# The Guide to VoyageMPD, BruteFIR and JACK

lich000king [at] yahoo [dot] de

April 26, 2012

## Contents

# 1 Introduction

VoyageMPD is a small Linux distribution with a realtime Kernel developed for audio applications. It has ALSA already installed and is well suited to install on flash drives. As a Debian Linux it allows to install software via apt-get and is therefore easily customizable. An X-server is not installed by default. This guide aims at describing the installation and configuration of a convolution engine that can run with next-to-no user interaction and that will also work as MPD server. It is intended for Linux newbies, but it may also contain valuable information for more advanced users. At the moment, it describes the steps to set up BruteFIR, JACK and MPD.

# 2 VoyageMPD setup

## 2.1 Installation

Guides how to do this can be found in [1] or [2]. However, here are step-by step instructions how it can be installed on an internal hard drive or on a USB stick.

### 2.1.1 Installation on a hard disk

- Download the Live CD here[3] and burn it to a CD.
- Boot from it.
- Log in. The username is "root", the password is "voyage".
- Type the following Linux commands:

```
mkdir /tmp/root
mount -o loop /live/image/live/filesystem.squashfs  /tmp/root
cd /tmp/root
mkdir /tmp/cf
```

Now we want to format the target drive. First we have to find out which drive it is. Type

```
mount
```

to see what is mounted. Internal hard drive are often /etc/sda, possibly /etc/hda. It es helpful to already have something on the drive. Then you can use the ls command to verify you are about to format the right drive.

The following command will format (and thus completely erase) the drive (assuming it is sda) and create an ext2 partition /dev/sda1:

```
/usr/local/sbin/format-cf.sh /dev/sda
```

Now we are going to start the installation script:

```
/usr/local/sbin/voyage.update
```

Continue with section 2.1.3!

### 2.1.2 Installation on a USB stick

This does not seem to work from voyagelinux. However, it should work with most other Linux distributions. So if you already have one installed somewhere, you should try it. Otherwise you can just use a live CD (Ubuntu, ...).

Make sure you have something you recognize on the stick. Plug it in. Check where it is mounted:

```
mount
```

Will list all mounted drives. It might say something like "/dev/sdb is mounted at /media/mountlocation". Check if this is your USB stick by doing

```
ls /media/mountlocation
```

Download the latest version of voyage-mpd.tar.bz2 here[3]. Unpack it:

```
sudo tar --numeric-owner -jxf /downloadlocation/voyage-mpd[...].tar.bz2
```

It should be extracted to your home folder.

If are certain which device is your stick and where it is mounted, you can now unmount it:

```
sudo umount /media/mountlocation
```

Use the helper script to format the stick (replace /dev/sda with your stick):

```
sudo /home/username/voyage-mpd[...]/./usr/local/sbin/format-cf.sh /dev/sda
```

Create a directory used later for the installation:

```
sudo mkdir /mnt/cf
```

Then you can start the installation script by doing:

```
sudo /home/username/voyage-mpd[...]/./usr/local/sbin/voyage.update
```

### 2.1.3 The installation script

The following input should do:

```
1
/tmp/root

2
6

3
/dev/sda [change this if necessary]
1 [we choose the first partition]
/tmp/cf [mount to the directory we created earlier]

4
grub
1

5
2

6
1 [use 2 when installing to a USB stick]

7

8
```

Now VoyageMPD is hopefully correctly installed and ready to go. Type:

```
reboot
```

## 2.2 Basic configuration

After installing VoyageMPD, we are going to change add and change a few things to make our life easier.

Log in as earlier. Type

```
remountrw
```

This will remount the file system as read-write. Now we do not want to login after every reboot, so we are going to configure the system to log in automatically. Starting from here, it is incredibly helpful to have the machine connected to the Internet. If the Internet connection is working, do the following.

```
apt-get update
```

This updates the apt-get database and we can now use apt-get to easily install software. Start with

```
apt-get install mingetty
```

### 2.2.1 Text editor

We can install a text editor as follows:

```
apt-get install nano
```

Here are the most important nano commands:

- Ctrl+O (saves the document at the location of your choice)
- Ctrl+X (quits nano)
- Ctrl+W (search)

Now we are going to use nano to edit the inittab file:

```
nano /etc/inittab
```

we replace the line

```
1:2345:respawn:/sbin/getty 38400 tty1
```

by

```
1:2345:respawn:/sbin/mingetty --autologin root  --noclear tty1
```

Then we reboot. A login should no longer be necessary.

