
ZoneMinder Documentation

<https://github.com/ZoneMinder/ZoneMinder/graphs/contributors>

Feb 19, 2020

Contents

1	Installation Guide	1
1.1	An Easy To Use Docker Image	1
1.2	All Distros - A Docker Way to Build ZoneMinder	1
1.3	Ubuntu	4
1.4	Debian	12
1.5	Redhat	16
1.6	Windows 10+ using WSL	21
1.7	Multi-Server Install	22
1.8	Dedicated Drive, Partition, or Network Share	23
2	User Guide	25
2.1	Introduction	25
2.2	Components	26
2.3	Getting Started	29
2.4	Defining Monitors	40
2.5	Defining Zones	50
2.6	Viewing Monitors	55
2.7	Filtering Events	56
2.8	Viewing Events	62
2.9	Options	63
2.10	Camera Control	95
2.11	Controlling Monitors	95
2.12	Mobile Devices	100
2.13	Logging	100
2.14	Configuration Files	103
3	API	105
3.1	Overview	105
3.2	API Wrappers	105
3.3	API evolution	105
3.4	Enabling API	106
3.5	Enabling secret key	106
3.6	Getting an API key	106
3.7	Using these keys with subsequent requests	107
3.8	Key lifetime (v1.0)	107
3.9	Key lifetime (v2.0)	108
3.10	Understanding access/refresh tokens (v2.0)	108

3.11	Understanding key security	108
3.12	Examples	109
3.13	API Version	109
3.14	Return a list of all monitors	110
3.15	Retrieve monitor 1	110
3.16	Change State of Monitor 1	110
3.17	Get Daemon Status of Monitor 1	110
3.18	Add a monitor	110
3.19	Edit monitor 1	111
3.20	Delete monitor 1	111
3.21	Arm/Disarm monitors	111
3.22	Return a list of all events	111
3.23	Retrieve event Id 1000	112
3.24	Edit event 1	112
3.25	Delete event 1	112
3.26	Return a list of events for a specific monitor Id =5	112
3.27	Return a list of events for a specific monitor within a specific date/time range	112
3.28	Return a list of events for all monitors within a specified date/time range	112
3.29	Return event count based on times and conditions	113
3.30	Return sorted events	113
3.31	Configuration Apis	113
3.32	Run State Apis	113
3.33	Create a Zone	114
3.34	PTZ Control Meta-Data APIs	114
3.35	Host APIs	114
3.36	Storage and Server APIs	115
3.37	Other APIs	116
3.38	Streaming Interface	116
3.39	Further Reading	118
4	FAQ	119
4.1	How can I stop ZoneMinder filling up my disk?	119
4.2	Math for Memory: Making sure you have enough memory to handle your cameras	121
4.3	I have enabled motion detection but it is not always being triggered when things happen in the camera view	122
4.4	Why can't ZoneMinder capture images (either at all or just particularly fast) when I can see my camera just fine in xawtv or similar?	123
4.5	Why can't I see streamed images when I can see stills in the zone window etc?	124
4.6	I have several monitors configured but when I load the Montage view in FireFox why can I only see two? or, Why don't all my cameras display when I use the Montage view in FireFox?	124
4.7	I can't see more than 6 monitors in montage on my browser	125
4.8	Why is ZoneMinder using so much CPU?	125
4.9	Why is the timeline view all messed up?	126
4.10	How much Hard Disk Space / Bandwidth do I need for ZM?	126
4.11	When I try and run ZoneMinder I get lots of audit permission errors in the logs and it won't start	126
4.12	How do I enable ZoneMinder's security?	127
4.13	Managing system load (with IP Cameras in mind)	127
4.14	Extending Zoneminder	129
4.15	Trouble Shooting	130
4.16	Miscellaneous	133
5	Contributing	135
6	Indices and tables	137

CHAPTER 1

Installation Guide

Todo: This entire guide needs to be checked/updated as needed

Contents:

1.1 An Easy To Use Docker Image

If you are interested in trying out ZoneMinder quickly, user Dan Landon maintains an easy to use docker image for ZoneMinder. With a few simple configuration changes, it also provides complete Event Notification Server and Machine Learning hook support. Please follow instructions in his repository. He maintains two repositories:

- If you want to run the latest stable release, please use his [zoneminder repository](#).
- If you want to run the latest zoneminder master, please use his [zoneminder master repository](#).

In both cases, instructions are provided in the repo README files.

If you are looking at building your own native (non docker) binary packages of ZoneMinder for your distro, please refer to the distro specific install guides or [All Distros - A Docker Way to Build ZoneMinder](#).

1.2 All Distros - A Docker Way to Build ZoneMinder

Note: If you are looking for an easy way to run ZoneMinder and not interested in building your own docker image, please refer to [An Easy To Use Docker Image](#).

Contents

- *All Distros - A Docker Way to Build ZoneMinder*
 - *Background*
 - *Procedure*
 - *Appendix A - Enable Qemu On the Host*

These instructions represent an alternative way to build ZoneMinder for any supported distro.

Advantages:

- Fewer steps and therefore much simpler
- Target distro agnostic - the steps are the same regardless of the target distro
- Host distro agnostic - the steps described here should work on any host Linux distro capable of running Bash and Docker

1.2.1 Background

These instructions leverage the power of the automated build system recently implemented in the ZoneMinder project. Behind the scenes, a project called [packpack](#) is utilized, to build ZoneMinder inside a Docker container.

1.2.2 Procedure

Step 1: Verify the target distro.

- Open the project's [.travis.yml](#) file and verify the distro you want to build ZoneMinder for appears in the build matrix. The distros shown in the matrix are those known to build on ZoneMinder. If the distro you desire is in the list then continue to step two.
- If the desired distro is not in the first list, then open the [packpack project README](#) and check if the desired distro is theoretically supported. If it is, then continue to step 2 with the understanding that you are heading out into uncharted territory. There could be problems.
- If the desired distro does not appear in either list, then unfortunately you cannot use the procedure described here.
- If the desired distro architecture is arm, refer to [Appendix A - Enable Qemu On the Host](#) to enable qemu emulation on your amd64 host machine.

Step 2: Install Docker.

You need to have a working installation of Docker so head over to the [Docker site](#) and get it working. Before continuing to the next step, verify you can run the Docker "Hello World" container as a normal user. To run a Docker container as a normal user, issue the following:

```
sudo gpasswd -a <username> docker
newgrp docker
```

Where <username> is, you guessed it, the user name you log in with.

Step 3: Git clone the ZoneMinder project.

Clone the ZoneMinder project if you have not done so already.

```
git clone https://github.com/ZoneMinder/ZoneMinder
cd ZoneMinder
```

Alternatively, if you have already cloned the repo and wish to update it, do the following.

```
cd ZoneMinder
git checkout master
git pull origin master
```

Step 4: Checkout the revision of ZoneMinder you wish to build.

A freshly cloned ZoneMinder git repo already points to the most recent commit in the master branch. If you want the latest development code then continue to the next step. If instead, you want to build a stable release then perform the following step.

```
git checkout <releasename>
```

Where <releasename> is one of the official ZoneMinder releases shown on the [releases page](#), such as 1.30.4.

Step 5: Build ZoneMinder

To start the build, simply execute the following command from the root folder of the local git repo:

```
OS=<distroname> DIST=<distrorel> utils/packpack/startpackpack.sh
```

Where <distroname> is the name of the distro you wish to build on, such as fedora, and <distrorel> is release name or number of the distro you wish to build on. Redhat distros expect a number for <distrorel> while Debian and Ubuntu distros expect a name. For example:

```
OS=fedora DIST=25 utils/packpack/startpackpack.sh
```

```
OS=ubuntu DIST=xenial utils/packpack/startpackpack.sh
```

Once you enter the appropriate command, go get a coffee while a ZoneMinder package is built. When the build finished, you can find the resulting packages under a subfolder called “build”.

Note that this will build packages with x86_64 architecture. This build method can also build on some distros (debian & ubuntu only at the moment) using i386 architecture. You can do that by adding “ARCH=i386” parameter.

```
OS=ubuntu DIST=xenial ARCH=i386 utils/packpack/startpackpack.sh
```

For advanced users who really want to go out into uncharted waters, it is theoretically possible to build arm packages as well, as long as the host architecture is compatible.

```
OS=ubuntu DIST=xenial ARCH=armhf utils/packpack/startpackpack.sh
```

Building arm packages in this manner has not been tested by us, however.

1.2.3 Appendix A - Enable Qemu On the Host

If you intend to build ZoneMinder packages for arm on an amd64 host, then Debian users can following these steps to enable transparent Qemu emulation:

```
sudo apt-get install binfmt-support qemu qemu-user-static
```

Verify arm emulation is enabled by issuing:

```
sudo update-binfmts --enable qemu-arm
```

You may get a message stating emulation for this processor is already enabled.

More testing needs to be done for Redhat distros but it appears Fedora users can just run:

```
sudo systemctl start systemd-binfmt
```

Todo: Verify the details behind enabling qemu emulation on redhat distros. Pull requests are welcome.

1.3 Ubuntu

Contents

- *Ubuntu*
 - *Easy Way: Ubuntu 18.04 (Bionic)*
 - *Easy Way: Ubuntu 16.04 (Xenial)*
 - *Easy Way: Ubuntu 14.x (Trusty)*
 - *Harder Way: Build Package From Source*
 - *Hints*
 - * *Make sure ZoneMinder and APIs work with security*
 - * *Socket_sendto or no live streaming*
 - * *Changed Default DB User*

1.3.1 Easy Way: Ubuntu 18.04 (Bionic)

These instructions are for a brand new ubuntu 18.04 system which does not have ZM installed.

It is recommended that you use an Ubuntu Server install and select the LAMP option during install to install Apache, MySQL and PHP. If you failed to do this you can achieve the same result by running:

```
sudo apt-get install tasksel
sudo tasksel install lamp-server
```

During installation it will ask you to set up a master/root password for the MySQL. Installing LAMP is not ZoneMinder specific so you will find plenty of resources to guide you with a quick search.

Step 1: Either run commands in this install using sudo or use the below to become root

```
sudo -i
```

Step 2: Update Repos

Latest Release

ZoneMinder is now part of the current standard Ubuntu repository, but sometimes the official repository can lag behind. To find out check our [releases page](#) for the latest release.

Alternatively, the ZoneMinder project team maintains a [PPA](#), which is updated immediately following a new release of ZoneMinder. To use this repository instead of the official Ubuntu repository, enter the following from the command line:

```
add-apt-repository ppa:iconnor/zoneminder-1.32
```

Update repo and upgrade.

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Step 3: Configure MySQL

Note

The MySQL default configuration file (/etc/mysql/mysql.cnf) is read through several symbolic links beginning with /etc/mysql/my.cnf as follows:

```
/etc/mysql/my.cnf -> /etc/alternatives/my.cnf
/etc/alternatives/my.cnf -> /etc/mysql/mysql.cnf
/etc/mysql/mysql.cnf is a basic file
```

Certain new defaults in MySQL 5.7 cause some issues with ZoneMinder < 1.32.0, the workaround is to modify the `sql_mode` setting of MySQL. Please note that these changes are NOT required for ZoneMinder 1.32.0 and some people have reported them causing problems in 1.32.0.

To better manage the MySQL server it is recommended to copy the sample config file and replace the default `my.cnf` symbolic link.

```
rm /etc/mysql/my.cnf (this removes the current symbolic link)
cp /etc/mysql/mysql.conf.d/mysqld.cnf /etc/mysql/my.cnf
```

To change MySQL settings:

```
nano /etc/mysql/my.cnf
```

In the `[mysqld]` section add the following

```
sql_mode = NO_ENGINE_SUBSTITUTION
```

CTRL+o then [Enter] to save

CTRL+x to exit

Restart MySQL

```
systemctl restart mysql
```

Step 4: Install ZoneMinder

```
apt-get install zoneminder
```

Step 5: Configure the ZoneMinder Database

This step should not be required on ZoneMinder 1.32.0.

```
mysql -uroot -p < /usr/share/zoneminder/db/zm_create.sql
mysql -uroot -p -e "grant lock tables,alter,drop,select,insert,update,delete,create,
↳index,alter routine,create routine, trigger,execute on zm.* to 'zmuser'@localhost_
↳identified by 'zmpass';"
```

Step 6: Set permissions

Set /etc/zm/zm.conf to root:www-data 740 and www-data access to content

```
chmod 740 /etc/zm/zm.conf
chown root:www-data /etc/zm/zm.conf
chown -R www-data:www-data /usr/share/zoneminder/
```

Step 7: Configure Apache correctly:

```
a2enmod cgi
a2enmod rewrite
a2enconf zoneminder
```

You may also want to enable the following modules to improve caching performance

```
a2enmod expires
a2enmod headers
```

Step 8: Enable and start Zoneminder

```
systemctl enable zoneminder
systemctl start zoneminder
```

Step 9: Edit Timezone in PHP

```
nano /etc/php/7.2/apache2/php.ini
```

Search for [Date] (Ctrl + w then type Date and press Enter) and change date.timezone for your time zone, see [this](<https://www.php.net/manual/en/timezones.php>). **Don't forget to remove the ; from in front of date.timezone**

```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = America/New_York
```

CTRL+o then [Enter] to save

CTRL+x to exit

Step 10: Reload Apache service

```
systemctl reload apache2
```

Step 11: Making sure ZoneMinder works

1. Open up a browser and go to `http://hostname_or_ip/zm` - should bring up ZoneMinder Console
2. (Optional API Check) Open up a tab in the same browser and go to `http://hostname_or_ip/zm/api/host/getVersion.json`

If it is working correctly you should get version information similar to the example below:

```
{
  "version": "1.29.0",
  "apiversion": "1.29.0.1"
}
```

Congratulations Your installation is complete

PPA install may need some tweaking of ZMS_PATH in ZoneMinder options. *Socket_sendto or no live streaming*

1.3.2 Easy Way: Ubuntu 16.04 (Xenial)

These instructions are for a brand new ubuntu 16.04 system which does not have ZM installed.

It is recommended that you use an Ubuntu Server install and select the LAMP option during install to install Apache, MySQL and PHP. If you failed to do this you can achieve the same result by running:

```
sudo tasksel install lamp-server
```

During installation it will ask you to set up a master/root password for the MySQL. Installing LAMP is not ZoneMinder specific so you will find plenty of resources to guide you with a quick search.

Step 1: Either run commands in this install using sudo or use the below to become root

```
sudo -i
```

Step 2: Update Repos

Latest Release

ZoneMinder is now part of the current standard Ubuntu repository, but sometimes the official repository can lag behind. To find out check our [releases page](#) for the latest release.

Alternatively, the ZoneMinder project team maintains a [PPA](#), which is updated immediately following a new release of ZoneMinder. To use this repository instead of the official Ubuntu repository, enter the following from the command line:

```
add-apt-repository ppa:iconnor/zoneminder
add-apt-repository ppa:iconnor/zoneminder-1.32
```

Update repo and upgrade.

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

Step 3: Configure MySQL

Note

The MySQL default configuration file (/etc/mysql/mysql.cnf) is read through several symbolic links beginning with /etc/mysql/my.cnf as follows:

```
/etc/mysql/my.cnf -> /etc/alternatives/my.cnf
```

```
/etc/alternatives/my.cnf -> /etc/mysql/mysql.cnf
/etc/mysql/mysql.cnf is a basic file
```

Certain new defaults in MySQL 5.7 cause some issues with ZoneMinder < 1.32.0, the workaround is to modify the `sql_mode` setting of MySQL. Please note that these changes are NOT required for ZoneMinder 1.32.0 and some people have reported them causing problems in 1.32.0.

To better manage the MySQL server it is recommended to copy the sample config file and replace the default `my.cnf` symbolic link.

```
rm /etc/mysql/my.cnf (this removes the current symbolic link)
cp /etc/mysql/mysql.conf.d/mysqld.cnf /etc/mysql/my.cnf
```

To change MySQL settings:

```
nano /etc/mysql/my.cnf
```

In the `[mysqld]` section add the following

```
sql_mode = NO_ENGINE_SUBSTITUTION
```

CTRL+o then [Enter] to save

CTRL+x to exit

Restart MySQL

```
systemctl restart mysql
```

Step 4: Install ZoneMinder

```
apt-get install zoneminder
```

Step 5: Configure the ZoneMinder Database

This step should not be required on ZoneMinder 1.32.0.

```
mysql -uroot -p < /usr/share/zoneminder/db/zm_create.sql
mysql -uroot -p -e "grant lock tables,alter,drop,select,insert,update,delete,create,
↳index,alter routine,create routine, trigger,execute on zm.* to 'zmuser'@localhost_
↳identified by 'zmpass';"
```

Step 6: Set permissions

Set `/etc/zm/zm.conf` to `root:www-data 740` and `www-data` access to content

```
chmod 740 /etc/zm/zm.conf
chown root:www-data /etc/zm/zm.conf
chown -R www-data:www-data /usr/share/zoneminder/
```

Step 7: Configure Apache correctly:

```
a2enmod cgi
a2enmod rewrite
a2enconf zoneminder
```

You may also want to enable to following modules to improve caching performance

```
a2enmod expires
a2enmod headers
```

Step 8: Enable and start Zoneminder

```
systemctl enable zoneminder
systemctl start zoneminder
```

Step 9: Edit Timezone in PHP

```
nano /etc/php/7.0/apache2/php.ini
```

Search for [Date] (Ctrl + w then type Date and press Enter) and change date.timezone for your time zone, see [this](<https://www.php.net/manual/en/timezones.php>). **Don't forget to remove the ; from in front of date.timezone**

```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = America/New_York
```

CTRL+o then [Enter] to save

CTRL+x to exit

Step 10: Reload Apache service

```
systemctl reload apache2
```

Step 11: Making sure ZoneMinder works

1. Open up a browser and go to `http://hostname_or_ip/zm` - should bring up ZoneMinder Console
2. (Optional API Check) Open up a tab in the same browser and go to `http://hostname_or_ip/zm/api/host/getVersion.json`

If it is working correctly you should get version information similar to the example below:

```
{
  "version": "1.29.0",
  "apiversion": "1.29.0.1"
}
```

Congratulations Your installation is complete

PPA install may need some tweaking of ZMS_PATH in ZoneMinder options. *Socket_sendto or no live streaming*

1.3.3 Easy Way: Ubuntu 14.x (Trusty)

These instructions are for a brand new ubuntu 14.x system which does not have ZM installed.

Step 1: Either run commands in this install using sudo or use the below to become root

```
sudo -i
```

Step 2: Install ZoneMinder

```
add-apt-repository ppa:iconnor/zoneminder
apt-get update
apt-get install zoneminder
```

(just press OK for the prompts you get)

Step 3: Set up DB

```
mysql -uroot -p < /usr/share/zoneminder/db/zm_create.sql
mysql -uroot -p -e "grant select,insert,update,delete,create,alter,index,lock tables,
↳on zm.* to 'zmuser'@localhost identified by 'zmpass';"
```

Step 4: Set up Apache

```
a2enconf zoneminder
a2enmod rewrite
a2enmod cgi
```

Step 5: Make zm.conf readable by web user.

```
sudo chown www-data:www-data /etc/zm/zm.conf
```

Step 6: Edit Timezone in PHP

```
nano /etc/php5/apache2/php.ini
```

Search for [Date] (Ctrl + w then type Date and press Enter) and change date.timezone for your time zone, see [this](<https://www.php.net/manual/en/timezones.php>). **Don't forget to remove the ; from in front of date.timezone**

```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = America/New_York
```

CTRL+o then [Enter] to save

CTRL+x to exit

Step 7: Restart Apache service and start ZoneMinder

```
service apache2 reload
service zoneminder start
```

Step 8: Making sure ZoneMinder works

1. Open up a browser and go to `http://hostname_or_ip/zm` - should bring up ZoneMinder Console
2. (Optional API Check) Open up a tab in the same browser and go to `http://hostname_or_ip/zm/api/host/getVersion.json`

If it is working correctly you should get version information similar to the example below:

```
{
  "version": "1.29.0",
  "apiversion": "1.29.0.1"
}
```

Congratulations Your installation is complete

1.3.4 Harder Way: Build Package From Source

(These instructions assume installation from source on a ubuntu 15.x+ system)

Step 1: Grab the package installer script

```
wget https://raw.githubusercontent.com/ZoneMinder/ZoneMinder/master/utils/do_debian_
↪package.sh
chmod a+x do_debian_package.sh
```

Step 2: Update the system

```
sudo apt-get update
```

Step 3 Create the package

To build the latest master snapshot:

```
./do_debian_package.sh --snapshot=NOW --branch=master --type=local
```

To build the latest stable release:

```
./do_debian_package.sh --snapshot=stable --type=local
```

Note that the distribution will be guessed using `lsb_release -a 2>/dev/null | grep Codename | awk '{print $2}'` which simply extracts your distribution name - like “vivid”, “trusty” etc. You can always specify it using `--distro=your distro name` if you know it. As far as the script goes, it checks if your distro is “trusty” in which case it pulls in pre-systemd release configurations and if its not “trusty” it assumes its based on systemd and pulls in systemd related config files.

(At the end the script will ask if you want to retain the checked out version of ZoneMinder. If you are a developer and are making local changes, make sure you select “y” so that the next time you do the build process mentioned here, it keeps your changes. Selecting any other value than “y” or “Y” will delete the checked out code and only retain the package)

This should now create a bunch of .deb files

Step 4: Install the package

```
sudo gdebi zoneminder_<version>_<arch>.deb
(example sudo gdebi zoneminder_1.29.0-vivid-2016012001_amd64.deb)
```

This will report DB errors - ignore - you need to configure the DB and some other stuff

Step 5: Post install configuration

Now that you have installed from your own package you can resume following the standard install guide for your version, start at the step after Install Zoneminder.

1.3.5 Hints

Make sure ZoneMinder and APIs work with security

1. Enable OPT_AUTH in ZoneMinder
2. Log out of ZoneMinder in browser
3. Open a new tab in the *same browser* (important) and go to `http://localhost/zm/api/host/getVersion.json` - should give you “Unauthorized” along with a lot more of text
4. Go to another tab in the SAME BROWSER (important) and log into ZM
5. Repeat step 3 and it should give you the ZM and API version

Socket_sendto or no live streaming

After you have setup your camera make sure you can view Monitor streams, if not check some of the common causes:

- Check Apache cgi module is enabled.
- Check Apache `/etc/apache2/conf-enabled/zoneminder.conf` `ScriptAlias` matches `PATH_ZMS`.

`ScriptAlias /zm/cgi-bin /usr/lib/zoneminder/cgi-bin`

From console go to `Options->Path` and make sure `PATH_ZMS` is set to `/zm/cgi-bin/nph-zms`.

Changed Default DB User

If you have changed your DB login/password from `zmuser/zmpass`, you need to update these values in `zm.conf`.

1. Edit `zm.conf` to change `ZM_DB_USER` and `ZM_DB_PASS` to the values you used.

1.4 Debian

Contents

- *Debian*
 - *Easy Way: Debian Stretch*
 - *Easy Way: Debian Jessie*

1.4.1 Easy Way: Debian Stretch

This procedure will guide you through the installation of ZoneMinder on Debian 9 (Stretch). This section has been tested with ZoneMinder 1.32.3 on Debian 9.8.

Step 1: Make sure your system is up to date

Open a console and use `su` command to become Root.

```
apt update
apt upgrade
```

Step 2: Setup Sudo (optional but recommended)

By default Debian does not come with `sudo`, so you have to install it and configure it manually. This step is optional but recommended and the following instructions assume that you have setup `sudo`. If you prefer to setup ZoneMinder as root, do it at your own risk and adapt the following instructions accordingly.

```
apt install sudo
usermod -a -G sudo <username>
exit
```

Now your terminal session is back under your normal user. You can check that you are now part of the `sudo` group with the command `groups`, “`sudo`” should appear in the list. If not, run `newgrp sudo` and check again with `groups`.

Step 3: Install Apache and MySQL

These are not dependencies for the ZoneMinder package as they could be installed elsewhere. If they are not installed yet in your system, you have to trigger their installation manually.

```
sudo apt install apache2 mysql-server
```

Step 4: Add ZoneMinder's Package repository to your apt sources

ZoneMinder's Debian packages are not included in Debian's official package repositories. To be able to install ZoneMinder with APT, you have to edit the list of apt sources and add ZoneMinder's repository.

```
sudo nano /etc/apt/sources.list
```

Add the following to the bottom of the file

```
# ZoneMinder repository
deb https://zmrepo.zoneminder.com/debian/release stretch/
```

CTRL+o and <Enter> to save CTRL+x to exit

Because ZoneMinder's package repository provides a secure connection through HTTPS, apt must be enabled for HTTPS.

```
sudo apt install apt-transport-https
```

Finally, download the GPG key for ZoneMinder's repository:

```
wget -O - https://zmrepo.zoneminder.com/debian/archive-keyring.gpg | sudo apt-key add
```

Step 5: Install ZoneMinder

```
sudo apt update
sudo apt install zoneminder
```

Step 6: Read the Readme

The rest of the install process is covered in the README.Debian, so feel free to have a read.

```
zcat /usr/share/doc/zoneminder/README.Debian.gz
```

Step 7: Enable ZoneMinder service

```
sudo systemctl enable zoneminder.service
```

Step 8: Configure Apache

The following commands will setup the default /zm virtual directory and configure required apache modules.

```
sudo a2enconf zoneminder
sudo a2enmod rewrite
sudo a2enmod cgi # this is done automatically when installing the package. Redo this
↳ command manually only for troubleshooting.
```

Step 9: Edit Timezone in PHP

Automated way:

```
sudo sed -i "s;/date.timezone =/date.timezone = $(sed 's/\\/\\\\\\\\/' /etc/timezone)/g" /
↳ etc/php/7.0/apache2/php.ini
```

Manual way

```
sudo nano /etc/php/7.0/apache2/php.ini
```

Search for [Date] (Ctrl + w then type Date and press Enter) and change date.timezone for your time zone. Don't forget to remove the ; from in front of date.timezone.

```
[Date]
; Defines the default timezone used by the date functions
; http://php.net/date.timezone
date.timezone = America/New_York
```

CTRL+o then [Enter] to save

CTRL+x to exit

Step 10: Start ZoneMinder

Reload Apache to enable your changes and then start ZoneMinder.

```
sudo systemctl reload apache2
sudo systemctl start zoneminder
```

You are now ready to go with ZoneMinder. Open a browser and type either `localhost/zm` on the local machine or `{ IP-OF-ZM-SERVER } /zm` if you connect from a remote computer.

1.4.2 Easy Way: Debian Jessie

Step 1: Setup Sudo

By default Debian does not come with sudo. Log in as root or use su command. N.B. The instructions below are for setting up sudo for your current account, you can do this as root if you prefer.

```
apt-get update
apt-get install sudo
usermod -a -G sudo <username>
exit
```

Logout or try `newgrp` to reload user groups

Step 2: Run sudo and update

Now run session using sudo and ensure system is updated.

```
sudo -i
apt-get upgrade
```

Step 3: Install Apache and MySQL

These are not dependencies for the package as they could be installed elsewhere.

```
apt-get install apache2 mysql-server
```

Step 4: Edit sources.list to add jessie-backports

```
nano /etc/apt/sources.list
```

Add the following to the bottom of the file

```
# Backports repository
deb http://archive.debian.org/debian/ jessie-backports main contrib non-free
```

CTRL+o and <Enter> to save CTRL+x to exit

Run the following

```
echo 'Acquire::Check-Valid-Until no;' > /etc/apt/apt.conf.d/99no-check-valid-until
```

Step 5: Install ZoneMinder

```
apt-get update
apt-get install zoneminder
```

Step 6: Read the Readme

The rest of the install process is covered in the README.Debian, so feel free to have a read.

```
zcat /usr/share/doc/zoneminder/README.Debian.gz
```

Step 7: Setup Database

Install the zm database and setup the user account. Refer to Hints in Ubuntu install should you choose to change default database user and password.

```
cat /usr/share/zoneminder/db/zm_create.sql | sudo mysql --defaults-file=/etc/mysql/
↪debian.cnf
echo 'grant lock tables,alter,create,select,insert,update,delete,index on zm.* to
↪'zmuser'@localhost identified by "zmpass";' | sudo mysql --defaults-file=/etc/
↪mysql/debian.cnf mysql
```

Step 8: zm.conf Permissions

Adjust permissions to the zm.conf file to allow web account to access it.

```
chgrp -c www-data /etc/zm/zm.conf
```

Step 9: Setup ZoneMinder service

```
systemctl enable zoneminder.service
```

Step 10: Configure Apache

The following commands will setup the default /zm virtual directory and configure required apache modules.

```
a2enconf zoneminder
a2enmod cgi
a2enmod rewrite
```

Step 11: Edit Timezone in PHP

```
nano /etc/php5/apache2/php.ini
```

Search for [Date] (Ctrl + w then type Date and press Enter) and change date.timezone for your time zone. **Don't forget to remove the ; from in front of date.timezone**

```
[Date]
; Defines the default timezone used by the date functions
```

(continues on next page)

(continued from previous page)

```
; http://php.net/date.timezone
date.timezone = America/New_York
```

CTRL+o then [Enter] to save

CTRL+x to exit

Step 12: Please check the configuration

Zoneminder 1.32.x

1. Check path of ZM_PATH in ‘etc/zm/conf.d/zmcustom.conf’ is ZM_PATH_ZMS=/zm/cgi-bin/nph-zms

```
:: cat /etc/zm/conf.d/zmcustom.conf
```

2. Check config of /etc/apache2/conf-enabled/zoneminder.conf has the same ScriptAlias /zm/cgi-bin that is configured in ZM_PATH. The part /nph-zms has to be left out of the ScriptAlias

```
ScriptAlias /zm/cgi-bin "/usr/lib/zoneminder/cgi-bin" <Directory "/usr/lib/zoneminder/cgi-bin">
```

```
:: cat /etc/apache2/conf-enabled/zoneminder.conf
```

Step 13: Start ZoneMinder

Reload Apache to enable your changes and then start ZoneMinder.

```
systemctl reload apache2
systemctl start zoneminder
```

Step 14: Making sure ZoneMinder works

1. Open up a browser and go to http://hostname_or_ip/zm - should bring up ZoneMinder Console
2. (Optional API Check) Open up a tab in the same browser and go to http://hostname_or_ip/zm/api/host/getVersion.json

If it is working correctly you should get version information similar to the example below:

```
{
  "version": "1.29.0",
  "apiversion": "1.29.0.1"
}
```

Congratulations Your installation is complete

1.5 Redhat

Contents

- *Redhat*
 - *Background: RHEL, CentOS, and Clones*
 - *Background: Fedora*
 - *How To Avoid Known Installation Problems*
 - *How to Install ZoneMinder*

- *How to Install Nightly Development Builds*
- *How to Change from Zmrepo to RPM Fusion*
- *How to Build Your Own ZoneMinder Package*
 - * *Background*
 - * *Set Up Your Environment*
 - * *Build from SRPM*
 - * *Installation*
 - * *How to Create Your Own Source RPM*

These instructions apply to all Redhat distros and their clones, including but not limited to: Fedora, RHEL, CentOS, Scientific Linux, and others. While the installation instructions are the same for each distro, the reason why one might use one distro over the other is different. A short description follows, which is intended to help you choose what distro best fits your needs.

