver

```
# service moosefs-pro-chunkserver start
```

Repeat the same steps for each chunk server you want to use for storing data in MooseFS system.

Now at `http://192.168.1.1:9425` full information about the system is available, including the master server and chunk servers.

3.6 Users' computers installation

In order to mount a file system based on MooseFS, it is necessary that users' computers have FUSE package (at least in version 2.6, recommended >=2.7.2). If it is not present, you have to install it. One of the options is to compile it from sources, or you can install it from repositories on Debian-based systems with following command:

Listing 3.21: FUSE installing

```
# apt-get install fuse
```

`mfsmount` can be installed in the same way as other MooseFS components:

Listing 3.22: Installing mfsmount

```
# apt-get install moosefs-pro-client
```

Let's assume that you'll mount the system in a `/mnt/mfs` folder on a client's machine. Issue the following commands:

Listing 3.23: Mounting the Moose File System

```
# mkdir -p /mnt/mfs
# mfsmount /mnt/mfs -H mfsmaster
```

Now after issuing the `df -h | grep mfs` command you should get information similar to this:

Listing 3.24: Result of `df -h | grep mfs`

```
/storage/mfschunks/mfschunks1
    2.0G    69M     1.9G    4%  /mnt/mfschunks1
/storage/mfschunks/mfschunks2
    2.0G    69M     1.9G    4%  /mnt/mfschunks2
mfs#mfsmaster:9421
    3.2G    0       3.2G    0%  /mnt/mfs
```

Chapter 4

Basic MooseFS use

To create `folder1` in `/mnt/mfs`, in which you will store files in one copy (setting `goal=1`), issue the following command:

Listing 4.1: Making directory #1

```
mkdir -p /mnt/mfs/folder1
```

To create `folder2`, in which you will store files in two copies (setting `goal=2`), issue the following command:

Listing 4.2: Making directory #2

```
mkdir -p /mnt/mfs/folder2
```

The number of copies for the folder is set with the `mfssetgoal -r` command:

Listing 4.3: `mfssetgoal -r command`

```
# mfssetgoal -r 1 /mnt/mfs/folder1
/mnt/mfs/folder1:
inodes with goal changed:                1
inodes with goal not changed:            1
inodes with permission denied:           0

# mfssetgoal -r 2 /mnt/mfs/folder2
/mnt/mfs/folder2:
inodes with goal changed:                1
inodes with goal not changed:            0
inodes with permission denied:           0
```

Create and copy a file to both folders:

Listing 4.4: Creating and copying a file to newly created folders

```
echo "test" > testmfs
cp testmfs /mnt/mfs/folder1
cp testmfs /mnt/mfs/folder2
```

To check in how many copies a file is stored, use the `mfscheckfile` command. In `folder1` you have one copy stored in one chunk:

Listing 4.5: Checking amount of copies

```
# mfscheckfile /mnt/mfs/folder1/testmfs
/mnt/mfs/folder1/testmfs:
chunks with 1 copy:              1
```

And in the `folder2` the file `testmfs` is stored in two copies:

Listing 4.6: Checking amount of copies

```
# mfscheckfile /mnt/mfs/folder2/testmfs
/mnt/mfs/folder2/testmfs:
chunks with 2 copies:            1
```

Notice, that if you set a goal for a file higher than the total number of working chunk servers, this file will be saved in only as many copies as there are chunk servers. This is because one chunk server will store no more than one copy of any chunk/file.

You can find more information about MooseFS usage and commands on this page:
`http://www.moosefs.org`

It is also recommended to read the FAQ page:
`http://www.moosefs.org/moosefs-faq.html`

Chapter 5

Stopping MooseFS

In order to safely stop the MooseFS cluster you have to perform the following steps:

- Unmount the file system on all machines using umount command (in our examples it would be: `umount /mnt/mfs`)
- Stop the chunk server processes: `service moosefs-pro-chunkserver stop`
- Stop the master server process(es): `service moosefs-pro-master stop`
- Stop the metalogger process(es) (if any): `service moosefs-pro-metalogger stop`

Chapter 6

Supplement: Setting up DNS server on Debian/Ubuntu

In this extra chapter you'll use bind9 as your DNS server.
Notice: You can find out more about DNS server e.g. on these pages:

- `https://help.ubuntu.com/community/BIND9ServerHowto`
- `http://ubuntuforums.org/showthread.php?t=236093`

6.1 Setting up DNS server

1. The very first thing to do is installing bind9 and DNS utils. You can do this by running the following command:

   Listing 6.1: installing bind9

   ```
   # sudo apt-get install bind9 dnsutils
   ```

   Main configuration files are placed in /etc/bind/ directory.