### 2.2.2 Read-write file system

Actually, we do not want to type "remountrw" every time we restart either. So we do:

```
remountrw
```

(one last time) followed by:

```
nano /etc/init.d/voyage-util
```

Search for (Ctrl+W) "read-only", comment out the echo line and the line "/usr/local/sbin/remountro by adding a # at the beginning of these lines (which makes Voyage ignore these lines). Save (Ctrl+O) and close (Ctrl+X). Reboot again and the file system should be read-write. You can test this by creating a directory:

```
mkdir /root/testdirectoy
```

Check, if your directory has been created:

```
ls /root/
```

Remove the directory:

```
rmdir /root/testdirectory
```

### 2.2.3 Boot timeout

The GRUB boot loader has a timeout which we can set to zero:

```
nano /boot/grub/menu.lst
```

### 2.2.4 Killall command

We also want to have the incredibly useful "killall command" at our disposal, so we do:

```
apt-get install psmisc
```

We will use killall later to end processes. Now our VoyageMPD is ready to use.

# 3 BruteFIR installation

Of course we could now go ahead and apt-get BruteFIR. But we want the most up-to-date version , so we are going to build it from source, along with fftw. In order to do this, we first have to get some stuff:

```
apt-get install build-essential flex libasound2-dev libjack-dev libfftw3-dev
update-rc.d -f mpd remove
```

Now we have to download the source code for BruteFIR[4] and fftw[5]. I will assume the sources have been unpacked in the directories "brutefir-1.0k" and "fftw-3.2.2" respectively in the root directory of a USB stick that is mounted in /media/usb0 (which is where a stick is likely to get mounted automatically when plugged in). If you want to use a Windows machine to download stuff in order to use it in VoyageMPD later, you should use a USB stick formatted to e.g. fat32, Windows does not read ext file systems.

Now we are going to create directories where we will build the software later:

```
mkdir /root/apps
mkdir /root/apps/brutefir
mkdir /root/apps/fftw_single
mkdir /root/apps/fftw_double
```

We create two separate directories for fftw, because we will install it in both single and double precision. Now we can copy from the USB stick to the newly created directories:

```
cp -r /media/usb0/brutefir-1.0k/* /root/apps/brutefir
cp -r /media/usb0/fftw-3.2.2/* /root/apps/fftw_single
cp -r /media/usb0/fftw-3.2.2/* /root/apps/fftw_double
```

The -r switch ensures all directories and subdirectories are also copied. The "" is needed so only the content of e.g. the brutefir-1.0k directory is copied and no new directory of this name is created (try what happens without it!). Now we go to the first directory:

```
cd /root/apps/fftw_single
```

And configure fftw to prepare it for compilation in single precision:

```
./configure --enable-float
```

After some time this has hopefully finished without error and we can compile (this will take some time):

```
make
```

and install

```
make install
```

Finally, we clean up:

```
make clean
```

If this was successful, we can repeat the same steps in the other directory (of course we omit the "–enable-float" switch this time):

```
cd /root/apps/fftw_double
./configure
make
make install
make clean
```

Now we are finally ready to build BruteFIR ("./configure" is not needed here):

```
cd /root/apps/brutefir
make
make install
make clean
```

BruteFIR is now installed and ready to use. You can do

```
brutefir
```

BruteFIR will show up quickly and complain about missing files.

# 4 BruteFIR configuration

After a successful setup, we can configure our machine as convolution engine that will take an input signal (from a CD player, say), convolve the signals with FIR filters (as created by acourate) and output the result to the sound card. We want the convolution to start automatically and we want the ability to switch between different configs with as little effort as possible. First we can create a few directories:

```
mkdir /root/brutefir
mkdir /root/brutefir/start
mkdir /root/brutefir/config
mkdir /root/brutefir/filter
```

