kmotion

v1.1xb Documentation
Last updated 08/04/2008

kmotion documentation

The following documentation provides a detailed breakdown of the internals of kmotion. It is aimed primarily at developers.

1:0 Installing kmotion

1:1 Server requirements ...

kmotion has been developed and tested on Ubuntu 7.10. Although kmotion has modest requirement your mileage may vary on different distros. Summary of the key server requirements.

Disk space : min 5 GB

The amount of disk space that kmotion uses can be adjusted. See the FAQ

1:2 Using the automated install script (ubuntu 7.10 only) ...

Download the tarball and unzip it. Decide where you want the kmotion directory to live and move it there. kmotion can be installed almost anywhere but $HOME is as good as any.

cd kmotion
sudo ./install.py
.....
<press enter to start the install>
.....

And with any luck kmotion has installed. If everything is showing green [OK]'s go to the 'configuring kmotion' section.

1:3 Manual installation ...

So you need do do it the hard way huh ?

Download the tarball and unzip it. Decide where you want the kmotion directory to live and move it there. kmotion can be installed almost anywhere but $HOME is as good as any.

cd kmotion/daemons
./install_int.py
lpr daemon.rc

'install_int.py' sets the correct paths for the internal directories, generates a 'kmotion_vhost' file with the correct paths and generates 'kmotion' and 'kmotion_reload' with the correct paths. If you can print the daemons.rc it will be usefull for later reference.

now install 'motion', 'apache2' and 'apache2 PHP'
and its a good idea to also install 'ntp' & 'screen'

as root ...
ln -s <apache2_config_dir from daemon.rc>/kmotion_vhost /etc/apache2/<vhost directory>
/etc/init.d/apache2 restart

mv kmotion /usr/bin/
mv kmotion_reload /usr/bin

'kmotion' and 'kmotion_reload' are 'bin' files that have the correct paths to access the kmotion

daemons. They were generated by 'install_int.py'.

Add the following line to the startup file for your distro eg /etc/rc.local
'sudo -u <your login name> kmotion &'

This starts kmotion on a reboot.

And with any luck kmotion has been installed. If everything went smoothly go to the 'configuring kmotion' section.

2:0 Configuring kmotion

kmotion uses the same configuration file as motion i.e. 'motion.conf'. 95% of all configuration for kmotion is the same as motion with a few exceptions.

In order for kmotion to display live jpegs from motion and offer event playback motion has to be configured in a certain way.

(1) Firstly motion.conf must be configured in thread mode even if you only have one camera.

(2) Secondly kmotion has to override five motion.conf options. Namely: jpeg_filename, snapshot_filename, on_event_start, on_event_end, in_picture_save and target_dir.

(3) Last an extra option. To get kmotion to display the cameras description use the kmotion option '#ktext <text>' inside 'motion.conf' or its threads. So if camera 3 is of a garage, in thread 3 you would add the line:

#ktext messy garage

So now you know the restrictions, go configure motion.conf as you normally would e.g. setting 'snapshot_interval' and 'framerate'. kmotion looks for 'motion.conf' in the same places as motion looks i.e. ./motion.conf, /etc/motion/motion.conf, ~/.motion/motion.conf
and /usr/local/etc/motion.conf

Then cd to the kmotion directoy edit 'kmotion.rc'. Look for the subsection '[www]' and the option 'www = 2.0'. This option changes the web interface style, it can be either '2.0' or 'classic'.

When done do not start motion just type 'kmotion', it in turns starts 'motion'. If you change 'motion.conf' type 'kmotion_reload' to get kmotion to reconfigure.

Restart apache2, normally with '/etc/init.d/apache2 restart' to get it to see kmotions website.

Finally point your browser (hopefully Firefox) to 'http://localhost:8085' or 'http://xx.xx.xx.xx:8085' the default username is 'kmotion' the default password is 'kmotion' and have fun !

3:0 FAQ

3:1 'Its not working are there any diagnostic tools ?'

