pDLNA

Installation, Configuration and Debugging Guide

Stefan Heumader

version 0.53.0

August 15, 2012

Contents

Chapter 1

Introduction

This document gives detailed information and instructions regarding the requirements, the installation and the configuration of *pDLNA* in its version 0.53.0.

Chapter 8 gives an overview regarding debugging or reporting problems of *pDLNA* on different operating systems.

In the end of this document, chapter 9 contains a list of known issues, which affect this version of *pDLNA* and possible workarounds, to fix the issue temporarily.

If you have any questions at all please do not hesitate to contact me.

Chapter 2

Requirements

This chapter gives an overview regarding the supported Perl versions, necessary Perl modules and additional/optional third party software, which are required for specific functionalities of *pDLNA*.

2.1 Perl

Currently, *pDLNA* has been tested with the following Perl versions:

- 5.10

Your installed Perl version can be determined by executing:

```
pdlna@mediaserver:~$ perl -v
```

Additionally, for those without interest in their Perl version, *pDLNA* has been tested with the following Linux distrubutions:

- Debian GNU/Linux 6 (squeeze)
- CentOS 6

NOTE: *pDLNA* propably works with any other Perl version or Linux distrubution. Please feel free to contact me about your installation environment.

2.2 Perl modules

In the following table, column one gives an overview about the necessary Perl modules, which must be installed. While column two lists the Debian GNU/Linux 6 (and its variants) package names, column three lists the CentOS 6 package names of these modules.

As already mentioned you need to install these Perl modules. You are able to install these modules using *CPAN (Comprehensive Perl Archive Network)*[1] or even via the package management of your favourite Linux distribution.

[1] `http://cpan.perl.org/`

Perl module name	Debian 6 package name	CentOS 6 package name
Audio::FLAC::Header	libaudio-flac-header-perl	
Audio::Wav	libaudio-wav-perl	
Audio::WMA	libaudio-wma-perl	
Config		
Config::ApacheFormat	libconfig-apacheformat-perl	
Data::Dumper		
Date::Format		
Devel::Size	libdevel-size-perl	
Digest::MD5	libdigest-md5-perl	
Digest::SHA1	libdigest-sha1-perl	perl-Digest-SHA1.i686
Fcntl		
File::Basename		
File::Glob		
File::MimeInfo	libfile-mimeinfo-perl	
GD	libgd-gd2-perl	perl-GD.i686
Getopt::Long::Descriptive	libgetopt-long-descriptive-perl	
Image::Info	libimage-info-perl	perl-Image-Info.noarch
IO::Interface	libio-interface-perl	
IO::Select		
IO::Socket		
IO::Socket::INET		
IO::Socket::Multicast	libio-socket-multicast-perl	
LWP::UserAgent		
MP3::Info	libmp3-info-perl	
MP4::Info	libmp4-info-perl	
Net::Address::Ethernet		
Net::Interface		
Net::IP	libnet-ip-perl	perl-Net-IP.noarch
Net::Netmask	libnet-netmask-perl	
Movie::Info		
Ogg::Vorbis::Header	libogg-vorbis-header-perl	
POSIX		
Proc::ProcessTable	libproc-processtable-perl	
Socket		
Sys::Hostname		
Sys::Syslog	libsys-syslog-perl	
threads		
threads::shared		
URI::Split		
XML::Simple	libxml-simple-perl	perl-XML-Simple.noarch

Table 2.1: Necessary Perl modules (FIXME)

For example, installing the *XML::Simple* Perl module can be installed via *CPAN* by using the following command

```
pdlna@mediaserver:~$ sudo cpan
cpan[1]> install XML::Simple
```

or by executing the following command

```
pdlna@mediaserver:~$ sudo apt-get install libxml-simple-perl
```

on the Debian GNU/Linux distribution and its variants.

For Perl modules without a package provided by your Linux distribution, you need to install it via *CPAN*.

2.3 Third party software

pDLNA requires for specific functionalities third party software, which is open source software.

2.3.1 MPlayer

`MPlayer` (`http://www.mplayerhq.hu` is needed, when even one of the following conditions are configured:

- if `EnableVideoThumbnails` is enabled (see section 6.4.4 for detailed information)
- if `LowResourceMode` is disabled (see section 6.4.5 for detailed information)

The `MPlayer` source code or binaries can be obtained from the project's official website `http://www.mplayerhq.hu` or can be installed via the package management of your favorite Linux distribution. The following command will install the `MPlayer` package on the Debian GNU/Linux distribution and its variants.

```
pdlna@mediaserver:~$ sudo apt-get install mplayer
```

The official website of `MPlayer` does also provide binaries for your Windows operating system.

2.3.2 FFmpeg

For enabling transcoding[2] of video and audio files, `FFmpeg` (`http://ffmpeg.org`) is required.

[2] Transcoding is converting specific media items on the fly to a (by the DLNA aware device) supported media format.

If no Transcoding Profiles (see section 6.7 for detailed information) are configured, `FFmpeg` is not required. If `LowResourceMode` is enabled, configured Transcoding Profiles will be ignored.

The `FFmpeg`'s source code or binaries can be obtained from the project's official website `http://ffmpeg.org` or can be installed via the package management of your favorite Linux distribution. The following command will install the `FFmpeg` package on the Debian GNU/Linux distribution and its variants.

```
pdlna@mediaserver:~$ sudo apt-get install ffmpeg
```

The official website of `FFmpeg` does also provide information, where to get binaries for your Windows operating system.

Chapter 3

Installing on Linux

This chapter gives an overview about the different methods for installing *pDLNA* on your favorite Linux distribution. The first section gives an overview about installing the necessary prerequisites for various Linux distributions. The second section describes the installation steps via the official `git` repository, while the third section describes installing *pDLNA* from a packed tarball.

