tvpvrd

Table of Contents

Name

tvpvrd — TV Personal Video Recorder Server/Daemon

Synopsis

`tvpvrd` [options...]

Description

The tvpvrd(1) command starts a video recording server that will make use of one or more installed TV-Capture cards and, following a specified schedule, will record broadcasted and encoded streams from the capture card to disk. In effect a digital version of a programmable VHS. In a similar way as a VCR the daemon also have the capability to shutdown and power on the server in time for recordings (see the auto-shutdown section below). In the same way as the old-fashioned VHS it has the capability to turn off the power of the server and wake it up in time for a recording in a fully automated way (by using the BIOS wakeup alarm).

The original HW MPEG2 encoded videofile (from the capture card) can then be optionally transcoded (using ffmpeg(1)) to a more efficient format by the daemon. The specific details on both the initial HW encoder settings as well as the transcoding options are bundled in a user configurable 'profile'. The daemon supports the usage of multiple profiles.

Each recording is done using one (or multiple) profiles. Each recording will always have at least one profile, the primary profile. In this way it is possible to create multiple transcoded versions of one recording, for example one high quality version for viewing on a computer and one low quality versions for viewing on a mobile device. By default the distribution includes six profiles 'high', 'normal', 'low' without transcoding and 'thigh', 'tnormal', 'tlow' with transcoding enabled. Additional profiles can easily be added by the user. A profile is specified in the traditional ini-file text file format and is stored under the 'profiles' directory which is a subdirectory to the settings directory (normally `/etc/tvpvrd`). Each setting in the profile is well documented in the profiles.

The recordings specified are stored in a database which is a plain text file in XML format using UTF-8 character encoding.

Recorded and transcoded files are stored in a predefined directory structure. The base directory for this directory structure is defined in the configuration file, see also the section "Directory Structure" below.

Arguments

The following command line arguments are supported when starting the daemon

`-d`, `--daemon` [*y*|*n*]
: Determine whether the server should be disconnected from the controlling terminal (a.k.a. running as a daemon). If the server is run as a daemon the logfile cannot be '*stdout*'. This command line option will override the '*daemon*' setting in the ini-file. The default setting is to run as a daemon.

`-f`, `--xmldbfile` `filename`
: Use the specified file as the database file where all the specified recordings are stored in XML format. This command line option will override the '*xmldbfile*' setting in the ini-file. The default location is '<data-root>/xmldb/tvpvrd_db.xml'

`-h`, `--help`
: Print a short help and exit.

`-i`, `--inifile` `filename`
: Use the specified file as configuration file. This will override the location specified at compile time. The default location is '`/etc/tvpvrd/tvpvrd.conf`'

`-l`, `--logfile` `filename`
: Define the name of the logfile. Two special names can be used, `stdout` which will cause the log to be written to the *stdout* and `syslog` which will cause the logs to be written using the default system logger.

This command line option will override the '*logfile*' setting in the ini-file. The default location is '`/tmp/tvpvrd.log`'

`-p`, `--port` *nnnn*
Specify what TCP/IP port number that the server will listening on. This will override the '*port*' setting in the ini-file. By default this is port 9300. The port number of the WEB-interface will be *nnnn*+1 so by default the WEB-interface will be available on port 9301.

`-s`, `--slave`
Run the server in slave configuration. This will allow the server to be running on a machine with no video cards and be restricted to only do video transcodings.

`-V`, `--verbose` [*0*|*1*|*2*|*3*]
Log verbosity level. 0 will only log error messages, 1 will log error and info messages, 2 will log errors, info and additional notices finally 3 will also log (possibly) high volume debug messages.

`-v`, `--version`
Print program version information and exit.

`-x`, `--xawtvrc` `filename`
Use the specified file to read the station names. The station name file maps the standardized frequency channel names to the more commonly used station names. See the section "Stations, channels and frequencies" below for more information. This will override the setting in the ini-file. The default location is '`/etc/X11/xawtvrc`'

The default arguments corresponds to starting the server with

```
tvpvrd -d y -f tvpvrd_db.xml -i /etc/tvpvrd/tvpvrd.conf -p 9300 -V 1 -x /etc/X11/xawtvrc
```

Starting and stopping the server

While strictly speaking not necessary the server should normally be started by the root user. As soon as the server doesn't need root privileges it will switch to run as the user specified in the ini-file, by default this is the user *tvpvrd*. This user should for security reasons not be the 'root' user. This user should be a system user without login shell. Such a user can normally be created by running the command '*useradd*'. (If the server is installed via the RPM package this user will be automatically created).

Note: If you want to use the automatic shutdown/wakeup functionality the daemon must be run as root (since only root has privileges to access the BIOS wakeup functionality)

```
$> useradd -r tvpvrd
```

To start the server as root the simplest way is to just call

```
$root> tvpvrd
```

In order to stop the server and allow it to do a clean shutdown the SIGHUP (or SIGSTOP) signal should be sent to the process running the server (assuming tvpvrd has been started as a daemon). The distribution includes two support scripts to help with this.

The first script will be automatically installed in "`/etc/init.d/tvpvrd`" and follows the normal guidelines for a "`init.d`" service start/stopping script according to LSB (Linux Standard Base). Using this script to start the server the command

```
/etc/init.d/tvpvrd start
```

is given and to stop the server the command

```
/etc/init.d/tvpvrd stop
```

should be given. The startup script also supports the "restart" and "status" options.

The second script is '`stop-tvpvrd`' which will read the server process pid from '`/var/run/tvpvrd.pid`' and send the SIGHUP signal.

In order to stop the server first switch to root user and then use the command

```
$root> stop-tvpvrd
```

Finally if for some reason you don't want to use any of these scripts it is possible to manually kill the daemon process by sending the SIGHUP signal via the '*kill*'command as

```
$> kill -SIGHUP <pid>
```

where <pid> is the process id of the *tvpvrd* daemon. Use 'ps xa' to find the pid or if the server was started as root look in the file '`/var/run/tvpvrd.pid`'

When the server is shutdown, depending on the "Kill transcoding flag" (see the `-ktf` command), any on-going transcodings will be shutdown. Any potential ongoing recordings will be aborted and the recordings up to the time the recording is shutdown will be left in the '`<data-dir>/vtmp`' directory and must be removed manually.

If the server gives the error message : "*Server can not be started. See system log*" then more information about the details of the problem can be found in the system log, normally `/var/log/messages`. One common issue might be a stale lock file (in `/var/run/tvpvrd.pid`) in case the server is not started from the root user. The reason is that in order to write these lock files the server must run as a root. Since normally the server only runs as a root at the startup phase (and then switches over to the selected user) the lock/pid file cannot be removed. This is not a problem if the server is again started as root but if the server is started from the command line as a non-root user it will complain that the lock file already exists. The solution in this case is to manually remove the lockfile (as root).

Interfacing with the server

The primary interface to the server is via a TCP/IP connection to the server running '*tvpvrd*' via a dedicated port (by default 9300). The server is controlled via a simple (but efficient) command language which allows manipulation of future recordings and various other administrative tasks. All commands are extensively described in the section 'Command language' below.

The preferred way of interacting with the server on the command level is by using the supplied utility shell 'tvpsh'. This supports history and full command line editing and is recommended over a plain 'telnet' connections to the server.

As an optional way there is also a native basic WEB-interface that allows a limited number of commands such as adding of new recording, deleting existing recordings, viewing status, showing upcoming recordings etc. The WEB-inerface contains enough controls and command to handle most day-to-day needs for a digital VCR and it is anticipated that most users should use this interface as there main port of accessing the server.

The WEB-interface must first be enabled in the config-file since it is by default disabled. The WEB interface is accessed on port number (port+1) i.e one more than the port number specified for the command interface. This means that by default the WEB-interface is available on port 9301. The WEB-interface can optionally require a separate login as specified in the configuration file.

To access the WEB interface from a browser simple give the server address and port number. For example to access the daemon on the server "`localhost`" with the default port number use URL "`http://localhost:9301`".

The look and feel (themes) of the WEB-interface is entirely controlled by a CSS style sheet stored in the configuration directory (usually specified as `/etc`). This is the same place as where the configuration file is stored. This means it is straightforward to change and adjust the look and feel from the default colors and layout in the supplied themes. All that is necessary is basic knowledge of CSS stylesheets.

It is also possible to control some functional aspects uin the WEB-UI such as what controls are displayed in the Web UI by adjusting config file settings. Supported settings are:

- display_advanced_rec_control, This setting hides/displays two extra controls for how naming of repeated recordings are named.
- web_display_theme, If the theme switching control should be displayed.
- web_autodisplay_transc, If the transcoded information bar should be hidden if there are no ongoing transcodings.
- web_display_qadd, If the section for "Quick Recording", should be displayed in the Web-UI or not.

The WEB interface will also automatically adapt when it detects that a mobile phone browser is trying to access the interface to a format more suitable for a smaller screen. By necessity this means that only an even smaller subset of the WEB-commands are available in the browser on a phone in order to keep the interface simple. The detection is based on the header that the browser sends and should detect most phones. However, it is possible that some browsers used on mobile phone will not be detected as mobile browsers and then the normal WEB interface will be used.

See the section "Connecting to the server" below for more detailed examples.