1.5.1 Background: RHEL, CentOS, and Clones

These distributions are classified as enterprise operating systems and have a long operating lifetime of many years. By design, they will not have the latest and greatest versions of any package. Instead, stable packages are the emphasis.

Replacing any core package in these distributions with a newer package from a third party is expressly verboten. The ZoneMinder development team will not do this, and neither should you. If you have the perception that you've got to have a newer version of php, mysql, gnome, apache, etc. then, rather than upgrade these packages, you should instead consider using a different distribution such as Fedora.

The ZoneMinder team will not provide support for systems which have had any core package replaced with a package from a third party.

1.5.2 Background: Fedora

One can think of Fedora as RHEL or CentOS Beta. This is, in fact, what it is. Fedora is primarily geared towards development and testing of newer, sometimes bleeding edge, packages. The ZoneMinder team uses this distro to determine the interoperability of ZoneMinder with the latest and greatest versions of packages like mysql, apache, systemd, and others. If a problem is detected, it will be addressed long before it makes it way into RHEL.

Fedora has a short life-cycle of just 6 months. However, Fedora, and consequently ZoneMinder, is available on armv7 architecture. Rejoice, Raspberry Pi users!

If you desire newer packages than what is available in RHEL or CentOS, you should consider using Fedora.

1.5.3 How To Avoid Known Installation Problems

The following notes are based on real problems which have occurred by those who came before you:

- Zmrepo assumes you have installed the underlying distribution **using the official installation media for that distro**. Third party "Spins" may not work correctly.
- ZoneMinder is intended to be installed in an environment dedicated to ZoneMinder. While ZoneMinder will play well with many applications, some invariably will not. Asterisk is one such example.
- Be advised that you need to start with a clean system before installing ZoneMinder.

- If you have previously installed ZoneMinder from-source, then your system is **NOT** clean. You must manually search for and delete all ZoneMinder related files first (look under /usr/local). Issuing a “make uninstall” helps, but it will not do this for you correctly. You **WILL** have problems if you ignore this step.
- Unlike Debian/Ubuntu distros, it is not necessary, and not recommended, to install a LAMP stack ahead of time.
- Disable any other third party repos and uninstall any of ZoneMinder’s third party dependencies, which might already be on the system, especially ffmpeg and vlc. Attempting to install dependencies yourself often causes problems.
- Each ZoneMinder rpm includes a README file under /usr/share/doc. You must follow all the steps in this README file, precisely, each and every time ZoneMinder is installed or upgraded. **Failure to do so is guaranteed to result in a non-functional system.**

1.5.4 How to Install ZoneMinder

ZoneMinder releases are now being hosted at RPM Fusion. New users should navigate the [RPM Fusion site](#) then follow the instructions to enable that repo. RHEL/CentOS users must also navigate to the [EPEL Site](#) and enable that repo as well. Once enabled, install ZoneMinder from the commandline:

```
sudo dnf install zoneminder
```

Note that RHEL/CentOS 7 users should use yum instead of dnf.

Once ZoneMinder has been installed, it is critically important that you read the README file under /usr/share/doc/zoneminder. ZoneMinder will not run without completing the steps outlined in the README.

1.5.5 How to Install Nightly Development Builds

ZoneMinder development packages, which represent the most recent build from our master branch, are available from [zmrepo](#).

The feedback we get from those who use these development packages is extremely helpful. However, please understand these packages are intended for testing the latest master branch only. They are not intended to be used on any production system. There will be new bugs, and new features may not be documented. This is bleeding edge, and there might be breakage. Please keep that in mind when using this repo. We know from our user forum that this can’t be stated enough.

1.5.6 How to Change from Zmrepo to RPM Fusion

As mentioned above, the place to get the latest ZoneMinder release is now [RPM Fusion](#). If you are currently using ZoneMinder release packages from Zmrepo, then the following steps will change you over to RPM Fusion:

- Navigate to the [RPM Fusion site](#) and enable RPM Fusion on your system
- Now issue the following from the command line:

```
sudo dnf remove zmrepo
sudo dnf update
```

Note that RHEL/CentOS 7 users should use yum instead of dnf.

1.5.7 How to Build Your Own ZoneMinder Package

If you are looking to do development or the available packages just don't suit you, then you can follow these steps to build your own ZoneMinder RPM.

Background

The following method documents how to build ZoneMinder into an RPM package, for Fedora, Redhat, CentOS, and other compatible clones. This is exactly how the RPMS in `zmrepo` are built.

The method documented below was chosen because:

- All of ZoneMinder's dependencies are downloaded and installed automatically
- Cross platform capable. The build host does not have to be the same distro or release version as the target.
- Once your build environment is set up, few steps are required to run the build again in the future.
- Troubleshooting becomes easier if we are all building ZoneMinder the same way.

IMPORTANT Certain commands in these instructions require root privileges while other commands do not. Pay close attention to this. If the instructions below state to issue a command without a "sudo" prefix, then you should *not* be root while issuing the command. Getting this incorrect will result in a failed build, or worse a broken system.

Set Up Your Environment

Before you begin, set up an `rpmbuild` environment by following [this guide](#) by the CentOS developers.

In addition, make sure RPM Fusion is enabled as described in the previous section [How to Install ZoneMinder](#).

With RPM Fusion enabled, issue the following command:

```
sudo yum install mock-rpmsfusion-free mock
```

Add your user account to the group `mock`:

```
sudo gpasswd -a {your account name} mock
```

Your build environment is now set up.

Build from SRPM

To continue, you need a ZoneMinder SRPM. If you wish to rebuild a ZoneMinder release, then browse the [RPM Fusion site](#). If instead you wish to rebuild the latest source rpm from our master branch then browse the [Zmrepo site](#).

For this example, I'll use one of the source rpms from `zmrepo`:

```
wget -P ~/rpmbuild/SRPMS http://zmrepo.zoneminder.com/el/7/SRPMS/zoneminder-1.31.1-1.
↪el7.centos.src.rpm
```

Now comes the fun part. To build ZoneMinder, issue the following command:

```
mock -r epel-7-x86_64-rpmsfusion-free ~/rpmbuild/SRPMS/zoneminder-1.31.1-1.el7.centos.
↪src.rpm
```

Want to build ZoneMinder for Fedora, instead of CentOS, from the same host? Once you download the Fedora SRPM, issue the following:

```
mock -r fedora-26-x86_64-rpmsfusion-free ~/rpmbuild/SRPMS/zoneminder-1.31.1-1.el7.  
↪centos.src.rpm
```

Notice that the mock tool requires the following parameters:

```
mock -r MOCKCONFIG ZONEMINDER_SRPM
```

The list of available Mock config files are available here:

```
ls /etc/mock/*rpmsfusion-free.cfg
```

You choose the config file based on the desired distro (e.g. el7, f29, f30) and basearch (e.g. x86, x86_64, arhmfhp). Notice that, when specifying the Mock config as a commandline parameter, you should leave off the “.cfg” filename extension.

Installation

Once the build completes, you will be presented with a message stating where the newly built rpms can be found. It will look similar to this:

```
INFO: Results and/or logs in: /var/lib/mock/fedora-26-x86_64/result
```

Copy the newly built ZoneMinder RPMs to the desired system, enable RPM Fusion as described in [How to Install ZoneMinder](#), and then install the rpm by issuing the appropriate yum/dnf install command. Finish the installation by following the zoneminder setup instructions in the distro specific readme file, named README.{distriname}, which will be installed into the /usr/share/doc/zoneminder* folder.

Finally, you may want to consider editing the rpmsfusion repo file under /etc/yum.repos.d and placing an “exclude=zoneminder*” line into the config file. This will prevent your system from overwriting your manually built RPM with the ZoneMinder RPM found in the repo.

How to Create Your Own Source RPM

In the previous section we described how to rebuild an existing ZoneMinder SRPM. The instructions which follow show how to build the ZoneMinder git source tree into a source rpm, which can be used in the previous section to build an rpm.

Make sure git and rpmdevtools are installed:

```
sudo yum install git rpmdevtools
```

Now clone the ZoneMinder git repository from your home folder:

```
cd  
git clone https://github.com/ZoneMinder/zoneminder  
cd zoneminder
```

This will create a sub-folder called ZoneMinder, which will contain the latest development source code.

If you have previously cloned the ZoneMinder git repo and wish to update it to the most recent, then issue these commands instead:

```
cd ~/zoneminder  
git pull origin master
```

Get the crud submodule tarball:


```
spectool -f -g -R -s 1 ~/zoneminder/distros/redhat/zoneminder.spec
```

At this point, you can make changes to the source code. Depending on what you want to do with those changes, you generally want to create a new branch first:

```
cd ~/zoneminder
git checkout -b mynewbranch
```

Again, depending on what you want to do with those changes, you may want to commit your changes:

```
cd ~/zoneminder
git add .
git commit
```

Once you have made your changes, it is time to turn your work into a new tarball, but first we need to look in the rpm specfile:

```
less ~/zoneminder/distros/redhat/zoneminder.spec
```

Scroll down until you see the Version field. Note the value, which will be in the format x.xx.x. Now create the tarball with the following command:

```
cd ~/zoneminder
git archive --prefix=zoneminder-1.33.4/ -o ~/rpmbuild/SOURCES/zoneminder-1.33.4.tar.
→gz HEAD
```

Replace “1.33.4” with the Version shown in the rpm specfile.

From the root of the local ZoneMinder git repo, execute the following:

```
cd ~/zoneminder
rpmbuild -bs --nodeps distros/redhat/zoneminder.spec
```

This step will create a source rpm and it will tell you where it was saved. For example:

```
Wrote: /home/abauer/rpmbuild/SRPMS/zoneminder-1.33.4-1.fc26.src.rpm
```

Now follow the previous instructions *Build from SRPM* which describe how to build that source rpm into an rpm.

1.6 Windows 10+ using WSL

With Windows 10, Microsoft released the [Window Subsystem for Linux \(WSL\)](#) that enables you to run native Linux tools directly on Windows, alongside your traditional Windows desktop and modern store apps. To install WSL, please refer to the [installation guide from Microsoft](#).

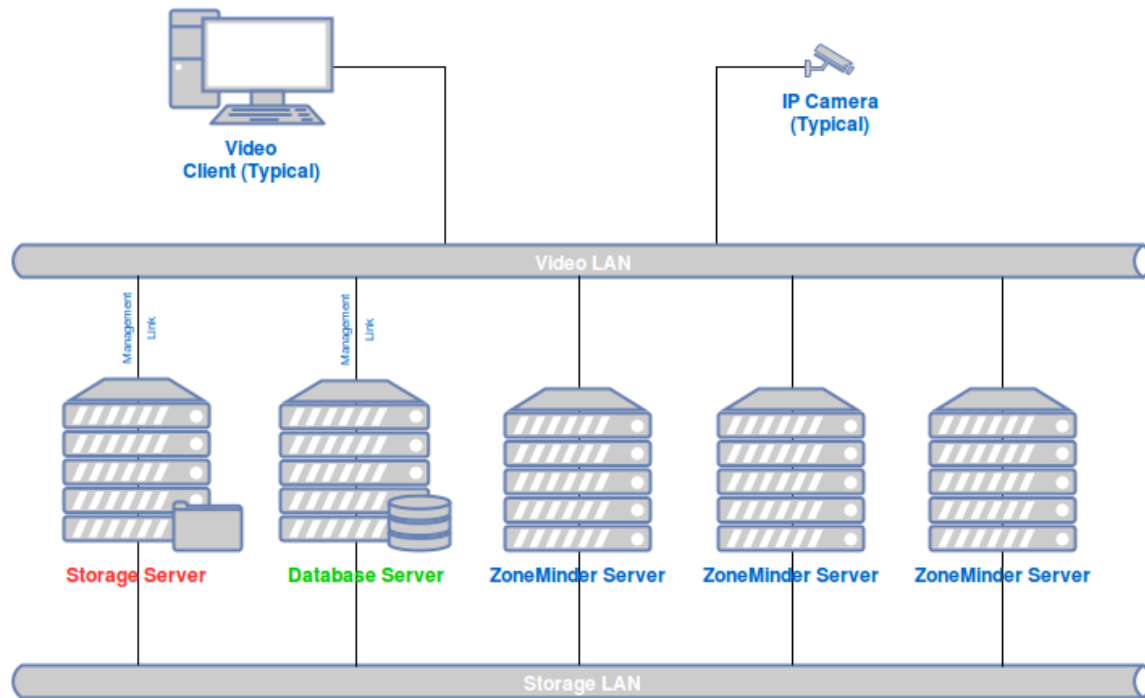
ZoneMinder now runs on Windows 10+ systems which have WSL enabled. This guide will explain how to go about installing an Ubuntu ZM image on Windows 10+.

Todo: Isaac to add instructions.

1.7 Multi-Server Install

It is possible to run multiple ZoneMinder servers and manage them from a single interface. To achieve this each zoneminder server is connected to a single shared database server and shares file storage for event data.

ZoneMinder Multi-Server Network Topology



1.7.1 Topology Design Notes

1. Device symbols represent separate logical functions, not necessarily separate hardware. For example, the Database Server and a ZoneMinder Server, can reside on the same physical hardware.
2. Configure each ZoneMinder Server to use the same, remote Database Server (Green).
3. The Storage Server (Red) represents shared storage, accessible by all ZoneMinder Servers, mounted under each server's events folder.
4. Create at least two networks for best performance. Dedicate a Storage LAN for communication with the Storage and Database Servers. Make use of multipath and jumbo frames if possible. Keep all other traffic off the Storage

LAN! Dedicate the second LAN, called the Video LAN in the diagram, for all other traffic.

1.7.2 New installs

1. Follow the normal instructions for your distro for installing ZoneMinder onto all the ZoneMinder servers in the normal fashion. Only a single database will be needed either as standalone, or on one of the ZoneMinder Servers.
2. On each ZoneMinder server, edit `zm.conf`. Find the `ZM_DB_HOST` variable and set it to the name or ip address of your Database Server. Find the `ZM_SERVER_HOST` and enter a name for this ZoneMinder server. Use a name easily recognizable by you. This name is not used by ZoneMinder for dns or any other form of network connectivity.
3. Copy the file `/usr/share/zoneminder/db/zm_create.sql` from one of the ZoneMinder Servers to the machine targeted as the Database Server.
4. Install `mysql/mariadb` server onto the Database Server.
5. It is advised to run “`mysql_secure_installation`” to help secure the server.
6. Using the password for the root account set during the previous step, create the ZoneMinder database and configure a database account for ZoneMinder to use:

```
mysql -u root -p < zm_create.sql
mysql -uroot -p -e "grant all on zm.* to 'zmuser'@localhost identified by 'zmpass';"
mysqladmin -u root -p reload
```

The database account credentials, `zmuser/zmpass`, are arbitrary. Set them to anything that suits your environment. Note that these commands are just an example and might not be secure enough for your environment.

7. If you have chosen to change the ZoneMinder database account credentials to something other than `zmuser/zmpass`, you must now update `zm.conf` on each ZoneMinder Server. Change `ZM_DB_USER` and `ZM_DB_PASS` to the values you created in the previous step.
8. All ZoneMinders Servers must share a common events folder. This can be done in any manner supported by the underlying operating system. From the Storage Server, share/export a folder to be used for ZoneMinder events.
9. From each ZoneMinder Server, mount the shared events folder on the Storage Server to the events folder on the local ZoneMinder Server.

NOTE: The location of this folder varies by distro. This folder is often found under “`/var/lib/zoneminder/events`” for RedHat based distros and “`/var/cache/zoneminder/events`” for Debain based distros. This folder is NOT a Symbolic Link!

10. Open your browser and point it to the web console on any of the ZoneMinder Servers (they will all be the same). Open Options, click the Servers tab, and populate this screen with all of your ZoneMinder Servers. Each server has a field for its name and its hostname. The name is what you used for `ZM_SERVER_HOST` in step 2. The hostname is the network name or ip address ZoneMinder should use.
11. When creating a new Monitor, remember to select the server the camera will be assigned to from the Server drop down box.

1.8 Dedicated Drive, Partition, or Network Share

One of the first steps the end user must perform after installing ZoneMinder is to dedicate an entire partition, drive, or network share for ZoneMinder’s event storage. The reason being, ZoneMinder will, by design, fill up your hard disk, and you don’t want to do that to your root volume!

The following steps apply to ZoneMinder 1.31 or newer, running on a typical Linux system, which uses systemd. If you are using an older version of ZoneMinder, please follow the legacy steps in the [ZoneMinder Wiki](#).

Step 1: Stop ZoneMinder

```
sudo systemctl stop zoneminder
```

Step 2: Mount your dedicated drive, partition, or network share to the local filesystem in any folder of your choosing. We recommend you use systemd to manage the mount points. Instructions on how to accomplish this can be found [here](#) and [here](#). Note that bind mounting ZoneMinder's images folder is optional. Newer version of ZoneMinder write very little, if anything, to the images folder. Verify the dedicated drive, partition, or network share is successfully mounted before proceeding to the next step.

Step 3: Set the owner and group to that of the web server user account. Debian based distros typically use “www-data” as the web server user account while many rpm based distros use “apache”.

```
sudo chown -R apache:apache /path/to/your/zoneminder/events/folder
sudo chown -R apache:apache /path/to/your/zoneminder/images/folder
```

Recall from the previous step, the images folder is optional.

Step 4: Create a config file under /etc/zm/conf.d using your favorite text editor. Name the file anything you want just as long as it ends in “.conf”. Add the following content to the file and save your changes:

```
ZM_DIR_EVENTS=/full/path/to/the/events/folder
```

Step 5: Start ZoneMinder and inspect the ZoneMinder log files for errors.

```
sudo systemctl start zoneminder
```

2.1 Introduction

Welcome to ZoneMinder, the all-in-one security camera solution for Linux with GPL License.

Commercial “security systems” are often designed as a monitoring system with little attention to recording quality. In such a system, locating and exporting relevant video can be challenging and often requires extensive human intervention. ZoneMinder was designed to provide the best possible record quality while allowing easy searching, filtering and exporting of security footage.

ZoneMinder is designed around a series of independent components that only function when necessary, limiting any wasted resource and maximising the efficiency of your machine. An outdated Pentium II PC can have multiple recording devices connected to it, and it is able to track one camera per device at up to 25 frames per second, which drops by approximately half for each additional camera on the same device. Additional cameras on devices that do not interact with other devices can maintain the 25 frame rate per second. Monitoring several cameras will not overload the CPU as frame processing is designed to synchronise with capture.

A fast video interface core, a user-friendly and comprehensive PHP based web interface allows ZoneMinder to be efficient, friendly and most importantly useful. You can control and monitor your cameras from home, at work, on the road, or a web-enabled cell phone. It supports variable web capabilities based on available bandwidth. The web interface also allows you to view events that your cameras have captured, which can be archived, reviewed or deleted. The web application directly interacts with the core daemons ensuring full co-operation at all times. ZoneMinder can also be installed as a system service to reboot a system remotely.

The core of ZoneMinder is the capture and analysis of images and a highly configurable set of parameters that eliminate false positives whilst ensuring minimum loss of footage. For example, you can define a set of ‘zones’ for each camera of varying sensitivity and functionality. This eliminates zones that you don’t wish to track or define areas that will alarm if various thresholds are exceeded in conjunction with other zones.

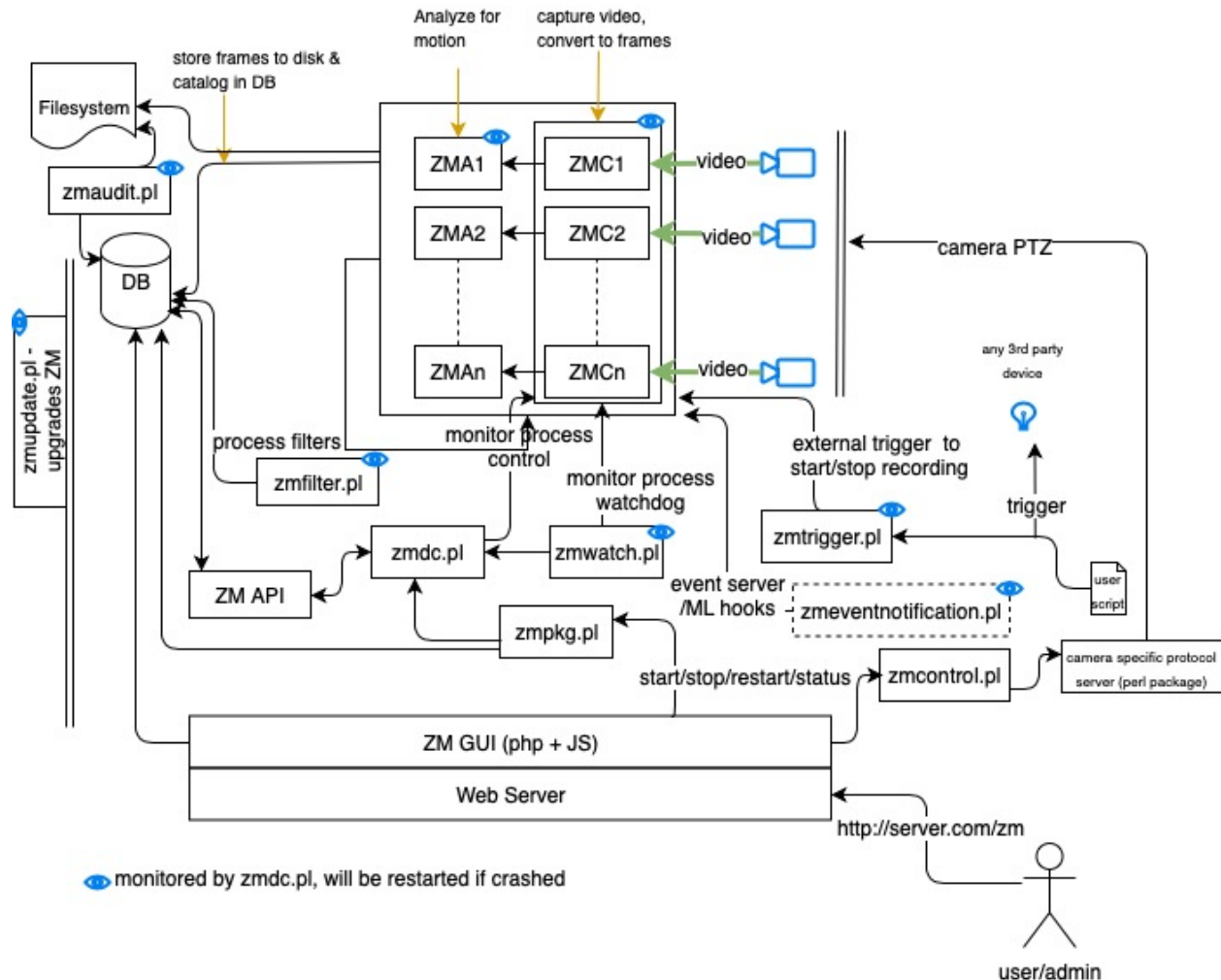
ZoneMinder is free under GPL License, but if you do find it useful, then please feel free to visit <https://zoneminder.com/donate/> and help us fund our future improvements.

2.2 Components

ZoneMinder is not a single monolithic application but is formed from several components. These components primarily include executable compiled binaries which do the main video processing work, perl scripts which usually perform helper and/or external interface tasks and php web scripts which are used for the web interface.

2.2.1 System Overview

Depicted below is a high level diagram of the ZoneMinder system with key components



A brief description of each of the principle components follows.

2.2.2 Binaries

zmc This is the ZoneMinder Capture daemon. This binary's job is to sit on a video device and suck frames off it as fast as possible, this should run at more or less constant speed.

zma This is the ZoneMinder Analysis daemon. This is the component that goes through the captured frames and checks them for motion which might generate an alarm or event. It generally keeps up with the Capture daemon but if very busy may skip some frames to prevent it falling behind.

zms This is the ZoneMinder Streaming server. The web interface connects with this to get real-time or historical streamed images. It runs only when a live monitor stream or event stream is actually being viewed and dies when the event finishes or the associate web page is closed. If you find you have several zms processes running when nothing is being viewed then it is likely you need a patch for apache (see the Troubleshooting section). A non-parsed header version of zms, called nph-zms, is also installed and may be used instead depending on your web server configuration.

zmu This is the ZoneMinder Utility. It's basically a handy command line interface to several useful functions. It's not really meant to be used by anyone except the web page (there's only limited 'help' in it so far) but can be if necessary, especially for debugging video problems.

2.2.3 PHP

As well as this there are the web PHP files in the web directory. Currently these consist of a single skin with Classic and Flat styles.

Classic Original ZoneMinder skin

Flat An updated version of Classic skin, retaining the same layout with a more modern style. Originally a skin this is now just a CSS style.

2.2.4 Perl

Finally some perl scripts in the scripts directory. These scripts all have some configuration at the top of the files which should be viewed and amended if necessary and are as follows.

zmpkg.pl This is the ZoneMinder Package Control script. This is used by the web interface and service scripts to control the execution of the system as a whole.

zmdc.pl This is the ZoneMinder Daemon Control script. This is used by the web interface and the zmpkg.pl script to control and maintain the execution of the capture and analysis daemons, amongst others. You should not need to run this script yourself, although you can use it to start/top individual ZM processes.

zmfilter.pl This script controls the execution of saved filters and will be started and stopped by the web interface based on whether there are filters that have been defined to be autonomous(background). This script is also responsible for the automatic uploading of events to a 3rd party server. Prior to 1.32 there was one zmfilter.pl process. In 1.32 onwards we start a zmfilter.pl process for each background filter so that the processing time of one filter doesn't delay the processing of another filter.

zmaudit.pl This script is used to check the consistency of the event file system and database. It can delete orphaned events, i.e. ones that appear in one location and not the other as well as checking that all the various event related tables are in line. It can be run interactively or in batch mode either from the command line or a cron job or similar. In the zmconfig.pl there is an option to specify fast event deletes where the web interface only deletes the event entry from the database itself. If this is set then it is this script that tidies up the rest. We do not recommend fast event deletion and we do not recommend having zmaudit.pl run in the background. It is a very ram cpu and disk io intensive program, constantly scanning every event. Please run it manually or from a cron job on weekends or something.

zmwatch.pl This is a simple script purely designed to keep an eye on the capture daemons and restart them if they lockup. It has been known for sync problems in the video drivers to cause this so this script makes sure that nothing important gets missed.

zmupdate.pl Currently this script is responsible for checking whether a new version of ZoneMinder is available and other miscellaneous actions related to upgrades and migrations. It is also intended to be a 'one stop shop' for any upgrades and will execute everything necessary to update your installation to a new version.

zmvideo.pl This script is used from the web interface to generate video files in various formats in a common way. You can also use it from the command line in certain circumstances but this is not usually necessary.

zmx10.pl This is an optional script that can be used to initiate and monitor X10 Home Automation style events and interface with an alarm system either by the generation of X10 signals on ZoneMinder events or by initiating ZoneMinder monitoring and capture on receipt of X10 signals from elsewhere, for instance the triggering of an X10 PIR. For example I have several cameras that don't do motion detection until I arm my alarm system whereupon they switch to active mode when an X10 signal is generated by the alarm system and received by ZoneMinder.

zmtrigger.pl This is an optional script that is a more generic solution to external triggering of alarms. It can handle external connections via either internet socket, unix socket or file/device interfaces. You can either use it 'as is' if you can interface with the existing format, or override connections and channels to customise it to your needs. The format of triggers used by zmtrigger.pl is as follows "<id><action><score><cause><text><showtext>" where

- 'id' is the id number or name of the ZM monitor.
- 'action' is 'on', 'off', 'cancel' or 'show' where 'on' forces an alarm condition on, 'off' forces an alarm condition off and 'cancel' negates the previous 'on' or 'off'. The 'show' action merely updates some auxiliary text which can optionally be displayed in the images captured by the monitor. Ordinarily you would use 'on' and 'cancel', 'off' would tend to be used to suppress motion based events. Additionally 'on' and 'off' can take an additional time offset, e.g. on+20 which automatically 'cancel's the previous action after that number of seconds.
- 'score' is the score given to the alarm, usually to indicate it's importance. For 'on' triggers it should be non-zero, otherwise it should be zero.
- 'cause' is a 32 char max string indicating the reason for, or source of the alarm e.g. 'Relay 1 open'. This is saved in the 'Cause' field of the event. Ignored for 'off' or 'cancel' messages.
- 'text' is a 256 char max additional info field, which is saved in the 'Description' field of an event. Ignored for 'off' or 'cancel' messages.
- 'showtext' is up to 32 characters of text that can be displayed in the timestamp that is added to images. The 'show' action is designed to update this text without affecting alarms but the text is updated, if present, for any of the actions. This is designed to allow external input to appear on the images captured, for instance temperature or personnel identity etc.

Note that multiple messages can be sent at once and should be LF or CRLF delimited. This script is not necessarily intended to be a solution in itself, but is intended to be used as 'glue' to help ZoneMinder interface with other systems. It will almost certainly require some customisation before you can make any use of it. If all you want to do is generate alarms from external sources then using the ZoneMinder::SharedMem perl module is likely to be easier.

zmcamtool.pl This optional script is new for the upcoming 1.27 release of ZoneMinder. It is intended to make it easy to do the following: bring in new ptz controls and camera presets, convert existing monitors into presets, and export custom ptz controls and presets. For the initial release, this script is not integrated into the UI and must be called from the command line. Type "zmcamtool.pl -help" from the command line to get an explanation of the different arguments one can pass to the script.

zmcontrol-*.pl These are a set of example scripts which can be used to control Pan/Tilt/Zoom class cameras. Each script converts a set of standard parameters used for camera control into the actual protocol commands sent to the camera. If you are using a camera control protocol that is not in the shipped list then you will have to create a similar script though it can be created entirely separately from ZoneMinder and does not need to be named as these scripts are. Although the scripts are used to action commands originated from the web interface they can also be used directly or from other programs or scripts, for instance to implement periodic scanning to different presets.

zmtrack.pl This script is used to manage the experimental motion tracking feature. It is responsible for detecting that an alarm is taking place and moving the camera to point to the alarmed location, and then subsequently returning it to a defined standby location. As well as moving the camera it also controls when motion detection is suspended and restored so that the action of the camera tracking does not trigger endless further alarms which are not justified.

zm This is the (optional) ZoneMinder init script, see below for details.

zmeventnotification.pl This is an optional 3rd party real time event notification server that also provides push notifications for zmNinja as well as machine learning powered object/face-detection. Please see [Event Notification Server Documentation](#) for more details (Note that the machine learning components are optional, and are developed in Python3)

Finally, there are also a number of ZoneMinder perl modules included. These are used by the scripts above, but can also be used by your own or 3rd party scripts. Full documentation for most modules is available in 'pod' form via 'perldoc' but the general purpose of each module is as follows.

ZoneMinder.pm This is a general ZoneMinder container module. It includes the Base.pm, Config.pm Debug.pm, Database.pm, and SharedMem.pm modules described below. It also exports all of their symbols by default. If you use the other modules directly you have request which symbol tags to import.

ZoneMinder/Base.pm This is the base ZoneMinder perl module. It contains only simple data such as version information. It is included by all other ZoneMinder perl modules

ZoneMinder/Config.pm This module imports the ZoneMinder configuration from the database.

ZoneMinder/Debug.pm This module contains the defined Debug and Error functions etc, that are used by scripts to produce diagnostic information in a standard format.