3.1 Install prerequisites

3.1.1 Debian GNU/Linux 6

On a fresh installed Debian GNU/Linux 6 operating system, you are able to execute the following command to install necessary Debian GNU/Linux packages and mostly all necessary Perl modules from the Debian repository:

```
pdlna@mediaserver:~$ sudo apt-get install \
 build-essential git \
 mplayer ffmpeg \
 libgetopt-long-descriptive-perl \
 libfile-copy-recursive-perl \
 libaudio-flac-header-perl libaudio-wav-perl \
 libaudio-wma-perl libconfig-apacheformat-perl \
 libdevel-size-perl libdigest-sha1-perl \
 libfile-mimeinfo-perl libgd-gd2-perl \
 libimage-info-perl libio-interface-perl \
 libio-socket-multicast-perl libmp3-info-perl \
 libmp4-info-perl libnet-ip-perl \
 libnet-netmask-perl libogg-vorbis-header-perl \
 libproc-processtable-perl
```

Afterwards you still need to install some more Perl modules via *CPAN* itself:

```
pdlna@mediaserver:~$ sudo cpan
cpan[1]> install Net::Address::Ethernet
cpan[2]> install Net::Interface
cpan[3]> install Movie::Info
```

3.1.2 CentOS 6

On a fresh CentOS 6 (mininal) installation, the following command installs necessary CentOS packages and some necessary Perl Modules from the official repository:

```
pdlna@mediaserver:~$ sudo yum install \
 git bind-utils sudo cpan make gcc \
 libogg-devel.i686 libogg.i686 libvorbis.i686 \
 libvorbis-devel.i686 vorbis-tools.i686 \
 perl-Digest-SHA1.i686 perl-GD.i686 \
 perl-Image-Info.noarch perl-Net-IP.noarch \
 perl-XML-Simple.noarch shared-mime-info.i686
```

IMPORTANT NOTE: In the default repositories of CentOS, there are no packages for `Mplayer` or `FFmpeg`, so *pDLNA* is able to run in `LowResourceMode` (see section 6.4.5 for detailed information).

Afterwards you still need to install a lot of Perl modules via *CPAN* itself, since they are not included in the official repository:

```
pdlna@mediaserver:~$ sudo cpan
cpan[1]> install YAML
cpan[2]> install Getopt::Long::Descriptive
cpan[3]> install Audio::FLAC::Header
cpan[4]> install Audio::Wav
cpan[5]> install Audio::WMA
cpan[6]> install Config::ApacheFormat
cpan[7]> install Date::Format
cpan[8]> install Devel::Size
cpan[9]> install File::MimeInfo
cpan[10]> install IO::Interface
cpan[11]> install IO::Socket::Multicast
cpan[12]> install MP3::Info
cpan[13]> install MP4::Info
cpan[14]> install Net::Address::Ethernet
cpan[15]> install Net::Interface
cpan[16]> install Net::Netmask
cpan[17]> install Movie::Info
cpan[18]> install Inline::MakeMaker
cpan[19]> install Ogg::Vorbis::Header
cpan[20]> install Proc::ProcessTable
```

3.2 Installing via git

Installing *pDLNA* via a `git clone` is a simple way to install *pDLNA* und keep it up to date. `git` is a distributed revision control system and is developed by Linus Torvalds. The official `git` repository is hosted on GitHub (`https://github.com/geuma/pDLNA/`), which is a web-based hosting service for software development projects.

3.2.1 Cloning the git repository

At first the `git` software must be installed on the computer, *pDLNA* should be run at. The `git` source code can be obtained from the project's official website `http://git-scm.com/` or can be installed via the package management of your favorite Linux distribution. The following command will install the `git` package on the Debian GNU/Linux distribution and its variants.

```
pdlna@mediaserver:~$ sudo apt-get install git
```

After installing `git` you need to clone the repository by executing the following command:

```
pdlna@mediaserver:~$ git clone git://github.com/geuma/pDLNA.git
```

In the end, a directory named *pDLNA* has been created, which should look like the following directory listing.

```
pdlna@mediaserver:~/pDLNA$ ls -lah
total 112K
drwxr-xr-x. 5 pdlna pdlna 4.0K Aug 15 00:26 .
drwx------. 3 pdlna pdlna 4.0K Aug 15 00:02 ..
-rwxrwxr-x. 1 pdlna pdlna  13K Aug 14 20:53 CHANGELOG
drwxrwxr-x. 2 pdlna pdlna 4.0K Aug 14 20:53 external_programs
drwxrwxr-x. 8 pdlna pdlna 4.0K Aug 14 20:53 .git
-rwxrwxr-x. 1 pdlna pdlna 1.4K Aug 14 20:53 INSTALL
-rwxrwxr-x. 1 pdlna pdlna 6.2K Aug 14 20:53 install.pl
-rwxrwxr-x. 1 pdlna pdlna  35K Aug 14 20:53 LICENSE
drwxrwxr-x. 2 pdlna pdlna 4.0K Aug 14 20:53 PDLNA
-rwxrwxr-x. 1 pdlna pdlna 7.3K Aug 14 20:53 pdlna.conf
-rwxrwxr-x. 1 pdlna pdlna 2.8K Aug 14 20:53 pDLNA.pl
-rwxrwxr-x. 1 pdlna pdlna 2.2K Aug 14 20:53 rc.pDLNA
-rwxrwxr-x. 1 pdlna pdlna  831 Aug 14 20:53 README
-rwxrwxr-x. 1 pdlna pdlna 1.3K Aug 14 20:53 TODO
-rwxrwxr-x. 1 pdlna pdlna   21 Aug 14 20:53 VERSION
```

3.2.2 Finalizing the installation

The easiest way to finalize the installation is to copy the default configuration file from the `git clone` to the */etc/* directory by executing the following commands. For copying the configuration file you might need superuser rights.