You may want to create subdirectories in /filter for the different filter sets you may have. Then copy the filters to the /filter directory. The BruteFIR config files go to the /config directory. If you do not have any BruteFIR config files yet, you will have to create them, have a look at[6]. In any event you will have to change the filter paths in your config files to point to the filters in the newly created directories. You can set "convolver_config" to /root/brutefir/convolver_config. Make sure you have lock_memory set to "false". You can also put your sound card config files into the /config/ directory. These scripts contain the sound card settings (sample rate, volume for each channel etc.) If you have used an RME sound card in spblinux, your file is likely called hdsp_spdif. The /start/ directory contains the start scripts that are used to run the sound card config scripts and start BruteFIR with appropriate parameters. You can later execute these start scripts simply by typing their name in the terminal. If you have a sound card config script called hdsp_spdif and want to create a start script called "go" that starts BruteFIR with a config file called "cd" you can do the following: Create the file by

```
nano /root/brutefir/start/go
```

and enter the following:

```
mpd /root/mpd/mpd_cd.conf --kill
killall jackd
killall brutefir

/root/brutefir/config/hdsp_spdif >/dev/null
brutefir -nodefault /root/brutefir/config/cd
```

If you have several different configs, these will all look very similar. The first two commands are not needed yet. But they do not harm, either. The sound card config scripts can produce a lot of unwanted text output. It is suppressed by means of the >/dev/null.

The script needs to be made executable as follows:

```
chmod 775 /root/brutefir/start/go
```

Now we are going to extend the PATH environment variable so the "go" command will be recognized. Do

```
nano /etc/profile
```

You will see a list of directories, separated by colons. Add ":/root/brutefir/start" to it. After a reboot, you can type "go" in the terminal and see what happens. If everything is configured correctly, BruteFIR will start convolving and the message "audio processing starts now" will be displayed. BruteFIR can be stopped by pressing Ctrl+C.

Now there is one thing left to do: we want the convolution to start automatically. This can be achieved by adding a startup script to rc.local:

```
nano /etc/rc.local
```

A script that will start the same config as above should look like this:

```
/root/brutefir/config/hdsp_spdif >/dev/null
/usr/bin/brutefir -nodefault -daemon /root/brutefir/config/cd
```

If BruteFIR crashes after a few seconds with a sample rate error message, it may be helpful to add a "sleep 10" line between the two above commands. The -daemon option makes BruteFIR run as daemon. This means we can't stop it by pressing Ctrl+C. We can, however, stop it by typing

```
killall brutefir
```

If we just want to run BruteFIR with another config file, we do not need to issue the above command first, we can directly run the new start script. It already contains the killall command and will thus stop the old instance of BruteFIR before starting a new one. In order for the certain command we add to /etc/rc.local to work, we need to include their locations in the PATH:

```
nano /etc/init.d/rc.local
```

Look for the PATH line and add

```
/usr/local
/usr/local/bin
```

separated with colons.

# 5 Installation of JACK and MPD

Note: You can also use the newest version of MPD, as described in appendix A. Furthermore it is possible to avoid JACK alltogether by using MPD's pipe output to feed bruteFIR directly (possibly with some disadvantages), see appendix B

MPD is already installed. We do not want MPD to start automatically, so we do

```
update-rc.d -f mpd remove
```

JACK is installed simply through

```
apt-get install jackd1
```

(Pay attention to the spelling in "jackd1", it is important).

Now is a good time to reboot.

# 6 MPD configuration

Create some new directories:

```
mkdir /root/mpd
mkdir /root/mpd/playlists
mkdir /mnt/music
```

## 6.1 mpd.conf file

Now create a mpd config file with the following content

```
music_directory "/mnt/music"
playlist_directory "/root/mpd/playlists"
db_file "/root/mpd/database"
log_file "/root/mpd/mpd.log"
pid_file "/root/mpd/mpd.pid"
state_file "/root/mpd/state"
user "root"
bind_to_address "any"
port "6600"

input {
```

```
plugin "curl"
}

audio_output {
type "jack"
name "jack_cd"
format "44100:16:2"
destination_ports "brutefir:input-0,brutefir:input-1"
}
```

and save it as /root/mpd/mpd_cd.conf

### 6.1.1 Additional Settings

There are many ways in which you can customize your config file[7]. Here are a few hints: If you want to make sure mpd does not touch the mixer, you can add:

```
mixer_type "disabled"
```

Resource Limitations to prevent MPD from using too many resources:

```
connection_timeout "180"
max_playlist_length "30000"
```

tag types to be extracted during the audio file discovery process:

```
metadata_to_use "artist,album,title,track,name,genre,date,disc"
```

## 6.2 Server Mount

If you have your music on a server, you can mount it to /mnt/music. You can do this by adding the appropriate line to /etc/fstab. The following line mounts a windows share:

```
//[server IP address]/[servername]/[directory on server] /mnt/music
cifs noserverino,ro,iocharset=utf8,password= 0 0
```

where cifs is the mount type, noserverino is a windows specific option, ro=readonly, iocharset=utf8 is useful for special characters. Linux shares may look different.