The best diagnostic tool is 'tail -f /var/log/motion'. In 95% of cases the problem is an incorrectly configured 'motion.conf'. To increase the verbosity of kmotion logging change the reporting level.

In addition cd to kmotion/daemons. Execute 'daemon_manual.py'. This gives you manual control and status information of the daemons.

3:2 'How do I change the default port from 8085 ?'

cd to the kmotion directory, edit 'kmotion.rc'. Look for the subsection '[misc]' and the option 'port = 8085'. Change the port number then execute 'kmotion_reload'. You now need to restart apache2 with 'sudo /etc/init.d/apache2 restart'.

3:3 'How do I change the default user name and password ?'

cd to the kmotion/apace2_config. The passwords are held in the 'users_digest' file. To change or add passwords see 'man htpasswd'. When you have changed the password you need to restart apache2 with 'sudo /etc/init.d/apache2 restart'.

3:4 'I don't want to log in on my LAN, do I have to ?'

No you don't. Simply save 'http://<user name>@<password>:<xx.xx.xx.xx>:8085' as a bookmark, when you return to the bookmark your browser will no longer request a login.

3:5 'How can I change where the image database is situated ?'

cd to the kmotion directoy edit 'kmotion.rc'. Look for the subsection '[dirs]' and the option 'images_dir = .../kmotion/images'. Change the path to point to your desired location. Then execute 'kmotion_reload'

3:6 'How can I change the maximum size of the images database ?'

cd to the kmotion director edit 'kmotion.rc'. Look for the subsection '[storage]' and the option 'max_size_gb = 5'. Change this value to your new maximum size in GB. When the image database grows to 90% of this value the oldest dated jpegs are deleted. Finally execute 'kmotion_reload'

3:7 'How can I change where directory X is situated ?'

In exactly the same way as you change where the image database is situated however if you move the daemons directory you will have to re 'sudo install.py'.

3:8 'How can I modify the kmotion_vhost file ?'

The 'kmotion_vhost' file is generated by kmotion from the 'kmotion_vhost_template' file. cd kmotion/apache2_config and edit 'kmotion_vhost_template' respecting %xxxx% strings. These strings are expanded to the correct paths when 'kmotion_vhost' is generated by 'install_int.py'. Execute 'kmotion_reload' and finally restart apache by executing 'sudo /etc/init.d/apache2 restart'.

3:9 'What is in the misc_config directory ?'

The 'misc_config' directory contains configs and files generated and used by kmotion. Do not change anything in this directory.

3:10 'Why are you not using plugin X to show video ?'

In order for kmotion to be viewable on any browser anywhere in the world it must rely on nothing more exotic than Javascript. Locked down cyber cafes will not let you execute a .jar file !

This being the case kmotion requires motion to save its recordings as a sequence of jpegs rather than video. This has two benefits, firstly any browser anywhere can view the images and secondly kmotion archive viewer can playback events at variable speed and in both directions.

3:11 'This is brilliant can I pay you some money ?'

Certainly :)

4:0 The Python back end

kmotion uses two python daemons as a back end to perform housekeeping duties, they are called 'kmotion_hkd1.py' and 'kmotion_hkd2.py'. The hkd part standing for House Keeping Daemon !

4:1 'kmotion_hkd1.py'

'kmotion_hkd1.py' is the master daemon running on a 15 minutes cycle. It calculates the size of today's jpegs in the images database and updates the 'dir_size' file for today. It then adds the values of all 'dir_size' files for all available days and culls the oldest images if the database has grown too large.

In addition 'kmotion_hkd1.py' checks that 'kmotion_hkd2.py' and 'motion' are both running, restarting them if for any reason they have ceased.

On a SIGHUP 'kmotion_hkd1.py' reloads its config from 'rc.daemon'

4:2 'kmotion_hkd2.py'

'kmotion_hkd2.py' checks and creates critical directories in the images database on a date change. These files would normally be created by 'motion' but it is possible that 'kmotion_hkd2.py' runs before 'motion' has a chance to create them..