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ sudo cp pdlna.conf /etc/
```

Additionally you should copy the sample initscript from the `git clone` to the */etc/init.d/* directory and setting the execute bit by executing the following commands. This step might also require superuser rights.

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ sudo cp rc.pDLNA /etc/init.d/
pdlna@mediaserver:~/pDLNA$ sudo chmod +x /etc/init.d/rc.pDLNA
```

In the end you need to change the *DIR* variable in the initscript (line 20) by editing the file */etc/init.d/rc.pDLNA* with your favourite editor like *vim*, *nano* or whatever you like. The following snippet shows you an example command:

```
pdlna@mediaserver:~$ sudo vim /etc/init.d/rc.pDLNA
```

After opening the file, jump to line 20 and edit the path for the *DIR* variable to the path, where the *pDLNA.pl* is stored. If you have cloned the `git` repository to `/home/pdlna/pDLNA/` you need to set the variable to the following value:

```
DIR="/home/pdlna/pDLNA/"
```

Additionally you need to check for the dependencies (see chapter 2) you are able to run the `install.pl` script with the *(-c or –checkrequirements)* parameter, which is stored in the `git` repository.

```
pdlna@mediaserver:~/pDLNA$ perl install.pl -c
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use Devel::Size;
ok 9 - use Digest::MD5;
ok 10 - use Digest::SHA1;
ok 11 - use Fcntl;
ok 12 - use File::Basename;
ok 13 - use File::Glob;
ok 14 - use File::MimeInfo;
ok 15 - use GD;
ok 16 - use Getopt::Long::Descriptive;
ok 17 - use Image::Info;
ok 18 - use IO::Interface;
ok 19 - use IO::Select;
ok 20 - use IO::Socket;
ok 21 - use IO::Socket::INET;
ok 22 - use IO::Socket::Multicast;
ok 23 - use LWP::UserAgent;
ok 24 - use MP3::Info;
ok 25 - use MP4::Info;
ok 26 - use Net::Address::Ethernet;
ok 27 - use Net::Interface;
ok 28 - use Net::IP;
ok 29 - use Net::Netmask;
ok 30 - use Movie::Info;
ok 31 - use Ogg::Vorbis::Header;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use Socket;
ok 35 - use Sys::Hostname;
ok 36 - use Sys::Syslog;
ok 37 - use threads;
ok 38 - use threads::shared;
ok 39 - use URI::Split;
ok 40 - use XML::Simple;
1..40
```

The output above shows the executed `install.pl` script with its output. As long as all of these checks return *ok*, all necessary requirements are fullfilled.

IMPORTANT NOTE: The checking will not include checking if `MPlayer` and `FFmpeg` are installed. For detailed information regarding the usage of `MPlayer` see section 2.3.1 or the usage of `FFmpeg` see section 2.3.2.

After the initial configuration (see chapter 6), you are able to start *pDLNA* by executing the following command:

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

3.2.3 Updating pDLNA

Because of the `git clone` updating *pDLNA* is pretty easy if there have not been any changes to your `git clone`. By executing the following commands

```
pdlna@mediaserver:~$ cd ~/pDLNA/
pdlna@mediaserver:~/pDLNA$ git pull
```

pDLNA will be updated to the latest version, which is currently pushed to the git repository. After checking for requirements by executing `install.pl -c` and restarting *pDLNA* via

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA restart
```

the new version is going to be started.

3.3 Installing the latest release

Installing the latest release of *pDLNA* is the simpliest way to install *pDLNA*. Every release of *pDLNA* will be packaged as a tarball and published on `https://github.com/geuma/pDLNA/downloads`.

3.3.1 Downloading latest release

For downloading the latest release of *pDLNA* you should visit `https://github.com/geuma/pDLNA/downloads` to look for the latest release. The following snippet shows the commands to download the latest version using `wget`:

```
pdlna@mediaserver:~$ cd /tmp/
pdlna@mediaserver:/tmp$ wget \
  https://github.com/downloads/geuma/pDLNA/pDLNA-$pDLNAversion.tgz
```

3.3.2 Installation

After downloading the latest version of *pDLNA* to the `/tmp` directory, you should extract the tarball and change to the extracted directory by executing the following commands:

```
pdlna@mediaserver:/tmp$ tar xzf pDLNA-$pDLNAversion.tgz
pdlna@mediaserver:/tmp$ cd pDLNA
```

In this directory, there is a script called `install.pl`, which supports checking for requirements *(-c or –checkrequirements)* and installing the application *(-i or –install)*. Additionally the help function *(-h or –help)* prints out more detailed information.

So the first step to install *pDLNA* should be to run the installation script to check for the necessary requirements (see chapter 2) by executing the following command:

```
pdlna@mediaserver:/tmp/pDLNA$ perl install.pl -c
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use Devel::Size;
ok 9 - use Digest::MD5;
ok 10 - use Digest::SHA1;
ok 11 - use Fcntl;
ok 12 - use File::Basename;
ok 13 - use File::Glob;
ok 14 - use File::MimeInfo;
ok 15 - use GD;
ok 16 - use Getopt::Long::Descriptive;
ok 17 - use Image::Info;
ok 18 - use IO::Interface;
ok 19 - use IO::Select;
ok 20 - use IO::Socket;
ok 21 - use IO::Socket::INET;
ok 22 - use IO::Socket::Multicast;
ok 23 - use LWP::UserAgent;
ok 24 - use MP3::Info;
ok 25 - use MP4::Info;
ok 26 - use Net::Address::Ethernet;
ok 27 - use Net::Interface;
ok 28 - use Net::IP;
ok 29 - use Net::Netmask;
ok 30 - use Movie::Info;
ok 31 - use Ogg::Vorbis::Header;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use Socket;
ok 35 - use Sys::Hostname;
ok 36 - use Sys::Syslog;
ok 37 - use threads;
ok 38 - use threads::shared;
ok 39 - use URI::Split;
ok 40 - use XML::Simple;
1..40
```

The attached output, shows the different tests and their results, which were performed by the installation script. For a complete overview regarding the requirements please see chapter 2.

IMPORTANT NOTE: The checking will not include checking if `MPlayer` and `FFmpeg` are installed. For detailed information regarding the usage of `MPlayer` see section 2.3.1 or the usage of `FFmpeg` see section 2.3.2.

When all requirements have been fixed and installed, you are able to rerun the installation script by executing the installation script with the *–install* parameter. By default, the script will install *pDLNA* to the `/opt` directory, which can be changed by setting the *–prefix=/path/to/your/directory* parameter. The following output shows the installation process, which is started by checking the requirements for *pDLNA*. While step 2 is checking for two more Perl modules for installation, step 3 installs the necessary files from *pDLNA* to the specified directories and step 4 modifies the installed files for the installation specific parts. In the end, step 5 verifies the installation.