Alternatively, you can also just copy an audio file to /mnt/music for testing.

### 6.3 Modified BruteFIR config file

Now you need to create a new config file to work with JACK. The input and output structures have to look as follows:

```
input "fleft", "fright" {
        device: "jack" { ports: "", ""; };
        sample: "AUTO";
channels: 2;

};

output "flhigh", "fllow", "frhigh", "frlow", "sub" {
        device: "jack" { ports: "system:playback_12", [...]; };
        sample: "AUTO";
channels: 5;
delay: 43,39,43,39,0;
dither: true;
};
```

Of course you have to change number of channels, ports, etc. according to your setup, i.e. replace [...] by additional channels as in the example. Save the new config file as /root/brutefir/config/mpd_cd

### 6.4 A first start

Now you should be ready to test MPD.

Start JACK:

```
jackd -dalsa -r44100 -dhw:0 -p256 &
```

Then start BruteFIR:

```
brutefir -nodefault -daemon /root/brutefir/config/mpd_cd
```

And finally create MPD's database and start the server:

```
mpd /root/mpd/mpd_cd.conf --create-db
```

Note that the –create-db option is only needed when starting MPD for the first time, or when the database was changed.

If all worked well so far, you should now be able to play music through MPD. You can do this by using the client of your choice, e.g. MPDroid for

Android or MPOD for iphone/ipad/ipodtouch. In the connection settings you have to enter the IP of the computer, where MPD is running and the port 6600 which we have configured in the mpd.conf file.

### 6.5 Startup script

We can change the startup script to run MPD. We can delete (or comment out with #) what we hat in /etc/rc.local earlier and add the following:

```
/root/brutefir/config/hdsp_spdif_3 >/dev/null
sleep 1
jackd -dalsa -r44100 -dhw:0 -p64 &
sleep 5
ps -A |grep -m 1 jackd |awk '{print$1}' |xargs -i schedtool -F -p 5 {}
sleep 2
brutefir -nodefault -daemon /root/brutefir/config/mpd_cd
sleep 2
mpd /root/mpd/mpd_cd.conf
sleep 1
ps -A |grep -m 1 mpd |awk '{print$1}' |xargs -i schedtool -F -p 2 {}
```

For this to work, we need to apt-get schedtool, too. This command allows us to change the scheduling policy and -priority of a particular process, or to simply view it. You can do schedtool –help for details. If you are interested what the weird-looking line starting wich "ps" does exactly, you can try it out first with one command, then two, etc., starting from the left. the "|" pipes the output of the command to its left to the command to its right. Later you may be able two lower the "256" (always powers of two). The sound card will have to be set to sync internally with the sample rate of the source material to be played. It is also a good idea to have a script that starts MPD when the computer is already running. We can e.g. create a script /root/brutefir/start/run that contains the following lines:

```
mpd /root/mpd/mpd_cd.conf --kill
killall jackd
killall brutefir
```

followed by the same lines we have just added to rc.local. This "run" script will again have to be made executable:

```
chmod 775 /root/brutefir/start/run
```

When you are listening if you want to be sure that the audio file is not affected by resample you can type the following command in the terminal:

```
cat /proc/asound/card0/pcm0p/sub0/hw_params
```

Maybe you will have to replace "card0" with "card1" or "pcm1" or "pcm0p" or "pcm1p"...

## 6.6 Optimizations

Here are a few ideas how to further optimize your system.

### 6.6.1 Turning off

To have the music server turn off itself when you push the power button exactly as it does with the "halt" command you first need to install "acpid"

```
apt-get install acpid
```

Then you will have to create a new file:

```
nano /etc/acpi/events/power
```

and copy in it the following lines:

```
# /etc/acpi/events/power
#This is called when the user presses the power button
event=button/power(PWR.||PBTN)
action=/sbin/poweroff
```

save the file then you can test if it works by typing:

```
/etc/init.d/acpid restart
```