ZoneMinder/Database.pm This module contains database access definitions and functions. Currently not a lot is in this module but it is included as a placeholder for future development.

ZoneMinder/Event.pm This module contains functions to load, manipulate, delete, copy, move events.

ZoneMinder/Filter.pm This module contains functions to load, execute etc filters.

ZoneMinder/SharedMem.pm This module contains standard shared memory access functions. These can be used to access the current state of monitors etc as well as issuing commands to the monitors to switch things on and off. This module effectively provides a ZoneMinder API.

ZoneMinder/ConfigAdmin.pm This module is a specialised module that contains the definition, and other information, about the various configuration options. It is not intended for use by 3rd parties.

ZoneMinder/Control/*.pm These modules contain implementations of the various PTZ protocols.

ZoneMinder/Trigger/*.pm These modules contain definitions of trigger channels and connections used by the zmtrigger.pl script. Although they can be used 'as is', they are really intended as examples that can be customised or specialised for different interfaces. Contributed modules for new channels or connections will be welcomed and included in future versions of ZoneMinder.

2.3 Getting Started

Having followed the [Installation Guide](#) for your distribution you should now be able to load the ZoneMinder web frontend. By default this will be with the Classic skin, below is an example of the page you should now see.

The screenshot shows the ZoneMinder web interface. At the top, there's a navigation bar with links: Console, Options, Log (highlighted), Groups, Filters, Cycle, Montage, Montage Review, Audit Events Report, and a RUNNING status button. Below the navigation bar, there's a status bar showing 'Low' priority, 'Load: 1.7', 'DB: 8/151', 'Storage: Default: 80% /dev/shm: 11%', and version 'v1.33.14'. Below the status bar, there's a search and filter section with fields for Name, Function, Status, Source, and Monitor. Below this is a table with columns: Name, Function, Source, Events, Hour, Day, Week, Month, Archived, Zones, and All. The table shows one row with '0B/s' in the Function column and '0' in the Events, Hour, Day, Week, Month, Archived, and Zones columns.

Name	Function	Source	Events	Hour	Day	Week	Month	Archived	Zones	All
	0B/s		0	0	0	0	0	0	0	

2.3.1 Setting Timezone

Previous versions of ZoneMinder required the user to set up Timezone correctly in `php.ini`. This is no longer the case. Starting 1.34, ZoneMinder allows you to specify the TimeZone in the UI. Please make sure it is set up correctly. The Timezone can be changed by selecting Options->System->Timezone

The screenshot shows the 'TIMEZONE' section of the ZoneMinder web interface. It features a dropdown menu with the selected value '(GMT-04:00) America, New York'. Below the dropdown, there's a note: 'The timezone that php should use. (?)'.

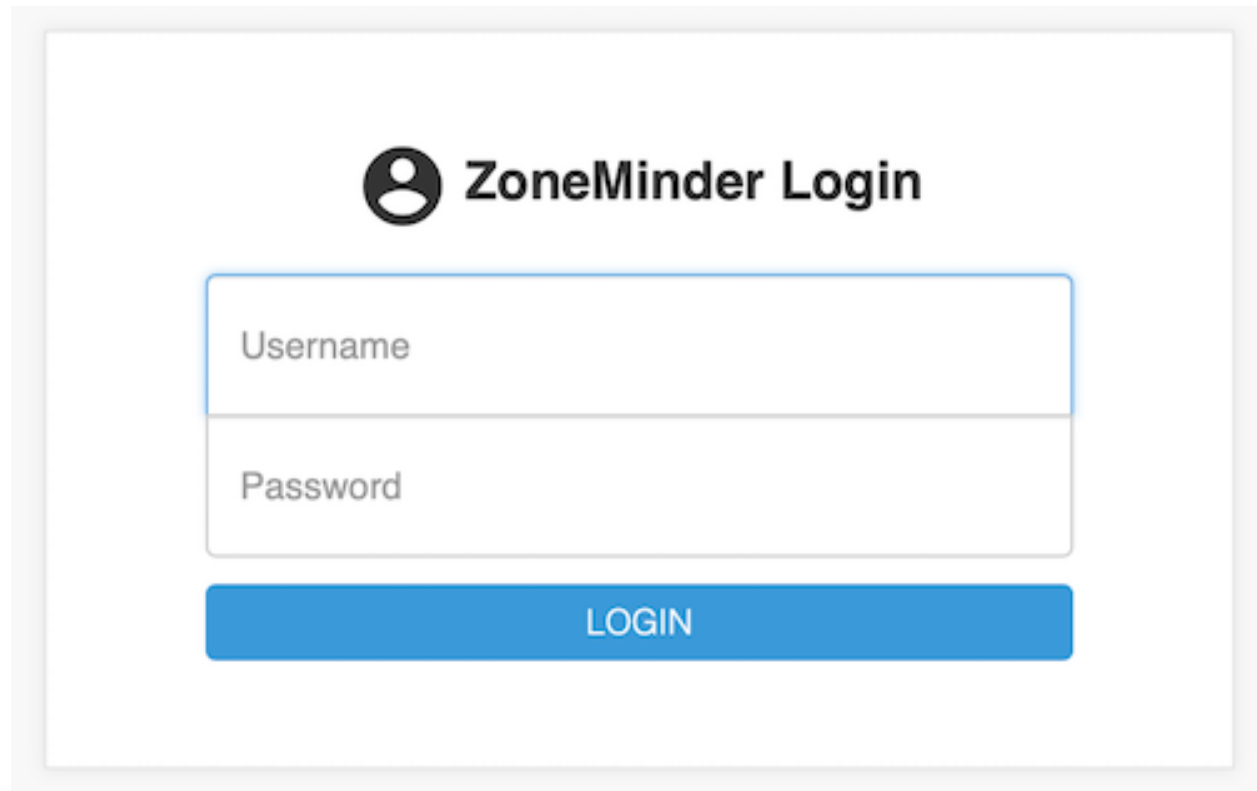
2.3.2 Enabling Authentication

We strongly recommend enabling authentication right away. There are some situations where certain users don't enable authentication, such as instances where the server is in a LAN not directly exposed to the Internet, and is only accessible via VPN etc., but in most cases, authentication should be enabled. So let's do that right away.

- Click on the Options link on the top bar of the web interface
- You will now be presented with a sidebar full of options. Click on the "System" link

The screenshot displays the ZoneMinder configuration interface. The left sidebar lists various system components, with 'System' currently selected. The main configuration area contains several settings, each with a label, a control element, and a description. Red arrows are drawn on the image to highlight specific settings that need to be modified for authentication: **OPT_USE_AUTH**, **AUTH_TYPE**, **AUTH_RELAY**, **AUTH_HASH_SECRET**, **AUTH_HASH_IPS**, and **AUTH_HASH_LOGINS**.

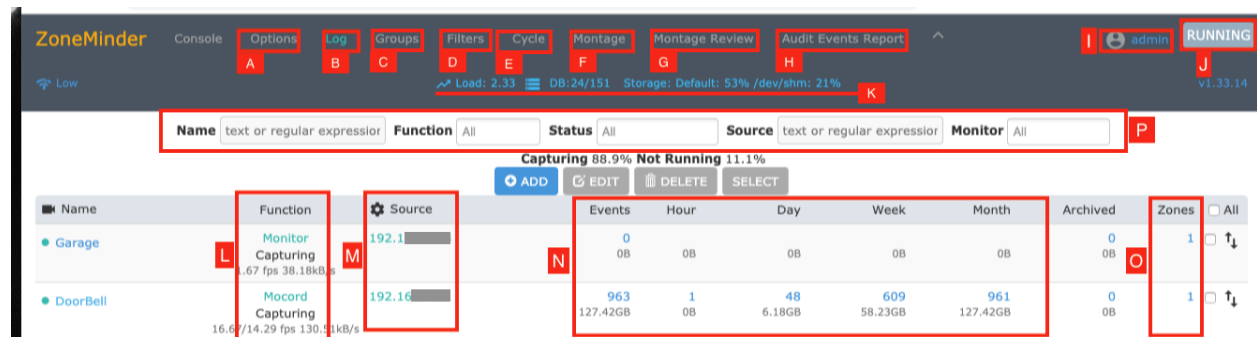
- The relevant portions to change are marked in red above
- Enable **OPT_USE_AUTH** - this automatically switches to authentication mode with a default user (more on that later)
- Select a random string for **AUTH_HASH_SECRET** - this is used to make the authentication logic more secure, so please generate your own string and make sure it is sufficiently randomized and long. Note that if you plan to use APIs with ZoneMinder (needed by zmNinja/other apps), it is mandatory that you have this field populated
- The other options highlighted above should already be set, but if not, please make sure they are
- Note that if you are planning to use zmNinja and plan to use ZM authentication, you must also:
 - set **AUTH_RELAY** to hashed
 - Enable **AUTH_HASH_LOGINS**
- Click on Save at the bottom and that's it! The next time you refresh that page, you will now be presented with a login screen. Job well done!



Note: The default login/password is “admin/admin”

2.3.3 Understanding the Web Console

Before we proceed, let's spend a few minutes understanding the key functions of the web console. For the sake of illustration, we are going to use a populated zoneminder configuration with several monitors and events.

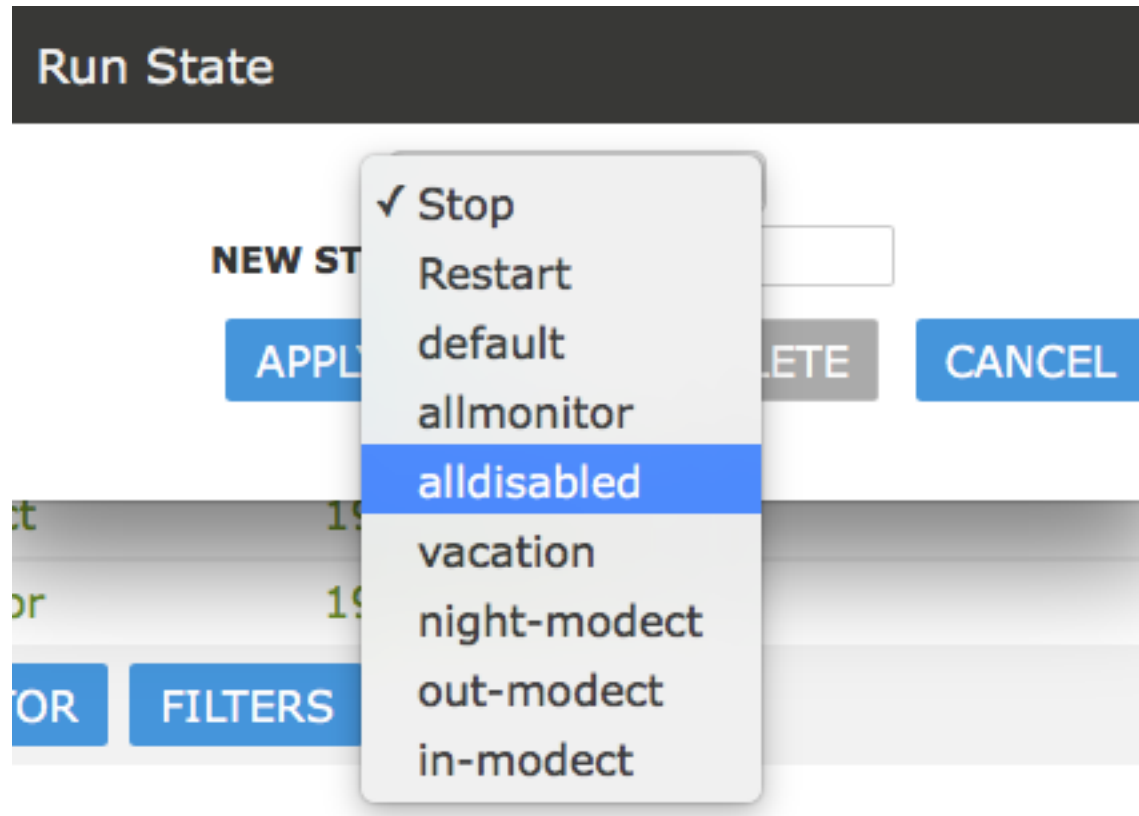


This screen is called the “console” screen in ZoneMinder and shows a summary of your monitors, associated events and more information.

- **A:** The options menu lets you configure many aspects of ZoneMinder. Refer to [Options](#).
- **B:** This brings up a color coded log window that shows various system and component level logs. This window is useful if you are trying to diagnose issues. Refer to [Logging](#).

- **C:** ZoneMinder allows you to group monitors for logical separation. This option lets you create new groups, associate monitors to them and edit/delete existing groups.
- **D:** Filters are a powerful mechanism to perform actions when certain conditions are met. ZoneMinder comes with some preset filters that keep a tab of disk space and others. Many users create their own filters for more advanced actions like sending emails when certain events occur and more. Refer to [Filtering Events](#).
- **E:** The Cycle option allows you to rotate between live views of each configured monitor.
- **F:** The Montage option shows a collage of your monitors. You can customize them including moving them around.
- **G:** Montage Review allows you to simultaneously view past events for different monitors. Note that this is a very resource intensive page and its performance will vary based on your system capabilities.
- **H:** Audit Events Report is more of a power user feature. This option looks for recording gaps in events and recording issues in mp4 files.
- **I:** This is the user you are currently logged in as.
- **J:** ZoneMinder allows you to maintain “run states”. If you click on the “Running” text, ZoneMinder brings up a popup that allows you to define additional “states” (referred to as runstates). A runstate is essentially a snapshot that records the state of each monitor and you can switch between states easily. For example, you might have a run state defined that switches all monitors to “monitor” mode in which they are not recording anything while another state that sets some of the monitors to “modect”. Why would you want this? A great example is to disable recording when you are at home and enable when you are away, based on time of day or other triggers. You can switch states by selecting an appropriate state manually, or do it automatically via cron jobs, for example. An example of using cron to automatically switch is provided in the [FAQ](#). More esoteric examples of switching run states based on phone location can be found [here](#).

Here is an example of multiple run states that I’ve defined. Each one of these runstates changes the mode of specific monitors depending on time of day and other conditions. Use your imagination to decide which conditions require state changes.



- **K:** This line shows you system health information
- **L:** This defines how Zoneminder will record events. There are various modes. In brief Modect == record if a motion is detected, Record = always record 24x7, Mocord = always record PLUS detect motion, Monitor = just provide a live view but don't record anytime, Nodect = Don't record till an external entity via zmtrigger tells Zoneminder to (this is advanced usage).
- **M:** This is the "source" column that tells you the type of the camera - if its an IP camera, a USB camera or more. In this example, they are all IP cameras. Green means the monitor is running. Red means there is something wrong with that camera.
- **N:** This is the core of ZoneMinder - recording events. It gives you a count of how many events were recorded over the hour, day, week, month.
- **O:** These are the "Zones". Zones are areas within the camera that you mark as 'hotspots' for motion detection. Simply put, when you first configure your monitors (cameras), by default Zoneminder uses the entire field of view of the camera to detect motion. You may not want this. You may want to create "zones" specifically for detecting motion and ignore others. For example, lets consider a room with a fan that spins. You surely don't want to consider the fan moving continuously a reason for triggering a record? Probably not - in that case, you'd leave the fan out while making your zones.
- **P:** This is a "visual filter" which lets you 'filter' the console display based on text you enter. While this may not be particularly useful for small systems, ZoneMinder is also used in mega-installations will well over 200+ cameras and this visual filter helps reduce the monitors you are seeing at one time.

2.3.4 Adding Monitors

Now that we have a basic understanding of the web console, lets go about adding a new camera (monitor). For this example, lets assume we have an IP camera that streams RTSP at LAN IP address 192.168.1.33.

Note

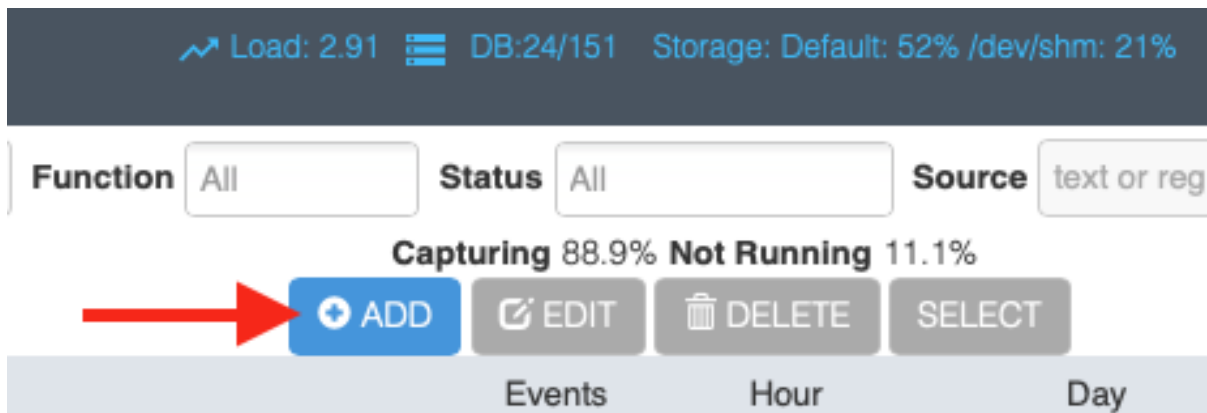
This is meant to be a simple example. For a more detailed explanation of other options available when creating a monitor, please see [Defining Monitors](#)

The first thing we will need to know is how to access that camera's video feed. You will need to consult your camera's manual or check their forum. Zoneminder community users also have a frequently updated list right [here](#) that lists information about many cameras. If you don't find your list there and can't seem to find it elsewhere, feel free to register and ask in the [user forums](#).

The camera we are using as an example here is a Foscam 9831W which is a 1280x960 RTSP camera, and the URL to access it's feed is `username:password@IPADDRESS:PORT/videoMain`

Let's get started:

Click on the "Add" button below:



This brings up the new monitor window:

Monitor - Garage (1)
Probe ONVIF Presets

General

Source

Storage

Timestamp

Buffers

Misc

Name	<input type="text" value="Garage"/>
Server	<div>None ▾</div>
Storage Area	<div>Default ▾</div>
Source Type	<div>Ffmpeg ▾</div>
Function	<div>Modect ▾</div>
Enabled	<input checked="" type="checkbox"/>
Linked Monitors (?)	<div>Select Some Op</div>
Groups	<div>Sel</div>
Analysis FPS	<div>5.00</div>
Maximum FPS (?)	<div></div>
Alarm Maximum FPS (?)	<div></div>
Reference Image Blend %ge	<div>6.25% (Indoor) ▾</div>
Alarm Reference Image Blend %ge	<div>6.25% ▾</div>
Triggers	None available

SAVE

CANCEL

- We've given it a name of 'Garage', because, well, its better than Monitor-1 and this is my Garage camera.
- There are various source types. As a brief introduction you'd want to use 'Local' if your camera is physically attached to your ZM server (like a USB camera, for example), and one of 'Remote', 'FFMpeg', 'Libvlc' or 'cURL' for a remote camera (not necessarily, but usually). For this example, let's go with 'FFMpeg'.

Note: As a thumb rule, if you have a camera accessible via IP and it does HTTP or RTSP, start with FFMpeg first and libvlc if it doesn't work (*Defining Monitors* covers other modes in more details). If you are wondering what 'File' does, well, ZoneMinder was built with compatibility in mind. Take a look at [this post](#) to see how file can be used for leisure reading.

- In this example, the Function is 'Modect', which means it will start recording if motion is detected on that camera feed. The parameters for what constitutes motion detected is specific in *Defining Zones*
- In Analysis FPS, we've put in 5FPS here. Note that you should not put an FPS that is greater than the camera

FPS. In my case, 5FPS is sufficient for my needs

Note: Leave Maximum FPS and Alarm Maximum FPS **empty** if you are configuring an IP camera. In older versions of ZoneMinder, you were encouraged to put a value here, but that is no longer recommended. Infact, if you see your feed going much slower than the feed is supposed to go, or you get a lot of buffering/display issues, make sure this is empty. If you need to control camera FPS, please do it directly on the camera (via its own web interface, for example)

- We are done for the General tab. Let's move to the next tab

Monitor - Garage (1)

General	Source	Storage	Timestamp	Buffers	Misc
Source Path		rtsp://[REDACTED]68.1.33:20005/			
Method (?)		TCP			
Options (?)					
DecoderHWAccelName (?)		cuda			
DecoderHWAccelDevice (?)		cuid			
Target colorspace		32 bit colour			
Capture Resolution (pixels)		1280 960 1280x960 960p			
Preserve Aspect Ratio		<input type="checkbox"/>			
Orientation		Normal			
Deinterlacing		Disabled			
<div>SAVE CA</div>					

ONLY IF YOU
A GPU AND FF
IS COMPILE
SUPPORT IT.
EMPTY IF NO

- Let's select a protocol of RTSP and a remote method of RTP/RTSP (this is an RTSP camera)
- Note that starting ZM 1.34, GPUs are supported. In my case, I have an NVIDIA GeForce GTX1050i. These `cuda` and `cuid` parameters are what my system supports to use the NVIDIA hardware decoder and GPU resources. If you don't have a GPU, or don't know how to configure your ffmpeg to support it, leave it empty for now. In future, we will add a section on how to set up a GPU

Todo: add GPU docs

That's pretty much it. Click on Save. We are not going to explore the other tabs in this simple guide.

You now have a configured monitor:

Name	Function	Source
Garage	Monitor Capturing 5.00 fps 71.94kB/s	192.168.1.33

And then, finally, to see if everything works, if you click on the garage monitor you just added, you should be able to see its live feed. If you don't, inspect your webserver logs and your ZoneMinder logs to see what is going on.

2.3.5 Switching to another theme

Todo: Fix theme text after I clearly understand that System->CSS is doing

When you first install ZoneMinder, you see is what is called a "classic" skin. Zoneminder has a host of configuration options that you can customize over time. This guide is meant to get you started the easiest possible way, so we will not go into all the details. However, it is worthwhile to note that Zoneminder also has a 'flat' theme that depending on your preferences may look more modern. So let's use that as an example of introducing you to the Options menu

- Click on the Options link on the top right of the web interface in the image above
- This will bring you to the options window as shown below. Click on the "System" tab and then select the "flat" option for CSS_DEFAULT as shown below

10.211.55.4/zm/?view=options

Name	Description	Value
SKIN_DEFAULT	Default skin used by web interface (?)	<input checked="" type="radio"/> classic <input type="radio"/> mobile <input type="radio"/> xml
CSS_DEFAULT	Default set of css files used by web interface (?)	<input type="radio"/> classic <input type="radio"/> dark <input checked="" type="radio"/> flat
LANG_DEFAULT	Default language used by web interface (?)	en_gb
OPT_UISC_AUTH	Authenticate user logins to ZoneMinder (?)	<input type="checkbox"/>

- Click Save at the bottom

Now, switch to the "Display" tab and also select "Flat" there like so:

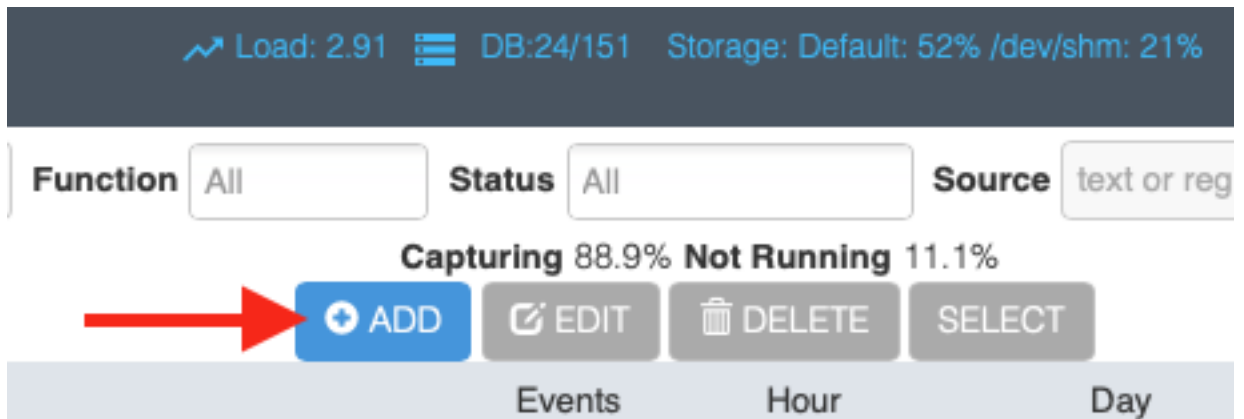
Options

Name	Description	Value
ZM_SKIN	Change the default skin for this computer	classic
ZM_CSS	Change the default css for this computer	flat

Save Cancel

Your screen will now look like this:

Congratulations! You now have a modern looking interface.



2.3.6 Conclusion

This was a quick ‘Getting Started’ guide where you were introduced to the very basics of how to add a monitor (camera). We’ve skipped many details to keep this concise. Please refer to [Defining Monitors](#) for many other customization details.

2.4 Defining Monitors

To use ZoneMinder properly you need to define at least one Monitor. Essentially, a monitor is associated with a camera and can continually check it for motion detection and such like.

You can access the monitor window by clicking on the “Add New Monitor” button, or by clicking on the “Source” column of a predefined monitor.

Monitor - FirstLevel (1)
Probe
Presets

General
Source
Timestamp
Buffers
Control

Misc

Name
FirstLevel

Source Type
Remote

Function
Modect

Enabled
☐

Garage

There are a small number of camera setups that ZoneMinder knows about and which can be accessed by clicking on the 'Presets' link. Selecting one of the presets will fill in the monitor configuration with appropriate values but you will still need to enter others and confirm the preset settings. Here is an example of the presets window:

Monitor Preset

Select an appropriate preset from the list below.

Please note that this may overwrite any values you already have configured for the current monitor.

Preset
Choose Preset
Axis IP, 320x240, mpjpeg
Axis IP, 320x240, mpjpeg, max 5 FPS
Axis IP, 320x240, jpeg
Axis IP, 320x240, jpeg, max 5 FPS
Axis IP, 640x480, mpjpeg

The options are divided into a set of tabs to make it easier to edit. You do not have to ‘save’ to change to different tab so you can make all the changes you require and then click ‘Save’ at the end. The individual options are explained in a little more detail below,

2.4.1 Monitor Tab

Name The name for your monitor. This should be made up of alphanumeric characters (a-z,A-Z,0-9) and hyphen (-) and underscore(_) only. Whitespace is not allowed.

Server Multi-Server implementation allows the ability to define multiple ZoneMinder servers sharing a single database. When servers are configured this setting allows you nominate the server for each monitor.

Source Type This determines whether the camera is a local one attached to a physical video or USB port on your machine, a remote network camera or an image source that is represented by a file (for instance periodically downloaded from a alternate location). Choosing one or the other affects which set of options are shown in the Source tab.

Function This essentially defines what the monitor is doing. This can be one of the following;

- None – The monitor is currently disabled. No streams can be viewed or events generated. Nothing is recorded.
- Monitor – The monitor is only available for live streaming. No image analysis is done so no alarms or events will be generated, and nothing will be recorded.
- Modect – or MOtion DEteCTtion. All captured images will be analysed and events generated with recorded video where motion is detected.
- Record – The monitor will be continuously recorded. Events of a fixed-length will be generated regardless of motion, analogous to a conventional time-lapse video recorder. No motion detection takes place in this mode.
- Mocord – The monitor will be continuously recorded, with any motion being highlighted within those events.
- Nodect – or No DEteCTtion. This is a special mode designed to be used with external triggers. In Nodect no motion detection takes place but events are recorded if external triggers require it.

Generally speaking it is best to choose ‘Monitor’ as an initial setting here.

Enabled The enabled field indicates whether the monitor should be started in an active mode or in a more passive state. You will nearly always want to check this box, the only exceptions being when you want the camera to be enabled or disabled by external triggers or scripts. If not enabled then the monitor will not create any events in response to motion or any other triggers.

Linked Monitors This field allows you to select other monitors on your system that act as triggers for this monitor. So if you have a camera covering one aspect of your property you can force all cameras to record while that camera detects motion or other events. You can either directly enter a comma separated list of monitor ids or click on ‘Select’ to choose a selection. Be very careful not to create circular dependencies with this feature however you will have infinitely persisting alarms which is almost certainly not what you want! To unlink monitors you can ctrl-click.

Maximum FPS

Warning: Unless you know what you are doing, please leave this field empty, especially if you are configuring a network camera. More often than not, putting a value here adversely affects recording.

On some occasions you may have one or more cameras capable of high capture rates but find that you generally do not require this performance at all times and would prefer to lighten the load on your server. This option permits you to limit the maximum capture rate to a specified value. This may allow you to have more cameras supported on your system by reducing the CPU load or to allocate video bandwidth unevenly between cameras sharing the same video device. This value is only a rough guide and the lower the value you set the less close the actual FPS may approach it especially on shared devices where it can be difficult to synchronise two or more different capture rates precisely. This option controls the maximum FPS in the circumstance where no alarm is occurring only.

This feature is limited and will only work under the following conditions:

1. Local cameras
2. Remote (IP) cameras in snapshot or jpeg mode **only**

Using this field for video streams from IP cameras will cause undesirable results when the value is equal to or less than the frame rate from the camera. Note that placing a value higher than the camera's frame rate is allowed and can help prevent cpu spikes when communication from the camera is lost.

Alarm Maximum FPS

Warning: Unless you know what you are doing, please leave this field empty, especially if you are configuring a network camera. More often than not, putting a value here adversely affects recording.

If you have specified a Maximum FPS it may be that you don't want this limitation to apply when your monitor is recording motion or other event. This setting allows you to override the Maximum FPS value if this circumstance occurs. As with the Maximum FPS setting leaving this blank implies no limit so if you have set a maximum fps in the previous option then when an alarm occurs this limit would be ignored and ZoneMinder would capture as fast as possible for the duration of the alarm, returning to the limited value after the alarm has concluded. Equally you could set this to the same, or higher (or even lower) value than Maximum FPS for more precise control over the capture rate in the event of an alarm.

IMPORTANT: This field is subject to the same limitations as the Maximum FPS field. Ignoring these limitations will produce undesirable results.

Reference Image Blend %ge Each analysed image in ZoneMinder is a composite of previous images and is formed by applying the current image as a certain percentage of the previous reference image. Thus, if we entered the value of 10 here, each image's part in the reference image will diminish by a factor of 0.9 each time round. So a typical reference image will be 10% the previous image, 9% the one before that and then 8.1%, 7.2%, 6.5% and so on of the rest of the way. An image will effectively vanish around 25 images later than when it was added. This blend value is what is specified here and if higher will make slower progressing events less detectable as the reference image would change more quickly. Similarly events will be deemed to be over much sooner as the reference image adapts to the new images more quickly. In signal processing terms the higher this value the steeper the event attack and decay of the signal. It depends on your particular requirements what the appropriate value would be for you but start with 10 here and adjust it (usually down) later if necessary.

Triggers This small section lets you select which triggers will apply if the run mode has been set to 'triggered' above. The most common trigger is X10 and this will appear here if you indicated that your system supported it during installation. Only X10 is supported as a shipped trigger with ZoneMinder at present but it is possible that other triggers will become available as necessary. You can also just use 'cron' jobs or other mechanisms to actually control the camera and keep them completely outside of the ZoneMinder settings. The zmtrigger.pl script is also available to implement custom external triggering.