Automatic shutdown and startup of the server

In the same way as a VCR (or a modern HD recorder) has the capability to shut down and power on in time for recording the tvpvrd can do this with the server it is running on. The daemon package has two radically different way to do this and depending on your preference (and/or HW) either way can be chosen. The server supports either the usage of the internal BIOS real time clock using a wakeup alarm or running a power management daemon (**tvpowerd**) on a separate server. If you have a reasonable modern server (say 2005 or newer) it will most likely support BIOS wakeup alarm and since that is the easiest method we strongly recommend you use that method.

Method 1: Using the BIOS wakeup alarm on the server running the tvpvrd daemon

To use this method it is first important to recognize that the daemon must be running as user 'root' in order to have the privilege to shut down the server. This is also the preferred method.

To enable this feature it must first be enabled in the configuration file under the section [shutdown] . By default this feature is disabled. There are then some basic parameters that should be checked according to preference. For example the parameter ignore_users will determine if the shutdown should still happen if there are user logged in to the server or not. All the available parameters are fully specified and documented in the default configuration file.

To make this work the daemon assumes that the RTC (BIOS Real Time Clock) presents an interface on the "/sys/class/rtc/rtc0/wakealarm" that can be used to set the time in seconds since Epoc when the server should wake up if it is powered off. In some BIOS it is also necessary to first enable the wakeup alarm by changing some default setting (check your BIOS documentation).

In order not to miss the recording the server is woken up a little bit before the actual recording time. By default this is 180 sec which should be enough time for even old servers to fully wake up. This pre-wakeup time is configurable in the configuration file.

The actual shutdown of the server is done by the "shutdown.sh" script stored in the configuration directory. In the distribution this is included as a template "shutdown.sh.template" which can then be augmented. By default the script initiates a complete server shutdown and if that is enough the template can just be copied and renamed.

If the server is reasonable new it might be possible to do a sleep (or hibernate) instead but in that case we would recommend using the second method of auto-shutdown using the separate "**tvpowerd**" as described below. Please note that there are some special precautions that must be taken to use sleep or hibernation mode since the ivtv-driver is not power management aware. So unless absolutely necessary please use the safe and always working method of just doing a complete shutdown. See tvpowerd(1) for more details.

Method 2: Using the tvpowerd-daemon on a separate server

Since some older motherboards does not have a working BIOS wakeup clock and for the case where you rather sleep (or hibernate) the server instead of shutting it down the package gives an alternative auto-shutdown method using the "**tvpowerd**" daemon. This is a separate daemon that must be run on a separate server. This daemon will then communicate with the server running the recording daemon and shut the server down and power it on according to the recording schedule. The power on will be done using Wakeup-On-LAN so this must be enabled in the BIOS on the recording server. There is a complete separate manual page describing all the details on how to use this daemon in tvpowerd(1)

When shall I use Method 1 and when shall I use Method 2 ?

If only one server is available and the motherboard supports the RTC wakeup alarm then method 1 is the preferred way. Even if you have multiple servers available it is still easier to use method 1. However there is one circumstance where it might be more practically to use the **tvpowerd** standalone daemon. For the common case where you have a NFS or SMB file share exposed from the recording server, for example to be able to view the videos on another computer, there are all kinds off issues if the SMB or NFS server is just shut down without the share unmounted. Especially the NFS protocol is notoriously bad in handling a disconnected server without a proper unmount. Running the separate **tvpowerd** daemon on the client will allow you to put in the necessary unmount commands in the shutdown script (assuming the tvpowerd is running on the server mounting shares).

In future releases of **tvpvrd** it will be possible to run user selectable scripts on clients to the recording server at the shutdown time. This can then be used to, for example, give the client notification that the recording server is shutting down and then be able to do any necessary unmounts or other cleanup activities.

In the case where you rather want to suspend to RAM/DISK then it is also necessary to run the separate **tvpowerd** daemon. The main reason here is that since the ivtv-drivers that resides in the kernel is not power management aware they must therefore be unloaded before the server is put into hibernation and then reloaded when the server is woken up. This can be (and is) all taken care of in the shutdown scripts for the tvpowerd daemon. However, there is really no practical reason to put the recording server in sleep/hibernation state compare with a full shutdown. The only thing gained is faster startup time which for these use-cases are not really interesting (nor necessary).

Running as master or slave

Normally the daemon will do necessary checks to verify that one or several video cards are present in order to startup. If no video cards are found the server will print an error message and terminate. However it is still possible to run the daemon on a server with no video cards by running the server in *slave* mode. This will restrict the functionality of the server to only transcoding and status information commands.

The primary purpose of running the server in slave configuration is to make it possible to use several servers to transcode video files. There are three commands "**tf**" (*transcode file*), "**tl**" (*transcode list*) and "**td**" (*transcode directory*) that can be used to transcode one or more files in slave mode (see the command section below for details).

In future version of the server there will be possibility to configure a load balancing setup among several servers for the transcoding but as of version 1.x this has to be done manually.

The server can be run in slave mode by either giving the "`-s`" option when starting the server or by setting the configuration alternative "`master=no`" in the configuration file.

Connecting to the server

Once the server is running the easiest way to connect to the server is by using the '*telnet*' program. Assuming that the *tvpvrd* server is running on the localhost the following command will connect to the server (assuming the default port 9300 has not been changed in the ini-file (e.g. `/etc/tvpvrd.conf`) or by a command line argument).

```
$> telnet localhost 9300
```

After a successful connection the server will reply with a short information message. The full returned information message at the connection time should look something similar to the following (your version and build might be different):

```
Welcome to tvpvrd-2.3.0  [Server] (build:20101225.191)
You are user number 1 out of 4 allowed.
Type 'exit' to disconnect.
Type 'h' to list available commands.
Connection will be closed after 30 min of inactivity.
```

If password protection (see section "Security" below) the welcome message is not be shown until the correct password has been specified. Instead when password protection is enabled the connection sequence looks like this

```
$> telnet localhost 9300
Password:xxxxxxx
Welcome to tvpvrd-2.3.0  [Server] (build:20101225.191)
You are user number 1 out of 4 allowed.
Type 'exit' to disconnect.
Type 'h' to list available commands.
Connection will be closed after 30 min of inactivity.
```

This means that the correct server password has to be given before the welcome message is seen. The server is now ready to accept commands on this connection.

To disconnect to the server just type '**exit**'

Using the tvpsh to connect to the daemon

Starting with version 3.1.0 the distribution also includes a full command line shell, **tvpsh**. This allows much better editing (emacs compatible editing) as well as full command line history. This shell can also be used as a system login shel and hereby making it possible to connect to the daemon using **ssh** for a more secure communication by creating a unique user with tvpsh as the login shell.

How to setup and use the tvpsh command line shell is fully described in tvpsh(1)

In its simplest for (assuming you are on the same sever as the dameon) the shell can be started with

```
$> tvpsh
```

Run **tvpsh -h** for a list of supported options.

The recording and transcoding process

The recording process refers to the process of saving the generated MPRG2 video stream from the capture card to a file.

The transcoding process refers to the re-coding of the original MPEG2 video stream to a more efficient format, usually an MPEG4 container using H.264 video codec (the default in the distributed profiles). Depending on the quality settings it is common to see reduction in original file size with 75%.

The transcoding process is very CPU-intensive and with the default settings in the distribution the normal profile will require anywhere between roughly 1.5 times the original recording length to transcode on a pentium dual core to roughly 0.5 times the recording length on a newer i3 or i5 Intel CPU. With a high end Intel multi-core i7 CPU and some basic setting the transcoding time can be down to ~0.3 times the length of the original recording.

In order not to make the machine totally unresponsive the ini-file specifies a maximum load on the server for a transcoding to start in order not to load the server with too many parallel transcoding processes. The transcoding process is also run with a 'nice' value of 20 to have the transcoding process interfere as little as possible with any interactive use on the server. If the load is too high the transcoding processes will be put in a queue waiting for the load to become acceptable. Note that the serer will not wait forever and if the load has not dropped below the threshold for a certain time (as specified by the `max_waiting_time_to_transcode` setting in the ini-file) the transcoding will be cancelled and the original MPEG2 file left alone.

When a recording is started a temporary directory will be created under '`<data-dir>/vtmp/vid<n>/`' where <n> is the number of the video card used. The name of the temporary directory will be the same as the base name of the recording. The MPEG2 stream is read from the video card and stored in a file in the temporary directory with the same name as the title (converted to lower case) with the suffix ".mpg".

If multiple profiles are specified there exists an ambiguity on what HW MPEG2 parameters to use in case they are different among the selected profiles. The server resolves this by using the HW parameters from the profile with the highest specified video bitrate. This is motivated by the fact that using high quality encoder settings on a low original bitrate can never give a good result.

When the recording has stopped the transcoding profile(s) is read and the transcoding is started after a check that the server load is not higher than the maximum allowed load in order to start a transcoding. If the load is too high the transcoding is put in a queue of waiting transcodings that will be started when the server load allows the transcoding to start. One transcoding will be made for each of the associated profiles of this recording.

As a special case no transcoding will be made if the '*ffmpeg*' profile bitrate is set to 0 Mbps in the profile. In this case only the original MPEG2 video file will be kept.