```
pdlna@mediaserver:/tmp/pDLNA$ perl install.pl -i
-------------------------------------------------------
Step 1:
Testing for necessary Perl Modules ...
-------------------------------------------------------
ok 1 - use Audio::FLAC::Header;
ok 2 - use Audio::Wav;
ok 3 - use Audio::WMA;
ok 4 - use Config;
ok 5 - use Config::ApacheFormat;
ok 6 - use Data::Dumper;
ok 7 - use Date::Format;
ok 8 - use Devel::Size;
ok 9 - use Digest::MD5;
ok 10 - use Digest::SHA1;
ok 11 - use Fcntl;
ok 12 - use File::Basename;
ok 13 - use File::Glob;
ok 14 - use File::MimeInfo;
ok 15 - use GD;
ok 16 - use Getopt::Long::Descriptive;
ok 17 - use Image::Info;
ok 18 - use IO::Interface;
ok 19 - use IO::Select;
ok 20 - use IO::Socket;
ok 21 - use IO::Socket::INET;
ok 22 - use IO::Socket::Multicast;
ok 23 - use LWP::UserAgent;
ok 24 - use MP3::Info;
ok 25 - use MP4::Info;
ok 26 - use Net::Address::Ethernet;
ok 27 - use Net::Interface;
ok 28 - use Net::IP;
ok 29 - use Net::Netmask;
ok 30 - use Movie::Info;
ok 31 - use Ogg::Vorbis::Header;
ok 32 - use POSIX;
ok 33 - use Proc::ProcessTable;
ok 34 - use Socket;
ok 35 - use Sys::Hostname;
ok 36 - use Sys::Syslog;
ok 37 - use threads;
ok 38 - use threads::shared;
ok 39 - use URI::Split;
ok 40 - use XML::Simple;
-------------------------------------------------------
```

```
Step 2:
Testing for necessary Perl Modules for installation ...
-------------------------------------------------------

ok 41 - use File::Copy;
ok 42 - use File::Copy::Recursive;
-------------------------------------------------------
Step 3:
Installing files ...
-------------------------------------------------------
ok 43 - Installed './PDLNA' to '/opt/pDLNA/PDLNA'.
ok 44 - Set rights for '/opt/pDLNA/PDLNA'.
ok 45 - Installed './external_programs' to
  '/opt/pDLNA/external_programs'.
ok 46 - Set rights for '/opt/pDLNA/external_programs'.
ok 47 - Installed './VERSION' to '/opt/pDLNA/VERSION'.
ok 48 - Set rights for '/opt/pDLNA/VERSION'.
ok 49 - Installed './pDLNA.pl' to '/opt/pDLNA/pDLNA.pl'.
ok 50 - Set rights for '/opt/pDLNA/pDLNA.pl'.
ok 51 - Installed './LICENSE' to '/opt/pDLNA/LICENSE'.
ok 52 - Set rights for '/opt/pDLNA/LICENSE'.
ok 53 - Installed './README' to '/opt/pDLNA/README'.
ok 54 - Set rights for '/opt/pDLNA/README'.
ok 55 - Installed './pdlna.conf' to '/etc/pdlna.conf'.
ok 56 - Set rights for '/etc/pdlna.conf'.
ok 57 - Installed './rc.pDLNA' to '/etc/init.d/rc.pDLNA'.
ok 58 - Set rights for '/etc/init.d/rc.pDLNA'.
-------------------------------------------------------
Step 4:
Setting of relevant paths ...
-------------------------------------------------------
ok 59 - Changed path for binary in '/etc/init.d/rc.pDLNA'.
ok 60 - Changed path for lib in '/opt/pDLNA/pDLNA.pl'.
-------------------------------------------------------
Step 5:
Checking for pDLNA Perl Modules ...
-------------------------------------------------------
ok 61 - use PDLNA::Config;
ok 62 - use PDLNA::ContentDirectory;
ok 63 - use PDLNA::ContentItem;
ok 64 - use PDLNA::ContentLibrary;
ok 65 - use PDLNA::Daemon;
ok 66 - use PDLNA::Device;
ok 67 - use PDLNA::DeviceList;
ok 68 - use PDLNA::DeviceService;
ok 69 - use PDLNA::DeviceServiceAction;
ok 70 - use PDLNA::DeviceUDN;
ok 71 - use PDLNA::HTTPServer;
ok 72 - use PDLNA::HTTPXML;
ok 73 - use PDLNA::Media;
ok 74 - use PDLNA::Log;
ok 75 - use PDLNA::SSDP;
ok 76 - use PDLNA::Status;
ok 77 - use PDLNA::Utils;
ok 78 - use PDLNA::WebUI;
1..78
```

After configuring your *pDLNA* installation (see chapter 6), you are able to start *pDLNA* by executing the following command:

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

3.3.3 Updating pDLNA

Currently, there is no simple way to update this *pDLNA* installation. Actually, you should do a backup of the `/etc/pdlna.conf` configuration file by executing for instance

```
pdlna@mediaserver:~$ sudo cp /etc/pdlna.conf /etc/pdlna.conf.bak
```

and rerun the installation process for the new version again. In the end, you are able to restore the old configuration file by simple executing the following command:

```
pdlna@mediaserver:~$ sudo cp /etc/pdlna.conf.bak /etc/pdlna.conf
```

Chapter 4

Installing on Windows

Currently installing and running *pDLNA* has not been tested on a Microsoft Windows operation system yet. Please contact me about your experiences regarding installation and/or executing *pDLNA* on Windows.

NOTE: If somebody is interrested into porting and maintaining *pDLNA* for Windows, please contact me and start doing it. Thanks.

Chapter 5

Installing on MacOS X

pDLNA has not been tested on a MacOS X operating system yet either. So please contact me about your experiences regarding installation and/or executing *pDLNA* on a Apple operating system.

NOTE: In fact, you should be able to install *pDLNA* similar as described in chapter 3.

Chapter 6

Configuration

This chapter gives an overview about the possible parameters to configure *pDLNA* and describes their functionality and their possible impact on the installation. By default, the configuration file is stored in `/etc/pdlna.conf`. If you would like to change the location of the configuration file, you need to change the following line in the initscript (by default `/etc/init.d/rc.pDLNA`) to the correct location:

```
CFGFILE="/etc/pdlna.conf"
```

Some of the available configuration parameters are binary values, which can be enabled or disabled. To enable one of those parameters, simple configure the parameter to one of the following values:

- `on`
- `true`
- `yes`
- `enabled`
- `enable`
- `1`

The parsing of these values is case insenstive and if you would like to disable one of these parameters, simple use another value (.e.g. `disabled` or `Off`).

6.1 Global parameters

6.1.1 FriendlyName

The *FriendlyName* configures a name, which will be shown be all capable DNLA devices to identify a running digital media server. By default, the *FriendlyName* will be set to `pDLNA v$VERSION on $HOSTNAME` or you are able to define your *FriendlyName* by configuring it like this:

```
FriendlyName 'pDLNA media server'
```

6.1.2 Check4Updates

If *Check4Updates* is enabled, the running *pDLNA* installation will check every 24 hours, if there is a new version of *pDLNA* available. Therefore, *pDLNA* will do a HTTP request to `http://www.pdlna.com/cgi-bin/status.pl` and transmits its **version number** including the information if it is a **beta release** and the configured or generated **UUID** as XML data to the server. The response from the server is XML data too, which will be evaluated by the running *pDLNA* installation and the result will be logged.

If you do not like *pDLNA* to check for a new version, you are able to deactivate this feature by adding the following line to your configuration file.

```
Check4Updates Off
```

6.1.3 PIDFile

This parameter specifies the location of the *PIDFile*, which is used to store the process ID of a running *pDLNA* installation. By default, the parameter is set to `/var/run/pdlna.pid` or you are able to specify a different location by configuring like this:

```
PIDFile /var/run/pdlna.pid
```

IMPORTANT NOTE: please ensure, that the user, which is running *pDLNA* has permissions to write to the configured path.

6.2 Network configuration

If you are not sure about the installed network interfaces and configured IP addresses, execute the following command:

```
pdlna@mediaserver:~$ sudo ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state
 UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc
 pfifo_fast state UNKNOWN qlen 1000
    link/ether 00:0c:29:bc:fc:da brd ff:ff:ff:ff:ff:ff
    inet 192.168.145.139/24 brd 192.168.145.255 scope global eth0
    inet6 fe80::20c:29ff:febc:fcda/64 scope link
       valid_lft forever preferred_lft forever
```

In the following two sections regarding the configuration of *ListenInterface* and *ListenIPAddress* this output will be taken as an example.

6.2.1 ListenInterface

The *ListenInterface* parameter specifies the network interface, *pDLNA* should be using. If the parameter is **not** specified, *pDLNA* will try to determine the first active network interface with an IP address.

In case, that *pDLNA* was not able to determine the correct network interface, you need to specify it by the following configuration line:

```
ListenInterface eth0
```

6.2.2 ListenIPAddress

The *ListenIPAddress* parameter specifies the IP address, *pDLNA* should use to communicate with the DLNA capable devices. If *ListenIPAddress* is **not** specified, *pDLNA* will try to determine the first IP address on the configured or even detected *ListenInterface*. Otherwise you are able to define it by configuring *ListenIPAddress* like this:

```
ListenIPAddress 192.168.145.139
```

6.2.3 HTTPPort

The *HTTPPort* parameter defines the TCP port, which should be used by the integrated HTTP server. By default it will be set to `8001` or you are able to define another by the following parameter:

```
HTTPPort 8001
```

IMPORTANT NOTE: Please ensure, that the configured TCP port is **not** used by any other application, otherwise *pDLNA* will not be able to start up correctly.

6.2.4 AllowedClients

For reasons of data privacy, you need to specify IP address(es) and/or subnet(s), which should be able to communicate with *pDLNA*. *SSDP* is limited to the local subnet because of the Multicast communication (as long as there is no Multicast Routing), but HTTP is not limited to the local subnet.

```
AllowedClients 192.168.145.2, 192.168.145.128/26, \
 192.168.145.192/255.255.255.252
```

IMPORTANT NOTE: if **none** are specified, the *ListenIPAddresses* local subnet will be configured. Please limit the number of hosts.

IMPORTANT NOTE: the hosts, which should access the WebUI (see chapter 7) have also to be added to the *AllowedClients* configuration parameter.

PARAMETER	DESCRIPTION
`Version3`	the hostname's MD5 checksum
`Version4`	random generated UUID
`Version4MAC`	random generated UUID including the MAC address of the configured *ListenInterface* in the end
`Version5`	the hostname's SHA-1 checksum
`<staticUUID>`	define a static UUID, which is formated like `56657273-696f-6e34-4d41-000c29bcfcda`

Table 6.1: Available UUID configuration parameters

6.3 SSDP configuration

6.3.1 CacheControl

CacheControl represents a parameter in the Simple Service Discovery Protocol (SSDP) and defines the time in seconds, clients will cache the server's information. The value has impact on the interval *pDLNA* is going to send out his SSDP alive messages.

Most devices and *pDLNA* use as default value `1800` seconds. If you would like to use a different value, define it in seconds like this:

```
CacheControl 1800
```

IMPORTANT NOTE: Changing this value **may** result into malfunction of *pDLNA*.

6.3.2 UUID

The Universally Unique Identifier (UUID) is used in the Simple Service Discovery Protocol (SSDP) as an unique identifier. RFC 4122[1] describes the format and the different methods to generate a UUID.

The available configuration parameters are listed in table 6.1. If **none** is specified, `Version4` will be used or you are able to change the parameter by the following line:

```
UUID Version4MAC
```

IMPORTANT NOTE: the method for generating the UUIDs are **not** (completely) compliant to RFC 4122 and are **only** pseudo-random.

[1] `http://www.ietf.org/rfc/rfc4122.txt`

6.4 DLNA configuration

6.4.1 BufferSize

The *BufferSize* defines the default size of a chunk, which is used to transfer streaming data. The default value for the *BufferSize* is `32768`. The following example shows how to change the value:

```
BufferSize 1337
```

IMPORTANT NOTE: Changing this value **may** result into malfunction of *pDLNA* or may even result into a crash of your system because of high memory usage.

6.4.2 SpecificViews

The *SpecificViews* configuration parameters enables for specific DLNA aware devices, like

- Samsung TV

some different method for directory listings. By default, this value is disabled. To enable this feature, simple configure it by adding the following line to your configuration file:

```
SpecificViews Off
```

IMPORTANT NOTE: Enabling *SpecificViews* in the current version will result in not being able to fulfill a directory listing request of a *Samsung TV*.

6.4.3 EnableImageThumbnails

By switching the parameter *EnableImageThumbnails* on or off, you are able to decide if preview thumbnails of images should be displayed on capabale devices. By default, this feature is deactivated.

IMPORTANT NOTE: Enabling *EnableImageThumbnails* might decrease the performance of directory listings.