2.4.2 Source Tab

FFmpeg

This is the **recommended** source type for most modern ip cameras.

Source Path Use this field to enter the full URL of the stream or file your camera supports. This is usually an RTSP url. There are several methods to learn this:

- Check the documentation that came with your camera
- Look for your camera in the hardware compatibility list in the [hardware compatibility wiki](#)
- Try ZoneMinder's new ONVIF probe feature
- Download and install the [ONVIF Device Manager](#) onto a Windows machine
- Use Google to find third party sites, such as ispy, which document this information

Source Colours Specify the amount of colours in the captured image. 32 bit is the preferred choice here. Unlike with local cameras changing this has no controlling effect on the remote camera itself so ensure that your camera is actually capturing to this palette beforehand.

Capture Width/Height Make sure you enter here the same values as they are in the remote camera's internal setting.

Keep aspect ratio As per local devices.

Orientation As per local devices.

LibVLC

The fields for the LibVLC source type are configured the same way as the ffmpeg source type. We recommend only using this source type if issues are experienced with the ffmpeg source type.

cURL

Local

Device Path/Channel Enter the full path to the device file that your camera is attached to, e.g. /dev/video0. Some video devices, e.g. BTTV cards support multiple cameras on one device so in this case enter the channel number in the Channel box or leave it at zero if you're using a USB camera or one with just one channel. Look in Supported Hardware section, how to see if your capture card or USB webcam is supported or not, and what extra settings you may have to do, to make it work.

Device Format Enter the video format of the video stream. This is defined in various system files (e.g. /usr/include/linux/videodev.h) but the two most common are 0 for PAL and 1 for NTSC.

Capture Palette Finally for the video part of the configuration enter the colour depth. ZoneMinder supports a handful of the most common palettes, so choose one here. If in doubt try 32 bit colour first, then 24 bit colour, then grey. If none of these work very well, and your camera is local, then YUV420P or one of the others probably will. There is a slight performance penalty when using palettes other than 32, 24, or grey palettes as an internal conversion is involved. Recent versions of ZoneMinder support 32bit colour. This capture palette provides a performance boost when used on all modern Intel-based processors.

Capture Width/Height The dimensions of the video stream your camera will supply. If your camera supports several just enter the one you'll want to use for this application, you can always change it later. However I would recommend starting with no larger than 320x240 or 384x288 and then perhaps increasing and seeing how performance is affected. This size should be adequate in most cases. Some cameras are quite choosy about the sizes you can use here so unusual sizes such as 197x333 should be avoided initially.

Keep aspect ratio When typing in the dimensions of monitors you can click this checkbox to ensure that the width stays in the correct ratio to the height, or vice versa. It allows height to be calculated automatically from width (or vice versa) according to preset aspect ratio. This is preset to 4:3 but can be amended globally via the Options->Config->ZM_DEFAULT_ASPECT_RATIO setting. Aside from 4:3 which is the usual for network and analog cameras another common setting is 11:9 for CIF (352x288) based sources.

Orientation If your camera is mounted upside down or at right angles you can use this field to specify a rotation that is applied to the image as it is captured. This incurs an additional processing overhead so if possible it is better to mount your camera the right way round if you can. If you choose one of the rotation options remember to switch the height and width fields so that they apply, e.g. if your camera captures at 352x288 and you choose 'Rotate Right' here then set the height to be 352 and width to be 288. You can also choose to 'flip' the image if your camera provides mirrored input.

Remote

Remote Protocol Choices are currently HTTP and RTSP. Before RTSP became the industry standard, many ip cameras streamed directly from their web portal. If you have an ip camera that does not speak RTSP then choose HTTP here. **If you camera does speak RTSP then you should change your source type to ffmpeg instead of selecting RTSP here.** The Remote -> RTSP method is no longer being maintained and may go away at some point in the future.

Remote Method When HTTP is the Remote Protocol, your choices are Simple and Regexp. Most should choose Simple. When RTSP is the Remote Protocol, your choices are RTP/Unicast, RTP/Multicast, RTP/RTSP, RTP,RTSP,HTTP. Try each of these to determine which works with your camera. Most cameras will use either RTP/Unicast (UDP) or RTP/RTSP (TCP).

Remote Host/Port/Path Use these fields to enter the full URL of the camera. Basically if your camera is at `http://camserver.home.net:8192/cameras/camera1.jpg` then these fields will be `camserver.home.net`, `8192` and `/cameras/camera1.jpg` respectively. Leave the port at 80 if there is no special port required. If you require authentication to access your camera then add this onto the host name in the form `<username>:<password>@<hostname>.com`. This will usually be 32 or 24 bit colour even if the image looks black and white. Look in Supported Hardware > Network Cameras section, how to obtain these strings that may apply to your camera.

Remote Image Colours Specify the amount of colours in the captured image. Unlike with local cameras changing this has no controlling effect on the remote camera itself so ensure that your camera is actually capturing to this palette beforehand.

Capture Width/Height Make sure you enter here the same values as they are in the remote camera's internal setting.

Keep aspect ratio As per local devices.

Orientation As per local devices.

For an example to setup a MPEG-4 camera see: [How_to_Setup_an_Axis211A_with_MPEG-4_streaming](#)

File

File Path Enter the full path to the file to be used as the image source.

File Colours Specify the amount of colours in the image. Usually 32 bit colour.

Capture Width/Height As per local devices.

Keep aspect ratio As per local devices.

Orientation As per local devices.

WebSite

This Source Type allows one to configure an arbitrary website as a non-recordable, fully interactive, monitor in ZoneMinder. Note that sites with self-signed certificates will not display until the end user first manually navigates to the site and accepts the unsigned certificate. Also note that some sites will set an X-Frame option in the header, which discourages their site from being displayed within a frame. ZoneMinder will detect this condition and present a warning in the log. When this occurs, the end user can choose to install a browser plugin or extension to work around this issue.

Website URL Enter the full http or https url to the desired website.

Width (pixels) Chose a desired width in pixels that gives an acceptable appearance. This may take some experimentation.

Height (pixels) Chose a desired height in pixels that gives an acceptable appearance. This may take some experimentation.

Web Site Refresh If the website in question has static content, optionally enter a time period in seconds for ZoneMinder to refresh the content.

2.4.3 Storage Tab

The storage section allows for each monitor to configure if and how video and audio are recorded.

Save JPEGs Records video in individual JPEG frames. Storing JPEG frames requires more storage space than h264 but it allows to view an event anytime while it is being recorded.

- Disabled – video is not recorded as JPEG frames. If this setting is selected, then “Video Writer” should be enabled otherwise there is no video recording at all.
- Frames only – video is recorded in individual JPEG frames.
- Analysis images only (if available) – video is recorded in individual JPEG frames with an overlay of the motion detection analysis information. Note that this overlay remains permanently visible in the frames.
- Frames + Analysis images (if available) – video is recorded twice, once as normal individual JPEG frames and once in individual JPEG frames with analysis information overlaid.

Video Writer Records video in real video format. It provides much better compression results than saving JPEGs, thus longer video history can be stored.

- Disabled – video is not recorded in video format. If this setting is selected, then “Save JPEGs” should be enabled otherwise there is no video recording at all.
- X264 Encode – the video or picture frames received from the camera are transcoded into h264 and stored as a video. This option is useful if the camera cannot natively stream h264.
- H264 Camera Passthrough – this option assumes that the camera is already sending an h264 stream. Video will be recorded as is, without any post-processing in zoneminder. Video characteristics such as bitrate, encoding mode, etc. should be set directly in the camera.

Recording Audio Check the box labeled “Whether to store the audio stream when saving an event.” in order to save audio (if available) when events are recorded.

2.4.4 Timestamp Tab

Timestamp Label Format This relates to the timestamp that is applied to each frame. It is a ‘strftime’ style string with a few extra tokens. You can add %f to add the decimal hundredths of a second to the frame timestamp,

so %H:%M:%S.%f will output time like 10:45:37.45. You can also use %N for the name of the monitor and %Q which will be filled by any of the 'show text' detailed in the zmtriggers.pl section.

Timestamp Label X/Y The X and Y values determine where to put the timestamp. A value of 0 for the X value will put it on the left side of the image and a Y value of 0 will place it at the top of the image. To place the timestamp at the bottom of the image use a value eight less than the image height.

2.4.5 Buffers Tab

Image Buffer Size This option determines how many frames are held in the ring buffer at any one time. The ring buffer is the storage space where the last 'n' images are kept, ready to be resurrected on an alarm or just kept waiting to be analysed. It can be any value you like with a couple of provisos, (see next options). However it is stored in shared memory and making it too large especially for large images with a high colour depth can use a lot of memory. A value of no more than 50 is usually ok. If you find that your system will not let you use the value you want it is probably because your system has an arbitrary limit on the size of shared memory that may be used even though you may have plenty of free memory available. This limit is usually fairly easy to change, see the Troubleshooting section for details.

Warm-up Frames This specifies how many frames the analysis daemon should process but not examine when it starts. This allows it to generate an accurate reference image from a series of images before looking too carefully for any changes. I use a value of 25 here, too high and it will take a long time to start, too low and you will get false alarms when the analysis daemon starts up.

Pre/Post Event Image Buffer These options determine how many frames from before and after an event should be preserved with it. This allows you to view what happened immediately prior and subsequent to the event. A value of 10 for both of these will get you started but if you get a lot of short events and would prefer them to run together to form fewer longer ones then increase the Post Event buffer size. The pre-event buffer is a true buffer and should not really exceed half the ring buffer size. However the post-event buffer is just a count that is applied to captured frames and so can be managed more flexibly. You should also bear in mind the frame rate of the camera when choosing these values. For instance a network camera capturing at 1FPS will give you 10 seconds before and after each event if you chose 10 here. This may well be too much and pad out events more than necessary. However a fast video card may capture at 25FPS and you will want to ensure that this setting enables you to view a reasonable time frame pre and post event.

Stream Replay Image Buffer The number of frames buffered to allow pausing and rewinding of the stream when live viewing a monitor. A value of 0 disables the feature. Frames are buffered to ZM_PATH_SWAP. If this path points to a physical drive, a lot of IO will be caused during live view / montage. If you experience high system load in those situations, either disable the feature or use a RAM drive for ZM_PATH_SWAP.

Alarm Frame Count This option allows you to specify how many consecutive alarm frames must occur before an alarm event is generated. The usual, and default, value is 1 which implies that any alarm frame will cause or participate in an event. You can enter any value up to 16 here to eliminate bogus events caused perhaps by screen flickers or other transients. Values over 3 or 4 are unlikely to be useful however. Please note that if you have statistics recording enabled then currently statistics are not recorded for the first 'Alarm Frame Count'-1 frames of an event. So if you set this value to 5 then the first 4 frames will be missing statistics whereas the more usual value of 1 will ensure that all alarm frames have statistics recorded.

2.4.6 Control Tab

Note: This tab and its options will only appear if you have selected the ZM_OPT_CONTROL option to indicate that your system contains cameras which are able to be controlled via Pan/Tilt/Zoom or other mechanisms. See the Camera Control section elsewhere in this document for further details on camera control protocols and methods.

Controllable Check this box to indicate your camera can be controlled.

Control Type Select the control type that is appropriate for your camera. ZoneMinder ships with a small number of predefined control protocols which will work with some cameras without modification but which may have to be amended to function with others. Choose the edit link to create new control types or to edit the existing ones.

Control Device This is the device that is used to control your camera. This will normally be a serial or similar port. If your camera is a network camera, you will generally not need to specify a control device.

Control Address This is the address of your camera. Some control protocols require that each camera is identified by a particular, usually numeric, id. If your camera uses addressing then enter the id of your camera here. If your camera is a network camera then you will usually need to enter the hostname or IP address of it here. This is ordinarily the same as that given for the camera itself.

Auto Stop Timeout Some cameras only support a continuous mode of movement. For instance you tell the camera to pan right and then when it is aligned correctly you tell it to stop. In some cases it is difficult to time this precisely over a web interface so this option allows you to specify an automatic timeout where the command will be automatically stopped. So a value of 0.25 here can tell the script to stop moving a quarter of a second after starting. This allows a more precise method of fine control. If this value is left blank or at zero it will be ignored, if set then it will be used as the timeout however it will only be applied for the lower 25% of possible speed ranges. In other words if your camera has a pan speed range of 1 to 100 then selecting to move at 26 or over will be assumed to imply that you want a larger movement that you can control yourself and no timeout will be applied. Selecting motion at lower speeds will be interpreted as requiring finer control and the automatic timeout will be invoked.

Track Motion This and the following four options are used with the experimental motion function. This will only work if your camera supports mapped movement modes where a point on an image can be mapped to a control command. This is generally most common on network cameras but can be replicated to some degree on other cameras that support relative movement modes. See the Camera Control section for more details. Check this box to enable motion tracking.

Track Delay This is the number of seconds to suspend motion detection for following any movement that the camera may make to track motion.

Return Location If your camera supports a 'home' position or presets you can choose which preset the camera should return to after tracking motion.

Return Delay This is the delay, in seconds, once motion has stopped being detected, before the camera returns to any defined return location.

2.4.7 X10 Tab

Note: This tab and its options will only appear if you have indicated that your system supports the X10 home automation protocol during initial system configuration.

X10 Activation String The contents of this field determine when a monitor starts and/or stops being active when running in 'Triggered' mode and with X10 triggers. The format of this string is as follows,

- **n** : If you simply enter a number then the monitor will be activated when an X10 ON signal for that unit code is detected and will be deactivated when an OFF signal is detected.
- **!n** : This inverts the previous mode, e.g. **!5** means that the monitor is activated when an OFF signal for unit code 5 is detected and deactivated by an ON.
- **n+** : Entering a unit code followed by **+** means that the monitor is activated on receipt of a ON signal for that unit code but will ignore the OFF signal and as such will not be deactivated by this instruction. If you prepend a **!** as per the previous definition it similarly inverts the mode, i.e. the ON signal deactivates the monitor.
- **n+<seconds>** : As per the previous mode except that the monitor will deactivate itself after the given number of seconds.

- **n-** : Entering a unit code followed by - means that the monitor is deactivated on receipt of a OFF signal for that unit code but will ignore the ON signal and as such will not be activated by this instruction. If you prepend a '!' as per the previous definition it similarly inverts the mode, i.e. the OFF signal activates the monitor.
- **n-<seconds>** : As per the previous mode except that the monitor will activate itself after the given number of seconds.

You can also combine several of these expressions to by separating them with a comma to create multiple circumstances of activation. However for now leave this blank.

X10 Input Alarm String This has the same format as the previous field but instead of activating the monitor with will cause a forced alarm to be generated and an event recorded if the monitor is Active. The same definition as above applies except that for activated read alarmed and for deactivated read unalarmed(!). Again leave this blank for now.

X10 Output Alarm String This X10 string also has the same format as the two above options. However it works in a slightly different way. Instead of ZoneMinder reacting to X10 events this option controls how ZoneMinder emits X10 signals when the current monitor goes into or comes out of the alarm state. Thus just entering a number will cause the ON signal for that unit code to be sent when going into alarm state and the OFF signal when coming out of alarm state. Similarly 7+30 will send the unit code 7 ON signal when going into alarm state and the OFF signal 30 seconds later regardless of state. The combination of the X10 instruction allows ZoneMinder to react intelligently to, and also assume control of, other devices when necessary. However the indiscriminate use of the Input Alarm and Output Alarm signals can cause some horrendous race conditions such as a light going on in response to an alarm which then causes an alarm itself and so on. Thus some circumspection is required here. Leave this blank for now anyway.

2.4.8 Misc Tab

Event Prefix By default events are named 'Event-<event id>', however you are free to rename them individually as you wish. This option lets you modify the event prefix, the 'Event-' part, to be a value of your choice so that events are named differently as they are generated. This allows you to name events according to which monitor generated them.

Section Length This specifies the length (in seconds) of any fixed length events produced when the monitor function is 'Record' or 'Mocord'. Otherwise it is ignored. This should not be so long that events are difficult to navigate nor so short that too many events are generated. A length of between 300 and 900 seconds I recommended.

Frame Skip This setting also applies only to the 'Record' or 'Mocord' functions and specifies how many frames should be skipped in the recorded events. The default setting of zero results in every captured frame being saved. Using a value of one would mean that one frame is skipped between each saved, two means that two frames are skipped between each saved frame etc. An alternate way of thinking is that one in every 'Frame Skip + 1' frames is saved. The point of this is to ensure that saved events do not take up too much space unnecessarily whilst still allowing the camera to capture at a fairly high frame rate. The alternate approach is to limit the capture frame rate which will obviously affect the rate at which frames are saved.

FPS Report Interval How often the current performance in terms of Frames Per Second is output to the system log. Not used in any functional way so set it to maybe 1000 for now. If you watch /var/log/messages (normally) you will see this value being emitted at the frequency you specify both for video capture and processing.

Default Scale If your monitor has been defined with a particularly large or small image size then you can choose a default scale here with which to view the monitor so it is easier or more visible from the web interface.

Web Colour Some elements of ZoneMinder now use colours to identify monitors on certain views. You can select which colour is used for each monitor here. Any specification that is valid for HTML colours is valid here, e.g. 'red' or '#ff0000'. A small swatch next to the input box displays the colour you have chosen.

Embed EXIF data into image: Embeds EXIF data into each jpeg frame

Todo: what about mp4s?

2.5 Defining Zones

The next important thing to do with a new monitor is set up Zones for it to use. By default you'll already have one generated for you when you created your monitor (the default zone is the full area captured by the monitor) but you might want to modify it or add others.

Click on the Zones column for your monitor and you should see a small popup window appear which contains an image from your camera overlain with a stippled pattern representing your zone. In the default case this will cover the whole image. The colour of the zones appearing here is determined by what type they are. The default zone is Active and so will be red, Inclusive zones are orange, exclusive zones are purple, preclusive zones are blue and inactive zones are white.

Beneath the zones image will be a table containing a listing of your zones. Clicking on either the relevant bit of the image or on the Id or Name in the table will bring up another window where you can edit the particulars for your Zones. For more information on defining or editing a zone, see Defining Zones.

Zone configuration and tuning are important when running in the motion detection modes to avoid storing, sorting through, or being alerted on uninteresting video data. Configuring a zone involves setting some basic parameters, as well as choosing an alarm check method and tuning their associated detection parameters.

The Zone view is split into two main areas, on the left is the options area and on the right is the zone drawing area. A default or new zone will cover the whole drawing area and will overlay any other zones you already have on there. Unlike the previous zones image, the current zone is coloured green, other zones will be orange regardless of type. The smaller the zone, the less processing time it takes to examine it.

2.5.1 Basic parameters

Name Each Zone can be named for reference purposes. It is used for logging and debugging. Choose a name that helps you identify your zones.

Type This is one of the more important concepts in ZoneMinder and there are six to choose from.

- **Active** Triggers an alarm when motion is detected within it. This is the zone type you'll use most often, and which will be set for your default zone. Only Active and Exclusive zones can trigger an alarm.
- **Inclusive** This zone type can be used for any zones that you want to trigger an alarm only if at least one other Active zone has already triggered one. This might be for example to cover an area of the image like a plant or tree which moves a lot and which would trigger lots of alarms. Perhaps this is behind an area you'd like to monitor though, in this case you'd create an active zone covering the non-moving parts and an inclusive zone covering the tree perhaps with less sensitive detection settings also. If something triggered an alarm in the Active zone and also in the Inclusive zone they would both be registered and the resulting alarm would be that much bigger than if you had blanked it out altogether.
- **Exclusive** Triggers an alarm when motion is detected within it, as long as no alarms have already been triggered in an Active zone. This is the most specialized of the zone types. For instance in the camera covering my garden I keep watch for a hedgehog that visits most nights and scoffs the food out of my cats bowls. By creating a sensitive Exclusive zone in that area I can ensure that a hedgehog alarm will only trigger if there is activity in that small area. If something much bigger occurs, like someone walking by it will trigger a regular alarm and not one from the Exclusive zone. Thus I can ensure I get alarms for big events and also special small events but not the noise in between.

- **Preclusive** This zone type is relatively recent. It is called a Preclusive zone because if it is triggered it actually precludes an alarm being generated for that image frame. So motion or other changes that occur in a Preclusive zone will have the effect of ensuring that no alarm occurs at all. The application for this zone type is primarily as a shortcut for detecting general large-scale lighting or other changes. Generally this may be achieved by limiting the maximum number of alarm pixels or other measure in an Active zone. However in some cases that zone may cover an area where the area of variable illumination occurs in different places as the sun and/or shadows move and it thus may be difficult to come up with general values. Additionally, if the sun comes out rapidly then although the initial change may be ignored in this way as the reference image catches up an alarm may ultimately be triggered as the image becomes less different. Using one or more Preclusive zones offers a different approach. Preclusive zones are designed to be fairly small, even just a few pixels across, with quite low alarm thresholds. They should be situated in areas of the image that are less likely to have motion occur such as high on a wall or in a corner. Should a general illumination change occur they would be triggered at least as early as any Active zones and prevent any other zones from generating an alarm. Obviously careful placement is required to ensure that they do not cancel any genuine alarms or that they are not so close together that any motion just hops from one Preclusive zone to another. Preclusive zones may also be used to reduce processing time by situating one over an Active zone. The Preclusive zone is processed first; if it is small, and is triggered, the rest of the zone/image will not be processed. See Extend Alarm Frame Count below for a way to hold the preclusive zone active for an extended period.
- **Inactive** Suppresses the detection of motion within it. This can be layered on top of any other zone type, preventing motion within the Inactive zone from being effective for any other zone type. Use inactive zones to cover areas in which nothing notable will ever happen or where you get false alarms that don't relate to what you are trying to monitor. Inactive zones may be overlaid on other zones to blank out areas, and are processed first (with the exception of Privacy zones, see below). As a general practice, you should try and make zones abut each other instead of overlapping to avoid repeated duplicate processing of the same area.
- **Privacy** Blackens the pixels within it. This can be used if you want to hide some regions in the image if the situation does not allow another solution. This zone type is different to all the others in that it gets processed as soon as possible during capture (even before the timestamp gets into the image) and not in the analyzing process. So if you add, change or delete a Privacy zone, you don't see the changes in the image until the capture process gets restarted. This will be done automatically, but needs a few seconds.

Preset The preset chooser sets sensible default values based on computational needs (fast v. best) and sensitivity (low, medium, high.) It is not required that you select a preset, and you can alter any of the parameters after choosing a preset. For a small number of monitors with ZoneMinder running on modern equipment, Best, high sensitivity can be chosen as a good starting point.

It is important to understand that the available presets are intended merely as a starting point. Since every camera's view is unique, they are not guaranteed to work properly in every case. Presets tend to work acceptably for indoor cameras, where the objects of interest are relatively close and there typically are few or no unwanted objects moving within the cameras view. Presets, on the other hand, tend to not work acceptably for outdoor cameras, where the field of view is typically much wider, objects of interest are farther away, and changing weather patterns can cause false triggers. For outdoor cameras in particular, you will almost certainly have to tune your motion detection zone to get desired results. Please refer to [this guide](#) to learn how to do this.

Units

- **Pixels** - Selecting this option will allow many of the following values to be entered (or viewed) in units of pixels.
- **Percentage** - Selecting this option will allow may of the following values to be entered (or viewed) as a percentage. The sense of the percentage values refers to the area of the zone and not the image as a whole. This makes trying to work out necessary sizes rather easier.

Region points



The sample region shown to the right shows a region defined by 6 control points. The shape of the region causes the check methods to ignore the sidewalk and areas of the porch wall that receive changing sunlight; two conditions that are not of interest in this zone.

A region is a part of the captured image that is of interest for this zone. By default, a region is configured to cover the whole captured image. Depending on the selected type of this zone, the shape of the region can be adjusted to accommodate multiple effects. This can be done by dragging the control points in the reference image around, or by altering the coordinates found in the controls below the reference image. Clicking on a control point in the reference image highlights the coordinates in the table below. Clicking the + button in a point row adds a control point between this point and the next; clicking the - button removes this control point. It is possible to accidentally place a control point outside of the valid coordinates of the image. This will prevent the monitor from working properly. You can make zones almost any shape you like; except that zones may not self-intersect (i.e. edges crossing over each other).

Alarm Colour These parameters can be used to individually colorize the zone overlay pattern. Alarms in this zone will be highlighted in the alarm colour. This option is irrelevant for Preclusive and Inactive zones and will be disabled.

Alarm Check Methods There are 3 Alarm Check Methods. They are sequential, and are layered: In AlarmedPixels mode, only the AlarmedPixel analysis is performed. In FilteredPixels mode, the AlarmedPixel analysis is performed first, followed by the FilteredPixel analysis. In the Blobs mode, all 3 analysis methods are performed in order. An alarm is only triggered if *all* of the enabled analysis modes are triggered. For performance reasons, as soon as the criteria for one of the analysis modes is not met, the alarm checking for the frame is complete. Since the subsequent modes each require progressively more computations, it is a good idea to tune the parameters in each of the activated layers.

For reference purposes, the Zone Area box shows the area of the entire region of interest. In percent mode, this is 100. In Pixels mode, this is the pixel count of the region. All 3 Min/Max Area parameter groups are based on the Zone Area as the maximum sensible value, and all 3 are interpreted in the units specified in the Units input.

AlarmedPixels Alarmed pixels is the first layer of analysis, and is always enabled. It is recommended that you start with this method and move on to the subsequent methods once the effects of the basic parameters are understood. In the AlarmedPixels mode, 2 parameter categories are available for tuning: Min/Max Pixel Threshold, and Min/Max Alarmed Area.

Min/Max Pixel Threshold (0-255) In the AlarmedPixel layer of analysis, each individual pixel of the image is compared to the current reference image. Pixels that are different from the reference image are considered alarmed pixels. However, small aberrations in lighting or auto exposure camera adjustments may cause the explicit value of a pixel to vary by small amounts from image to image. This parameter allows you to set the limits of what will be considered a changed pixel. For example, if your camera points to a blank white wall, and you raise a black colored item into view, then the change in any one pixel will be great, indeed, extreme. If however, you raise a white piece of paper, then the change in an individual pixel will be less.

The minimum pixel threshold setting should be high enough to cause minor lighting, imaging, or compression changes to be ignored. Setting the minimum value too high, may allow a white cat to walk undetected across the view of the white wall. A good starting point for the minimum pixel threshold is 40, meaning that the difference in pixel value from must be greater than 40. A good default for the maximum pixel threshold is 0 (indicating that all differences above the minimum threshold are considered a change.)

Min/Max Alarmed Area The count of alarmed pixels (or percentage of alarmed pixels relative to the pixel area of the region if in percent mode) is used in this first layer of analysis to determine if an alarm is triggered. If the count or percentage is above the minimum alarmed area, but less than the maximum alarmed area, an alarm is triggered. These settings depend on the size of the object you are trying to capture: a value too low may cause false alarms, while a value too high might not detect small objects. A good starting point for both the minimum and maximum are 0 and 0, indicating that any number of alarmed pixels (or any percentage) greater than 0 will trigger an alarm. The frame scores from logged events can then be used to bring the minimum up to a reasonable value. An alternative starting point for the minimum alarmed area (in percent) is 25% of the area that an object of interest takes up in the region. For example, if you approximate that a subject moving through the frame takes up 30% of the frame, then a good starting minimum area is about 7.5%.

FilteredPixels Selecting the FilteredPixels Alarm Check Method adds an additional layer of analysis to the Alarmed-Pixels check along with 2 additional parameter categories for tuning. This layer works by analyzing the alarmed pixels identified in the first layer. Alarmed pixels are disregarded, in this and future layers if enabled, if they are not in groups of a minimum small square size. Primarily, this filtering removes isolated alarmed pixels that may be artifacts of the camera, lens, or compression.

Filter Width/Height (pixels) This parameter is always specified in Pixels, even when Percentages are the selected units. It specifies the size of the group of pixels surrounding a given pixel that must be in alarmed pixels for the pixel itself to be considered an alarmed pixel. The width and height should always be an odd number. 3 x 3 is the default value, and 5 x 5 is also suggested as a sensible alternative. Avoid using large numbers for the width and height of the filter area. When using the Blobs Alarm Check Method, FilteredPixels can be effectively disabled by setting either the width or height to a value less than 1.

Min/Max Filtered Area Applying the filtering analysis results in an area that is less than or equal to the alarmed area. Thus the minimum and maximum filtered area parameters for alarm should be equal to or less than the corresponding alarm area parameters, or the FilteredPixels analysis will never trigger an alarm. In particular, it is useful to raise the minimum alarmed area parameter until false events from image artifacts disappear, and setting a minimum filtered area parameter less the minimum alarmed area parameter by enough to capture small events of interest.

Blobs



This image shows an image with 1 identified blob. The blob is outlined in the Alarm Colour specified above.

When two or more Filtered areas touch or share a boundary, it is sensible to evaluate the regions as one contiguous area instead of separate entities. A Blob is a contiguous area made up of multiple filtered areas. Whereas FilteredPixels is useful for excluding parts of the image that are not part of the actual scene, Blob filtering is better suited to disregarding areas of the actual scene that are not of interest.

Selecting the Blobs Alarm Check Method opens up all of the available parameters. Enabling Blobs adds one more layer of analysis to the AlarmedPixel and FilteredPixel checks in the determination of a valid alarm along along with 2 additional parameter categories for tuning: the size of the blobs, and the number of blobs. A Blob is not necessarily the whole object that may be of interest. In the example image, the subject is moving, but only a portion of him is marked as a blob. This is because as the subject moves, many pixels of the image do not change in value beyond the set threshold. A pixel that is representing the subject's shoulder in one frame may be representing his back in the next, however, the value of the pixel remains nearly the same.

Min/Max Blob Area The blob area parameters control the smallest and largest contiguous areas that are to be considered a blob. A good value for the maximum area is the default of 0. (There is no upper bound for the size of a contiguous area that will still be considered a blob.)

Min/Max Blobs Normally, you would want any positive number of blobs to trigger an event, so the default value of 1 should suffice. In some circumstances, it may benefit to have only one blob NOT trigger an event, in which case, setting this value to 2 or higher may serve some special purpose. A good value for the maximum blobs is the default of 0. (There is no upper bound for the number of blobs that will trigger an event. Use the maximum blobs parameter can be used to tune out events that show a high number of blobs.

Overload Frame Ignore Count This setting specifies the number of frames to NOT raise an alarm after an overload. In this context, overload is defined as a detected change too big to raise an alarm. Depending on the alarm check method that could be * Number of alarmed pixels > Max Alarmed Area or * Number of filtered pixels > Max Filtered Area or * Number of Blobs > Max Blobs The idea is that after a change like a light going on that is considered too big to count as an alarm, it could take a couple of frames for things to settle down again.

Extend Alarm Frame Count This field applies to Preclusive Zones only. Placing a value in this field holds the Preclusive zone active for the specified number of frames after the initial triggering event. This is useful in cases where a sudden change in light level triggers the Preclusive zone, but the zone needs to be held active for a few frames as the camera itself adjusts to that change in light level.

Other information

Refer to [this](#) user contributed Zone guide for additional information will illustrations if you are new to zones and need more help.