Once a transcoding has finished the resulting MP4 file will be moved either to the directory '`<data-dir>/mp4/<profile>/`' with the same file basename as the title but with the filename suffix ".mp4" or directly under `'<data-dir>/mp4/'` depending whether or not subdirectories should be used for each profile (see setting file).

When all the transcodings have been done and if the profile defines that the original MPEG2 video stream should be kept it is moved to the directory '`<data-dir>/mp2/`'. After that the temporary directory (and the original MPEG2 file) is deleted.

Another possibility is to do the recording first and later decide to transcode one or several video files. This is done by giving a separate transcode command to the server and specify the full file path to the file to be transcoded. In order to make this as convenient as possible there are three variants of this. One variant is to transcode one video file, the other is to specify a directory where all the files in that directory will be transcoded. The final way is to store a number of video file names in a text file and use that text file as an input list of the files to transcode. See the command description below for the commands "`tf`", "`td`" and "`tl`".

Note that it is possible to get notify by mail when a transcoding has finished. The email address used is specified in the `ini-file`

Directory structure

The root directory for where the video files (recordings) are stored is specified in the ini-file (e.g. `/etc/tvpvrd/tvpvrd.conf`). Under this root directory the following subdirectories and files are automatically created by the server:

`<dataroot>/mp2`

If the profile specifies that the original MP2 files recorded from the capture card should be saved they are stored under this directory.

`<dataroot>/mp4/<profile>`

The transcoded original MPEG2 files are by default placed under a subdirectory under MP4 corresponding to the name of the profile used for the transcoding. This can be changed to a flat structure instead which will cause all transcoded files to be placed directly under the `mp4/` ignoring the profile name. This is done by setting the option '`use_profile_directories`' to false in the ini-file.

`<dataroot>/stats/<profile>.stats`

The statistics recorded while transcoding is stored in a file corresponding to the name of the profile.

`<dataroot>/vtmp`

Temporary video directory used while recording and transcoding is performed. Each capture card used has its own subdirectory named '`vid`'*n*. Where '*n*' is the number for the video card in that folder.

`<dataroot>/xmldb/tvpvrd_db.xml`

The XML DB file that holds the database for all defined recordings.

Security

In order to prevent unauthorized access to the server the server have a basic password protection. This means that a password can be configured to be used before a client can start giving command to the server. This is configured in the ini-file. The configuration uses the same password for all clients so the server does not support multiple users/multiple password. Since the password is stored in plain text format the configuration file should have restricted access. This should not be considered strong security but rather a weak "stop-messing-with-my-recordings" protection.

The setting whether password should be used or not and the actual password is specified in the ini-file.

After connecting to the server when the password has been enabled the first string sent back from the server will be the single string:

```
Password:
```

The rest of the reply is the same as when no password is used.

For the WEB-interface it is also possible to specify a different user and password to allow the usage of the WEB GUI.

Stations, channels and frequencies

All TV stations are broadcasted on a local unique frequency. The frequency spectrum is a natural resource that divided in internationally standardized fixed width channels. A channel is the same as a specified frequency. The distance between each assigned frequency is different depending on both the frequency area and the TV standard used. This depends on the fact that different TV standard requires different bandwidth. This channel division is also different depending on the geographic location.

It is important to realize that these fixed frequencies are assigned according to international standards. In each city (and city area) the broadcasting company is then responsible for assigning local stations to appropriate frequency channels. This is the difference between station and channel names. The channel names are standardized names that have short name with 2 to 5 characters, for example 'SE14'. Even though the frequency division is standardized the exact channel names are not always given with the assigned names. For example a broadcasting company might claim (for example on their home page) that the station 'BBC1' in a certain area can be found on channel '14' when they actually should have said that the station can be found on channel 'SE14' which is the official name of this channel.

To further complicate matters different countries uses different standardized frequency maps. The program has built in knowledge of the frequency channels available in the following geographic areas

- `"europe-west"`, Western europe
- `"europe-east"`, Eastern Europe
- `"france"`, France
- `"ireland"`, Ireland
- `"italy"`, Italy
- `"australia"`, Australia

- "newzealand", New Zealand
- "us-bcast", United States air frequencies
- "us-cable", United States cable frequencies
- "us-cable-hrc", United States cable frequencies
- "us-cable-irc", United States cable frequencies
- "japan-bcast", Japan air frequencies
- "japan-cable", Japan cable frequencies
- "china-bcast", China air frequencies
- "southafrica", Southafrica air frequencies
- "argentina", Argentina air frequencies
- "australia-optus", Australia alternative air frequencies

Since the program has no way to automatically know which map to use this must be specified in the application ini-file. This map file lists all the defined frequencies and the official name for that country and frequency. The map to use should be specified with the name shown in the list above as one entry in the station-channel map file under the "global" section as tag "freqtab", for example as

```
[global]
freqtab = europe-west
```

In order to record a specific station the tuner on the capture card must be tuned to the correct frequency channel used by the station we want to record. Since it is a lot easier to use the station names, for example "BBC1", than the official name on the frequency channel, e.g. SE14, the program uses yet another mapping file so it is possible to use station names when recording instead of frequency channels. This is the way most people think about "TV-Stations".

In order for maximum re-use *tvpvrd* uses the same format of the station/channel mapping file as is done in '*xawtvrc*'. This is a plain text file using ini-file format. In the distribution an example file is included. Normally your TV provider provides a channel plan for your area. Unfortunately, to make things a bit more difficult some broadcasting company is not following the international naming convention.

If you are already viewing TV on your computer chances are very high that you already have this 'xawtvrc' ini-file since it is used by many of the available TV viewing programs. Common location for this file is either '/etc/X11/xawtvrc' or in your home directory '.xawtvrc'.

An example of a station-channel mapping file is shown below (this mapping file happens to be the maps for a large Swedish distributor in the Stockholm region):

```
[global]
freqtab = europe-west
[Kanal5]
channel = SE11
[Kanal8]
channel = 57
[Kanal9]
channel = S36
[MTV]
channel = SE16
[SVT24]
channel = E10
[SVTB/Kunsk]
```

```
channel = SE13
[TV1]
channel = E5
[TV2]
channel = E7
[TV3]
channel = E11
[TV4]
channel = E6
[TV4+]
channel = SE14
[TV6]
channel = E8
```

The mapping above says that for example the station "Kanal5" can be found on frequency channel SE11. This mapping will then make it possible to use the name "Kanal5" when specifying recordings which is a lot easier than to remember that this station is on channel "SE11".

Some cable providers are a bit naughty and does not follow the officially allocated channel names and frequencies. To handle this case there is also a possibility to specify the frequency directly in the station-channel mapping file. Please note that this is an extension to the original format of the `xawtvrc` file and is only usable by tvpvrd.

In order to provide a frequency directly the frequency is specified in place of the channel name as an integer prefixed by a '@' character. The frequency is given in kHz. An example of this is shown below

```
[StrangeStation]
channel = @134550
```

In some Unix distributions there are automatic frequency scanning helper programs (for example scantv(1)) which will search through a given frequency map and list all channels where a broadcasting was found (this is identified as a high enough signal-to-noise ratio). For example the SuSE distribution have a TV channel scanning setup in the yast2 configuration tool. The result of this channel scan is usually stored in a file named '`xawtvrc`'. This is normally stored in either '`/etc/X11/xawtvrc`' or in your home directory, as '`~/.awtvrc`'. However the actual station names you want to use for the channels must be manually given.

The somewhat strange name of the map file comes from an ancient TV viewer program under X11 which was/(is) called 'xawtv'. For historic reason this name has stayed.

While it is possible to use either the station name or the channel name in a recording there must be a '`xawtvrc`' file in order for the server to start properly. If the station file cannot be found an error message will be printed and the server will not start.

The location of this file is normally specified in the ini-file but the location can also be overridden with command line argument (`-x`) when the server is started (see Options) which will override the setting in the ini-file.

Selecting alternative video inputs as source for the encoding

A video capture card has normally several possible input to the HW MP2 encoder. When encoding TV the tuner is usually the selected input. However most cards also have one or several external connects that can be used to encode other video sources. For example Hauppauge analogue cards have one or more (depending on the card) S-Video inputs that could be used as source for the video encoding.

The first thing to do if one wants to use one of these alternative input sources is to determine exactly what input sources the combination of card and low level driver supports. This can be done with the help of the command **li** . This will list all the available input sources for the specified video card. For example if your server have a Hauppauge 350 installed as the first card giving the command

li 0

would generate output similar to

```
Hauppauge WinTV PVR-350
00 :*(tuner) Tuner 1
01 : (camera) S-Video 1
02 : (camera) Composite 1
03 : (camera) S-Video 2
04 : (camera) Composite 2
05 : (camera) Composite 3
```

This is a list of all possible video sources supported by the card. The current selected source is shown with an asterix just to the left of the type of input.

In order to use one of these inputs as the source for the recoding instead of the standard tuner a special station name is used "*_inpXY*". The 'X' should be replaced by the video card to be used (starting at 0) and the 'Y' should be replaced by the index for the wanted input source. So for example going back to the previous Hauppauge 350 above a meta station name of "`_inp04`" would use the composite 2 input source (assuming the card is the first card, i.e. 0).