```
EnableImageThumbnails On
```

6.4.4 EnableVideoThumbnails

By switching the parameter *EnableVideoThumbnails* on or off, you are able to decide if preview thumbnails of video files should be displayed on capabale devices. By default, this feature is deactivated.

IMPORTANT NOTE: Enabling *EnableVideoThumbnails* might decrease the performance of directory listings and will require `Mplayer`.

```
EnableVideoThumbnails On
```

6.4.5 LowResourceMode

By default, *pDLNA* opens every single video and audio file with `Mplayer` to determine its codecs, length and various other attributes. This behaviour is not really IO friendly and so crawling and indexing the media library is slow. Because of small devices, with limited resource like CPU and IO, enabling the `LowResourceMode` increases crawling and indexing the media library enormous.

By default, `LowResourceMode` is disabled.

```
LowResourceMode On
```

IMPORTANT NOTE: Enabling `LowResourceMode` will decrease the usability: No information like codecs, length and various other attributes will be gathered from the media itself. Because of this limited information, directory listings might not be as beautiful as when `LowResourceMode` is disabled.
When `LowResourceMode` is enabled, *pDLNA* will ignore `External` and `Transcode` configuration blocks. *pDLNA* will also not index audio or video streams as elements from a playlist. But `Mplayer` and `FFmpeg` are not required.

6.4.6 MPlayerBinaryPath

pDLNA has to use `MPlayer` for different various tasks (see section 2.3.1 for detailed information). In this cases, it is necessary to configure *MPlayerBinaryPath* with the correct path of `MPlayer`'s binary. The default path for `MPlayer`'s binary is set to `/usr/bin/mplayer`.

```
MPlayerBinaryPath /usr/bin/mplayer
```

6.4.7 FFmpegBinaryPath

pDLNA has to use `FFmpeg` for transcoding. So it is necessary to configure *FFmpegBinaryPath* with the correct path of `FFmpeg`'s binary. The default path for `FFmpeg`'s binary is set to `/usr/bin/ffmpeg`. For detailed information about `FFmpeg` please see section 2.3.2.

```
FFmpegBinaryPath /usr/bin/ffmpeg
```

PARAMETER	DESCRIPTION
`STDERR`	all logging output will be printed to *STDERR*
`SYSLOG`	all logging output will be logged to syslog
`<full path to log file>`	specify a full path to a file, *pDLNA* should use as a logfile (like `/var/log/pdlna.log`)

Table 6.2: Available LogFile configuration parameters

LOGLEVEL	DESCRIPTION
`0`	normal
`1`	debug
`2`	debug 2
`3`	debug 3

Table 6.3: Available LogLevel configuration parameters

6.5 Logging

6.5.1 LogFile

The *LogFile* configuration parameter defines the logging location of *pDLNA*. Table 6.2 gives an overview about the available configuration options.

IMPORTANT NOTE: please ensure, that the user, which is running *pDLNA* has permissions to write to the configured path.

If *LogFile* is not specified, *pDLNA* will print all logging output to *STDERR*.

```
LogFile /var/log/pdlna.log
```

6.5.2 LogFileMaxSize

pDLNA is able to keep track of the logfile's size and clear it, if the size exceeds a value specified by *LogFileMaxSize* in Megabytes. The configured value has to be **greater than 0 and less than 100**.

By default, *LogFileMaxSize* is set to 10 Megabytes.

```
LogFileMaxSize 10
```

6.5.3 LogLevel

pDLNA is capable to differentiate between different kind of log messages by their *LogLevel*. Specifying a higher *LogLevel* will result in more detailed logging messages including the messages of the lower *LogLevels*. Please see table 6.3 for detailed information about the different *LogLevel*

If *LogLevel* is **not** specified, it will be set to 0.

```
LogLevel 1
```

LogCategory	Description
`discovery`	SSDP messages
`httpdir`	messages from the directory listings via HTTP
`httpstream`	messages from the streaming via HTTP
`library`	messages from building the media library
`httpgeneric`	generic HTTP messages

Table 6.4: Available LogCategory configuration parameters

Variables	Description
`%m`	number of month
`%d`	numeric day of the month
`%H`	hour, 24 hour clock
`%I`	hour, 12 hour clock
`%p`	AM or PM
`%M`	minute
`%S`	second
`%s`	seconds since the epoch, UCT (aka unixtimestamps)
`%o`	ornate day of month – 1st, 2nd, 25th, etc.
`%Y`	year
`%Z`	timezone in ascii. eg: PST
`,-_`: and spaces	characters to format the date string

Table 6.5: Available DateFormat configuration variables

6.5.4 LogCategory

pDLNA is capable to differentiate between different kind of log messages by their *LogCategory*. To activate log messages of one of the *LogCategory* listed in table 6.4, simple configure them via a comma seperated list, like in the following example, which enables all categories.

```
LogCategory discovery,httpdir,httpstream,library,httpgeneric
```

By default, only some generic messages will be logged and **none** of these categories are enabled.

6.5.5 DateFormat

By configuring *DateFormat*, you are able to define the format of time and date in log or debug messages. The following table 6.5 show valid variables, their value and other valid characters to format the *DateFormat* string.

If *DateFormat* is **not** specified, it will be set to `%Y-%m-%d %H:%M:%S`, which will result for instance in `2012-12-21 13:37:00`.