2.6 Viewing Monitors


ZoneMinder allows you to view a live feed of your configured monitors. One can access this view by clicking on the “Name” column of any of the monitors

Name	Function	Source	Events	Hour	Day	Week
Garage	Monitor Capturing 1.67 fps 37.98kB/s	192.168.1.100	0 0B	0B	0B	0B
DoorBell	Mocord Capturing 14.29/14.29 fps 126.18kB/s	192.168.1.101	901 127.36GB	1 0B	39 6.61GB	614 57.75GB
Basement	Modect Capturing 5.00/5.00 fps 134.78kB/s	192.168.1.102	134 246.51MB	0 0B	6 12.55MB	74 135.15MB

Clicking on the name produces a view similar to this:

Low
Load: 2.85
DB:23/151
Storage: Default: 52% /dev/shm: 22%
v1.33.14

Garage
Scale: 1/4x
Back

Click to zoom, shift click to pan, ctrl click to zoom out


DISABLE ALARMS

State: Idle - 1.68 fps

||
▶
🔍

Mode: Live Zoom: 1x

FORCE ALARM

Id

Name

Time

Secs

Frames

Score

The image should be self-explanatory but if it looks like garbage it is possible that the video configuration is wrong so look in your system error log and check for or report anything unusual. The centre of the window will have a tiny

frame that just contains a status; this will be 'Idle', 'Alarm' or 'Alert' depending on the function of the Monitor and what's going on in the field of view. Idle means nothing is happening, Alarm means there is an alarm in progress and Alert means that an alarm has happened and the monitor is 'cooling down', if another alarm is generated in this time it will just become part of the same event. These indicators are colour coded in green, red and amber.

By default if you have minimised this window or opened other windows in front it will pop up to the front if it goes to Alarm state. This behaviour can be turned off in 'options' if required. You can also specify a sound file in the configuration, which will be played when an alarm occurs to alert you to the fact if you are not in front of your computer. This should be a short sound of only a couple of seconds ideally. Note that as the status is refreshed every few seconds it is possible for this not to alert you to every event that takes place, so you shouldn't rely on it for this purpose if you expect very brief events. Alternatively you can decrease the refresh interval for this window in the configuration though having too frequent refreshing may impact on performance.

Below the status is a list of recent events that have occurred, by default this is a listing of just the last 10 but clicking on 'All' will give you a full list and 'Archive' will take you to the event archive for this monitor, more on this later. Clicking on any of the column headings will sort the events appropriately.

From here you can also delete events if you wish. The events themselves are listed with the event id, and event name (which you can change), the time that the event occurred, the length of the event including any preamble and postamble frames, the number of frames comprising the event with the number that actually contain an alarm in brackets and finally a score. This column lists the average score per alarm frame as well as the maximum score that any alarm frame had.

The score is an arbitrary value that essentially represents the percentage of pixels in the zone that are in blobs divided by the square root of the number of blobs and then divided by the size of the zone. This gives a nominal maximum of 100 for a zone and the totals for each zone are added together, Active zones scores are added unchanged, Inclusive zones are halved first and Exclusive zones are doubled. In reality values are likely to be much less than 100 but it does give a simple indication of how major the event was.

2.7 Filtering Events

Filters allow you to define complex conditions with associated actions in ZoneMinder. Examples could include:

- Send an email each time a new event occurs for a specific monitor
- Delete events that are more than 10 days old

And many more.

The filter window can be accessed by tapping on the top level filter menu

You can use the filter window to create your own filters or to modify existing ones. You can even save your favourite filters to re-use at a future date. Filtering itself is fairly simple; you first choose how many expressions you'd like your filter to contain. Changing this value will cause the window to redraw with a corresponding row for each expression. You then select what you want to filter on and how the expressions relate by choosing whether they are 'and' or 'or' relationships. For filters comprised of many expressions you will also get the option to bracket parts of the filter to ensure you can express it as desired. Then if you like choose how you want your results sorted and whether you want to limit the amount of events displayed.

Here is what the filter window looks like

A

Use Filter Choose Filter ▼

B

Name

C

Alarm Frames ▼ equal to ▼

+ -

D

Sort by Id ▼ Asc ▼ Limit to first 100 results only

E

Archive all matches ☐

Update used disk space ☐

Create video for all matches ☐

Execute command on all matches ☐ 0

Delete all matches ☐

Copy all matches ☐

Move all matches ☐

Run filter in background ☐

Run filter concurrently ☐

F

LIST MATCHES EXPORT MATCHES EXECUTE SAVE SAVE A

- A: This is a dropdown list where you can select pre-defined filters. You will notice that ZoneMinder comes with a PurgeWhenFull filter that is configured to delete events if you reach 95% of disk space.
- B: If you are creating a new filter, you can type in a name for your filter here
- C: This is where you specify conditions that need to match before the filter is executed. You use the “+” and “-” buttons to add/delete conditions
- D: This allows you to perform sorting and limiting operations on the output before you take an action
- E: This is where you specify what needs to happen when the conditions match:
 - Archive all matches: sets the archive field to 1 in the Database for the matched events. Think of ‘archiving’ as grouping them under a special category - you can view archived events later and also make sure archived events don’t get deleted, for example

Todo: fill in what update used disk space, copy all matches, move all matches do. For the “create video” filter, put in more details on how it works, any dependencies etc.

- Update used disk space:
- Create video for all matches: creates a video file of all the events that match

- **Execute command on all matches:** Allows you to execute any arbitrary command on the matched events. You can use:
 - * Delete all matches: Deletes all the matched events.
- Copy all matches:
- Move all matches:
- Run filter in background: When checked, ZoneMinder will make sure the filter is checked regularly. For example, if you want to be notified of new events by email, you should make sure this is checked. Filters that are configured to run in the background have a “*” next to it.
- Run filter concurrently: Allows this filter to run in its own thread thereby letting other filters run in parallel.
- *F:* Use ‘List Matches’ to ‘test’ your matching conditions. This will just match and show you what filters match. Use ‘Execute’ to actually execute the action after matching your conditions. Use ‘Save’ to save the filter for future use and ‘Reset’ to clear your settings

Note: More details on filter conditions:

There are several different elements to an event that you can filter on, some of which require further explanation. These are as follows, * ‘Date/Time’ which must evaluate to a date and a time together, * ‘Date’ and ‘Time’ which are variants which may only contain the relevant subsets of this, * ‘Weekday’ which as expected is a day of the week.

All of the preceding elements take a very flexible free format of dates and time based on the PHP strtotime function (<https://www.php.net/manual/en/function.strtotime.php>). This allows values such as ‘last Wednesday’ etc to be entered. We recommend acquainting yourself with this function to see what the allowed formats are. However automated filters are run in perl and so are parsed by the Date::Manip package. Not all date formats are available in both so if you are saving your filter to do automatic deletions or other tasks you should make sure that the date and time format you use is compatible with both methods. The safest type of format to use is ‘-3 day’ or similar with easily parseable numbers and units are in English.

The other things you can filter on are all fairly self explanatory, except perhaps for ‘Archived’ which you can use to include or exclude Archived events. In general you’ll probably do most filtering on un-archived events. There are also two elements, Disk Blocks and Disk Percent which don’t directly relate to the events themselves but to the disk partition on which the events are stored. These allow you to specify an amount of disk usage either in blocks or in percentage as returned by the ‘df’ command. They relate to the amount of disk space used and not the amount left free. Once your filter is specified, clicking ‘submit’ will filter the events according to your specification. As the disk based elements are not event related directly if you create a filter and include the term ‘DiskPercent > 95’ then if your current disk usage is over that amount when you submit the filter then all events will be listed whereas if it is less then none at all will. As such the disk related terms will tend to be used mostly for automatic filters (see below). If you have created a filter you want to keep, you can name it and save it by clicking ‘Save’.

If you do this then the subsequent dialog will also allow you specify whether you want this filter automatically applied in order to delete events or upload events via ftp to another server and mail notifications of events to one or more email accounts. Emails and messages (essentially small emails intended for mobile phones or pagers) have a format defined in the Options screen, and may include a variety of tokens that can be substituted for various details of the event that caused them. This includes links to the event view or the filter as well as the option of attaching images or videos to the email itself. Be aware that tokens that represent links may require you to log in to access the actual page, and sometimes may function differently when viewed outside of the general ZoneMinder context. The tokens you can use are as follows.

- %EI% Id of the event

- %EN% Name of the event
- %EC% Cause of the event
- %ED% Event description
- %ET% Time of the event
- %EL% Length of the event
- %EF% Number of frames in the event
- %EFA% Number of alarm frames in the event
- %EST% Total score of the event
- %ESA% Average score of the event
- %ESM% Maximum score of the event
- %EP% Path to the event
- %EPS% Path to the event stream
- %EPI% Path to the event images
- %EPI1% Path to the first alarmed event image
- %EPIM% Path to the (first) event image with the highest score
- %EI1% Attach first alarmed event image
- %EIM% Attach (first) event image with the highest score
- %EV% Attach event mpeg video
- %MN% Name of the monitor
- %MET% Total number of events for the monitor
- %MEH% Number of events for the monitor in the last hour
- %MED% Number of events for the monitor in the last day
- %MEW% Number of events for the monitor in the last week
- %MEM% Number of events for the monitor in the last month
- %MEA% Number of archived events for the monitor
- %MOD% Path to image containing object detection
- %MP% Path to the monitor window
- %MPS% Path to the monitor stream
- %MPI% Path to the monitor recent image
- %FN% Name of the current filter that matched
- %FP% Path to the current filter that matched
- %ZP% Path to your ZoneMinder console

Finally you can also specify a script which is run on each matched event. This script should be readable and executable by your web server user. It will get run once per event and the relative path to the directory containing the event in question. Normally this will be of the form <MonitorName>/<EventId> so from this path you can derive both the monitor name and event id and perform any action you wish. Note that arbitrary commands are not allowed to be specified in the filter, for security the only thing it may contain is the full path to an executable. What that contains is entirely up to you however.

Filtering is a powerful mechanism you can use to eliminate events that fit a certain pattern however in many cases modifying the zone settings will better address this. Where it really comes into its own is generally in applying time filters, so for instance events that happen during weekdays or at certain times of the day are highlighted, uploaded or deleted. Additionally using disk related terms in your filters means you can automatically create filters that delete the oldest events when your disk gets full. Be warned however that if you use this strategy then you should limit the returned results to the amount of events you want deleted in each pass until the disk usage is at an acceptable level. If you do not do this then the first pass when the disk usage is high will match, and then delete, all events unless you have used other criteria inside of limits. ZoneMinder ships with a sample filter already installed, though disabled. The `PurgeWhenFull` filter can be used to delete the oldest events when your disk starts filling up. To use it you should select and load it in the filter interface, modify it to your requirements, and then save it making you sure you check the 'Delete all matches' option. This will then run in the background and ensure that your disk does not fill up with events.

2.7.1 Saving filters

When saving filters, if you want the filter to run in the background make sure you select the "Run filter in background" option. When checked, ZoneMinder will make sure the filter is checked regularly. For example, if you want to be notified of new events by email, you should make sure this is checked. Filters that are configured to run in the background have a "*" next to it.

2.7.2 How filters actually work

It is useful to know how filters actually work behind the scenes in ZoneMinder, in the event you find your filter not functioning as intended:

- the primary filter processing process in ZoneMinder is a perl file called `zmfilter.pl` which retrieves filters from the Filters database table
- `zmfilter.pl` runs every `FILTER_EXECUTE_INTERVAL` seconds (default is 20s, can be changed in Options->System)
- in each run, it goes through all the filters which are marked as "Run in Background" and if the conditions match performs the specified action
- **`zmfilter.pl` also reloads all the filters every `FILTER_RELOAD_DELAY` seconds (default is 300s/5mins, can be changed in**
 - So if you have just created a new filter, `zmfilter` will not see it till the next `FILTER_RELOAD_DELAY` cycle
 - This is also important if you are using "relative times" like 'now' - see [Caveat with Relative items](#)

2.7.3 Relative items in date strings

Relative items adjust a date (or the current date if none) forward or backward. The effects of relative items accumulate. Here are some examples:

```
* 1 year
* 1 year ago
* 3 years
* 2 days
```


The unit of time displacement may be selected by the string ‘year’ or ‘month’ for moving by whole years or months. These are fuzzy units, as years and months are not all of equal duration. More precise units are ‘fortnight’ which is worth 14 days, ‘week’ worth 7 days, ‘day’ worth 24 hours, ‘hour’ worth 60 minutes, ‘minute’ or ‘min’ worth 60 seconds, and ‘second’ or ‘sec’ worth one second. An ‘s’ suffix on these units is accepted and ignored.

The unit of time may be preceded by a multiplier, given as an optionally signed number. Unsigned numbers are taken as positively signed. No number at all implies 1 for a multiplier. Following a relative item by the string ‘ago’ is equivalent to preceding the unit by a multiplier with value -1.

The string ‘tomorrow’ is worth one day in the future (equivalent to ‘day’), the string ‘yesterday’ is worth one day in the past (equivalent to ‘day ago’).

The strings ‘now’ or ‘today’ are relative items corresponding to zero-valued time displacement, these strings come from the fact a zero-valued time displacement represents the current time when not otherwise changed by previous items. They may be used to stress other items, like in ‘12:00 today’. The string ‘this’ also has the meaning of a zero-valued time displacement, but is preferred in date strings like ‘this thursday’.

When a relative item causes the resulting date to cross a boundary where the clocks were adjusted, typically for daylight saving time, the resulting date and time are adjusted accordingly.

The fuzz in units can cause problems with relative items. For example, ‘2003-07-31 -1 month’ might evaluate to 2003-07-01, because 2003-06-31 is an invalid date. To determine the previous month more reliably, you can ask for the month before the 15th of the current month. For example:

```
$ date -R
Thu, 31 Jul 2003 13:02:39 -0700

$ date --date='-1 month' +'Last month was %B?'
Last month was July?

$ date --date="$(date +%Y-%m-15) -1 month" +'Last month was %B!'
Last month was June!
```

As this applies to ZoneMinder filters, you might want to search for events in a period of time, or maybe for example create a purge filter that removes events older than 30 days. For the later you would want at least two lines in your filter. The first line should be:

```
[<Archive Status> <equal to> <Unarchived Only>]
```

as you don’t want to delete your archived events.

Your second line to find events older than 30 days would be:

```
[and <Date><less than> -30 days]
```

You use “less than” to indicate that you want to match events before the specified date, and you specify “-30 days” to indicate a date 30 days before the time the filter is run. Of course you could use 30 days ago as well(?).

You should always test your filters before enabling any actions based on them to make sure they consistently return the results you want. You can use the submit button to see what events are returned by your query.

2.7.4 Caveat with Relative items

One thing to remember if you specify relative dates like “now” or “1 minute ago”, etc, they are converted to a specific date and time by Zoneminder’s filtering process (zmfilter.pl) when the filters are loaded. They are NOT recomputed each time the filter runs. Filters are re-loaded depending on the value specified by FILTER_RELOAD_DELAY variable in the Zoneminder Web Console->Options->System

This may cause confusion in the following cases, for example: Let's say a user specifies that he wants to be notified of events via email the moment the event "DateTime" is "less than" "now" as a filter criteria. When the filter first gets loaded by `zmfilter.pl`, this will translate to "Match events where Start Time < " + `localtime()` where local time is the time that is resolved when this filter gets loaded. Now till the time the filter gets reloaded after `FILTER_RELOAD_DELAY` seconds (which is usually set to 300 seconds, or 5 minutes), that time does not get recomputed, so the filter will not process any new events that occur after that computed date till another 5 minutes, which is probably not what you want.

2.7.5 Troubleshooting tips






If your filter is not working, here are some useful tips:

- Look at Info and Debug logs in Zoneminder
- Run `sudo zmfilter.pl -f <yourfiltername>` from command line and see the log output
- Check how long your action is taking - `zmfilter.pl` will wait for the action to complete before it checks again
- If you are using relative times like 'now' or '1 year ago' etc. remember that `zmfilter` converts that relative time to an absolute date only when it reloads filters, which is dictated by the `FILTER_RELOAD_DELAY` duration. So, for example, if you are wondering why your events are not being detected before intervals of 5 minutes and you have used such a relative condition, this is why
- In the event that you see your new filter is working great when you try it out from the Web Console (using the Submit or Execute button) but does not seem to work when its running in background mode, you might have just chanced upon a compatibility issue between how Perl and PHP translate free form text to dates/times. When you test it via the "Submit" or "Execute" button, you are invoking a PHP function for time conversion. When the filter runs in background mode, `zmfilter.pl` calls a perl equivalent function. In some cases, depending on the version of Perl and PHP you have, the results may vary. If you face this situation, the best thing to do is to run `sudo zmfilter.pl -f <yourfiltername>` from a terminal to make sure the filter actually works in Perl as well.

2.8 Viewing Events

From the monitor or filtered events listing you can now click on an event to view it in more detail.

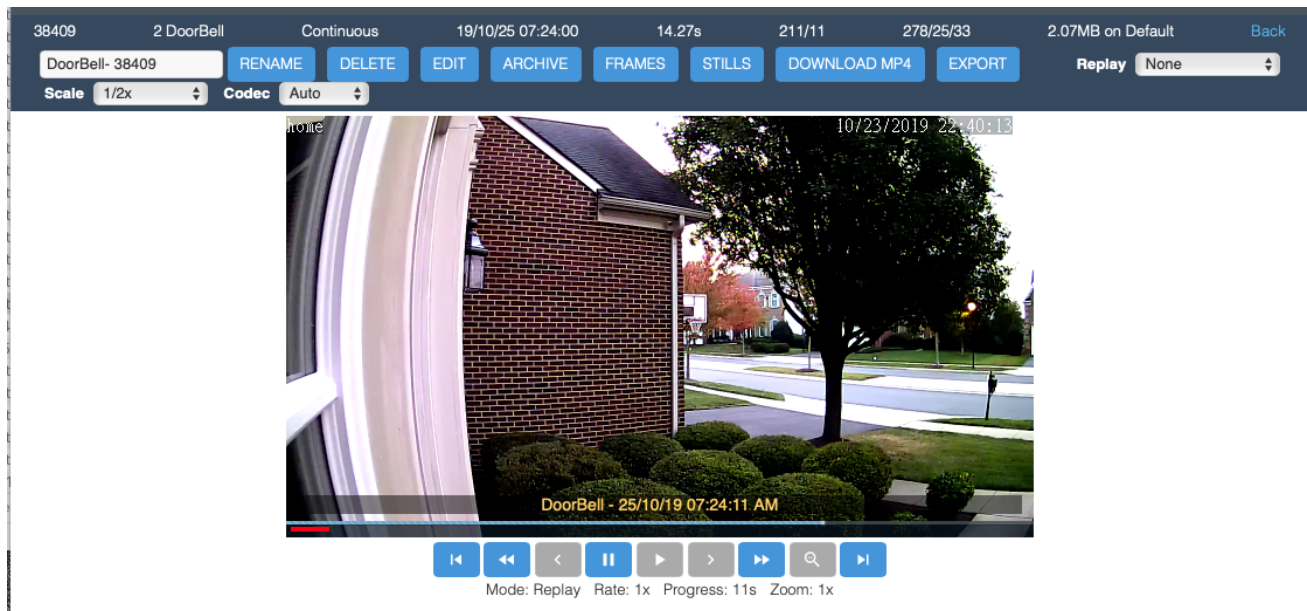
This is an example view that shows events for a specific monitor:

47 Events View All Close											
- 1 - 2 >											
Refresh	Show Filter Window						Show Timeline				
ID	NAME	MONITOR	CAUSE	TIME(V)	DURATION	FRAMES	ALARM FRAMES	TOTAL SCORE	AVG. SCORE	MAX. SCORE	THUMBNAIL
18280	Event-18280	unfinished	Motion	08/22 18:27:03	00:00:09	77	13	247	19	61	
18279	Event-18279	unfinished	Motion	08/22 18:26:54	00:00:09	74	13	177	13	47	
18274	Event-18274	unfinished	Motion	08/22 16:41:11	00:00:06	55	5	21	4	5	
18250	Event-18250	unfinished	Motion	08/22 10:29:06	00:00:09	70	20	344	17	54	
	Event-										

If you have streaming capability you will see a series of images that make up the event. Under that you should also see a progress bar. Depending on your configuration this will either be static or will be filled in to indicate how far through the event you are. By default this functionality is turned off for low bandwidth settings as the image delivery tends to not be able to keep up with real-time and the progress bar cannot take this into account. Regardless of whether the progress bar updates, you can click on it to navigate to particular points in the events.

You will also see a link to allow you to view the still images themselves. If you don't have streaming then you will be taken directly to this page. The images themselves are thumbnail size and depending on the configuration and bandwidth you have chosen will either be the full images scaled in your browser or actual scaled images. If it is the latter, if you have low bandwidth for example, it may take a few seconds to generate the images. If thumbnail images are required to be generated, they will be kept and not re-generated in future. Once the images appear you can mouse over them to get the image sequence number and the image score.

Here is an example of viewing an event stream:



The image above shows a typical window for an event that was recorded as an MP4 video

2.9 Options

The various options you can specify are displayed in a tabbed dialog with each group of options displayed under a different heading. Each option is displayed with its name, a short description and the current value. You can also click on the '?' link following each description to get a fuller explanation about each option. This is the same as you would get from zmconfig.pl. A number of option groups have a master option near the top which enables or disables the whole group so you should be aware of the state of this before modifying options and expecting them to make any difference.

If you have changed the value of an option you should then 'save' it. A number of the option groups will then prompt you to let you know that the option(s) you have changed will require a system restart. This is not done automatically in case you will be changing many values in the same session, however once you have made all of your changes you should restart ZoneMinder as soon as possible. The reason for this is that web and some scripts will pick up the new changes immediately but some of the daemons will still be using the old values and this can lead to data inconsistency or loss.

Note: If you are looking for Options->Paths documentation, it was moved to a configuration file starting

ZoneMinder 1.32. See [here](#).

2.9.1 Options - Display

This option screen allows user to select the skin for ZoneMinder. Currently available styles are:

Display

System

Config

API

Servers

Storage

Web

Skin: classic Change the skin for this session

CSS: base (selected), classic, dark

2.9.2 Options - System

This screen allows the admin to configure various core operations of the system.

A partial screenshot is shown below:

Display

System

Config

API

Servers

Storage

Web

Images

Logging

Network

Email

Upload

X10

High B/W

Medium B/W

Low B/W

Users

SKIN_DEFAULT: classic Default skin used by web interface (?)

CSS_DEFAULT: ☐ base ☐ classic ☒ dark Default set of css files used by web interface (?)

BANDWIDTH_DEFAULT: Low Default setting for bandwidth profile used by web interface (?)

LANG_DEFAULT: en_gb Default language used by web interface (?)

OPT_USE_AUTH: ☒ Authenticate user logins to ZoneMinder (?)

AUTH_TYPE: ☒ builtin ☐ remote What is used to authenticate ZoneMinder users (?)

AUTH_RELAY: ☒ hashed ☐ plain ☐ none Method used to relay authentication information (?)

AUTH_HASH_SECRET: Secret for encoding hashed authentication information (?)

AUTH_HASH_IPS: ☐ Include IP addresses in the authentication hash (?)

AUTH_HASH_TTL: 2 The number of hours that an authentication hash is valid for. (?)

SKIN_DEFAULT - ZoneMinder allows the use of many different web interfaces. This option allows you to set the default skin used by the website. Users can change their skin later, this merely sets the default.

CSS_DEFAULT - ZoneMinder allows the use of many different web interfaces, and some skins allow the use of different set of CSS files to control the appearance. This option allows you to set the default set of css files used by the website. Users can change their css later, this merely sets the default.

LANG_DEFAULT - ZoneMinder allows the web interface to use languages other than English if the appropriate language file has been created and is present. This option allows you to change the default language that is used from the shipped language, British English, to another language.

OPT_USE_AUTH - ZoneMinder can run in two modes. The simplest is an entirely unauthenticated mode where anyone can access ZoneMinder and perform all tasks. This is most suitable for installations where the web server access is limited in other ways. The other mode enables user accounts with varying sets of permissions. Users must login or authenticate to access ZoneMinder and are limited by their defined permissions. Authenticated mode alone should not be relied up for securing Internet connected ZoneMinder.

AUTH_TYPE - ZoneMinder can use two methods to authenticate users when running in authenticated mode. The first is a builtin method where ZoneMinder provides facilities for users to log in and maintains track of their identity. The second method allows interworking with other methods such as http basic authentication which passes an independently authentication 'remote' user via http. In this case ZoneMinder would use the supplied user without additional authentication provided such a user is configured ion ZoneMinder.

AUTH_RELAY - When ZoneMinder is running in authenticated mode it can pass user details between the web pages and the back end processes. There are two methods for doing this. This first is to use a time limited hashed string which contains no direct username or password details, the second method is to pass the username and passwords around in plaintext. This method is not recommend except where you do not have the md5 libraries available on your system or you have a completely isolated system with no external access. You can also switch off authentication relaying if your system is isolated in other ways.

AUTH_HASH_SECRET - When ZoneMinder is running in hashed authenticated mode it is necessary to generate hashed strings containing encrypted sensitive information such as usernames and password. Although these string are reasonably secure the addition of a random secret increases security substantially. Note that if you are using the new token based APIs, then this field is mandatory with ZM 1.34 and above

AUTH_HASH_IPS - When ZoneMinder is running in hashed authenticated mode it can optionally include the requesting IP address in the resultant hash. This adds an extra level of security as only requests from that address may use that authentication key. However in some circumstances, such as access over mobile networks, the requesting address can change for each request which will cause most requests to fail. This option allows you to control whether IP addresses are included in the authentication hash on your system. If you experience intermitent problems with authentication, switching this option off may help. It is recommended you keep this off if you use mobile apps like zmNinja over mobile carrier networks - several APNs change the IP very frequently which may result in authentication failure.

AUTH_HASH_TTL - Time before ZM auth will expire (does not apply to API tokens). The default has traditionally been 2 hours. A new hash will automatically be regenerated at half this value.

AUTH_HASH_LOGINS - The normal process for logging into ZoneMinder is via the login screen with username and password. In some circumstances it may be desirable to allow access directly to one or more pages, for instance from a third party application. If this option is enabled then adding an 'auth' parameter to any request will include a shortcut login bypassing the login screen, if not already logged in. As authentication hashes are time and, optionally, IP limited, this can allow short-term access to ZoneMinder screens from other web pages etc. In order to use this, the calling application will have to generate the authentication hash itself and ensure it is valid. If you use this option you should ensure that you have modified the ZM_AUTH_HASH_SECRET to something unique to your system.

ENABLE_CSRF_MAGIC - CSRF stands for Cross-Site Request Forgery which, under specific circumstances, can allow an attacker to perform any task your ZoneMinder user account has permission to perform. To accomplish this, the attacker must write a very specific web page and get you to navigate to it, while you are logged into the ZoneMinder web console at the same time. Enabling ZM_ENABLE_CSRF_MAGIC will help mitigate these kinds of attacks. If you are using zmNinja and face access issues, you might try turning this off.

OPT_USE_API - A global setting to enable/disable ZoneMinder APIs. If you are using mobile apps like zmNinja, this needs to be enabled

Note: If you are using zmNinja along with authentication, please make sure AUTH_HASH_LOGINS is enabled, OPT_USE_API is enabled, AUTH_RELAY is set to hashed, AUTH_HASH_IPS is off and a valid AUTH_HASHED_SECRET is specified.

OPT_USE_LEGACY_AUTH - Starting version 1.34.0, ZoneMinder uses a more secure Authentication mechanism using JWT tokens. Older versions used a less secure MD5 based auth hash. It is recommended you turn this off after you are sure you don't need it. If you are using a 3rd party app that relies on the older API auth mechanisms, you will have to update that app if you turn this off. Note that zmNinja 1.3.057 onwards supports the new token system.

OPT_USE_EVENT_NOTIFICATION - zmeventnotification is a 3rd party event notification server that is used to get notifications for alarms detected by ZoneMinder in real time. zmNinja requires this server for push notifications to mobile phones. This option only enables the server if its already installed. Please visit the [Event Notification Server project site](#) for installation instructions.

OPT_USE_GOOG_RECAPTCHA - This option allows you to include a google reCaptcha validation at login. This means in addition to providing a valid username and password, you will also have to pass the reCaptcha test. Please note that enabling this option results in the zoneminder login page reach out to google servers for captcha validation. Also please note that enabling this option may break 3rd party clients if they rely on web based logins (Note that zmNinja now uses the API based token method and will not be affected if reCAPTCHA is enabled). If you enable this, you also need to specify your site and secret key (please refer to context help in the ZoneMinder system screen)

SYSTEM_SHUTDOWN - this option decides if it is allowed to shutdown the full system via the ZM UI. The system will need to have sudo installed and the following added to /etc/sudoers:

```
www-data ALL=NOPASSWD: /sbin/shutdown
```

to perform the shutdown or reboot

OPT_FAST_DELETE - Normally an event created as the result of an alarm consists of entries in one or more database tables plus the various files associated with it. When deleting events in the browser it can take a long time to remove all of this if your are trying to do a lot of events at once. **NOTE:** It is recommended that you keep this option OFF, unless you are running on an old or low-powered system.

FILTER_RELOAD_DELAY - ZoneMinder allows you to save filters to the database which allow events that match certain criteria to be emailed, deleted or uploaded to a remote machine etc. The zmfilter daemon loads these and does the actual operation. This option determines how often in seconds the filters are reloaded from the database to get the latest versions or new filters. If you don't change filters very often this value can be set to a large value.

FILTER_EXECUTE_INTERVAL - ZoneMinder allows you to save filters to the database which allow events that match certain criteria to be emailed, deleted or uploaded to a remote machine etc. The zmfilter daemon loads these and does the actual operation. This option determines how often the filters are executed on the saved event in the database. If you want a rapid response to new events this should be a smaller value, however this may increase the overall load on the system and affect performance of other elements.

MAX_RESTART_DELAY - The zmdc (zm daemon control) process controls when processes are started or stopped and will attempt to restart any that fail. If a daemon fails frequently then a delay is introduced between each restart attempt. If the daemon stills fails then this delay is increased to prevent extra load being placed on the system by continual restarts. This option controls what this maximum delay is.

STATUS_UPDATE_INTERVAL - The zmstats daemon performs various db queries related to collecting system statistics that may take a long time in the background. This option decides how often this update is scheduled.

WATCH_CHECK_INTERVAL - The zmwatch daemon checks the image capture performance of the capture daemons to ensure that they have not locked up (rarely a sync error may occur which blocks indefinitely). This option determines how often the daemons are checked.

WATCH_MAX_DELAY - The `zmwatch` daemon checks the image capture performance of the capture daemons to ensure that they have not locked up (rarely a sync error may occur which blocks indefinitely). This option determines the maximum delay to allow since the last captured frame. The daemon will be restarted if it has not captured any images after this period though the actual restart may take slightly longer in conjunction with the check interval value above.

RUN_AUDIT - The `zmaudit` daemon exists to check that the saved information in the database and on the filesystem match and are consistent with each other. If an error occurs or if you are using 'fast deletes' it may be that database records are deleted but files remain. In this case, and similar, `zmaudit` will remove redundant information to synchronise the two data stores. This option controls whether `zmaudit` is run in the background and performs these checks and fixes continuously. It is recommended you keep this **OFF** in most systems.