'kmotion_hkd2.py' creates the 'journal_snap' files as detailed in the images database section. It deletes, moves or copies jpegs from the 'tmp' directory's to the 'video' directory's depending on the requested snapshot interval providing a 'sanitised' database consistent with the 'journal_snap' file.

On a SIGHUP 'kmotion_hkd2.py' reloads its config from 'daemon.rc' and updates 'journal_snap'
On a SIGKILL 'kmotion_hkd2.py' updates 'journal_snap' with a dummy string representing no further images are stored here i.e. #HHMMSS$86400. See the images database section for details.

4:3 Generating configs ...

kmotion can be installed anywhere on your system. One of the side effects of this is that certain config files have to be generated by kmotion on installation. The one constant config file is 'daemon.rc' which is located in the daemons directory and accessed by the daemons with a relative path './'

'gen_kmotion.py' generates the 'motion' file that can start kmotion from anywhere on the system .
'gen_kmotion_reload.py' generates the 'motion_reload' file that can restart kmotion from anywhere on the system.

'gen_vhost.py' reads 'daemon.rc' and creates a vhost file 'kmotion_vhost' from 'kmotion_vhost_template' with paths expanded as defined in 'daemon.rc'

'gen_int_rcs.py' reads in both 'daemon.rc' and 'motions' 'motion.conf'. It generates 'www.rc' for the PHP/Javascript front end and a modified version of 'motion.conf' for 'motion' to execute complete with paths expanded as defined in 'daemon.rc'

4:4 Diagnostic tools ...

The best diagnostic tool is 'tail -f /var/log/motion'.

To increase the verbosity of kmotion logging cd kmotion/daemons and edit 'daemon.rc'. Look for the subsection '[debug]' and the option 'log_level = WARN'. Change this value to EMERG, ALERT, CRIT, ERR, WARNING, NOTICE, INFO or DEBUG. Then manually kill all kmotion processes and restart with 'kmotion' (or just reboot).

In addition cd to kmotion/daemons. Execute 'daemon_manual.py'. This gives you manual control and status information of the daemons.

5:0 The images database

kmotion uses a custom database structure consisting of nested directories to store motion generated jpegs.

5:1 The images top level directory .../images/ contains

Multiple date directory's in the form 'YYYYMMDD'. kmotion catalogues all images at this level by their date of capture.

An 'events' directory. This directory holds flags that are created and deleted as motion is detected and cleared. The flags consist of number filenames with zero length. For example if the 'events' directory contains '2' and '5' motion has detected motion in threads 2 and 5. The flags are created and deleted by motions 'on_event_start' and 'on_event_end' directives.

A soft link called 'lastsnap.jpg'. This is a link generated by motion to point to the latest jpeg. It is not used by kmotion.

5:2 The images 1st level nested directory .../images/YYYYMMDD/ contains

Multiple thread directories in the form 'NN'. kmotion catalogues all images at this level by their thread or feed number as a two digit number.

A file called 'dir_size' containing an integer number. This number is the latest estimate of the number of bytes held in the current directory. The current days 'size' is updated every 15 mins by 'kmotion_hkd1.py'. The value of each 'dir_size' file in each of the available date directories is added together to decide if culling of the oldest dated directories is necessary. This is also performed by 'kmotion_hkd1.py'

5:3 The images 2nd level nested directory .../images/YYYYMMDD/NN/ contains

A file called 'av_file_size' containing an integer of the average number of bytes per jpeg file. This is calculated by averaging the size of the first 50 jpegs.

A file called 'av_dir_size' containing an integer of the average number of bytes per jpeg directory. This is calculated by averaging the size of the first 50 directories.

A file called 'last_jpeg' containing a path and filename. This points to the last jpeg for this particular thread or feed. It points to a jpeg in either the 'tmp' or 'video' directories. This file is updated by motions 'on_picture_save' directive. It is used by the PHP/Javascript front end to refresh the display with the latest jpeg.