```
DateFormat '%Y-%m-%d %H:%M:%S'
```

RescanMediaInterval	Description
`never`	never recrawl the media directories
`hourly`	media library will be marked as expired after 60 minutes
`halfdaily`	media library will be marked as expired after 12 hours
`daily`	media library will be marked as expired after 24 hours

Table 6.6: Available RescanMediaInterval configuration variables

6.6 Media Configuration

6.6.1 RescanMediaInterval

By configuring *RescanMediaInterval*, you are able to define the interval, when *pDLNA* recrawls the configured media directories. In fact, the media library will be marked as expired, when the media library is older than the *RescanMediaInterval*. If the media library is marked as expired and no devices are connected to *pDLNA*, the configured media directories and external media items will be recrawled.

IMPORTANT NOTE: If the media library is marked as expired, *pDLNA* will still handle all requests.

The following table 6.6 shows valid variables and their descriptions to configure *RescanMediaInterval*.

If *RescanMediaInterval* is **not** specified, it will be set to `never`.

```
RescanMediaInterval daily
```

6.6.2 Directory

By default and for reasons of data privacy, *pDLNA* will **not** scan automatically for media files and will **not** add them automatically to the media library.

Therefore you need to configure your media directories in the configuration file in a configuration block called `<Directory>`. The amount of `<Directory>` blocks is limited to `900` media directories. Each of these blocks can be configured separately with parameters, which are described in table 6.7.

The following configuration block will crawl the directory `/media/video/` **recursively** for only **video** files.

```
<Directory "/media/video/">
  MediaType      video
  Recursion      yes
</Directory>
```

Parameter	Information	Description
`MediaType`	obligatory	specify the type of media files, which should be crawled: `video`, `audio`, `image`, `all`
`Recursion`	optional, by default: `yes`	specify if these directories should be crawled recursively (by `yes` or `no`)
`ExcludeDirs`	optional	exclude a comma seperated list of directory names from being crawled and added to the media library
`ExcludeItems`	optional	exclude a comma seperated list of file names from being crawled and added to the media library
`AllowPlaylists`	optional, by default: `no`	enable AllowPlaylists configuration parameter to initialize playlist files

Table 6.7: Available Directory block configuration parameters

MediaType	MimeTypes
`video`	`video/x-msvideo`, `video/x-matroska`, `video/mp4`, `video/mpeg`, `video/x-flv`
`audio`	`audio/mpeg`, `audio/mp4`, `audio/x-ms-wma`, `audio/x-flac`, `audio/x-wav`, `video/x-theora`, `audio/ac3`, `audio/x-aiff`
`image`	`image/jpeg`, `image/gif`
`Playlist`	`audio/x-scpls`, `audio/x-mpegurl`, `application/vnd.apple.mpegurl`, `audio/x-ms-asx`, `video/x-ms-asf`, `application/xspf+xml`

Table 6.8: Supported MimeTypes

The next configuration snippet will crawl the directory `/media/music/` **recursively** for only **music** files. Additionally *pDLNA* looks for **Playlist files** and excludes directories which are called `Justin Bieber` or `Lady Gaga` and also ignores files with the following names: `justin_bieber.mp3` and `lady_gaga.mp3`.

```
<Directory "/media/music/">
  MediaType      audio
  ExcludeDirs    "Justin Bieber,Lady Gaga"
  ExcludeItems   "justin_bieber.mp3,lady_gaga.mp3"
  AllowPlaylists true
</Directory>
```

The third configuration example will crawl the directory `/media/images/` **recursively** for only **images**.

```
<Directory "/media/images/">
  MediaType      image
</Directory>
```

And the last snippet crawls the directory `/media/misc/` **not recursively** for all sort of media files excluded for **Playlist files**.

```
<Directory "/media/misc/">
  MediaType      all
  Recursion      no
  AllowPlaylists off
</Directory>
```

6.6.3 External

TODO

```
<External "mpegstream">
  Executable /media/external/mpegstream_src.pl
</External>
```

```
<External "FM4">
  StreamingURL http://mp3stream1.apasf.apa.at:8000/
</External>
```

6.7 Transcoding Profiles

Since not all DLNA aware devices support all different kinds of audio or video codecs, transcoding these media files allows playing these files on these devices anyway.

In / Out	AAC	AC3	FLAC	MP3	VORBIS	WAV	WMAV2
AAC							
AC3							
FLAC							
MP3							
VORBIS							
WAV							
WMAV2							

Table 6.9: Supported Audio Transcoding Profiles

Audio Codec	Media Container
flac	audio
mp3	audio
vorbis	ogg

Table 6.10: Audio Codecs and supported Media Container

Table 6.9 gives an overview about the currently tested and supported transcoding profiles for audio files. The necessary information regarding the Media Container can be found in table 6.10.

The green colored table cells show the currently supported and tested transcoding possiblites.

The following configuration snippet will transcode all media files, which do have an *ogg* Media Container and *vorbis* encoded audio data to a *flac* encoded audio stream.