..and pushing the power button

### 6.6.2 Removing unwanted programs at startup

Type the following command and type y (yes) if asked:

```
Type the following command and type y (yes) if asked:
apt-get remove cron dnsmasq hostapd klogd lirc lighttpd nfs-common dnsmasq-base
libdb-4.7 libevent-1.4-2 libfam0 libftdi1 libgssglue1 libnfsidmap2 libportaudio2
librpcsecgss3 perl libterm-readline-perl-perl perl-modules setserial portmap
watchdog
```

## References


[1] Voyage Linux getting started guide http://linux.voyage.hk/content/getting-started-v07x

[2] Voyage Linux Live CD getting started guide http://linux.voyage.hk/content/getting-started-live-cd-v07x

[3] VoyageMPD homepage http://linux.voyage.hk/voyage-mpd

[4] BruteFIR source code http://www.ludd.luth.se/~torger/brutefir.html#download

[5] fftw source code http://www.fftw.org/download.html

[6] Configuring and running BruteFIR http://www.ludd.luth.se/~torger/brutefir.html#config

[7] mpd.conf manpage http://linuxreviews.org/man/mpd.conf/

[8] MPD bug tracker http://www.musicpd.org/mantis/view.php?id=2796#c5316


## A Installing the most recent MPD version from the repositories

The benefit of using the most recent version may be fix of some small playback problems, playback of surround files, but new problems might arise, too.

First, we need to install git:

```
apt-get install git
```

Since the new stuff will be installed where our mpd files are, we back them up:

```
cp -r /root/mpd/* /root/oldmpd/
```

then we uninstall mpd and remove the directory

```
apt-get remove mpd
remove -r /root/mpd
```

Now we can download the new sources:

```
git clone git://git.musicpd.org/master/mpd.git
```

they will be dowloaded to /root/mpd. If you want to update it later, you can do this by

```
git pull
```

Before compiling, we need a few more packages (this list may vary, depending on the filetypes you want to play): autoconf automake libglib2.0-dev libasound2-dev libmad0-dev libid3tag0-dev libvorbis-dev libflac-dev libfaad-dev libshout3-dev libavcodec-dev libaudiofile-dev like

```
apt-get install package1 package2 ...
```

Finally:

```
./autogen.sh --enable-pipe-output
```

will build the new MPD version (you only need the "–enable-pipe-output" option if you plan on using pipe output). At the end you check if the formats you need are checked along with jack. If this is the case, type

```
make >/dev/null
make install
make clean
```

then you can simply copy back your old mpd files

```
cp -r /root/oldmpd/* /root/mpd
```

In case you want to run mor than 2 channels, you have to include something like (6 channels):

```
source_ports "A,B,C,D,E,F"
```

# B Pipe

If you prefer not to use JACK, you can do this by using MPD's pipe output plugin, thus feeding bruteFIR through the standard input. At this time, I do not recommend this for two reasons. First, the end of the last track in the playlist is cut off (the period that is cut off corresponds to bruteFIR's delay, so presumably this is irrelevant for minphase filters). Second, there may be some quality issues[8]. Furthermore, JACK is working great as far as I can tell.

If you want to try it all the same, here is how: First you have to build and install MPD with pipe output enabled as described in appendix A. Then you need to change the audio_output section in your mpd.conf to look something like(make sure you don't have a line break in command"..."):

```
audio_output {
    type            "pipe"
    name            "mpd-brutefir"
    command            "/usr/bin/brutefir -nodefault
    /root/brutefir/config/mpd_pipe"
    samplerate_converter    "0"
    format            "44100:16:2"
}
```

In your bruteFIR config file, modify your input/output section similar to:

```
## INPUT, OUTPUT ##

input "fleft", "fright" {
        device: "file" {path:"/dev/stdin"; };
        sample: "S16_LE";
    channels: 2;

};

output "flhigh", "fllow", "frhigh", "frlow", "sub" {
        device: "alsa" {device: "hw:0";};
        sample: "S24_4LE";
    channels: 26/11,10,15,14,9;
    delay: 190,186,190,186,0;
    dither: true;
```

```
};
```

Finally you can create a start script, e.g. brutefir/start/pipe (which you will have to make executable as before):

```
#!/bin/bash

killall mpd
killall jackd
killall brutefir

/root/brutefir/config/hdsp_spdif_3 >/dev/null

mpd /root/mpd/mpd_pipe.conf
sleep 0.5
ps -A |grep -m 1 mpd |awk '{print$1}' |xargs -i schedtool -F -p 2 {}
```