In order for the server to know which input belongs to the tuner this is specified in the config file with the setting *tuner_input_index*, by default this is 0. Since there is only one setting this implies that all cards use the same index for the tuner. This is a limitation of the current implementation. If you have several cards where the tuner is located at different input indexes then the server can only work correctly on one card. This is a very minor limitation since the number of users with multiple cards with tuners on other indexes than 0 can be assumed to be very small.

A final example will make this crystal clear. Assume we want to record a CCTV camer that is connected to composite input 1 on our Hauppauge 350 card. To immediately start recording for 30 min using the default profile we could give the command

```
q _inp02 00:30 cctv_camera_one
```

the command above would immediately start a recording using card 0 and input with index 2, i.e.. the "`Composite 1`" input. The recording will be stored with the name "`cctv_camera_one.mp4`" (assuming the default profile is setup to do transcoding to MP4 format).

There is also an option to handle channel switching on a cable box. This is described in the "Adding user "hooks" for special processing" section below.

Controlling an external cable box

When an external cable box is connected to the TV-card there are some special considerations that must be taken into account since channel switching must be done by the cable box and not the TV-card itself.

In order to enable the special processing described below the configuration file setting `external_switch` must be set to "`yes`".

When a cable box is connected to the TV-card the following facts must be taken into account:

- The cable box is connected to the TV-card either to the tuner on a specific station or to the S-video or composite input on the card. In any case this renders the usual channel switching on the tuner obsolete and wrong. To specify which input to be used for the cable box the configuration file setting `external_input` is used. This should specify the numerical value of the input channel. If the cable box is connected to the tuner on a fixed station this station must also be provided. This is specified in the configuration file as the setting `external_tuner_station`

- The actual channel switching must be done on the cable box. To help with this it is possible to connect an IR-sender to the computer running **tvpvrd** and control this IR-sender from the daemon. To help with this it is possible to have the **tvpvrd** daemon call an external script which in turn will receive the needed channel as argument. The script is then responsible for issuing the low level commands that control the IR sender based on the input to the script. In the distribution a template script for this is provided in the "`switchchannel.sh.template`" script. In the configuration script it should also be specified the name of the script used to do channel switching.

It is unfortunately beyond the scope of this manual to describe how to setup an infrared sender on a Linux server (see `http://www.lirc.org/`for more information).

Note: If you are unsure of what numerical value the input source on your card have use the **tvpvrd** command "**li**" (list iputs).

Finally we list exactly what happens when a new recording is made with external switching enabled:

1. The correct input source according to the `external_input` setting is activated.
2. If the input source is the tuner the tuner is then set to the channel specified in `external_tuner_station` setting
3. The switch channel script is then called with the arguments indicating on what station (the station name not the channel name!) the recording should be made on. The name of the script is specified in the configuration file by the `external_switch_script` setting. This script should now call the low level IR routines to send the correct IR blast to the cable box to do the actual channel switching. The recording will start once the script has finished. This indicates that in order to avoid recording the actual switching of channel a short delay should be introduced in the script to make sure the cable box has really had time to switch channel when the recording starts.

The rest of the recording is handled exactly as for a "normal" recording.

Managing notifications mails

When configuring the daemon it is possible to set options that will send notification mails on important events in the daemon. This could be notifications on errors that occurred but also notifications on that a transcoding was finished or that the server was shutdown automatically while waiting for the next recording.

All mail settings are specified in the "`[mail]`" section of the configuration file.

Sending mails can be done by either using a specified SMTP outgoing server (in the configuration file) or by using the systems own "**mail**" command. However, if you chose the latter you cannot ask the server to send HTML formatted mails (by setting the configuration "`use_html=yes`"), only plain text mail with the systems built in "**mail**" command.

The actual text (and formatting in case of HTML mails) are controlled by templates that are stored in the general configuration directory. All templates are prefixed by "`mail_*`". The templates exists in both a "`*.txt`" and a "`*.html`" version. The text version is used when sending plain text messages and the HTML version when sending HTML formatted mails. In complying with internet mail standards the plain version is also included in the mail even when sending HTML formatted mails so that mail clients that are unable to display HTML format will use the plain text version instead. (HTML mails are formatted according to internet standard MIME attachment rules.)

At the time of this writing the following templates are used by the daemon

- `mail_shutdown.*` Used when sending notifications on automatic server shutdown
- `mail_transcend.*` Used when sending notifications when transcoding has finished
- `mail_error.*` Used when sending notifications on errors

In these templates you can see a number of keys (in the format of `[KEYNAME]`) with different names that are replaced with the actual values at the time the mail should be sent. The name of the keys should make it obvious what value they will be replaced with is.

If you like to change the appearance of any of these templates, then by all means, go ahead !

Other mail sending options

In addition to sending notification mail it is also possible to have the server send HTML formatted mails with various layouts of the list of upcoming recordings. These lists can either be formatted as a list of all upcoming individual recordings one by one or as a concise format where the repeating recordings are presented as one line. The repeat interval and the number of occurrences left are stated in the table. Using this format the mail will contain two tables; one with the repeating recordings and one table with the single recordings.

The style of the HTML (look & feel) tables used can be modified with CSS format statement which are stored in the config file under the **[format]** section. By default they have a rather strict and as some may think, boring, color schema.

Read more below regarding the '**lmr**' and '**lm**' commands.

Please also note that sending these mails in HTML format require the config option "`use_html`" to be set to "`yes`" as well as a configured outgoing SMTP server (in the mail config section of the configuration file)

Adding user "hooks" for special processing

To maintain maximum flexibility the daemon will check for the existence of several (optional) shell scripts that can be provided by the user to take special actions on server shutdown, server startup, channel switching etc. The package provides templates for all of them that can be tweaked and then stored in the configuration directory (by default `/etc/tvpvrd`). The available hooks are listed below with a note if they are mandatory or not. To use the hooks they must first be renamed by removing the "`*.template`" extension. For the majority of the these script the daemon will call them with optional arguments. The template code shows how to make use fo these arguments, for example the full filename of a video that has just been recorded.

After the list we give some examples on how these hooks can be used

- `postrec_processing.sh.template` , (*optional*) This script is called just after a recording has been done and before any (optional) transcoding is taking place. The usage (or not) of this script is controlled in the configuration `use_postrec_processing`

- `posttransc_processing.sh.template` , (*optional*) This script is called just after a transcoding has been finished. The usage (or not) of this script is controlled in the configuration `use_posttransc_processing`

- `shutdown.sh.template`, (*optional*) This script is called when automatic shutdown has been enabled (`shutdown::enable=yes`) , By default it will just do a full system shutdown. This script is mandatory if automatic shutdown has been enabled.

- `startup.sh.template` , (*optional*) This script is called right after the daemon has started but before any recordings are started.

- `switchchannel.sh.template`, (*optional*) This script is called when the channel should be changed. This is controlled in the configuration file (`external_switch=yes`) . See section above on "*Controlling an external cable box*" for more information on this particular script.

Some typical usages of these hooks are:

- Handling automatic mounting/unmounting of filesystems. If the server running the daemon have exported filesystems to allow clients to fetch the videos by either SMB or NFS mounting there will be problem if those filesystems are still mounted if the server initiates an automatic shutdown. NFS is notoriously bad in handling a server no longer available and it can cause hangs and freezes in the client when it waits indefinitely for the server to respond.

  This can be handled by allowing the recording server to execute mounting/unmounting scripts on clients (via ssh). To allow ssh execution without passwords the recording server must be allowed to access a client via ssh and using public keys without passwords. This means that the recording servers public key (`id_rsa.pub`) must be stored in the clients "`authorized_keys`" file (usually in the home directory under `.ssh/`)

In the shutdown script one has then to add calls to each client to unmount the filesystems and in the startup script one can call the clients to re-mount all the necessary filesystems.

- Some video cards require some postprocessing on the MP2 video before it is transcoded to MP4, This postprocessing can be done in the `postrec_processing.sh` script.

Just remember to rename the templates by removing the "`*.template`" extension so they can be found by the daemon.

Recording repeating series by seasons using "SnnEmm" naming

A common use case is to record a, say weekly, recurring series that continues for several seasons. One way of naming each episode in such a series is by giving the core series name a suffix of "`SnnEmm`" where "`nn`" is the season number and "`mm`" is the episode number.

Starting with version 4.3.0 tvpvrd make this type of naming schema easier with the existing repeating recording command ("**ar**").

The way to handle this is to first specify that the name mangling for each recording in the series should add "`Emm`" to the core name. By default the episode number starts at "01". This is done by the "**sm**" command as "**sm 2**". The correct season number "`Snn`" is manually included when the name of the series is given. The default name mangling to be used by the server is also specified in the configuration file so if this is already set to "2" then there is no need to give this command. By default in the distribution the name mangling is set to add the date as a suffix to each recordings name.

If the recording of the series is started in the middle of the season the starting episode number can be specified with the "**ss**" command.

Example

To record the second season of a program called "Nature" starting with episode 5 the following commands would have to be given:

```
sm 2
ss 5
ar w 20 tv1 tue 21:00 21:56 Nature S02
```

This will in the end create recordings that are named "Nature_S02E05","Nature_S02E06", and so on.

These types of recording can also be done from the Web-interface (perhaps even easier since there is no need to remember the exact command sequence).