```
<Transcode "ogg2flac">
  ContainerIn    ogg
  Containerout   audio
  AudioCodecIn   vorbis
  AudioCodecOut  flac
</Transcode>
```

IMPORTANT NOTE: Transcoding depends on the supported audio and video codecs of your `FFmpeg` installation and some validation in the source code to ensure its functionality.

Chapter 7

WebUI

TODO

7.1 Configured media

7.2 Connected devices

7.3 Statistics

7.3.1 Process Info

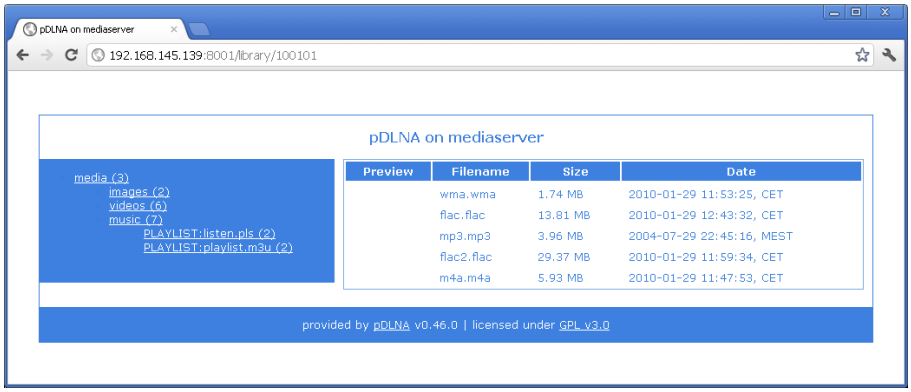

Figure 7.1: WebUI

Chapter 8

Debugging and reporting problems

This is not (yet) a real troubleshooting guide to *pDLNA*. Currently it is more a small handbook to do some general debugging and how to gather some necessary information on a running Linux operating system. You are also able to forward this information to me.

8.1 Debugging

When starting *pDLNA*, the process will parse the configuration file[1] and will not start until everything is configured correctly. In some cases *pDLNA* might not be able to determine the correct *ListenInterface*, where you need to configure those information by hand. The following command starts *pDLNA*.

```
pdlna@mediaserver:~$ sudo /etc/init.d/rc.pDLNA start
```

Once *pDLNA* is running successfully, you are able to verify this by running the following two commands. The first one checks all the running processes for the *pDLNA* process, while the second one prints out the stored PID. If both PID match each other, everything should be fine.

```
pdlna@mediaserver:~$ sudo ps -ef | grep pDLNA
root     13162     1  3 09:18 pts/2    00:00:50
  /usr/bin/perl ./pDLNA.pl -f /etc/pdlna.conf
```

```
pdlna@mediaserver:~$ sudo cat /var/run/pdlna.pid
13162
```

The following listings are based on the network configuration listed in 6.2. So, the next step to identify a lowlevel network problem of *pDLNA* will be to check if *pDLNA* is listening to the two necessary network ports. On the one

[1] Please see chapter 6 for more details.

hand, there is **port 1900 UDP** needed for the SSDP communication and **port 8001 TCP**[2] for the DLNA communication via HTTP. The following command checks the network statistics and filters for the lines including the PID of the running *pDLNA* installation.

```
pdlna@mediaserver:~$ sudo netstat -taunp | grep 13162
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address
 Foreign Address         State       PID/Program name
tcp        0      0 192.168.145.139:8001
 0.0.0.0:*               LISTEN      13162/perl
udp        0      0 192.168.145.139:37302
 239.255.255.250:1900    ESTABLISHED 13162/perl
udp    53368      0 0.0.0.0:1900
 0.0.0.0:*                           13162/perl
```

If your network statistics output shows, that *pDLNA* is correctly listening to **port 1900 UDP** and the configured *HTTPPort*, the network communication should be working.

IMPORTANT NOTE: *AllowedClients* is configured to the local subnet by default.

IMPORTANT NOTE: Please also check your firewall coniguration.

As a simple test you can visit the WebUI (see chapter 7), which can be accessed by the following URL: `http://ListenIPAddress:HTTPPort/webui/`. If you are receiving an *HTTP Error Code 403*, your host is not configured as an *AllowedClients*. If your Browser is running in a *timeout*, there might be a problem with your configuration.

After checking the general process and network information, the next step is about increasing the *LogLevel* and configuring the necessary *LogCategory* from table 6.4. At first you need to configure *LogLevel* to the highest *LogLevel* available in table 6.3 for the most detailed log messages.

8.1.1 My device is not able to discover *pDLNA*

If your DLNA capable device is not able to discover *pDLNA*, please set the configuration parameter *LogCategory* to `discovery, httpgeneric` and restart *pDLNA*. Additionally you are able to do a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -i ListenInterface \
 -s 1500 -w capture_full.pcap
```

Please ensure to fill in the configured *ListenInterface*.

[2]Please ensure that this port can be changed in the configuration file and *8001* is the default one.

8.1.2 My media directories are not read correctly

In the case, that *pDLNA* was not able to read in your shared directories correctly, enable the *LogCategory* `library` in the configuration file and restart the installed version of *pDLNA*. An easy way to navigate quickly through the media library and check for problems is by using the WebUI (see chapter 7).

8.1.3 My device is not able to list the directories/items shared by *pDLNA*

If browsing the shared media directories is not working properly by your DLNA aware devices, configure the *LogCategory* to `httpgeneric, httpdir`, restart *pDLNA* and do a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -p HTTPPort -s \
 1500 -w capture_http.pcap
```

Please ensure to fill in the configured *HTTPPort* (by default set to `8001`).

8.1.4 *pDLNA* is not able to stream media items to my device

In any case, that streaming of videos, music or images is not working properly, please set the *LogCategory* parameter to `httpgeneric, httpstream`, restart *pDLNA* and start a packet capture with the following command:

```
pdlna@mediaserver:~$ sudo tcpdump -p HTTPPort \
 -s 1500 -w capture_http.pcap
```

Please ensure to fill in the configured *HTTPPort* (by default set to `8001`).

8.2 Reporting problems

If you are not able to fix the problem or just want me to take a closer look, please supply all mentioned information, like network statistics, packet captures, logs and so on.

In case of having a DLNA aware device, which is not working properly with *pDLNA* please supply a full packet capture, including the whole SSDP and DLNA packets of this specific device communicating with another digital media server (which is working properly). In this case I might be able to determmine the differences and supply a new version of *pDLNA* which is supporting this device.

Chapter 9

Known Issues

9.1 AllowedClients auto detection on FreeBSD is defect

9.1.1 Describtion

Apparently *pDLNA* dies with the following error message on *FreeBSD* if *AllowedClients* auto detection is enabled[1]:

```
AllowedClients(ip_address_sv) at PDLNA/Config.pm line 219.
```

This seems to be a problem with the usage of `AllowedClients(ip_address_sv)` on *FreeBSD*.

9.1.2 Workaround

As a workaround you are able to define the *AllowedCLients* configuration parameter in the conifguration file. For detailed information see section 6.2.4.

[1] `https://github.com/geuma/pDLNA/issues/4`