AUDIT_CHECK_INTERVAL - The `zmaudit` daemon exists to check that the saved information in the database and on the filesystem match and are consistent with each other. If an error occurs or if you are using 'fast deletes' it may be that database records are deleted but files remain. In this case, and similar, `zmaudit` will remove redundant information to synchronise the two data stores. The default check interval of 900 seconds (15 minutes) is fine for most systems however if you have a very large number of events the process of scanning the database and filesystem may take a long time and impact performance. In this case you may prefer to make this interval much larger to reduce the impact on your system. This option determines how often these checks are performed.

AUDIT_MIN_AGE - The `zmaudit` daemon exists to check that the saved information in the database and on the filesystem match and are consistent with each other. Event files or db records that are younger than this setting will not be deleted and a warning will be given

OPT_CONTROL - ZoneMinder includes limited support for controllable cameras. A number of sample protocols are included and others can easily be added. If you wish to control your cameras via ZoneMinder then select this option otherwise if you only have static cameras or use other control methods then leave this option off.

OPT_TRIGGERS - ZoneMinder can interact with external systems which prompt or cancel alarms. This is done via the `zmtrigger.pl` script. This option indicates whether you want to use these external triggers. Most people will say no here.

CHECK_FOR_UPDATES - From ZoneMinder version 1.17.0 onwards new versions are expected to be more frequent. To save checking manually for each new version ZoneMinder can check with the `zoneminder.com` website to determine the most recent release. These checks are infrequent, about once per week, and no personal or system information is transmitted other than your current version number. If you do not wish these checks to take place or your ZoneMinder system has no internet access you can switch these check off with this configuration variable

TELEMETRY_DATA - Enable collection of usage information of the local system and send it to the ZoneMinder development team. This data will be used to determine things like who and where our customers are, how big their systems are, the underlying hardware and operating system, etc. This is being done for the sole purpose of creating a better product for our target audience. This script is intended to be completely transparent to the end user, and can be disabled from the web console under Options. For more details on what information we collect, please refer to Zoneminder's privacy statement (available in the contextual help of **TELEMETRY_DATA** on your installation).

UPDATE_CHECK_PROXY - If you use a proxy to access the internet then ZoneMinder needs to know so it can access `zoneminder.com` to check for updates. If you do use a proxy enter the full proxy url here in the form of `http://<proxy host>:<proxy port>/`

SHM_KEY - ZoneMinder uses shared memory to speed up communication between modules. To identify the right area to use shared memory keys are used. This option controls what the base key is, each monitor will have it's Id or'ed with this to get the actual key used. You will not normally need to change this value unless it clashes with another instance of ZoneMinder on the same machine. Only the first four hex digits are used, the lower four will be masked out and ignored.

COOKIE_LIFETIME - This will affect how long a session will be valid for since the last request. Keeping this short helps prevent session hijacking. Keeping it long allows you to stay logged in longer without refreshing the view. We recommend you keep this to the default of 3600 if you are not sure.

2.9.3 Options - Config

The config screen allows the admin to change various configuration parameters related to image capturing and storage. A partial screenshot is shown below:

Display	TIMESTAMP_ON_CAPTURE	<input checked="" type="checkbox"/>	Timestamp images as soon as they are captured (?)
System	TIMESTAMP_CODE_CHAR	<input data-bbox="738 451 1076 493" type="text" value="%"/>	Character to used to identify timestamp codes (?)
Config	CPU_EXTENSIONS	<input checked="" type="checkbox"/>	Use advanced CPU extensions to increase performance (?)
API	FAST_IMAGE_BLENDS	<input checked="" type="checkbox"/>	Use a fast algorithm to blend the reference image (?)
Servers	OPT_ADAPTIVE_SKIP	<input checked="" type="checkbox"/>	Should frame analysis try and be efficient in skipping frames (?)
Storage	MAX_SUSPEND_TIME	<input data-bbox="738 798 836 840" type="text" value="30"/>	Maximum time that a monitor may have motion detection suspended (?)
Web	STRICT_VIDEO_CONFIG	<input checked="" type="checkbox"/>	Allow errors in setting video config to be fatal (?)
Images	LD_PRELOAD	<input data-bbox="738 976 1076 1018" type="text"/>	Path to library to preload before launching daemons (?)
Logging	V4L_MULTI_BUFFER	<input type="checkbox"/>	Use more than one buffer for Video 4 Linux devices (?)
Network			
Email			
Upload			
X10			
High B/W			
Medium B/W			
Low B/W			

TIMESTAMP_ON_CAPTURE - ZoneMinder can add a timestamp to images in two ways. The default method, when this option is set, is that each image is timestamped immediately when captured and so the image held in memory is marked right away. The second method does not timestamp the images until they are either saved as part of an event or accessed over the web. The timestamp used in both methods will contain the same time as this is preserved along with the image. The first method ensures that an image is timestamped regardless of any other circumstances but will result in all images being timestamped even those never saved or viewed. The second method necessitates that saved images are copied before being saved otherwise two timestamps perhaps at different scales may be applied. This has the (perhaps) desirable side effect that the timestamp is always applied at the same resolution so an image that has scaling applied will still have a legible and correctly scaled timestamp.

TIMESTAMP_CODE_CHAR - There are a few codes one can use to tell ZoneMinder to insert data into the timestamp of each image. Traditionally, the percent (%) character has been used to identify these codes since the current character codes do not conflict with the strftime codes, which can also be used in the timestamp. While this works well for Linux, this does not work well for BSD operating systems. Changing the default character to something else, such as an exclamation point (!), resolves the issue. Note this only affects the timestamp codes built into ZoneMinder. It has no effect on the family of strftime codes one can use.

CPU_EXTENSIONS - When advanced processor extensions such as SSE2 or SSSE3 are available, ZoneMinder can use them, which should increase performance and reduce system load. Enabling this option on processors that do not support the advanced processors extensions used by ZoneMinder is harmless and will have no effect.

FAST_IMAGE_BLENDS - To detect alarms ZoneMinder needs to blend the captured image with the stored reference image to update it for comparison with the next image. The reference blend percentage specified for the monitor controls how much the new image affects the reference image. There are two methods that are available for this. If this option is set then fast calculation which does not use any multiplication or division is used. This calculation is

extremely fast, however it limits the possible blend percentages to 50%, 25%, 12.5%, 6.25%, 3.25% and 1.5%. Any other blend percentage will be rounded to the nearest possible one. The alternative is to switch this option off and use standard blending instead, which is slower.

OPT_ADAPTIVE_SKIP - In previous versions of ZoneMinder the analysis daemon would attempt to keep up with the capture daemon by processing the last captured frame on each pass. This would sometimes have the undesirable side-effect of missing a chunk of the initial activity that caused the alarm because the pre-alarm frames would all have to be written to disk and the database before processing the next frame, leading to some delay between the first and second event frames. Setting this option enables a newer adaptive algorithm where the analysis daemon attempts to process as many captured frames as possible, only skipping frames when in danger of the capture daemon overwriting yet to be processed frames. This skip is variable depending on the size of the ring buffer and the amount of space left in it. Enabling this option will give you much better coverage of the beginning of alarms whilst biasing out any skipped frames towards the middle or end of the event. However you should be aware that this will have the effect of making the analysis daemon run somewhat behind the capture daemon during events and for particularly fast rates of capture it is possible for the adaptive algorithm to be overwhelmed and not have time to react to a rapid build up of pending frames and thus for a buffer overrun condition to occur.

MAX_SUSPEND_TIME - ZoneMinder allows monitors to have motion detection to be suspended, for instance while panning a camera. Ordinarily this relies on the operator resuming motion detection afterwards as failure to do so can leave a monitor in a permanently suspended state. This setting allows you to set a maximum time which a camera may be suspended for before it automatically resumes motion detection. This time can be extended by subsequent suspend indications after the first so continuous camera movement will also occur while the monitor is suspended.

STRICT_VIDEO_CONFIG - With some video devices errors can be reported in setting the various video attributes when in fact the operation was successful. Switching this option off will still allow these errors to be reported but will not cause them to kill the video capture daemon. Note however that doing this will cause all errors to be ignored including those which are genuine and which may cause the video capture to not function correctly. Use this option with caution.

LD_PRELOAD - Some older cameras require the use of the v4l1 compat library. This setting allows the setting of the path to the library, so that it can be loaded by `zmdc.pl` before launching `zmc`.

V4L_MULTI_BUFFER - Performance when using Video 4 Linux devices is usually best if multiple buffers are used allowing the next image to be captured while the previous one is being processed. If you have multiple devices on a card sharing one input that requires switching then this approach can sometimes cause frames from one source to be mixed up with frames from another. Switching this option off prevents multi buffering resulting in slower but more stable image capture. This option is ignored for non-local cameras or if only one input is present on a capture chip. This option addresses a similar problem to the **ZM_CAPTURES_PER_FRAME** option and you should normally change the value of only one of the options at a time. If you have different capture cards that need different values you can override them in each individual monitor on the source page.

CAPTURES_PER_FRAME - If you are using cameras attached to a video capture card which forces multiple inputs to share one capture chip, it can sometimes produce images with interlaced frames reversed resulting in poor image quality and a distinctive comb edge appearance. Increasing this setting allows you to force additional image captures before one is selected as the captured frame. This allows the capture hardware to 'settle down' and produce better quality images at the price of lesser capture rates. This option has no effect on (a) network cameras, or (b) where multiple inputs do not share a capture chip. This option addresses a similar problem to the **ZM_V4L_MULTI_BUFFER** option and you should normally change the value of only one of the options at a time. If you have different capture cards that need different values you can override them in each individual monitor on the source page.

FORCED_ALARM_SCORE - The 'zmu' utility can be used to force an alarm on a monitor rather than rely on the motion detection algorithms. This option determines what score to give these alarms to distinguish them from regular ones. It must be 255 or less.

BULK_FRAME_INTERVAL - Traditionally ZoneMinder writes an entry into the Frames database table for each frame that is captured and saved. This works well in motion detection scenarios but when in a DVR situation ('Record' or 'Mocord' mode) this results in a huge number of frame writes and a lot of database and disk bandwidth for very little additional information. Setting this to a non-zero value will enable ZoneMinder to group these non-alarm frames

into one ‘bulk’ frame entry which saves a lot of bandwidth and space. The only disadvantage of this is that timing information for individual frames is lost but in constant frame rate situations this is usually not significant. This setting is ignored in Modect mode and individual frames are still written if an alarm occurs in Mocord mode also.

EVENT_CLOSE_MODE - When a monitor is running in a continuous recording mode (Record or Mocord) events are usually closed after a fixed period of time (the section length). However in Mocord mode it is possible that motion detection may occur near the end of a section. This option controls what happens when an alarm occurs in Mocord mode. The ‘time’ setting means that the event will be closed at the end of the section regardless of alarm activity. The ‘idle’ setting means that the event will be closed at the end of the section if there is no alarm activity occurring at the time otherwise it will be closed once the alarm is over meaning the event may end up being longer than the normal section length. The ‘alarm’ setting means that if an alarm occurs during the event, the event will be closed once the alarm is over regardless of when this occurs. This has the effect of limiting the number of alarms to one per event and the events will be shorter than the section length if an alarm has occurred.

CREATE_ANALYSIS_IMAGES - By default during an alarm ZoneMinder records both the raw captured image and one that has been analysed and had areas where motion was detected outlined. This can be very useful during zone configuration or in analysing why events occurred. However it also incurs some overhead and in a stable system may no longer be necessary. This parameter allows you to switch the generation of these images off.

WEIGHTED_ALARM_CENTRES - ZoneMinder will always calculate the centre point of an alarm in a zone to give some indication of where on the screen it is. This can be used by the experimental motion tracking feature or your own custom extensions. In the alarmed or filtered pixels mode this is a simple midpoint between the extents of the detected pixels. However in the blob method this can instead be calculated using weighted pixel locations to give more accurate positioning for irregularly shaped blobs. This method, while more precise is also slower and so is turned off by default.

EVENT_IMAGE_DIGITS - As event images are captured they are stored to the filesystem with a numerical index. By default this index has three digits so the numbers start 001, 002 etc. This works for most scenarios as events with more than 999 frames are rarely captured. However if you have extremely long events and use external applications then you may wish to increase this to ensure correct sorting of images in listings etc. Warning, increasing this value on a live system may render existing events unviewable as the event will have been saved with the previous scheme. Decreasing this value should have no ill effects.

DEFAULT_ASPECT_RATIO - When specifying the dimensions of monitors you can click a checkbox to ensure that the width stays in the correct ratio to the height, or vice versa. This setting allows you to indicate what the ratio of these settings should be. This should be specified in the format <width value>:<height value> and the default of 4:3 normally be acceptable but 11:9 is another common setting. If the checkbox is not clicked when specifying monitor dimensions this setting has no effect.

USER_SELF_EDIT - Ordinarily only users with system edit privilege are able to change users details. Switching this option on allows ordinary users to change their passwords and their language settings

2.9.4 Options - API

Note: The ZoneMinder web interface does not use APIs and therefore, the tokens discussed here don’t apply to the ZoneMinder UI. These only apply to apps that use the ZoneMinder API, like zmNinja.

The API option screen allows you enable/disable APIs on a per user basis. Furthermore, it also allows you to “revoke” tokens allotted to users. Starting ZoneMinder 1.34, the API ecosystem was overhauled and we now support JWT tokens with a concept of refresh tokens and access tokens. This allows for authentication without the need for sending passwords with each authentication request. For a more detailed understanding of how this works, please refer to [API](#). Over time, more control will be added to this screen.

UPDATE

REVOKE ALL TOKENS

Username	Revoke Token	API Enabled
admin	<input type="checkbox"/>	<input checked="" type="checkbox"/>
home	<input type="checkbox"/>	<input checked="" type="checkbox"/>
zmes	<input type="checkbox"/>	<input type="checkbox"/>

The “Revoke All Tokens” button can be used to globally invalidate access tokens for all users. If tokens are revoked, the user(s) will need to re-authenticate with login and password. As of today, refresh tokens last for 24 hours and access tokens for 1 hour.

2.9.5 Options - Servers

Todo: needs to be refreshed

Options

Display	System	Config	Servers	Paths	Web	Images	Logging	Network	Email	Upload	X10	High B/W	Medium B/W	Low B/W	Phone B/W	eyeZm	Users
name										Mark							
Main										<input type="checkbox"/>							

AddNewServer Delete Cancel

Servers tab is used for setting up multiple ZoneMinder servers sharing the same database and using a shared file share for all event data. To add a new server use the Add Server button. All that is required is a Name for the Server and Hostname.

To delete a server mark that server and click the Delete button.

Please note that all servers must have a functional web UI as the live view must come from the monitor’s host server.

On each server, you will have to edit /etc/zm/zm.conf and set either ZM_SERVER_NAME=

2.9.6 Options - Storage

Id	Name	Path	Type	Scheme	Server	DiskSpace	Mark
1	Default	/var/cache/zoneminder/events	local	Medium	Default	20.93GB of 29.21GB	<input type="checkbox"/>
2	S3	/media/S3	s3fs	Medium	S3	20.2MB of 256TB	<input type="checkbox"/>

Storage tab is used to setup storage areas for recorded Events. To add a new area use the Add New Storage button.

By default storage on local drive is automatically set up on installation. When no area is specified events will be stored to a default built-in location, which for example on Ubuntu is /var/cache/zoneminder/events.

Name: Storage names - can be anything

Path: String path to storage location for example /media/Videos

Url: Used for S3 communication - format `s3fs://ACCESS_KEY_ID:SECRET_ACCESS_KEY@s3.ca-central-1.amazonaws.com/bucket-name/events`

Supported storage types:

- Local - Local/mounted or network storage in local network
- s3fs - S3 mounted drive

Some users may require more advanced storage such as S3 provided by amazon or others.

2.9.7 S3 storage setup

You must use s3fs to mount the S3 bucket in your fs tree. Telling ZoneMinder that the location is S3 will let it use more efficient code to send and delete the event data.

Refer to this guide for installation and configuration of s3fs - <https://github.com/s3fs-fuse/s3fs-fuse>

Adding credentials to passwd_file

Create credentials file `echo ACCESS_KEY_ID:SECRET_ACCESS_KEY > /etc/passwd-s3fs`

Set file permissions `chmod 600 /etc/passwd-s3fs`

S3 mounting with fstab `s3fs#bucket_name /media/S3 fuse _netdev,allow_other,uid=33, url=https://s3.ca-central-1.amazonaws.com,passwd_file=/etc/passwd-s3fs, umask=022 0 0`

Setting up storage.

1. Click on Add new Storage
2. Set path to `/media/S3`
3. Add `Url` `s3fs://username:password@s3.ca-central-1.amazonaws.com/bucket-name/events`
4. Set type to `s3fs`
5. Save settings and monitor logs for errors

2.9.8 Options - Web

This screen lets you customize several aspects of the web interface of ZoneMinder. A partial screenshot is shown below:

Display	WEB_TITLE	<input type="text" value="ZoneMinder"/>	The title displayed wherever the site references itself. (?)
System	WEB_TITLE_PREFIX	<input type="text" value="ZM"/>	The title prefix displayed on each window (?)
Config	HOME_URL	<input type="text" value="https://yourserver/zm/"/>	The url used in the home/logo area of the navigation bar. (?)
API	HOME_CONTENT	<input type="text" value="ZoneMinder"/>	The content of the home button. (?)
Servers	WEB_CONSOLE_BANNER	<input type="text"/>	Arbitrary text message near the top of the console (?)
Storage			
Web			
Images			
Logging			
Network			
Email			

WEB_TITLE -

Todo: not quite sure what this does. Seems to change the “target” name - not sure what effect it is supposed to have.

WEB_TITLE_PREFIX - If you have more than one installation of ZoneMinder it can be helpful to display different titles for each one. Changing this option allows you to customise the window titles to include further information to aid identification.

HOME_URL - the link to navigate to, when a user clicks on the top left title.

HOME_CONTENT - The actual text that is shown on the top left corner. You can choose to leave it empty and put in a logo in a custom CSS as well.

WEB_CONSOLE_BANNER - Allows the administrator to place an arbitrary text message near the top of the web console. This is useful for the developers to display a message which indicates the running instance of ZoneMinder is a development snapshot, but it can also be used for any other purpose as well.

WEB_EVENT_DISK_SPACE - Adds another column to the listing of events showing the disk space used by the event. This will impart a small overhead as it will call du on the event directory. In practice this overhead is fairly small but may be noticeable on IO-constrained systems.

WEB_RESIZE_CONSOLE - Traditionally the main ZoneMinder web console window has resized itself to shrink to a size small enough to list only the monitors that are actually present. This is intended to make the window more unobtrusive but may not be to everyone's tastes, especially if opened in a tab in browsers which support this kind of layout. Switch this option off to have the console window size left to the user's preference.

WEB_ID_ON_CONSOLE - Some find it useful to have the monitor id always visible on the console. This option will add a column listing it. Note that if it is disabled, you can always hover over the monitor to see the id as well.

WEB_POPUP_ON_ALARM - When viewing a live monitor stream you can specify whether you want the window to pop to the front if an alarm occurs when the window is minimised or behind another window. This is most useful if your monitors are over doors for example when they can pop up if someone comes to the doorway.

WEB_SOUND_ON_ALARM - When viewing a live monitor stream you can specify whether you want the window to play a sound to alert you if an alarm occurs.

WEB_ALARM_SOUND - You can specify a sound file to play if an alarm occurs whilst you are watching a live monitor stream. So long as your browser understands the format it does not need to be any particular type. This file should be placed in the sounds directory defined earlier.

WEB_COMPACT_MONTAGE - The montage view shows the output of all of your active monitors in one window. This include a small menu and status information for each one. This can increase the web traffic and make the window larger than may be desired. Setting this option on removes all this extraneous information and just displays the images.

WEB_EVENT_SORT_FIELD - Events in lists can be initially ordered in any way you want. This option controls what field is used to sort them. You can modify this ordering from filters or by clicking on headings in the lists themselves. Bear in mind however that the 'Prev' and 'Next' links, when scrolling through events, relate to the ordering in the lists and so not always to time based ordering.

WEB_EVENT_SORT_ORDER - Events in lists can be initially ordered in any way you want. This option controls what order (ascending or descending) is used to sort them. You can modify this ordering from filters or by clicking on headings in the lists themselves. Bear in mind however that the 'Prev' and 'Next' links, when scrolling through events, relate to the ordering in the lists and so not always to time based ordering.

WEB_EVENTS_PER_PAGE - In the event list view you can either list all events or just a page at a time. This option controls how many events are listed per page in paged mode and how often to repeat the column headers in non-paged mode.

WEB_LIST_THUMBS - Ordinarily the event lists just display text details of the events to save space and time. By switching this option on you can also display small thumbnails to help you identify events of interest. The size of these thumbnails is controlled by the following two options.

WEB_LIST_THUMB_WIDTH - This options controls the width of the thumbnail images that appear in the event lists. It should be fairly small to fit in with the rest of the table. If you prefer you can specify a height instead in the next option but you should only use one of the width or height and the other option should be set to zero. If both width and height are specified then width will be used and height ignored.

WEB_LIST_THUMB_HEIGHT - This options controls the height of the thumbnail images that appear in the event lists. It should be fairly small to fit in with the rest of the table. If you prefer you can specify a width instead in the previous option but you should only use one of the width or height and the other option should be set to zero. If both width and height are specified then width will be used and height ignored.

WEB_USE_OBJECT_TAGS - There are two methods of including media content in web pages. The most common way is use the EMBED tag which is able to give some indication of the type of content. However this is not a standard part of HTML. The official method is to use OBJECT tags which are able to give more information allowing the correct media viewers etc to be loaded. However these are less widely supported and content may be specifically tailored to a particular platform or player. This option controls whether media content is enclosed in EMBED tags only or whether, where appropriate, it is additionally wrapped in OBJECT tags. Currently OBJECT tags are only used in a limited number of circumstances but they may become more widespread in the future. It is suggested that you leave this option on unless you encounter problems playing some content.

WEB_XFRAME_WARN - When creating a Web Site monitor, if the target web site has X-Frame-Options set to sameorigin in the header, the site will not display in ZoneMinder. This is a design feature in most modern browsers. When this condition occurs, ZoneMinder will write a warning to the log file. To get around this, one can install a browser plugin or extension to ignore X-Frame headers, and then the page will display properly. Once the plugin or extension has ben installed, the end user may choose to turn this warning off

WEB_FILTER_SOURCE - This option only affects monitors with a source type of Ffmpeg, Libvlc, or WebSite. This setting controls what information is displayed in the Source column on the console. Selecting 'None' will not filter anything. The entire source string will be displayed, which may contain sensitive information. Selecting 'NoCredentials' will strip out usernames and passwords from the string. If there are any port numbers in the string and they are common (80, 554, etc) then those will be removed as well. Selecting 'Hostname' will filter out all information except for the hostname or ip address. When in doubt, stay with the default 'Hostname'. This feature uses the php function 'url_parts' to identify the various pieces of the url. If the url in question is unusual or not standard in some way, then filtering may not produce the desired results.

2.9.9 Options - Images

This screen lets you control various image quality settings for live and recorded events. A partial screenshot is shown below:

Load: 1.37 DB:24/151 Storage: Default: 52% /dev/shm: 21%

COLOUR_JPEG_FILES	<input type="checkbox"/>	Colourise greyscale JPEG files (?)
ADD_JPEG_COMMENTS	<input type="checkbox"/>	Add jpeg timestamp annotations as file header comments (?)
JPEG_FILE_QUALITY	70	Set the JPEG quality setting for the saved event files (1-100) (?)
JPEG_ALARM_FILE_QUALITY	0	Set the JPEG quality setting for the saved event files during an alarm (1-100) (?)
JPEG_STREAM_QUALITY	70	Set the JPEG quality setting for the streamed 'live' images (1-100) (?)
MPEG_TIMED_FRAMES	<input checked="" type="checkbox"/>	Tag video frames with a timestamp for more realistic streaming (?)
MPEG_LIVE_FORMAT	swf	What format 'live' video streams are played in (?)

COLOUR_JPEG_FILES - Cameras that capture in greyscale can write their captured images to jpeg files with a corresponding greyscale colour space. This saves a small amount of disk space over colour ones. However some tools such as ffmpeg either fail to work with this colour space or have to convert it beforehand. Setting this option to yes uses up a little more space but makes creation of MPEG files much faster.

ADD_JPEG_COMMENTS - JPEG files may have a number of extra fields added to the file header. The comment field may have any kind of text added. This options allows you to have the same text that is used to annotate the image additionally included as a file header comment. If you archive event images to other locations this may help you locate images for particular events or times if you use software that can read comment headers.

JPEG_FILE_QUALITY - When ZoneMinder detects an event it will save the images associated with that event to files. These files are in the JPEG format and can be viewed or streamed later. This option specifies what image quality should be used to save these files. A higher number means better quality but less compression so will take up more disk space and take longer to view over a slow connection. By contrast a low number means smaller, quicker to view, files but at the price of lower quality images. This setting applies to all images written except if the capture image has caused an alarm and the alarm file quality option is set at a higher value when that is used instead.

JPEG_ALARM_FILE_QUALITY - This value is equivalent to the regular jpeg file quality setting above except that it only applies to images saved while in an alarm state and then only if this value is set to a higher quality setting than

the ordinary file setting. If set to a lower value then it is ignored. Thus leaving it at the default of 0 effectively means to use the regular file quality setting for all saved images. This is to prevent accidentally saving important images at a worse quality setting.

JPEG_STREAM_QUALITY - When viewing a 'live' stream for a monitor ZoneMinder will grab an image from the buffer and encode it into JPEG format before sending it. This option specifies what image quality should be used to encode these images. A higher number means better quality but less compression so will take longer to view over a slow connection. By contrast a low number means quicker to view images but at the price of lower quality images. This option does not apply when viewing events or still images as these are usually just read from disk and so will be encoded at the quality specified by the previous options.

MPEG_TIMED_FRAMES - When using streamed MPEG based video, either for live monitor streams or events, ZoneMinder can send the streams in two ways. If this option is selected then the timestamp for each frame, taken from it's capture time, is included in the stream. This means that where the frame rate varies, for instance around an alarm, the stream will still maintain it's 'real' timing. If this option is not selected then an approximate frame rate is calculated and that is used to schedule frames instead. This option should be selected unless you encounter problems with your preferred streaming method.

MPEG_LIVE_FORMAT - When using MPEG mode ZoneMinder can output live video. However what formats are handled by the browser varies greatly between machines. This option allows you to specify a video format using a file extension format, so you would just enter the extension of the file type you would like and the rest is determined from that. The default of 'asf' works well under Windows with Windows Media Player but I'm currently not sure what, if anything, works on a Linux platform. If you find out please let me know! If this option is left blank then live streams will revert to being in motion jpeg format.

MPEG_REPLAY_FORMAT - When using MPEG mode ZoneMinder can replay events in encoded video format. However what formats are handled by the browser varies greatly between machines. This option allows you to specify a video format using a file extension format, so you would just enter the extension of the file type you would like and the rest is determined from that. The default of 'asf' works well under Windows with Windows Media Player and 'mpg', or 'avi' etc should work under Linux. If you know any more then please let me know! If this option is left blank then live streams will revert to being in motion jpeg format.

RAND_STREAM - Some browsers can cache the streams used by ZoneMinder. In order to prevent this a harmless random string can be appended to the url to make each invocation of the stream appear unique.

OPT_CAMBOZOLA - Cambozola is a handy low fat cheese flavoured Java applet that ZoneMinder uses to view image streams on browsers such as Internet Explorer that don't natively support this format. If you use this browser it is highly recommended to install this from [this link](#) however if it is not installed still images at a lower refresh rate can still be viewed. Note that practically, if you are not using an old version of IE, you will likely not need this.

PATH_CAMBOZOLA - Leave this as 'cambozola.jar' if cambozola is installed in the same directory as the ZoneMinder web client files.

RELOAD_CAMBOZOLA - Cambozola allows for the viewing of streaming MJPEG however it caches the entire stream into cache space on the computer, setting this to a number > 0 will cause it to automatically reload after that many seconds to avoid filling up a hard drive.

OPT_FFMPEG - ZoneMinder can optionally encode a series of video images into an MPEG encoded movie file for viewing, downloading or storage. This option allows you to specify whether you have the ffmpeg tools installed. Note that creating MPEG files can be fairly CPU and disk intensive and is not a required option as events can still be reviewed as video streams without it.

PATH_FFMPEG - This path should point to where ffmpeg has been installed.

FFMPEG_INPUT_OPTIONS - Ffmpeg can take many options on the command line to control the quality of video produced. This option allows you to specify your own set that apply to the input to ffmpeg (options that are given before the -i option). Check the ffmpeg documentation for a full list of options which may be used here.

FFMPEG_OUTPUT_OPTIONS - Ffmpeg can take many options on the command line to control the quality of video produced. This option allows you to specify your own set that apply to the output from ffmpeg (options that are given

after the `-i` option). Check the `ffmpeg` documentation for a full list of options which may be used here. The most common one will often be to force an output frame rate supported by the video encoder.

FFMPEG_FORMATS - `Ffmpeg` can generate video in many different formats. This option allows you to list the ones you want to be able to select. As new formats are supported by `ffmpeg` you can add them here and be able to use them immediately. Adding a `*` after a format indicates that this will be the default format used for web video, adding `**` defines the default format for phone video.

FFMPEG_OPEN_TIMEOUT - When `Ffmpeg` is opening a stream, it can take a long time before failing; certain circumstances even seem to be able to lock indefinitely. This option allows you to set a maximum time in seconds to pass before closing the stream and trying to reopen it again.

2.9.10 Options - Logging

ZoneMinder has a powerful logging system. Understanding how to configure logging will help you track issues better. The logging options are accessed via `Options->Logging`. Let's follow along with an example. But before that, here is a basic construct of how logging works:

- Every component of ZoneMinder can generate different types of logs. Typically, `ERR` refers to an error condition that you should look at (in some cases, they are transient during startup/shutdown in which case they are usually benign). `INF` logs are informational, `WAR` are warning logs that might have a potential to cause issues, whilst `DBG` are debug logs that are useful when you need to debug a problems
- You can decide where these logs are written. Typically ZoneMinder writes logs to multiple sources: `* Syslog *` `Database *` individual files belonging to each component inside the logging folder configured

Consider for example, that you are trying to figure out why your “zmc 11” (i.e. Monitor 11) is not working. Obviously, you need to enable debug logs if you are not able to figure out what is going on with standard info logs. But you wouldn't want to write debug logs to the Database. Maybe, you also don't want it polluting your syslog and only want to write debug logs to the debug file of `_that_` component (`/var/log/zm/zmc_m11.log` for example). That is where customizing your logging is useful.