Commands (in alphabetic order)

The following is a list and detailed explanation of the commands that are recognized by the server. Commands are specified by 1 to 3 letters followed by any necessary arguments. It is possible to get more detailed help/information for each command by giving the help command 'h' followed by the command, e.g. 'h l', will give more detailed help on the 'list' command.

Since there are quite a few commands we start with an overview with a one line explanation of each command. Further details can be found below

- **a** - add new recording command
- **ac** - adjust audio controls (treble,bass)
- **af** - add recordings from file

- **ar** - add new repeating series recording
- **d** - delete single recording
- **df** - display used disk space for data directory
- **dp** - display all settings for specified profile
- **dr** - delete all repeated recording
- **h** - help
- **ic** - Set image controls (saturation,hue,brightness,contrast)
- **kt** - kill all ongoing transcoding(s)
- **ktf** - set/unset kill transcoding flag at shutdown
- **l [n]** - list all pending recordings, possibly restricted to the next <n> recordings
- **[lf]** - list all defined frequency tables
- **lh [n]** - list all pending record in a shorter more human readable format
- **lm [n]** - list all pending record and send them in a mail (in HTML format)
- **lmr [n]** - list all pending repeating and single recordings and send them in a mail (in HTML format)
- **lc** - list all controls for the capture card
- **li [n]** - list all input sources available for the capture card
- **log <n>** -show the last n lines of the logfile
- **ls** - list all stations
- **lp** - list all profiles
- **lq <n>** - list queued transcodings for video card <n>
- **lr** - list all repeating and single recordings
- **mlg** - mail the log file
- **n** - list the immediate next recording on each video
- **o** - list the ongoing recording(s)
- **ot** - list the ongoing transcoding(s)
- **q** - quick recording, start the recording immediately and continue the specified time
- **rh** - record history (list last N made recordings)
- **rst** - reset all statistics
- **rp** - refresh transcoding profiles from file
- **s** - print server status
- **sm** - set default name mangling for repeating series recording
- **sp** - set transcoding profile for specified recording

- **ss** - set episode start number for repeating series recording
- **st** - print profile statistics
- **t** - print server time
- **tf** - transcode specified file
- **tl** - read list of videos to transcode from file
- **td** - transcode all videos in directory
- **u** - force update of database with recordings
- **v** - print version
- **va** - adjust volume
- **vc <n>** - print information on TV-Card <n>
- **wt** - list waiting transcodings
- **x** - view database (in XML format) with recordings
- **z** - display all settings from ini-file
- **! <n>** - cancel ongoing recording on video card <n>

The input and output has a format suitable for both human interactive use as well as a programatically interfacing the server in order to build a UI that interacts with the server.

'a' - **Add new recording command**

Add new recording. The date/time logic is fairly complete and will for example correctly handle date crossings when recording starts and ends on different sides on the midnight. For all recording formats both title and profile might be omitted. In such case the title will be constructed from the station/channel name and the start date/time.

If no end time is given the recording will have the default recording length as defined in the ini-file. In the distribution this is set to 0:59 hours.

For all variants the start and end time is specified as

hh [:*mm*[:*ss*]]

The maximum length for one recording is 4 hours in order to keep the file size manageable.

The primary formats are described below. The syntactic element used have the following meaning:

<*ch*> = Channel or station name

<*starttime*> = Time for recording start

<*endtime*> = Time for recording end

<*title*> = Title of recording. This will also be used as the filename of the resulting file. Note that the filename will always be converted to lower case.

<*@profile*> = Name of the profile(s) to be used to the recording. Up to 4 profiles can be specified for each recording.

The three main variants of the add command are:

a <ch> <starttime> [<title>] [<@profile]

Specification of only start time. The end time will match the default recording length as specified in the ini-file.

If the start time is after the current time the date will be assumed to be today. If the time is earlier than the current time the date is assumed to be tomorrow.

If no title is specified the title will be constructed from the station/channel name and the start date and time.

If no profile is specified then the default profile specified in the ini-file will be used.

a <ch> <starttime> <endtime> [<title>] [<@profile]

Specification of both start and end time.

If the start time is after the current time the date will be assumed to be today. If the time is earlier than the current time the date is assumed to be tomorrow.

If no title is specified the title will be constructed from the station/channel name and the start date and time.

If no profile is specified then the default profile specified in the ini-file will be used.

a <ch> <startdate> <starttime> <endtime> [<title>] [<@profile]

Full date and time specification. The date can be specified as either a numeric ordinary date (note the order!):

yyyy-mm-dd

Numerical date yyyy=year, mm=month [1-12],dd=day [1-31]

or as week day in the coming week relative from todays date

[*today*|*tomorrow*|*mon*|*tue*|*wed*|*thu*|*fri*|*sat*|*sun*]

Relative date from today. The weekday name refers to the coming 7 days. If the same day as the current day is given then this refers to 7 days in the future.

If the start time is after the current time the date will be assumed to be today. If the time is earlier than the current time the date is assumed to be tomorrow.

If no title is specified the title will be constructed from the station/channel name and the start date and time.

If no profile is specified then the default profile specified in the ini-file will be used.

Examples:

```
a bbc1 19:30
```

Start a recording at 19:30 on station BBC1 which will have the default duration, default title and profile.

```
a bbc1 tue 19:30 News
```

Start a recording at 19:30 on station BBC1 the coming Tuesday which will have the default duration with title 'News' and profile.

```
a bbc1 19:30 21:15 News
```

Start a recording at 19:30 which will last until 21:15 on BBC1 with title set to "News". The default profile will be used.

```
a ch4 wed 22:00 23:45 The Movie
```

Start a recording the coming Wednesday at 22:00 until 23:45 on CH4 with title "The Movie". The default profile will be used.

```
a bbc2 tomorrow 18 20 "World in Focus" @mobile
```

Start a recording tomorrow at 18:00 until 20:00 on bbc2 with title "World in Focus". Using the mobile profile.

```
a bbc2 tomorrow 18 20 "World in Focus" @mobile @normal
```

Start a recording tomorrow at 18:00 until 20:00 on bbc2 with title "World in Focus". Using both the mobile and the normal profile.

ac *[<video-card>] <b|t> <val>* - **Adjust audio controls**

Adjust audio controls 'treble' and 'bass'. The value range is [-50,50] where 0 being the "neutral" value. If the video card number is omitted the value will be set for all available cards.

Example:

```
ac 0 b 20
ac b 20
```

'**af** *<filename>*' - **Add recordings from file**

Add recordings (specified with either the '**a**' or the '**ar**' command) from a file. The file is simply a list with one command per line that is formatted exactly as if the command would have been entered in the teminal directly. The purpose of this is primarily to allow other programs to generate a list of future recordings that can be easily added to the daemon.

The filename specified must be give as a full path filename since the daemon has no concept of a current directory.

Example:

```
af /tmp/newrecs.txt
```

The file `/tmp/newrecs.txt` could then have the following lines (for example)

```
a ch4 21:00 21:59 Child of our time @thigh
a bbc1 20:00 20:59 Walking with dinosaurs
a bbb1 21:00 21:29 lastnews @tlow
```

'**ar**' - **Add repeating series recording command**

Add a new repeated recording . The syntax is identical to the normal add command with the exception of the first two arguments which indicates the type of recurrence and the number of repeated occurrences or the end date. The full syntax is

ar <repeat-type> <repeat-count | end-date> <normal add arguments>

repeat-type

This specifies how often the repetition should be done. This can be specified as one of the following arguments:

'`d`' - Repeat daily (can also be specified as '1')

'`w`' - Repeat weekly (can also be specified as '2')

'`m`' - Repeat monthly (can also be specified as '3')

'`f`' - Repeat Monday to Friday (can also be specified as '4')

'`s`' - Repeat every Saturday to Sunday (can also be specified as '5')

'`t`' - Repeat every Monday to Thursday (can also be specified as '6')

'`n`' - Repeat every Tuesday to Friday (can also be specified as '7')

'`e`' - Repeat every Wednesday to Friday (can also be specified as '8')

'`i`' - Repeat every Tuesday to Thursday (can also be specified as '9') *repeat-cnt* | *end-date*

This specifies the number of repeated recordings by either specifying the number of repeats (as an integer) or by specifying the end date. If an end date is specified this includes the full final day up to 23:59:59

Since only one title is given for multiple recording the titles will be mangled with the recording date and time. In the recording list the title will usually be shown as "Base title (xx/yy)" where xx is the sequence number and yy is the total number of recordings in this sequence. This is controlled by the mangling parameter.

By default a series of recordings specified with this command is stored under a subdirectory with the same name as the basename of the recording. If you instead prefer to have all recordings stored in a flat structure directly under the recordings directory then set the config setting (in the config file) '`use_repeat_rec_basedir`' to '`no`'.