A file called 'journal_snap' containing a coded string with the format #<start second>$<interval seconds> i.e. #193735$300, possibly repeated several times. This equation defines the motion snapshot jpeg filenames. i.e. #193735$300 would mean that the following jpegs exist 193735.jpg, 194035.jpg, 194335.jpg etc. (An exception being if a video directory has already been created, see below). 'journal_snap' is updated by 'kmotion_hkd2'. Multiple instances of coded strings in 'journal_snap' occur when kmotion's configs have been re-loaded.

5:4 The images 3rd level nested directory .../images/YYYYMMDD/NN/tmp contains

A 'tmp' directory. This directory contains between 4 and 6 ... 8 jpegs and acts as a buffer. motion snapshots are generated every second and stored 'tmp' by the 'snapshot_filename' operator. The 'last_jpeg' file may well point to a file in 'tmp'. 'kmotion_hkd2' 'sanitises' the time stamps on these jpegs and, providing there are no video directories already created and the snapshot interval is correct, moves the jpegs to the video directory. The 'sanitisation' ensures that the 'journal_snap' equations predicted jpegs always exist. In addition the buffering allows motions 'jpeg_filename' to lay down video directories first before 'kmotion_hkd2' attempts to move the jpeg and it ensures that any jpegs pointed to by 'last_jpeg' are around long enough to be used by the PHP/Javascript front end.

5:5 The images 4th level nested directory .../images/YYYYMMDD/NN/video contains

Multiple time directory's in the form 'HHMMSS' and multiple jpegs in the form 'HHMMSS.jpg'. The jpegs are motion snapshots every interval seconds as defined in the 'journal_snap' coded string. jpegs whose filenames clash with a directory in the form 'HHMMSS' are skipped.. The 'last_jpeg' file may well point to a file in 'video'.

5:6 The images 5th level nested directory .../images/YYYYMMDD/NN/HHMMSS contains

Multiple jpegs in the form NN.jpg. These are created by motions 'jpeg_filename' directive and represent high frame rate 'video' when motion has been detected. The 'last_jpeg' file may well point to a file in 'HHMMSS'.

6:0 The PHP/Javascript front end

First a disclaimer. I have never coded in PHP or javascript before, this is my first project. The code works and is robust elegant it ain't !

kmotion's main web page 'index.php' control's the right hand side buttons and a large iFrame where several other pages that are loaded.

In addition there are a couple of AJAX PHP calls which communicate data blobs between the server and the browser.

6:1 'index.php'

'index.php' holds the front ends state variables because it is the only page that is static. All other pages are loaded into the iFrame within 'index.php'. All state variables are in the name space 'state' and are accessible by all other pages.

6:1 'archive_list.php'

Called as 'archive_list.php?archive_dir=<archive_dir>' it returns a coded data blob holding information on the selected video directory data structure.

Ideally a complete list of all jpeg file names would be returned but this could be tens of thousands of files long and incur a considerable delay. To avoid this the data blob is coded to drastically reduce the data size and increase responsiveness.

'archive_list.php' scans the directory and returns a data blob with the following format.

#s<filename of first snapshot with this config>$<seconds till next snapshot> ...
#es<dir name of first dir in seq>$<number of jpegs in dir> ...
#ef<dir name of last dir in seq>$<number of jpegs in dir> ...
#ec<number of jpegs in 'mid' seq> ...

From this the browsers Javascript can regenerate an accurate directory structure complete with file and directory filenames.

6:1 'feed_status.php'

Called as 'feed_status.php' with no parameters returns a coded data blob holding information on all video feeds and any events (i.e. motion activity as resisted in the 'events' directory).

'feed_status.php' looks for the latest jpeg filenames and any events in progress returning a data blob in the following format.

#<latest filename camera 1>#<latest filename camera 2>...#<latest filename camera 16>
$<event number>$<event number>

From this the browsers Javascript knows which jpeg to request for each camera and which cameras have an active event in progress.

7:0 Contacting the developer

If you find any errors or omissions in this document please let me know. In fact contact me anyway even if you can't find any errors, if you love kmotion or think it could be improved it would be good to get feedback. I don't bite honest :)

Cheers

Dave
dave6502@googlemail.com