Logging example








LOG_LEVEL_SYSLOG	<div>Info</div> <div>Save logging output to the system log (?)</div>
LOG_LEVEL_FILE	<div>Debug</div> <div>Save logging output to component files (?)</div>
LOG_LEVEL_WEBLOG	<div>Fatal</div> <div>Save logging output to the weblog (?)</div>
LOG_LEVEL_DATABASE	<div>Fatal</div> <div>Save logging output to the database (?)</div>
LOG_DATABASE_LIMIT	<div>7 day</div> <div>Maximum number of log entries to retain (?)</div>
LOG_FFMPEG	<div><input type="checkbox"/></div> <div>Log FFMPEG messages (?)</div>
LOG_DEBUG	<div><input checked="" type="checkbox"/></div> <div>Switch debugging on (?)</div>
LOG_DEBUG_TARGET	<div>_zmesdetectl_zmc_m11</div> <div>What components should have extra debug enabled (?)</div>
LOG_DEBUG_LEVEL	<div>2</div> <div>What level of extra debug should be enabled (?)</div>
LOG_DEBUG_FILE	<div></div> <div>Where extra debug is output to (?)</div>

In the example above, I've configured my logging as follows:

- I only want to log INFO level logs to Syslog
- I want DEBUG logs to only go to the component file

- When it comes to my WEBLOG (what I see in the ZM Log window) and Database log, I only want FATAL logs (you may want to set this to WAR or INF)
- I don't want to save FFMPEG logs (this was a new feature added). FFMPEG generates a log of logs on its own that you should only enable if you are trying to figure out video playback related issues
- I have enabled LOG_DEBUG (unless you enable this, DEBUG logs won't be logged)
- The LOG_DEBUG_TARGET is useful if you don't want to enable DEBUG logs for every component. In this case, I'm only interested in debugging the ZM Event Server and Monitor 11. Nothing else will have debug logs enabled.
- I prefer to keep the LOG_DEBUG_FILE to empty. This creates nicely separate files in my log folder with component names

The other logging parameters are left to their defaults, like so:

LOG_CHECK_PERIOD	<div>900 </div> <div>Time period used when calculating overall system health (?)</div>
LOG_ALERT_WAR_COUNT	<div>1 </div> <div>Number of warnings indicating system alert state (?)</div>
LOG_ALERT_ERR_COUNT	<div>1 </div> <div>Number of errors indicating system alert state (?)</div>
LOG_ALERT_FAT_COUNT	<div>0 </div> <div>Number of fatal error indicating system alert state (?)</div>
LOG_ALARM_WAR_COUNT	<div>100 </div> <div>Number of warnings indicating system alarm state (?)</div>
LOG_ALARM_ERR_COUNT	<div>10 </div> <div>Number of errors indicating system alarm state (?)</div>
LOG_ALARM_FAT_COUNT	<div>1 </div> <div>Number of fatal error indicating system alarm state (?)</div>
RECORD_EVENT_STATS	<div><input checked="" type="checkbox"/></div> <div>Record event statistical information, switch off if too slow (?)</div>
RECORD_DIAG_IMAGES	<div><input type="checkbox"/></div> <div>Record intermediate alarm diagnostic images, can be very slow (?)</div>
DUMP_CORES	<div><input type="checkbox"/></div> <div>Create core files on unexpected process failure. (?)</div>
RECORD_DIAG_IMAGES_FIFO	<div><input type="checkbox"/></div> <div>Recording intermediate alarm diagnostic use fifo instead of files (faster) (?)</div>

A more comprehensive explanation of the various log options

LOG_LEVEL_SYSLOG - ZoneMinder logging is now more integrated between components and allows you to specify the destination for logging output and the individual levels for each. This option lets you control the level of logging output that goes to the system log. ZoneMinder binaries have always logged to the system log but now scripts and web logging is also included. To preserve the previous behaviour you should ensure this value is set to Info or Warning. This option controls the maximum level of logging that will be written, so Info includes Warnings and Errors etc. To disable entirely, set this option to None. You should use caution when setting this option to Debug as it can severely affect system performance. If you want debug you will also need to set a level and component below

LOG_LEVEL_FILE - ZoneMinder logging is now more integrated between components and allows you to specify the destination for logging output and the individual levels for each. This option lets you control the level of logging output that goes to individual log files written by specific components. This is how logging worked previously and although useful for tracking down issues in specific components it also resulted in many disparate log files. To preserve this behaviour you should ensure this value is set to Info or Warning. This option controls the maximum level of logging that will be written, so Info includes Warnings and Errors etc. To disable entirely, set this option to None. You should use caution when setting this option to Debug as it can severely affect system performance though file output has less impact than the other options. If you want debug you will also need to set a level and component below

LOG_LEVEL_WEBLOG - ZoneMinder logging is now more integrated between components and allows you to specify the destination for logging output and the individual levels for each. This option lets you control the level of logging output from the web interface that goes to the httpd error log. Note that only web logging from PHP and JavaScript files is included and so this option is really only useful for investigating specific issues with those components. This option controls the maximum level of logging that will be written, so Info includes Warnings and Errors etc. To disable entirely, set this option to None. You should use caution when setting this option to Debug as it can severely affect system performance. If you want debug you will also need to set a level and component below

LOG_LEVEL_DATABASE - ZoneMinder logging is now more integrated between components and allows you to specify the destination for logging output and the individual levels for each. This option lets you control the level of logging output that is written to the database. This is a new option which can make viewing logging output easier and more intuitive and also makes it easier to get an overall impression of how the system is performing. If you have a large or very busy system then it is possible that use of this option may slow your system down if the table becomes very large. Ensure you use the **LOG_DATABASE_LIMIT** option to keep the table to a manageable size. This option controls the maximum level of logging that will be written, so Info includes Warnings and Errors etc. To disable entirely, set this option to None. You should use caution when setting this option to Debug as it can severely affect system performance. If you want debug you will also need to set a level and component below

LOG_DATABASE_LIMIT - If you are using database logging then it is possible to quickly build up a large number of entries in the Logs table. This option allows you to specify how many of these entries are kept. If you set this option to a number greater than zero then that number is used to determine the maximum number of rows, less than or equal to zero indicates no limit and is not recommended. You can also set this value to time values such as '<n> day' which will limit the log entries to those newer than that time. You can specify 'hour', 'day', 'week', 'month' and 'year', note that the values should be singular (no 's' at the end). The Logs table is pruned periodically so it is possible for more than the expected number of rows to be present briefly in the meantime.

LOG_DEBUG - ZoneMinder components usually support debug logging available to help with diagnosing problems. Binary components have several levels of debug whereas more other components have only one. Normally this is disabled to minimise performance penalties and avoid filling logs too quickly. This option lets you switch on other options that allow you to configure additional debug information to be output. Components will pick up this instruction when they are restarted.

LOG_DEBUG_TARGET - There are three scopes of debug available. Leaving this option blank means that all components will use extra debug (not recommended). Setting this option to '_<component>', e.g. `_zmc`, will limit extra debug to that component only. Setting this option to '_<component>_<identity>', e.g. `_zmc_m1` will limit extra debug to that instance of the component only. This is ordinarily what you probably want to do. To debug scripts use their names without the .pl extension, e.g. `_zmvideo` and to debug issues with the web interface use `_web`. You can specify multiple targets by separating them with 'l' characters.

LOG_DEBUG_LEVEL - There are 9 levels of debug available, with higher numbers being more debug and level 0 being no debug. However not all levels are used by all components. Also if there is debug at a high level it is usually likely to be output at such a volume that it may obstruct normal operation. For this reason you should set the level carefully and cautiously until the degree of debug you wish to see is present. Scripts and the web interface only have one level so this is an on/off type option for them.

LOG_DEBUG_FILE - This option allows you to specify a different target for debug output. All components have a default log file which will normally be in /tmp or /var/log and this is where debug will be written to if this value is empty. Adding a path here will temporarily redirect debug, and other logging output, to this file. This option is a simple filename and you are debugging several components then they will all try and write to the same file with undesirable consequences. Appending a '+' to the filename will cause the file to be created with a '<pid>' suffix containing your process id. In this way debug from each run of a component is kept separate. This is the recommended setting as it will also prevent subsequent runs from overwriting the same log. You should ensure that permissions are set up to allow writing to the file and directory specified here.

LOG_CHECK_PERIOD - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to indicate what period of historical events are used in this calculation. This value is expressed in seconds and is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALERT_WAR_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many warnings must have occurred within the defined time period to generate an overall system alert state. A value of zero means warnings are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALERT_ERR_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many errors must have occurred within the defined time period to generate an overall system alert state. A value of zero means errors are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALERT_FAT_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many fatal errors (including panics) must have occurred within the defined time period to generate an overall system alert state. A value of zero means fatal errors are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALARM_WAR_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many warnings must have occurred within the defined time period to generate an overall system alarm state. A value of zero means warnings are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALARM_ERR_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many errors must have occurred within the defined time period to generate an overall system alarm state. A value of zero means errors are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

LOG_ALARM_FAT_COUNT - When ZoneMinder is logging events to the database it can retrospectively examine the number of warnings and errors that have occurred to calculate an overall state of system health. This option allows you to specify how many fatal errors (including panics) must have occurred within the defined time period to generate an overall system alarm state. A value of zero means fatal errors are not considered. This value is ignored if LOG_LEVEL_DATABASE is set to None.

RECORD_EVENT_STATS - This version of ZoneMinder records detailed information about events in the Stats table. This can help in profiling what the optimum settings are for Zones though this is tricky at present. However in future

releases this will be done more easily and intuitively, especially with a large sample of events. The default option of 'yes' allows this information to be collected now in readiness for this but if you are concerned about performance you can switch this off in which case no Stats information will be saved.

RECORD_DIAG_IMAGES - In addition to recording event statistics you can also record the intermediate diagnostic images that display the results of the various checks and processing that occur when trying to determine if an alarm event has taken place. There are several of these images generated for each frame and zone for each alarm or alert frame so this can have a massive impact on performance. Only switch this setting on for debug or analysis purposes and remember to switch it off again once no longer required.

RECORD_DIAG_IMAGES_FIFO - Adds fifo options for diagnostic images for much lower impact diagnostics mode. Diagnostic images are only written when there is a client (like a web browser) listening for them. If there is no active client connected, FIFO images are skipped. Note that this feature also needs **RECORD_DIAG_IMAGES** to be on. **Note:** Your monitor needs to be in some recording mode (modect/mocord/etc.)

In addition to creating diagnostic images, this feature also adds a json stream for the detection data so you can see in real time the pixels or blobs detected for the motion. This allows for easy real time stream of both delta and reference images (as video streams) along with the detection numbers.

Once you turn on **RECORD_DIAG_IMAGES** and the new **RECORD_DIAG_IMAGES_FIFO** in the logging options you can then use 3 new remote stream urls:

- The delta images as an MJPEG stream (great to see where it is seeing the motion!): `https://portal/zm/cgi-bin/nph-zms?mode=jpeg&bitrate=2&buffer=0&source=fifo&format=delta&monitor=1&maxfps=` (change monitor, portal to your values. <auth> could be &user=user&pass=pass or &auth=authval or &token=access_token)
- The reference images as an MJPEG stream: `https://portal/zm/cgi-bin/nph-zms?mode=jpeg&bitrate=2&buffer=0&source=fifo&format=reference&monitor=1&maxfps=5<auth>` (change monitor, portal to your values. <auth> could be &user=user&pass=pass or &auth=authval or &token=access_token)
- text json raw stream: `https://portal/zm/cgi-bin/nph-zms?&buffer=0&source=fifo&format=raw&monitor=1<auth>` (change monitor, portal to your values, <auth> could be &user=user&pass=pass or &auth=authval or &token=access_token)

This will output a text stream on the browser like:

```
{ "zone":5, "type": "ALRM", "pixels":778661, "avg_diff":50}
{ "zone":5, "type": "FILT", "pixels":762704}
{ "zone":5, "type": "RBLB", "pixels":728102, "blobs":5}
{ "zone":5, "type": "FBLB", "pixels":728021, "blobs":2}
{ "zone":6, "type": "ALRM", "pixels":130844, "avg_diff":44}
{ "zone":6, "type": "FILT", "pixels":128608}
```

There are four types of events right now: Alarm (ALRM), Filter (FILT), Raw Blob (RBLB) and Filtered Blobs (FBLB) that correspond to those stages of analysis. It will show the number of pixels detected (along with average pixel difference against the threshold) and number of blobs at each stage.

For example, here is a delta image stream from one of my monitors showing in live mode:

`https://myserver/cgi-bin/nph-zms?mode=jpeg&bitrate=2&buffer=0&source=fifo&format=delta&mon.`

DUMP_CORES - When an unrecoverable error occurs in a ZoneMinder binary process it has traditionally been trapped and the details written to logs to aid in remote analysis. However in some cases it is easier to diagnose the error if a core file, which is a memory dump of the process at the time of the error, is created. This can be interactively analysed in the debugger and may reveal more or better information than that available from the logs. This option is recommended for advanced users only otherwise leave at the default. Note using this option to trigger core files will mean that there will be no indication in the binary logs that a process has died, they will just stop, however the zmdc

log will still contain an entry. Also note that you may have to explicitly enable core file creation on your system via the 'ulimit -c' command or other means otherwise no file will be created regardless of the value of this option.

2.9.11 Options - Network

Display System Config API Servers Storage Web Images Logging Network Email Upload X10	HTTP_VERSION <div> <input type="radio"/> 1.1 <input checked="" type="radio"/> 1.0 <small>The version of HTTP that ZoneMinder will use to connect (?)</small> </div>
	HTTP_UA <div> <input type="text" value="ZoneMinder"/> <small>The user agent that ZoneMinder uses to identify itself (?)</small> </div>
	HTTP_TIMEOUT <div> <input type="text" value="2500"/> <small>How long ZoneMinder waits before giving up on images (milliseconds) (?)</small> </div>
	<div> only if you plan on using multi-port → MIN_STREAMING_PORT <div> <input type="text" value="30000"/> <small>Alternate port range to contact for streaming video. (?)</small> </div> </div>
	MIN_RTP_PORT <div> <input type="text" value="40200"/> <small>Minimum port that ZoneMinder will listen for RTP traffic on (?)</small> </div>
	MAX_RTP_PORT <div> <input type="text" value="40499"/> <small>Maximum port that ZoneMinder will listen for RTP traffic on (?)</small> </div>

HTTP_VERSION - ZoneMinder can communicate with network cameras using either of the HTTP/1.1 or HTTP/1.0 standard. A server will normally fall back to the version it supports with no problem so this should usually be left at the default. However it can be changed to HTTP/1.0 if necessary to resolve particular issues.

HTTP_UA - When ZoneMinder communicates with remote cameras it will identify itself using this string and it's version number. This is normally sufficient, however if a particular camera expects only to communicate with certain browsers then this can be changed to a different string identifying ZoneMinder as Internet Explorer or Netscape etc.

HTTP_TIMEOUT - When retrieving remote images ZoneMinder will wait for this length of time before deciding that an image is not going to arrive and taking steps to retry. This timeout is in milliseconds (1000 per second) and will apply to each part of an image if it is not sent in one whole chunk.

MIN_STREAMING_PORT - ZoneMinder supports a concept called multi-port streaming. The core concept is that modern browsers like Chrome limit the number of simultaneous connections allowed from a specific domain (host name+port). In the case of Chrome this value is 6, which means you can't see more than 6 simultaneous streams from your server at one time. However, if the streams originated from different ports (or sub domains), this limitation would not apply. When you enable this option with a value (in this case, 30000), the streams from the monitors will originate from 30000 plus the monitor ID, effectively overcoming this limitation. **Note that this also needs additional setup your webserver configuration before this can start to work.** Please refer to [this article](#) on how to setup multi port streaming on Apache.

MIN_RTP_PORT - When ZoneMinder communicates with MPEG4 capable cameras using RTP with the unicast method it must open ports for the camera to connect back to for control and streaming purposes. This setting specifies the minimum port number that ZoneMinder will use. Ordinarily two adjacent ports are used for each camera, one for control packets and one for data packets. This port should be set to an even number, you may also need to open up a hole in your firewall to allow cameras to connect back if you wish to use unicasting.

MAX_RTP_PORT - When ZoneMinder communicates with MPEG4 capable cameras using RTP with the unicast method it must open ports for the camera to connect back to for control and streaming purposes. This setting specifies the maximum port number that ZoneMinder will use. Ordinarily two adjacent ports are used for each camera, one for control packets and one for data packets. This port should be set to an even number, you may also need to open up a

hole in your firewall to allow cameras to connect back if you wish to use unicasting. You should also ensure that you have opened up at least two ports for each monitor that will be connecting to unicasting network cameras.

2.9.12 Options - Email

Display System Config API Servers Storage Web Images Logging Network Email Upload X10 High B/W Medium B/W Low B/W Users	OPT_EMAIL	<input type="checkbox"/>	Should ZoneMinder email you details of events that match corresponding filters (?)
	EMAIL_ADDRESS	<input type="text"/>	The email address to send matching event details to (?)
	EMAIL_SUBJECT	ZoneMinder: Alarm - %MN%- %EI% (%E!	The subject of the email used to send matching event details (?)
	EMAIL_BODY	Hello, An alarm has been detected on your installation of the ZoneMinder. The details are as follows :-	The body of the email used to send matching event details (?)
	OPT_MESSAGE	<input type="checkbox"/>	Should ZoneMinder message you with details of events that match corresponding filters (?)
	MESSAGE_ADDRESS	<input type="text"/>	The email address to send matching event details (?)
	MESSAGE_SUBJECT	ZoneMinder: Alarm - %MN%- %EI%	The subject of the message used to send matching event details (?)
	MESSAGE_BODY	ZM alarm detected - %EL% secs, %EF%/%EFA% frames, t%EST%/m%ESM%/a%ESA% score.	

OPT_EMAIL - In ZoneMinder you can create event filters that specify whether events that match certain criteria should have their details emailed to you at a designated email address. This will allow you to be notified of events as soon as they occur and also to quickly view the events directly. This option specifies whether this functionality should be available. The email created with this option can be any size and is intended to be sent to a regular email reader rather than a mobile device.

EMAIL_ADDRESS - This option is used to define the email address that any events that match the appropriate filters will be sent to.

EMAIL_SUBJECT - This option is used to define the subject of the email that is sent for any events that match the appropriate filters.

EMAIL_BODY - This option is used to define the content of the email that is sent for any events that match the appropriate filters.

Todo: check if any other tags have been added

Token	Description
%EI%	Id of the event
%EN%	Name of the event
%EC%	Cause of the event
%ED%	Event description
%ET%	Time of the event
%EL%	Length of the event
%EF%	Number of frames in the event
%EFA%	Number of alarm frames in the event
%EST%	Total score of the event
%ESA%	Average score of the event
%ESM%	Maximum score of the event
%EP%	Path to the event
%EPS%	Path to the event stream
%EPI%	Path to the event images
%EPI1%	Path to the first alarmed event image
%EPIM%	Path to the (first) event image with the highest score
%EI1%	Attach first alarmed event image
%EIM%	Attach (first) event image with the highest score
%EIMOD%	Attach image of object detected. Requires event notfn. server setup and machine learning hooks
%EV%	Attach event mpeg video
%MN%	Name of the monitor
%MET%	Total number of events for the monitor
%MEH%	Number of events for the monitor in the last hour
%MED%	Number of events for the monitor in the last day
%MEW%	Number of events for the monitor in the last week
%MEM%	Number of events for the monitor in the last month
%MEA%	Number of archived events for the monitor
%MP%	Path to the monitor window
%MPS%	Path to the monitor stream
%MPI%	Path to the monitor recent image
%FN%	Name of the current filter that matched
%FP%	Path to the current filter that matched
%ZP%	Path to your ZoneMinder console

OPT_MESSAGE - In ZoneMinder you can create event filters that specify whether events that match certain criteria should have their details sent to you at a designated short message email address. This will allow you to be notified of events as soon as they occur. This option specifies whether this functionality should be available. The email created by this option will be brief and is intended to be sent to an SMS gateway or a minimal mail reader such as a mobile device or phone rather than a regular email reader.

MESSAGE_ADDRESS - This option is used to define the short message email address that any events that match the appropriate filters will be sent to.

MESSAGE_SUBJECT - This option is used to define the subject of the message that is sent for any events that match the appropriate filters.

MESSAGE_BODY - This option is used to define the content of the message that is sent for any events that match the appropriate filters.

NEW_MAIL_MODULES - Traditionally ZoneMinder has used the `MIME::Entity` perl module to construct and send notification emails and messages. Some people have reported problems with this module not being present at all or flexible enough for their needs. If you are one of those people this option allows you to select a new mailing method using `MIME::Lite` and `Net::SMTP` instead. This method was contributed by Ross Melin and should work for everyone but has not been extensively tested so currently is not selected by default.

EMAIL_HOST - If you have chosen SMTP as the method by which to send notification emails or messages then this option allows you to choose which SMTP server to use to send them. The default of localhost may work if you have the sendmail, exim or a similar daemon running however you may wish to enter your ISP's SMTP mail server here.

FROM_EMAIL - The emails or messages that will be sent to you informing you of events can appear to come from a designated email address to help you with mail filtering etc. An address of something like ZoneMinder@your.domain is recommended.

URL - The emails or messages that will be sent to you informing you of events can include a link to the events themselves for easy viewing. If you intend to use this feature then set this option to the url of your installation as it would appear from where you read your email, e.g. `http://host.your.domain/zm/index.php`.

SSMTP_MAIL - SSMTP is a lightweight and efficient method to send email. The SSMTP application is not installed by default. **NEW_MAIL_MODULES** must also be enabled. Please visit the ZoneMinder [SSMTP Wiki page](#) for setup and configuration help.

SSMTP_PATH - The path to the SSMTP application. If path is not defined. Zoneminder will try to determine the path via shell command. Example path: `/usr/sbin/ssmtp`.

2.9.13 Options - Upload

A partial screenshot of the upload options is shown below:

Display	OPT_UPLOAD	<input type="checkbox"/>	Should ZoneMinder support uploading events from filters (?)
System	UPLOAD_ARCH_FORMAT	<input checked="" type="radio"/> tar <input type="radio"/> zip	What format the uploaded events should be created in. (?)
Config	UPLOAD_ARCH_COMPRESS	<input type="checkbox"/>	Should archive files be compressed (?)
API	UPLOAD_ARCH_ANALYSE	<input type="checkbox"/>	Include the analysis files in the archive (?)
Servers	UPLOAD_PROTOCOL	<input checked="" type="radio"/> ftp <input type="radio"/> sftp	What protocol to use to upload events (?)
Storage	UPLOAD_HOST	<input type="text"/>	The remote server to upload events to (?)
Web	UPLOAD_PORT	<input type="text"/>	The port on the remote upload server, if not the default (SFTP only) (?)
Images			
Logging			
Network			
Email			
Upload			
X10			

OPT_UPLOAD - In ZoneMinder you can create event filters that specify whether events that match certain criteria should be uploaded to a remote server for archiving. This option specifies whether this functionality should be available.

UPLOAD_ARCH_FORMAT - Uploaded events may be stored in either .tar or .zip format, this option specifies which. Note that to use this you will need to have the Archive::Tar and/or Archive::Zip perl modules installed.

UPLOAD_ARCH_COMPRESS - When the archive files are created they can be compressed. However in general since the images are compressed already this saves only a minimal amount of space versus utilising more CPU in their creation. Only enable if you have CPU to waste and are limited in disk space on your remote server or bandwidth.

UPLOAD_ARCH_ANALYSE - When the archive files are created they can contain either just the captured frames or both the captured frames and, for frames that caused an alarm, the analysed image with the changed area highlighted. This option controls files are included. Only include analysed frames if you have a high bandwidth connection to the

remote server or if you need help in figuring out what caused an alarm in the first place as archives with these files in can be considerably larger.

UPLOAD_PROTOCOL - ZoneMinder can upload events to a remote server using either FTP or SFTP. Regular FTP is widely supported but not necessarily very secure whereas SFTP (Secure FTP) runs over an ssh connection and so is encrypted and uses regular ssh ports. Note that to use this you will need to have the appropriate perl module, either Net::FTP or Net::SFTP installed depending on your choice.

UPLOAD_HOST - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the name, or ip address, of the server to use.

UPLOAD_PORT - You can use filters to instruct ZoneMinder to upload events to a remote server. If you are using the SFTP protocol then this option allows you to specify a particular port to use for connection. If this option is left blank then the default, port 22, is used. This option is ignored for FTP uploads.

UPLOAD_USER - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the username that ZoneMinder should use to log in for transfer.

UPLOAD_PASS - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the password that ZoneMinder should use to log in for transfer. If you are using certificate based logins for SFTP servers you can leave this option blank.

UPLOAD_LOC_DIR - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the local directory that ZoneMinder should use for temporary upload files. These are files that are created from events, uploaded and then deleted.

UPLOAD_REM_DIR - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the remote directory that ZoneMinder should use to upload event files to.

UPLOAD_TIMEOUT - You can use filters to instruct ZoneMinder to upload events to a remote server. This option indicates the maximum inactivity timeout (in seconds) that should be tolerated before ZoneMinder determines that the transfer has failed and closes down the connection.

UPLOAD_STRICT - You can require SFTP uploads to verify the host key of the remote server for protection against man-in-the-middle attacks. You will need to add the server's key to the known_hosts file. On most systems, this will be ~/.ssh/known_hosts, where ~ is the home directory of the web server running ZoneMinder.

UPLOAD_FTP_PASSIVE - You can use filters to instruct ZoneMinder to upload events to a remote ftp server. This option indicates that ftp transfers should be done in passive mode. This uses a single connection for all ftp activity and, whilst slower than active transfers, is more robust and likely to work from behind firewalls. This option is ignored for SFTP transfers.

UPLOAD_DEBUG - You can use filters to instruct ZoneMinder to upload events to a remote server. If you are having (or expecting) troubles with uploading events then setting this to 'yes' permits additional information to be generated by the underlying transfer modules and included in the logs.

2.9.14 Options - X10

OPT_X10	<input type="checkbox"/>	Support interfacing with X10 devices (?)
X10_DEVICE	<input type="text" value="/dev/ttyS0"/>	What device is your X10 controller connected on (?)
X10_HOUSE_CODE	<input type="text" value="A"/>	What X10 house code should be used (?)
X10_DB_RELOAD_INTERVAL	<input type="text" value="60"/>	How often (in seconds) the X10 daemon reloads the monitors from the database (?)

OPT_X10 - If you have an X10 Home Automation setup in your home you can use ZoneMinder to initiate or react to X10 signals if your computer has the appropriate interface controller. This option indicates whether X10 options will be available in the browser client.

X10_DEVICE - If you have an X10 controller device (e.g. XM10U) connected to your computer this option details which port it is connected on, the default of /dev/ttyS0 maps to serial or com port 1.

X10_HOUSE_CODE - X10 devices are grouped together by identifying them as all belonging to one House Code. This option details what that is. It should be a single letter between A and P.

X10_DB_RELOAD_INTERVAL - The zmx10 daemon periodically checks the database to find out what X10 events trigger, or result from, alarms. This option determines how frequently this check occurs, unless you change this area frequently this can be a fairly large value.

2.9.15 Options - High, Medium and Low B/W

There are a number of options that are grouped into bandwidth categories, this allows you to configure the ZoneMinder client to work optimally over the various access methods you might to access the client. You may want to use different modes depending on your network to preserve bandwidth.

A partial screenshot is shown below:

Display	WEB_H_REFRESH_MAIN	60	How often (in seconds) the main console window should refresh itself (?)
System	WEB_H_REFRESH_NAVBAR	10	How often (in seconds) the navigation header should refresh itself (?)
Config	WEB_H_REFRESH_CYCLE	10	How often (in seconds) the cycle watch window swaps to the next monitor (?)
API	WEB_H_REFRESH_IMAGE	20	How often (in seconds) the watched image is refreshed (if not streaming) (?)
Servers	WEB_H_REFRESH_STATUS	1	How often (in seconds) the status refreshes itself in the watch window (?)
Storage	WEB_H_REFRESH_EVENTS	5	How often (in seconds) the event listing is refreshed in the watch window (?)
Web	WEB_H_CAN_STREAM	<input checked="" type="radio"/> auto <input type="radio"/> yes <input type="radio"/> no	Override the automatic detection of browser streaming capability (?)
Images			
Logging			
Network			
Email			
Upload			
X10			
High B/W			
Medium B/W			
Low B/W			

The following options are available in H, M and L options. These 3 groups control what happens when the client is running in 'high', 'medium' and 'low' bandwidth mode respectively. In most cases the default values will be suitable as a starting point.

High - You should set these options for when accessing the ZoneMinder client over a local network or high speed link.

Medium - You should set these options for when accessing the ZoneMinder client over a slower cable or DSL link.

Slow - You should set these options for when accessing Zoneminder client over a slow network link.

WEB_H_REFRESH_MAIN, WEB_M_REFRESH_MAIN, WEB_L_REFRESH_MAIN - How often (in seconds) the main console window should refresh itself. The main console window lists a general status and the event totals for all monitors. This is not a trivial task and should not be repeated too frequently or it may affect the performance of the rest of the system.

WEB_H_REFRESH_CYCLE, WEB_M_REFRESH_CYCLE, WEB_L_REFRESH_CYCLE - How often (in seconds) the cycle watch window swaps to the next monitor. The cycle watch window is a method of continuously cycling between images from all of your monitors. This option determines how often to refresh with a new image.

WEB_H_REFRESH_IMAGE, WEB_M_REFRESH_IMAGE, WEB_L_REFRESH_IMAGE - How often (in seconds) the watched image is refreshed (if not streaming). The live images from a monitor can be viewed in either streamed or stills mode. This option determines how often a stills image is refreshed, it has no effect if streaming is selected.

WEB_H_REFRESH_STATUS, WEB_M_REFRESH_STATUS, WEB_L_REFRESH_STATUS - How often (in seconds) the status refreshes itself in the watch window. The monitor window is actually made from several frames. The one in the middle merely contains a monitor status which needs to refresh fairly frequently to give a true indication. This option determines that frequency.

WEB_H_REFRESH_EVENTS, WEB_M_REFRESH_EVENTS, WEB_L_REFRESH_EVENTS - How often (in seconds) the event listing is refreshed in the watch window. The monitor window is actually made from several frames. The lower frame contains a listing of the last few events for easy access. This option determines how often this is refreshed.

WEB_H_CAN_STREAM, WEB_M_CAN_STREAM, WEB_L_CAN_STREAM - If you know that your browser can handle image streams of the type 'multipart/x-mixed-replace' but ZoneMinder does not detect this correctly you can set this option to ensure that the stream is delivered with or without the use of the Cambozola plugin. Selecting 'yes' will tell ZoneMinder that your browser can handle the streams natively.

WEB_H_STREAM_METHOD, WEB_M_STREAM_METHOD, WEB_L_STREAM_METHOD - ZoneMinder can be configured to use either mpeg encoded video or a series of still jpeg images when sending video streams. This option defines which is used. If you choose mpeg you should ensure that you have the appropriate plugins available on your browser whereas choosing jpeg will work natively on Mozilla and related browsers and with a Java applet on Internet Explorer.

WEB_H_DEFAULT_SCALE, WEB_M_DEFAULT_SCALE, WEB_L_DEFAULT_SCALE - Normally ZoneMinder will display 'live' or 'event' streams in their native size. However if you have monitors with large dimensions or a slow link you may prefer to reduce this size, alternatively for small monitors you can enlarge it. This option lets you specify what the default scaling factor will be. It is expressed as a percentage so 100 is normal size, 200 is double size etc.

WEB_H_DEFAULT_RATE, WEB_M_DEFAULT_RATE, WEB_L_DEFAULT_RATE - Normally ZoneMinder will display 'event' streams at their native rate, i.e. as close to real-time as possible. However if you have long events it is often convenient to replay them at a faster rate for review. This option lets you specify what the default replay rate will be. It is expressed as a percentage so 100 is normal rate, 200 is double speed etc.