Examples:

```
ar f 10 bbc1 18 18:30 News
```

Start recording every Mon-Tue at BBC1 between 18:00 and 18:30

```
ar w 20 tv1 tue 21:15 22:10 John Adams
```

Start a recording once a week on Tuesday evening at 21:15 until 22:10 with base title "John Adams".

```
ar s 10 BBC1 01:00 02:30 Night movie
```

Add a recording every weekend night between 01:00 and 02:30 with the base title "Night movie". The recording will be done for the next 5 weekends (2 recordings each weekend).

```
ar t 2010-05-10 channel3 19:00 19:59 House
```

This will make recordings every Mon-Thur at 19:00 until the 10:th of May 2010

'**d** *nnn*' - **Delete specified recording**

Delete specified recording. *nnn* is the recoding id shown when a recording is added or by the list command. If the identified recording is part of a recurrent sequence the other recordings in the same sequence are untouched.

'**df**' - **Display total and used disk space**

Displays in both absolute and relative terms the amount of disk space on the partition containing the specified data directory (as specified in the config file). The size is given in "human" readable format which means the use of M = Megabyte, G = Gigabyte and T = Terabyte.

Output example:

```
612G of 4,1T (15 percent used)
```

'**dp** [*@profile*]' - Display profile information command

Display information on the settings of the specified profile.

Output example:

```
name                  : normal
ENCODER:
video_bitrate         : 3700000
video_peak_bitrate    : 4700000
audio_sampling        : 44.1
audio_bitrate         : 192
aspect                : 4x3
size                  : pal
FFMPEG:
```

```
use_transcoding      : 0
video_bitrate        : 800
video_peak_bitrate   : 1100
vcodec               : libx264
vpre                 : medium
pass                 : 1
acodec               : libfaac
audio_bitrate        : 128
video_size           :
crop                 : l=0, r=0, t=0, b=0
ffmpeg_extra_options :
file_extension       : .mp4
```

'**dr** *nnn*' - **Delete all repeated recordings**

Delete all repeated recording that are part of the same repeating sequence as the id of the recording specified.

'**exit - Disconnect from the server**

This command terminates the client connection to the server. It will not affect any of the ongoing recordings or transcodings.

'**h**' - **Help command**

Print a summary of all commands available.

ic *[<video-card>] <s|c|h|b> <val>* - **Adjust video image controls**

Adjust image controls 'saturation', 'contrast', 'hue', 'brightness'. The value range is [-50,50] with 0 being the 'neutral' value. If the video card number is omitted the value will be set for all available cards.

Example:

```
ic 0 c 10
ic 0 b -20
```

'**kt**' *n* - **Kill all ongoing transcoding command**

Stop specified ongoing transcoding processes. The index of the ongoing transcoding processes can be found via the **ot** command. If no specific index is given then all ongoing transcodings will be stopped so be careful !

'**ktf** [*y*|*n*]' - **Kill transcoding flag set/unset command**

The kill transcoding flag determines whether or not the transcoding processes should be killed when the server is stopped. By default all transcoding processes are killed when the tvpvrd server is killed. By setting this flag to 'n' (No) the transcoding processes will be allowed to keep running after the server is stopped.

Output example:

```
ktf n
killflag=n
```

'**l**' *n* - **List future recordings command**

List coming recordings. If *n* is specified as number in the range [1,99] then the list will be limited to *n* recordings. Otherwise all pending recordings will be listed.

Each record is listed with 7 fields. Each record starts with a '[' and ends with a ']' character. Each field is separated with a bar '|'. The order of fields are

```
<id#> <station/channel name> <start-date> <start-time> <end-time> <title> <profile>
```

Output example:

```
[160|tv4    |2010-03-09|21:00|21:59|House (16/20)                  |@normal]
[135|kanal5 |2010-03-11|21:00|21:59|Criminal minds (16/20)         |@normal]
[038|kanal5 |2010-03-11|22:00|22:59|Fringe (17/20)                 |@normal]
[088|kanal5 |2010-03-14|22:25|23:19|Supernatural (18/21)           |@normal]
```

```
[161|tv4    |2010-03-16|21:00|21:59|House (17/20)                  |@normal]
[136|kanal5 |2010-03-18|21:00|21:59|Criminal minds (17/20)         |@normal]
[039|kanal5 |2010-03-18|22:00|22:59|Fringe (18/20)                 |@normal]
```

'lf' - list all defined frequency tables

List all defined frequency tables.

Output example:

```
01 europe-west
02 europe-east
03 france
04 ireland
05 italy
06 australia
07 newzealand
08 us-bcast
09 us-cable
10 us-cable-hrc
11 us-cable-irc
12 japan-bcast
13 japan-cable
14 china-bcast
15 southafrica
16 argentina
17 australia-optus
```

'**lh**' [*n*] - **List future recordings in human readable format**

This is the same command as 'l' but with the output shortened and ore easily read by humans

Output example:

```
Fri Oct 01 21:00-21:55 disc     Mountains
Fri Oct 01 22:05-23:15 bbc1     Changes
```

'**lm**' [*n*] - **List future recordings and send them in a mail**

This command will create a mail (in HTML) format and send them to the mail address already specified in the config file. The mail will be formatted as both a HTML mail as well as in plain text format. This will allow mail clients both with and without HTML rendering capability to read the mail. This command assumes the setup if the SMTP mail client and will return an error of this has not been done.

(Note: The format style of the HTML table is controlled by CSS specified in the tvpvrd config file.)

'**lmr**' [*n*] - **List future repeating and single recordings and send them in a mail**

This command will send a mail with two HTML lists. One with repeating recordings and the second list consisting of all single recordings, i.e. all recordings not included in the repeats.

(Note: The format style of the HTML table is controlled by CSS specified in the tvpvrd config file.)

'**log**' [*n*] - **Show last n lines of the log file**

If the logfile has been specified as a file (in `tvpvrd.conf`) then this command will show the last n lines of the log. If no number is given it will default to show the last 10 lines.

li [*n*] - **List all video input sources command**

List a all video sources available for the specified card. The current selected input source is marked with an asterisk. The type of input is shown in parenthesis in front of the textual name of the input. The type is either "tuner" or "camera". If no card is specified then information on all installed card will be listed.

Output example:

```
Hauppauge WinTV PVR-350
```

```
00 :*(tuner) Tuner 1
01 : (camera) S-Video 1
02 : (camera) Composite 1
03 : (camera) S-Video 2
04 : (camera) Composite 2
05 : (camera) Composite 3
```

'**lp**' - **List profiles command**

Print a list of all defined transcoding profiles numbered one per line(use '**dp**' command to display details of a specified profile)

Output example:

```
#01 : mobile
#02 : low
#03 : cropnormal
#04 : tnormal
#05 : high
#06 : normal
#07 : wnormal
```

'**ls**' - **List stations command**

List all defined station names. The stations are specified in the `xawtvrc` file and maps station names to channel names. When defining the frequency for a recording either the defined station name or channel (frequency) name can be used.

Output example:

```
SE11: kanal5
  57: kanal8
 S36: kanal9
SE16: mtv
 E10: svt24
SE13: svtb/kunsk
  E5: tv1
  E7: tv2
 E11: tv3
  E6: tv4
SE14: tv4+
  E8: tv6
```

'**lc** [*#card*]' - **List all video capture card controls command**

This can be considered a low level command of limited use for an end user. This command will list all user controllable video/audio controls that can be adjusted on the specified video card. This is directly related to the discovered 'v4l2' interfaces. Since the output is quite long we do not include any example output here. The output is intentionally modeled to be of the same format as the one used in 'v4l2' output.

(This command is probably of little use for anything else than troubleshooting a video card and driver.)

'**lq** [*#filelist*]' - **List information on the ongoing transcoding of file list #filelist**

List some useful information on the specified transcoding file list. The transcoding file list is a list with filenames that are queued up for transcoding. A transcoding file list is added with the **tr** command. Up to five simultaneous lists may be added. However, since transcoding is such a heavy job there is little real reason to add more than one lists at a time since a list with, say 20 full length movie video files, can easily take up more than two days to transcode.

Output example:

```
       Filelist: #01
          Total: 17 videos in list
```

```
      Processed: 02 (11%) files
    Job started: Mon Jan 25 16:29:53 2010
   Running time: 00 days 00:14 h
 Est. time left: -1 days ~-1 h (approx.)

Processed:
  * /data/pvr/mp2/normal/svu_001.mpg
  * /data/pvr/mp2/normal/svu_2010-01-20_00.40.mpg

Not processed:
  + /data/pvr/mp2/normal/svu_2010-01-21_00.40.mpg
  + /data/pvr/mp2/normal/svu_2010-01-22_00.40.mpg
  + /data/pvr/mp2/normal/svu2ggr.mpg
  + /data/pvr/mp2/normal/svu_k9_001.mpg
  + /data/pvr/mp2/normal/svu-k9_2010-01-21_00.05.mpg
  + /data/pvr/mp2/normal/svu-k9_2010-01-22_00.05.mpg
  + /data/pvr/mp2/normal/svu.mpg
  + /data/pvr/mp2/normal/two_and_half_men_001.mpg
  + /data/pvr/mp2/normal/two_and_half_men_002.mpg
  + /data/pvr/mp2/normal/two_and_half_men.mpg
  + /data/pvr/mp2/normal/världens_undergång.mpg
  + /data/pvr/mp2/normal/vita_huset_001.mpg
  + /data/pvr/mp2/normal/vita_huset.mpg
  + /data/pvr/mp2/normal/white_noise.mpg
  + /data/pvr/mp2/normal/WildThings.mpg
```

The fist set of information gives general information about the chosen file list. The last line of the first information block gives a rough estimate of the remaining time left needed to transcode the list. This is a very approximate estimation based on the time it has taken to transcode the files in the list up to now. In the example above the estimate is shown as -1 indicating that there are too few completed transcodings for any estimate to be done. At least 2 transcoding must have been completed in order for the estimate to be shown.