2. The second thing you have to do is edit in your favorite editor (e.g. `nano` or `vim`) file named "`named.conf.local`". You need to add there your new zone, e.g.:

   Listing 6.2: New zone in named.conf.local

   ```
   zone "mfsnetwork.lan" {
           type master;
           file "/etc/bind/mfsnetwork.lan";
   };
   ```

   In this file you can decide whether it is master or slave server and select path to zone's config file.

3. After that create the file you've pointed to in the zone configuration (user `bind` must have permissions to read it) and paste there the following code:

Listing 6.3: mfsnetwork.lan configuration file

```
$TTL 3600
$ORIGIN mfsnetwork.lan.
@   IN  SOA dns.mfsnetwork.lan. root.mfsnetwork.lan. (
                2014070901 ; serial numer YYYMMDDSS
                10800 ; refresh
                3600 ; retry
                604800 ; expire
                10800 ; negative TTL
)

@               IN  NS  dns.mfsnetwork.lan.
@               IN  A   192.168.0.1 ; address of bind9
dns             IN  A   192.168.0.1 ; address of bind9

mfsmaster       IN  A   192.168.1.1 ; address of mfsmaster01
mfsmaster       IN  A   192.168.1.2 ; address of mfsmaster02

mfsmaster01     IN  A   192.168.1.1 ; address of mfsmaster01
mfsmaster02     IN  A   192.168.1.2 ; address of mfsmaster02

chunkserver01   IN  A   192.168.1.101 ; address of chunkserver01
chunkserver02   IN  A   192.168.1.102 ; address of chunkserver02
chunkserver03   IN  A   192.168.1.103 ; address of chunkserver03
```

4. Next thing to do is to edit file `/etc/bind/named.conf.options`. You should use here your ISP's DNS servers, or you can use OpenDNS servers – IP addresses are presented below:

Listing 6.4: `named.conf.options` configuration file

```
forwarders {
        208.67.222.222;
        208.67.220.220;
};
```

5. Last thing to do is restarting bind9 DNS server (to let it load new configuration):

Listing 6.5: Restarting bind9

```
# service bind9 restart
```

6.2 Setting up revDNS server

Reverse DNS server is used by MooseFS and all network services in general to translate IP addresses to human-readable form (e.g. `192.168.1.1` to `mfsmaster01`). Ater installing and properly configuring DNS server you need to do 3 more things to have revDNS set up:

- In `/etc/bind` directory create an empty file named `rev.168.192.in-addr.arpa` and paste into it the following code:

Listing 6.6: Content of `rev.168.192.in-addr.arpa` file

```
@   IN      SOA     dns.coretech.lan. root.coretech.lan. (
                    2014070901
                    28800
                    604800
                    604800
                    86400
)
```

```
168.192.in-addr.arpa.   IN  NS  dns.mfsnetwork.lan.

1.1                     IN  PTR mfsmaster01.mfsnetwork.lan.
2.1                     IN  PTR mfsmaster02.mfsnetwork.lan.

101.1                   IN  PTR chunkserver1.mfsnetwork.lan.
102.1                   IN  PTR chunkserver2.mfsnetwork.lan.
103.1                   IN  PTR chunkserver3.mfsnetwork.lan.
```

- Add the following code to `/etc/bind/named.conf.local` file:

  Listing 6.7: Extra code to add to `/etc/bind/named.conf.local` file

```
zone "168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/rev.168.192.in-addr.arpa";
};
```

- Run `service bind9 restart` command:

  Listing 6.8: Running `service bind9 restart` command

```
# service bind9 restart
```