The next set of information is the "`Processed:`" files. This is the files that have been processed, meaning it has been submitted for transcoding. This means that they have either been completely transcoded or is in the process of being transcoded.

The final block of information, the "`Not processed:`" block is the remainder of the files which are still waiting to be processed. No files are submitted for processing until the system load has fallen below the maximum limit for transcoding to start as specified in the `tvpvrd.conf` file.

'**lr**' - **List repeating and single recordings**

This command will print a concise list with repeating recordings in one table and single recordings in a separate table. The date given for the repeating recordings will be the date for the next recording. See also '**lmr**' command which will mail a HTML version of these tables.

Output example:

```
Generated by: tvpvrd 3.0.5, Thu Mar 10 19:44:39 2011
====================================================================================
#   Ch      Date       Start End   Repeat   Next/Tot Title                     Profil
====================================================================================
001 ch12    Thu Mar 10 23:35 00:10 Mon-Thu  028/102  Mash                      @tnorm
002 ch12    Fri Mar 11 03:00 04:05 Tue-Fri  008/102  Amy                       @tnorm

Generated by: tvpvrd 3.0.5, Thu Mar 10 19:44:39 2011
====================================================================================
#   Ch      Date       Start End   Title                                       Profil
====================================================================================
                        (No recordings)
```

'**mlg**' - **Mail the log file as an attachment**

Mail the entire current logfile as an attachment. The address used is the system address specified in the config file.

'**n**' - **Next recording command**

List the immediate next recording(s) scheduled for the available video card(s). The first number in parenthesis is how long time until the recording will start.

Output example:

```
(01:37) : [002|tv3    |2009-11-20|18:00|18:59|NCIS_18 (02/20)               |@normal
(52:37) : [143|tv4    |2009-11-22|21:00|22:00|Shooter 1                     |@normal
```

'**o**' - **Ongoing recordings command**

List all ongoing recording in ordinary list format for all video cards. If no recordings are ongoing the output will show "None"

Output example:

```
Video #0: [175|tv1    |2009-11-20|16:26|16:27|tv1_20091120_1626             |@normal
Video #1: None.
```

'**ot**' - **Ongoing transcodings command**

List all ongoing transcodings. The output format is

```
<#id> <start-time> <(duration)> <filename> <profile>
```

The start time identifies when the transcoding was started and duration shows for how long the transcoding has been running.

Output example:

```
[#00|16:27|(00:00)|tv1_20091120_1626.mpg              |normal]
```

'**otl**' - **Ongoing transcoding long command**

List all ongoing transcoding in long format which includes the detailed 'ffmpeg' command used which is printed on the second line.

Output example:

```
[#00|16:27|(00:00)|tv1_20091120_1626.mpg              |normal]
(cmd: /usr/bin/ffmpeg -v 0 -i tv1_20091120_1626.mpg -threads 0
-vcodec libx264 -vpre normal -b 700k -bt 1000k  -croptop 8 -cropbottom 8
-cropleft 2 -cropright 2  -acodec copy -ab 196k  -y
tv1_20091120_1626.mp4 > /dev/null 2>&1)
```

'**q**' - **Quick recording command**

Add new recording that will start immediately. This is basically the same as a simplified '**a**' command where the start time and start date is always the current date and time.

The format of the command are

q <ch> [<duration>] [<title>] [<@profile]

The recording will start within a few seconds (depending on the resolution time specified in the ini-file) and will last for the duration specified. If no duration is specified the default duration time will be used. The duration is specified with hours and minutes as

hh:mm

If no title is specified the title will be constructed from the station/channel name and the start date and time.

If no profile is specified then the default profile specified in the ini-file will be used.

Examples:

```
q bbc2
```

Start a recording immediately on bbc2 using the default duration, title and profile.

```
q bbc2 1:30
```

Start a recording immediately on bbc2 with duration 1 hour 30 minutes, using default title and default profile.

```
q bbc2 News
```

Start a recording immediately on bbc2 with default duration using the title "News" and default profile.

```
q ch4 1:45 Newsfocus @high
```

Start a new recording on station 'ch4' with title 'Newsfocus' using the 'high' profile.

'**rh**' - **Record history. Show a list of previous made recordings**

List previously made recodings/transcodings with some basci information. There is a hard limit so that a maximum of 20 past recordings are shown in the list. This is a compile time setting. (The ini-file already have too many settings...)

Output example:

The columns in the listing are *<startdate> <start time> <title> <filepath> <profile>*

```
Thu Jun 09 21:55 Screen at two   /mnt/omega/mm/pvr/mp4/screen_at_two.mp4       tnorm
Tue Jun 07 19:30 Departed mems   /mnt/omega/mm/pvr/mp4/departed_mems.mp4       tnorm
Wed Jun 01 00:50 Once more       /mnt/omega/mm/pvr/mp4/once_more.mp4           tnorm
```

'**rhm**' - **Mail the history list of recordings**

The history list (sam as from command '**rh**') is mailed to the predefined system address in the configuration file

'**rst**' - **Reset statistics command**

Reset all statistics, i.e. set all statistics to 0.

'**rp**' - **Refresh profiles command**

This command will cause the profiles to be re-read from profile files. This can be used to refresh the profiles without having to restart the server.

'**sm** *n*' - **Set default name mangling type**

Set default name mangling type for series recording

Supported types:

```
0 = Add date, e.g. "myrec_2012-11-22"
1 = Number and total number of recordings, e.g. "myrec_02-12"
2 = Add episode number 'Enn' to the end of the name usefull for "SxxEyy" style namin
```

See '**ss**' command for controlling the episode numbering.

'**sp** *nnn @profile*' - **Set profile command**

Specify the profiles to be used on the specified recording. This will replace any previous defined profiles for this recording. Up to four profiles per recording can be defined.

Examples:

```
sp 87 @normal @mobile
```

Use profiles *normal* and *mobile* for recording *#87* '**ss** *nn*' - Set episode start number

Specify the start episode number used for specifying title and file name for the recording. Se the '**sm**' command for example on how the episode number is used.

'**s**' - **Server status command**

Print a summary of the server status. This includes information on when the tvpvrd was started, how long it has been running for, the current server load, current server time, allocated virtual memory, number of running threads in the server and finally a list of clients currently connected to the server.

Output example:

```
Current time    : Sun Oct 10 09:48:24 2010
tvpvrd started  : Fri Oct  8 08:53:00 2010
tvpvrd uptime   : 02 days 00 hours 55 min
Server uptime   : 03 days 02 hours 37 min
Server load     : 0.1 0.1 0.0
Virtual memory  : 56796 kB
Threads         : 4
Clients         : #01: 192.168.0.31, Sun Oct 10 09:48:24 2010
```

'**st**' - **Print profile statistics**

Print transcoding statistics for the defined profiles. This statistics is gathered for every transcoding made. The statistics includes among other things the running time and average size per minute of the original MP2 and the transcoded file. The statistics for the transcoding can be used to predict how long time a transcoding will take. However since it is highly dependent on the load on the server it can be difficult to accurately predict the needed transcoding time.

Output example:

```
profile_name            : 'normal'
transcoding_speed       : 44 s transcoded / min
mp2size_1min            : 27.5 Mb/min
mp4size_1min            : 6.4 Mb/min
comp_ratio              : 4.3
total_ttime             : 3481 min
total_mp2time           : 1779 min
total_mp2files          : 31
total_mp4files          : 31
```

The different fields have the following meaning

profile_name

The name of the profile that this statistics apply to.

transcoding_speed

How much play time the server on average can transcode per minute run time of 'ffmpeg'.

mp2size_1min

The average size of 1 minute play time in the original MP2 file.

mp4size_1min

The average size of 1 minute of play time in the resulting transcoded MP4 size.

comp_ratio

The compression ration between the original MP2 size and the MP4 transcoded file.

total_ttime

Total running time that has been spent transcoding files.

total_mp2time

Total recorded original MP2 time.

total_mp2files

Total number of mp2files recorded with this profile.

total_mp4files

Total number of transcoded MP4 files with this profile.

't' - Current Time and Date command

Print current time and date on the server where 'tvpvrd' is running

Output example:

```
Fri Nov 20 15:51:20 2009
```

'tf *filename* [*@profile*]' - **Transcode specified file using the named optional profile**

This command is used to transcode any file within the file system using a specified transcoding profile. If the profile name is not given the the system default transcoding profile will be used.

filename is the full path to the (normally MP2) video file that should be encoded. The resulting transcoded file is stored in the `<datadir>/mp4/<profile>/` directory. During the transcoding a temporary directory under `<datadir>/vtmp/` is created and used as a working directory. The original file is not chnaged in any ways. During the transcoding a symbolic link is created in the temporary directory which makes it in theory possible to have the original file on another file system.

Example:

```
tf /data/pvr/mp2/normal/oldreccording.mpg @high
```

'tl *filename* [*@profile*]' - **Queue all files listed in file filename for transcoding using the named optional profile**

This commands will queue all (video) files listed in the file specified in the command for transcoding.

The list file consists of a number of lines (each line ending with a newline) with either

a fully specified video file name or

a filepath which will be used for the following files as prefix until the next filepath is specified. A filepath is identified by having the first character on the row being a ':' (colon) character.

Example input file:

```
:/data/pvr/mp2/normal
svu_001.mpg
svu_2010-01-20_00.40.mpg
svu_2010-01-21_00.40.mpg
svu_2010-01-22_00.40.mpg
svu2ggr.mpg
svu_k9_001.mpg
svu-k9_2010-01-21_00.05.mpg
svu-k9_2010-01-22_00.05.mpg
svu.mpg
two_and_half_men_001.mpg
two_and_half_men_002.mpg
two_and_half_men.mpg
vita_huset_001.mpg
vita_huset.mpg
white_noise.mpg
WildThings.mpg
```

In the above file all the video files will be read from the directory `/data/pvr/mp2/normal`. The result of the transcoding will (as usual) be stored under the corresponding mp4 profile directory. Up to 5 different file lists can be added simultaneous. There is no guarantee that files from all queued file lists will be fairly processed, i.e. one file from the first list, and when that is done, a file from the second file lists and so on. This is completely random. The only thing that is guaranteed is that the video files in each file lists will be processed sequentially.

Remember that transcoding multiple files can take a long time. For example a file list with 10 video files can easily take ~2 days (~4 hours transcoding time per video).

To view the status of ongoing list transcodings the "**lq**" (list-queue) command can be used.

'**td** *directory* [*@profile*]' - **Transcode all files in the specified directory**

This command will search the specified directory for files with recognized video suffix and submit all found files to transcoding.

'**u**' - **Update database file command**

Force update of the database file from memory. Normally this command is not necessary since in versions above v1.2 of tvpvrd the database is automatically updated whenever a new recording is added or removed. This command is kept for backwards compatibility but should not be used.

'**v**' - **Version command**

Print program version and name as well as the build date. The following lines will give information on how the server was compiled.

Output example:

```
tvpvrd 1.1.2 [client] (Jan 26 2010)
Compiled with Large File Support (files > 2GB).
```

va *[<video-card>] value*- **Adjust volume**

Adjust volume on video card. Valid volume is in the range [0,100]. If the video card number is omitted the value will be set for all available cards.

Example:

```
va 0 85
va 50
```

'**vc** [*0*|*1*|*2*|*3*|*4*]' - **Video Card information command**

Print information on the specified video/capture cards. The command will print information of all available cards if no specific card number is specified. The information shown is the name of the card, driver and driver version.

Output example:

```
Card 00: Hauppauge WinTV PVR-350, driver=ivtv v1.4.1
Card 01: Hauppauge WinTV PVR-150, driver=ivtv v1.4.1
```

'**wt**' - **List all transcoding waiting to start**

A transcoding is not started until the server load is below the specified treshold in the ini-file. By default this treshold is 4. This command will generate a list of all transcoding that are waiting..

The listing uses the same format as the **ot** command and contains (in order from left to right)

1. Sequence number

2. Time when item was enqueued

3. How long (hh:mm) the item has been waiting to be transcoded

4. Name of video file to be transcoded

5. Profile name to be used for the transcoding

Note: There is no guarantee that the files will be transcoded in the order indicated by the list..

Output example:

```
[#01|21:37|(00:04)|news_at_eight.mpg                          |@tnormal]
[#02|21:38|(00:03)|commander.mpg                              |@tnormal]
```

'x' - View XML database command

View the current specified recordings in raw XML DB format. See tvpvrd(5) for details about the format.

'z' - Display all ini-file settings command

This will list all settings made in the ini-file as well as the location of the used ini file.

Output example:

```
datadir                       : /data/pvr
use_profile_directories       : 0
use_repeat_rec_basedir        : 1
logfile                       : /tmp/tvpvrd.log
verbose_log                   : 3
inifile_name                  : /etc/tvpvrd/tvpvrd.conf
sendmail_on_error             : 1
sendmail_on_transcode_end     : 1
sendmail_on_shutdown          : 1
sendmail_address              : johan162@gmail.com
locale_name                   : sv_SE.UTF8
smtp_use                      : 1
use_html                      : 1
smtp_server                   : 192.168.0.31
smtp_user                     : ljp
daemonize                     : 1
enable_webinterface           : 1
require_web_password          : 1
web_user                      : ljp
web_password                  : samba23
weblogin_timeout              : 0
username                      : root
xmldbfile_name                : /data/pvr/xmldb/tvpvrd_db.xml
max_video                     : 2
max_entries                   : 1024
max_clients                   : 4
client_idle_time              : 1800
port                          : 9300
time_resolution               : 3
video_buffer_size             : 307200 (0.3MB)
default_recording_time        : 00:59 (h:min)
xawtv_station file            : /etc/X11/xawtvrc
default_profile               : tnormal
frequency_map                 : europe-west
allow_profiles_adj_encoder    : 1
external_switch               : 0
external_input                : 1
external_switch_script        : switchchannel.sh
use_postrec_processing        : 1
postrec_processing_script     : postrec_processing.sh
shutdown_enable               : 1
shutdown_min_time             : 180
shutdown_ignore_users         : 0
shutdown_min_uptime           : 180
```

```
shutdown_pre_startup_time     : 180
shutdown_script_name          : shutdown.sh
```

'**!** [*#videocard*]' - Stop recording on specified video card

Stop the recording on the specified video card. The aborted recording will be left in the '<data-dir>/vtmp/vid<n>/' directory and must be manually removed.

Files

Note: The system configuration directory to be used is a compile time settings and specified with the '*configuration*' argument '--sysconfdir'.

/etc/X11/xawtvrc

A list in ini-file format which maps the commonly known station name to a frequency channel. This is the default location used by the package xawtv

<log-directory>/tvpvrd.log

The logfile. The log-directory is specified in the ini-file. It can also be specified as a command line argument (`-l, --logfile`) which will then override the ini-file setting. By default the log file will be stored under `/tmp/tvpvrd.log` .(The reason not to use `/var/log/tvpvrd.log` is the fact that the server will not normally run as root and hence will not have write access to this directory.)

/var/run/tvpvrd.pid

The pid of the started 'tvpvrd' server. This is used to easily find the pid in order to shutdown the server. This is used by the *stop-tvpvrd* script. Note that in order to use this directory the daemon has to be started by the root user.

/etc/tvpvrd/tvpvrd.conf

Default configuration file

/etc/tvpvrd/profiles/

Directory with all profiles

<data-directory>/

Directory where all recordings and temporary files are stored. The root directory is specified in the configuration file.

<data-directory>/vtmp

Video temporary directory. Used to store ongoing recordings and as working directories while doing transcoding

<data-directory>/mp2

If the profile specifies that the original mp2 files should be kept they are stored in this directory.

<data-directory>/mp4

The transcoded files are stored under this folder in a subfolder with the name of the profile used for the transcoding.

<data-directory>/xmldb

The database files are stored under this directory (future recordings and history). The format is structured XML.

<data-directory>/stats

The statistics for each profile is stored in a file in XML format with the same name as the profile and with the suffix '.stats'.

Limitations

- Numerical dates must be specified in European format, yyyy-mm-dd

- Not all international defined frequency maps are included. Supportedc maps can be found by the "**lf**" command.

  Frequency maps are a simple vector defining the frequency as an integer and a string that defines the channel name. This is a compile time setting and adding new maps will require recompilation of the server.

- The `/var/run/tvpvrd.pid` file is not removed after the program has shut down if the effective user is any other than 'root', (by default the server will switch to the 'tvpvrd' user) since only root has access to `/var/run`

- Maximum recording time is 4 hours (in order to keep maximum MPEG2 file size manageable). This can result in up to ~8GB MPEG2 files depending on the HW bitrate used.

- All information, error messages and loggings are hard coded in English.

- The server assumes that all installed capture cards uses the same index to specify the tuner as input among all the possible input sources (as specified by the configuration file setting *tuner_input_index*)

Bugs

- The configuration file has too many settings.

- Theoretically if the server is extremely busy some newer recorded files could be transcoded before older recordings. This is due to the fact that there is not a central queue for the transcodings. Each recording just backs off and waits a (random) time until it tries to see if it can start the transcoding (when the server load is below the set threshold).

- The '**i**' command is not yet implemented.

Notes

Any errors that are discovered in the startup phase before a logfile has been established will be written to the default system log (normally `/var/log/messages`)

The formal grammar for the XML based DB file is specified as an RNG (Relax New Grammar) XML specification included in the distribution in the docs directory.

Resources

- SourceForge: <http://sourceforge.net/projects/tvpvrd/>

- ivtv-driver: <http://ivtvdriver.org/>

- Video for Linux: <http://linux.bytesex.org/v4l2/>

Author

Written and maintained by Johan Persson <johan162@gmail.com>

Copyright

Free use of this software is granted under the terms of the GNU General Public License (GPL v3)., see the file COPYING included in the distribution archive for details.

See also

tvpsh(1), tvpowerd(1), ffmpeg(1), scantv(1), v4l-info(1), v4l-conf(1), v4lctl(1), xawtv(1)