WEB_H_VIDEO_BITRATE, WEB_M_VIDEO_BITRATE, WEB_L_VIDEO_BITRATE - When encoding real video via the ffmpeg library a bit rate can be specified which roughly corresponds to the available bandwidth used for the stream. This setting effectively corresponds to a 'quality' setting for the video. A low value will result in a blocky image whereas a high value will produce a clearer view. Note that this setting does not control the frame rate of the video however the quality of the video produced is affected both by this setting and the frame rate that the video is produced at. A higher frame rate at a particular bit rate results in individual frames being at a lower quality.

WEB_H_VIDEO_MAXFPS, WEB_M_VIDEO_MAXFPS, WEB_L_VIDEO_MAXFPS - When using streamed video the main control is the bitrate which determines how much data can be transmitted. However a lower bitrate at high frame rates results in a lower quality image. This option allows you to limit the maximum frame rate to ensure that video quality is maintained. An additional advantage is that encoding video at high frame rates is a processor intensive task when for the most part a very high frame rate offers little perceptible improvement over one that has a more manageable resource requirement. Note, this option is implemented as a cap beyond which binary reduction takes place. So if you have a device capturing at 15fps and set this option to 10fps then the video is not produced at 10fps, but rather at 7.5fps (15 divided by 2) as the final frame rate must be the original divided by a power of 2.

WEB_H_SCALE_THUMBS, WEB_M_SCALE_THUMBS, WEB_L_SCALE_THUMBS - If unset, this option sends the whole image to the browser which resizes it in the window. If set the image is scaled down on the server before sending a reduced size image to the browser to conserve bandwidth at the cost of cpu on the server. Note that ZM can only perform the resizing if the appropriate PHP graphics functionality is installed. This is usually available in the php-gd package.

WEB_H_EVENTS_VIEW, WEB_M_EVENTS_VIEW, WEB_L_EVENTS_VIEW - Stored events can be viewed in either an events list format or in a timeline based one. This option sets the default view that will be used. Choosing one view here does not prevent the other view being used as it will always be selectable from whichever view is currently being used.

WEB_H_SHOW_PROGRESS, WEB_M_SHOW_PROGRESS, WEB_L_SHOW_PROGRESS - When viewing events an event navigation panel and progress bar is shown below the event itself. This allows you to jump to specific points in the event, but can also dynamically update to display the current progress of the event replay itself. This progress is calculated from the actual event duration and is not directly linked to the replay itself, so on limited band-

width connections may be out of step with the replay. This option allows you to turn off the progress display, whilst still keeping the navigation aspect, where bandwidth prevents it functioning effectively.

WEB_H_AJAX_TIMEOUT, WEB_M_AJAX_TIMEOUT, WEB_L_AJAX_TIMEOUT - The newer versions of the live feed and event views use Ajax to request information from the server and populate the views dynamically. This option allows you to specify a timeout if required after which requests are abandoned. A timeout may be necessary if requests would otherwise hang such as on a slow connection. This would tend to consume a lot of browser memory and make the interface unresponsive. Ordinarily no requests should timeout so this setting should be set to a value greater than the slowest expected response. This value is in milliseconds but if set to zero then no timeout will be used.

2.9.16 Options - Users

Username	Language	Enabled	Stream	Events	Control	Monitors	Groups	System	Bandwidth	Monitor	API Enabled	Mark
admin*	default	Yes	View	Edit	Edit	Edit	Edit	Edit			Yes	<input type="checkbox"/>
home	default	Yes	View	View	None	View	None	View		DoorBell, DeckCamera, Driveway, FrontLawn	Yes	<input type="checkbox"/>
zmes	default	Yes	None	View	None	None	None	None			No	<input type="checkbox"/>

[ADD NEW USER](#)
[DELETE](#)

In this section you will see a list of the current users defined on the system. You can also add or delete users from here. It is recommended you do not delete the admin user unless you have created another fully privileged user to take over the same role. Each user is defined with a name and password (which is hidden) as well as an enabled setting which you can use to temporarily enable or disable users, for example a guest user for limited time access. As well as that there is a language setting that allows you to define user specific languages. Setting a language here that is different than the system language will mean that when that user logs in they will have the web interface presented in their own language rather than the system default, if it is available.

This screen allows you to configure various permissions on a per user basis. The permissions as of today are defined as follows:

- Streams - None: the user has no access to view live streams from the defined monitors - View: the user has access to only view live streams from the defined monitors - Edit: the user has access to edit live streams from the defined monitors
- Events - These permissions relate to the ability to view events from the defined monitors. The permission levels are the same as the Streams permissions, except that they apply to recorded events
- Control - These permissions relate to the ability to control Pan/Tilt/Zoom (PTZ) of the defined monitors. The permission levels are the same as the Streams permissions, except that they apply to PTZ
- Monitors - specifies whether a user can see the current monitor settings and change them. The permissions levels are the same as the Streams permissions, except that they apply to monitor settings
- Groups - specifies whether a user can see monitor groups and change them. The permissions levels are the same as the Streams permissions, except that they apply to groups
- System - Determines whether a user can view or modify the system settings as a whole, such as options and users or controlling the running of the system as a whole. The permissions levels are the same as the Streams permissions, except that they apply to groups.

Note: if you are using zmNinja, users are required to have 'View' access to system because multi-server information is only available as part of this permission

- Bandwidth - Specifies the maximum bandwidth that this user can configure (Low, Medium or High)

- API enabled - Specifies if the ZoneMinder API is enabled for this user (needs to be on, if you are using a mobile app such as zmNinja)

Finally, you can specify a list of monitors this user is allowed to access using the 'Restricted Monitors' list. You can select multiple monitors by shift+click(or command+click) on multiple monitors. If a user with 'Monitors' edit privileges is limited to specific monitors here they will not be able to add or delete monitors but only change the details of those they have access to. If a user has 'System' privileges then the 'Monitors Ids' setting is ignored and has no effect.

Here is an example of a restricted user, for example:

User - home

Username	<input type="text" value="home"/>
New Password	<input type="password"/>
Confirm Password	<input type="password"/>
Language	<input type="button" value=""/>
Enabled	<input type="button" value="Yes"/>
Stream	<input type="button" value="View"/>
Events	<input type="button" value="View"/>
Control	<input type="button" value="None"/>
Monitors	<input type="button" value="View"/>
Groups	<input type="button" value="None"/>
System	<input type="button" value="View"/>
Max Bandwidth	<input type="button" value=""/>
Restricted Monitors	<div><div>Garage</div><div>DoorBell</div><div>Basement</div><div>DeckCamera</div></div>
API Enabled	<input type="button" value="Yes"/>

This user “home” is enabled, can view live streams and events, but only from “DoorBell” and “DeckCamera”. This user also cannot control PTZ.

2.10 Camera Control

ZoneMinder provides the facility to control cameras from the web interface and to some extent automatically. Pan/Tilt/Zoom (PTZ) cameras have a wide range of capabilities and use a large number of different protocols making any kind of generic control solution potentially very difficult. To address this ZoneMinder uses two key approaches to get around this problem.

Definition of Capabilities For each camera model you use, an entry in the camera capabilities table must be created. These indicate what functions the camera supports and ensure that the interface presents only those capabilities that the camera supports. There are a very large number of capabilities that may be supported and it is very important that the entries in this table reflect the actual abilities of the camera. A small number of example capabilities are included in ZoneMinder, these can be used ‘as is’ or modified.

Control Scripts ZoneMinder itself does not generally provide the ability to send commands to cameras or receive responses. What it does is mediate motion requests from the web interface into a standard set of commands which are passed to a script defined in the control capability. Example scripts are provided in ZoneMinder which support a number of serial or network protocols but it is likely that for many cameras new scripts will have to be created. These can be modelled on the example ones, or if control commands already exist from other applications, then the script can just act as a ‘glue’ layer between ZoneMinder and those commands.

It should be emphasised that the control and capability elements of ZoneMinder are not intended to be able to support every camera out of the box. Some degree of development is likely to be required for many cameras.

2.11 Controlling Monitors

If you have defined your system as having controllable monitors and you are looking at a monitor that is configured for control, then clicking on the ‘Control’ link along the top of the window will change the short event listing area to a control area. The capabilities you have defined earlier determine exactly what is displayed in this window. Generally you will have a Pan/Tilt control area along with one or subsidiary areas such as zoom or focus control to the side. If you have preset support then these will be near the bottom of the window. The normal method of controlling the monitor is by clicking on the appropriate graphics which then send a command via the control script to the camera itself. This may sometimes take a noticeable delay before the camera responds.

It is usually the case that the control arrows are sensitive to where you click on them. If you have a camera that allows different speeds to be used for panning or zooming etc then clicking near the point of the arrow will invoke the faster speed whilst clicking near the base of the arrow will be slower. If you have defined continuous motion then ongoing activities can be stopped by clicking on the area between the arrows, which will either be a graphic in the case of pan/tilt controls or a word in the case of zoom and focus controls etc.

Certain control capabilities such as mapped motion allow direct control by clicking on the image itself when used in browsers which support streamed images directly. Used in this way you can just click on the area of the image that interests you and the camera will centre on that spot. You can also use direct image control for relative motion when the area of the image you click on defines the direction and the distance away from the centre of the image determines the speed. As it is not always very easy to estimate direction near the centre of the image, the active area does not start until a short distance away from the centre, resulting in a ‘dead’ zone in the middle of the image.

2.11.1 Control Flow

Having a basic understanding of how camera control works in ZoneMinder will go a long way in debugging issues in the future. It is important to note that many of the ‘camera control’ scripts are user contributed and it is entirely possible that they break in a future version upgrade.

- ZoneMinder relies on ‘control protocols’ for specific camera models. These ‘control’ protocols are nothing but perl packages located in `/usr/share/perl5/ZoneMinder/Control/` (in Ubuntu distributions) that are invoked by ZoneMinder when you invoke a PTZ operation
- When you associate a ‘protocol’ for PTZ for a camera, you are effectively letting ZoneMinder know where to locate the perl file that will eventually control the camera movement
- Let’s for example, assume that you are configuring a Foscam 9831W camera and have associated the ‘9831w’ protocol to that camera. This basically means when you move the camera via ZoneMinder, it will pass on the movements to `FI9831w.pm` in `/usr/share/perl5/ZoneMinder/Control/`
- ZoneMinder also maintains protocol configuration parameters in a table called `Controls` in the DB. This table is used to store parameters like whether the camera supports continuous move, zoom etc.
- The `Controls` table is used by ZoneMinder to build its PTZ web interface. For example, an FI9831W camera does not support Zoom -> so when you open the PTZ interface of ZoneMinder via the Web Console and navigate to the FI9831W camera, the Zoom option will not be shown. It knows not to show this because the `Control` table entry for FI9831W specifies it does not support Zoom. Note that you edit these parameters via Source->Control->Control Type->Edit in the web console
- If you ever look at any of the control protocol files, you will notice it has functions like `moveRelUp` or `moveConLeft` etc. -> these are the functions that eventually get invoked to move the camera around and it is expected that contributors who implement missing camera profiles fill in these functions with the appropriate camera specific commands. This way, the core ZoneMinder code does not need to worry about camera specific commands. All it needs to know is the features of a camera and accordingly invoke abstract commands in the protocol perl file and it is the responsibility of the perl file for that camera to implement the specifics. So, if you are facing problems with PTZ not working, these protocol files are what you should be debugging.

2.11.2 Control Capabilities

If you have a camera that supports PTZ controls and wish to use it with ZoneMinder then the first thing you need to do is ensure that it has an accurate entry in the capabilities table. To do this you need to go to the Control tab of the Monitor configuration dialog and select ‘Edit’ where it is listed by the Control Type selection box. This will bring up a new window which lists, with a brief summary, the existing capabilities. To edit an existing capability to modify select the Id or Name of the capability in question, or click on the Add button to add a new control capability. Either of these approaches will create a new window, in familiar style, with tabs along the top and forms fields below. In the case of the capabilities table there are a large number of settings and tabs, the mean and use of these are briefly explained below.

Main Tab

Name This is the name of the control capability, it will usually make sense to name capabilities after the camera model or protocol being used.

Type Whether the capability uses a local (usually serial) or network control protocol.

Command This is the full path to a script or application that will map the standard set of ZoneMinder control commands to equivalent control protocol command. This may be one of the shipped example `zmcontrol-*.pl` scripts or something else entirely.

Can Wake This is the first of the actual capability definitions. Checking this box indicates that a protocol command exists to wake up the camera from a sleeping state.

Can Sleep The camera can be put to sleep.

Can Reset The camera can be reset to a previously defined state.

Move Tab

Can Move The camera is able move, i.e. pan or tilt.

Can Move Diagonally The camera can move diagonally. Some devices can move only vertically or horizontally at a time.

Can Move Mapped The camera is able internally map a point on an image to a precise degree of motion to centre that point in the image.

Can Move Absolute The camera can move to an absolute location.

Can Move Relative The camera can move to a relative location, e.g. 7 point left or up.

Can Move Continuous The camera can move continuously in a defined direction until told to stop or the movement limits are reached, e.g. left.

Pan Tab

Can Pan The camera can pan, or move horizontally.

Min/Max Pan Range If the camera supports absolute motion this is the minimum and maximum pan co-ordinates that may be specified, e.g. -100 to 100.

Min/Max Pan Step If the camera supports relative motion, this is the minimum and maximum amount of movement that can be specified.

Has Pan Speed The camera supports specification of pan speeds.

Min/Max Pan Speed The minimum and maximum pan speed supported.

Has Turbo Pan The camera supports an additional turbo pan speed.

Turbo Pan Speed The actual turbo pan speed.

Tilt Tab

Definition of Tilt capabilities, fields as for 'Pan' tab.

Zoom Tab

Can Zoom The camera can zoom.

Can Zoom Absolute The camera can zoom to an absolute position.

Can Zoom Relative The camera can zoom to a relative position.

Can Zoom Continuous The camera can zoom continuously in or out until told to stop or the zoom limits are reached.

Min/Max Zoom Range If the camera supports absolute zoom this is the minimum and maximum zoom amounts that may be specified.

Min/Man Zoom Step If the camera supports relative zoom, this is the minimum and maximum amount of zoom change that can be specified.

Has Zoom Speed The camera supports specification of zoom speed.

Min/Max Zoom Speed The minimum and maximum zoom speed supported.

Focus Tab

Definition of Focus capabilities, fields as for 'Zoom' tab, but with the following additional capability.

Can Auto Focus The camera can focus automatically.

White Tab

Definition of White Balance capabilities, fields as for 'Focus' tab.

Iris Tab

Definition of Iris Control capabilities, fields as for 'Focus' tab.

Presets Tab

Has Presets The camera supports preset positions.

Num Presets How many presets the camera supports. If the camera supports a huge number of presets then it makes sense to specify a more reasonable number here, 20 or less is recommended.

Has Home Preset The camera has a defined 'home' position, usually in the mid point of its range.

Can Set Presets The camera supports setting preset locations via its control protocol.

2.11.3 Control Scripts

The second key element to controlling cameras with ZoneMinder is ensuring that an appropriate control script or application is present. A small number of sample scripts are included with ZoneMinder and can be used directly or as the basis for development. Control scripts are run atomically, that is to say that one requested action from the web interface results in one execution of the script and no state information is maintained. If your protocol requires state information to be preserved then you should ensure that your scripts do this as ZoneMinder has no concept of the state of the camera in control terms.

If you are writing a new control script then you need to ensure that it supports the parameters that ZoneMinder will pass to it. If you already have scripts or applications that control your cameras, the ZoneMinder control script will just act as glue to convert the parameters passed into a form that your existing application understands. If you are writing a script to support a new protocol then you will need to convert the parameters passed into the script to equivalent protocol commands. If you have carefully defined your control capabilities above then you should only expect commands that correspond to those capabilities.

The standard set of parameters passed to control scripts is defined below,

—device=<device> : This is the control device from the monitor definition. Absent if no device is specified.

—address=<address> : This is the control address from the monitor definition. This will usually be a hostname or ip address for network cameras or a simple numeric camera id for other cameras.

`-autostop=<timeout>` : This indicates whether an automatic timeout should be applied to “stop” the given command. It will only be included for “continuous” commands, as listed below, and will be a timeout in decimal seconds, probably fractional.

`—command=<command>` : This specifies the command that the script should execute. Valid commands are given below.

`-xcoord=<x>, -ycoord=<y>` : This specifies the x and/or y coordinates for commands which require them. These will normally be absolute or mapped commands.

`—width=<width>’, ‘-height=<height>` : This specifies the width and height of the current image, for mapped motion commands where the coordinates values passed must have a context.

`-speed=<speed>` : This specifies the speed that the command should use, if appropriate.

`—panspeed=<speed>’, ‘-tiltspeed=<speed>` : This indicates the specific pan and tilt speeds for diagonal movements which may allow a different motion rate for horizontal and vertical components.

`-step=<step>` : This specifies the amount of motion that the command should use, if appropriate. Normally used for relative commands only.

`—panstep=<step>’, ‘-tiltstep=<step>` : This indicates the specific pan and tilt steps for diagonal movements which may allow a different amount of motion for horizontal and vertical components.

`-preset=<preset>` : This specifies the particular preset that relevant commands should operate on.

The *command* option listed above may take one of the following commands as a parameter.

wake Wake the camera.

sleep Send the camera to sleep.

reset Reset the camera.

move_map Move mapped to a specified location on the image.

move_pseudo_map As *move_map* above. Pseudo-mapped motion can be used when mapped motion is not supported but relative motion is in which case mapped motion can be roughly approximated by careful calibration.

move_abs_<direction> Move to a specified absolute location. The direction element gives a hint to the direction to go but can be omitted. If present it will be one of “up”, “down”, “left”, “right”, “upleft”, “upright”, “downleft” or “downright”.

move_rel_<direction> Move a specified amount in the given direction.

move_con_<direction> Move continuously in the given direction until told to stop.

move_stop Stop any motion which may be in progress.

zoom_abs_<direction> Zoom to a specified absolute zoom position. The direction element gives a hint to the direction to go but can be omitted. If present it will be one of “tele” or “wide”.

zoom_rel_<direction> Zoom a specified amount in the given direction.

zoom_con_<direction> Zoom continuously in the given direction until told to stop.

zoom_stop Stop any zooming which may be in progress.

focus_auto Set focusing to be automatic.

focus_man Set focusing to be manual.

focus_abs_<direction> Focus to a specified absolute focus position. The direction element gives a hint to the direction to go but can be omitted. If present it will be one of “near” or “far”.

focus_rel_<direction> Focus a specified amount in the given direction.

focus_con_<direction> Focus continuously in the given direction until told to stop.

focus_stop Stop any focusing which may be in progress.

white_<subcommand> As per the focus commands, except that direction may be “in” or “out”.

iris_<subcommand> As per the focus commands, except that direction may be “open” or “close”.

preset_set Set the given preset to the current location.

preset_goto Move to the given preset.

preset_home Move to the “home” preset.

2.12 Mobile Devices

Here are some options for using ZoneMinder on Mobile devices:

2.12.1 Third party mobile clients

- **zmNinja** ([source code](#), **needs APIs to be installed to work**)
 - Available in App Store, Play Store and for Desktops - [website](#)

2.12.2 Using the existing web console

- You can directly use the ZoneMinder interface by launching a browser and going to the ZoneMinder server just like you do on the Desktop

2.12.3 Discontinued clients

The following are a list of clients that do not work and have not been updated:

- eyeZM
- zmView

2.13 Logging

Note: Understanding how logging works in ZoneMinder is key to being able to isolate/pinpoint issues well. Please refer to [Options - Logging](#) to read about how to customize logging.

Most components of ZoneMinder can emit informational, warning, error and debug messages in a standard format. These messages can be logged in one or more locations. By default all messages produced by scripts are logged in `<script name>.log` files which are placed in the directory defined by the `ZM_PATH_LOGS` configuration variable. This is initially defined as `/var/log/zm` (on debian based systems) though it can be overridden to a custom path (the path is usually defined in `/etc/zm/conf.d/01-system-paths.conf`, but to override it, you should create your own config file, not overwrite this file). So for example, the `zmdc.pl` script will output messages to `/var/log/zmdc.log`, an example of these messages is:


```
10/24/2019 08:01:19.291513 zmdc[6414].INF [ZMServer:408] [Starting pending process, ↵
↵zma -m 2]
10/24/2019 08:01:19.296575 zmdc[6414].INF [ZMServer:408] ['zma -m 2' starting at 19/
↵10/24 08:01:19, pid = 15740]
10/24/2019 08:01:19.296927 zmdc[15740].INF [ZMServer:408] ['zma -m 2' started at 19/
↵10/24 08:01:19]
```

where the first part refers to the date and time of the entry, the next section is the name (or an abbreviated version) of the script, followed by the process id in square brackets, a severity code (INF, WAR, ERR or DBG) and the debug text. If you change the location of the log directory, ensure it refers to an existing directory which the web user has permissions to write to. Also ensure that no logs are present in that directory the web user does not have permission to open. This can happen if you run commands or scripts as the root user for testing at some point. If this occurs then subsequent non-privileged runs will fail due to being unable to open the log files.

As well as specific script logging above, information, warning and error messages are logged via the system syslog service. This is a standard component on Linux systems and allows logging of all sorts of messages in a standard way and using a standard format. On most systems, unless otherwise configured, messages produced by ZoneMinder will go to the `/var/log/messages` or `/var/log/syslog` file. On some distributions they may end up in another file, but usually still in `/var/log`. Messages in this file are similar to those in the script log files but differ slightly. For example the above event in the system log file looks like:

```
Jan  3 13:46:00 shuttle52 zmpkg[11148]: INF [Command: start]
```

where you can see that the date is formatted differently (and only to 1 second precision) and there is an additional field for the hostname (as syslog can operate over a network). As well as ZoneMinder entries in this file you may also see entries from various other system components. You should ensure that your syslogd daemon is running for syslog messages to be correctly handled.

2.13.1 Customizing logging properly in ZoneMinder

2.13.2 Other Notes

A number of users have asked how to suppress or redirect ZoneMinder messages that are written to this file. This most often occurs due to not wanting other system messages to be overwhelmed and obscured by the ZoneMinder produced ones (which can be quite frequent by default). In order to control syslog messages you need to locate and edit the `syslog.conf` file on your system. This will often be in the `/etc` directory. This file allows configuration of syslog so that certain classes and categories of messages are routed to different files or highlighted to a console, or just ignored. Full details of the format of this file is outside the scope of this document (typing `man syslog.conf` will give you more information) but the most often requested changes are easy to implement.

The syslog service uses the concept of priorities and facilities where the former refers to the importance of the message and the latter refers to that part of the system from which it originated. Standard priorities include 'info', 'warning', 'err' and 'debug' and ZoneMinder uses these priorities when generating the corresponding class of message. Standard facilities include 'mail', 'cron' and 'security' etc but as well this, there are eight 'local' facilities that can be used by machine specific message generators. ZoneMinder produces its messages via the 'local1' facility.

So armed with the knowledge of the priority and facility of a message, the `syslog.conf` file can be amended to handle messages however you like.

So to ensure that all ZoneMinder messages go to a specific log file you can add the following line near the top of your `syslog.conf` file:

```
# Save ZoneMinder messages to zm.log
local1.* /var/log/zm/zm.log
```

which will ensure that all messages produced with the local1 facility are routed to the `/var/log/zm/zm.log` file. However this does not necessarily prevent them also going into the standard system log. To do this you will need to modify the line that determines which messages are logged to this file. This may look something like:

```
# Log anything (except mail) of level info or higher.
# Don't log private authentication messages!
*.info;mail.none;news.none;authpriv.none;cron.none    /var/log/messages
```

by default. To remove ZoneMinder messages altogether from this file you can modify this line to look like:

```
*.info;local1.!*;mail.none;news.none;authpriv.none;cron.none    /var/log/messages
```

which instructs syslog to ignore any messages from the local1 facility. If however you still want warnings and errors to occur in the system log file, you could change it to:

```
*.info;local1.!*;local1.warning;mail.none;news.none;authpriv.none;cron.none    /var/
↪ log/messages
```

which follows the ignore instruction with a further one to indicate that any messages with a facility of local1 and a priority of warning or above should still go into the file.

These recipes are just examples of how you can modify the logging to suit your system, there are a lot of other modifications you could make. If you do make any changes to `syslog.conf` you should ensure you restart the syslogd process or send it a HUP signal to force it to reread its configuration file otherwise your changes will be ignored.

The discussion of logging above began by describing how scripts produce error and debug messages. The way that the binaries work is slightly different. Binaries generate information, warning and error messages using syslog in exactly the same way as scripts and these messages will be handled identically. However debug output is somewhat different. For the scripts, if you want to enable debug you will need to edit the script file itself and change the `DBG_LEVEL` constant to have a value of 1. This will then cause debug messages to be written to the `<script>.log` file as well as the more important messages. Debug messages however are not routed via syslog. Scripts currently only have one level of debug so this will cause any and all debug messages to be generated. Binaries work slightly differently and while you can edit the call to `zmDbgInit` that is present in every binary's 'main' function to update the initial value of the debug level, there are easier ways.

The simplest way of collecting debug output is to click on the Options link from the main ZoneMinder console view and then go to the Debug tab. There you will find a number of debug options. The first thing you should do is ensure that the `ZM_EXTRA_DEBUG` setting is switched on. This enables debug generally. The next thing you need to do is select the debug target, level and destination file using the relevant options. Click on the '?' by each option for more information about valid settings. You will need to restart ZoneMinder as a whole or at least the component in question for logging to take effect. When you have finished debugging you should ensure you switch debug off by unchecking the `ZM_EXTRA_DEBUG` option and restarting ZoneMinder. You can leave the other options as you like as they are ignored if the master debug option is off.

Once you have debug being logged you can modify the level by sending `USR1` and `USR2` signals to the relevant binary (or binaries) to increase or decrease the level of debug being emitted with immediate effect. This modification will not persist if the binary gets restarted however.

If you wish to run a binary directly from the command line to test specific functionality or scenarios, you can set the `ZM_DBG_LEVEL` and `ZM_DBG_LOG` environment variables to set the level and log file of the debug you wish to see, and the `ZM_DBG_PRINT` environment variable to 1 to output the debug directly to your terminal.

All ZoneMinder logs can now be rotated by logrotate. A sample logrotate config file is shown below:

```
/var/log/zm/*.log {
    missingok
    notifempty
    sharedscripts
```

(continues on next page)

(continued from previous page)

```

postrotate
    /usr/local/bin/zmpkg.pl logrot 2> /dev/null > /dev/null || true
endscript
}

```

2.14 Configuration Files

This section describes configuration files that ZoneMinder uses beyond the various Web UI options.

2.14.1 System Path Configurations

At one point of time, ZoneMinder stored various system path configurations under the Web UI (Options->Paths). This was removed a few versions ago and now resides in a configuration file. The motivation for this change can be read in [this discussion](#).

Typically, path configurations now reside in `/etc/zm`.

Here is an example of the file hierarchy:

```

/etc/zm
├── conf.d
│   ├── 01-system-paths.conf
│   ├── 02-multiserver.conf
│   ├── 03-custom.conf #optional
│   └── README
├── objectconfig.ini # optional
├── zm.conf
└── zmeventnotification.ini #optional

```

The roles of the files are as follows:

- `zm.conf` contains various base configuration entries. You should not edit this file as it may be overwritten on an upgrade.
- `zmeventnotification.ini` is only present if you have installed the ZoneMinder Event Notification Server.
- `objectconfig.ini` is only present if you have installed the machine learning hooks for the Event Notification Server.
- `conf.d` contains additional configuration items as follows:
 - `01-system-paths.conf` contains all the paths that were once part of Options->Paths in the Web UI. You should not edit this file as it may be overwritten on an upgrade
 - `02-multiserver.conf` file consists of custom variables if you are deploying ZoneMinder in a multi-server configuration (see [Multi-Server Install](#))
 - `03-custom.conf` is an custom config file that I created to override specific variables in the path files. **This is the recommended way to customize entries.** Anything that you want to change should be in a new file inside `conf.d`. Note that ZoneMinder will sort all the files alphabetically and run their contents in ascending order. So it doesn't really matter what you name them, as long as you make sure your changes are not overwritten by another file in the sorting sequence. It is therefore good practice to prefix your file names by `nn-` where `nn` is a monotonically increasing numerical sequence `01-` `02-` `03-` and so forth, so you know the order they will be processed.

2.14.2 Timezone Configuration

Earlier versions of ZoneMinder relied on `php.ini` to set Date/Time Zone. This is no longer the case. You can (and must) set the Timezone via the Web UI, starting ZoneMinder version 1.34. See [here](#).

2.14.3 Database Specific Configuration

Todo: do we really need to have this section? Not sure if its generic and not specific to ZM

While the ZoneMinder specific database config entries reside in `/etc/zm/zm.conf` and related customizations discussed above, general database configuration items can be tweaked in `/etc/mysql` (or whichever path your DB server is installed)

This document will provide an overview of ZoneMinder's API.

3.1 Overview

In an effort to further 'open up' ZoneMinder, an API was needed. This will allow quick integration with and development of ZoneMinder.

The API is built in CakePHP and lives under the `/api` directory. It provides a RESTful service and supports CRUD (create, retrieve, update, delete) functions for Monitors, Events, Frames, Zones and Config.

3.2 API Wrappers

- `pym` is a python wrapper for the ZoneMinder APIs. It supports both the legacy and new token based API, as well as ZM logs/ZM shared memory support. See [its project site](#) for more details. Documentation is [here](#).

3.3 API evolution

The ZoneMinder API has evolved over time. Broadly speaking the iterations were as follows:

- Prior to version 1.29, there really was no API layer. Users had to use the same URLs that the web console used to 'mimic' operations, or use an XML skin
- Starting version 1.29, a v1.0 CakePHP based API was released which continues to evolve over time. From a security perspective, it still tied into ZM auth and required client cookies for many operations. Primarily, two authentication modes were offered:
 - You use cookies to maintain session state (`ZM_SESS_ID`)
 - You use an authentication hash to validate yourself, which included encoding personal information and time stamps which at times caused timing validation issues, especially for mobile consumers

- Starting version 1.34, ZoneMinder has introduced a new “token” based system which is based JWT. We have given it a ‘2.0’ version ID. These tokens don’t encode any personal data and can be statelessly passed around per request. It introduces concepts like access tokens, refresh tokens and per user level API revocation to manage security better. The internal components of ZoneMinder all support this new scheme now and if you are using the APIs we strongly recommend you migrate to 1.34 and use this new token system (as a side note, 1.34 also moves from MYSQL PASSWORD to Bcrypt for passwords, which is also a good reason why you should migrate).
- Note that as of 1.34, both versions of API access will work (tokens and the older auth hash mechanism), however we no longer use sessions by default. You will have to add a `stateful=1` query parameter during login to tell ZM to set a COOKIE and store the required info in the session. This option is only available if `OPT_USE_LEGACY_API_AUTH` is set to ON.

Note: For the rest of the document, we will specifically highlight v2.0 only features. If you don’t see a special mention, assume it applies for both API versions.

3.4 Enabling API

ZoneMinder comes with APIs enabled. To check if APIs are enabled, visit `Options->System`. If `OPT_USE_API` is enabled, your APIs are active. For v2.0 APIs, you have an additional option right below it:

- `OPT_USE_LEGACY_API_AUTH` which is enabled by default. When enabled, the `login.json` API (discussed later) will return both the old style (`auth=`) and new style (`token=`) credentials. The reason this is enabled by default is because any existing apps that use the API would break if they were not updated to use v2.0. (Note that `zmNinja` 1.3.057 and beyond will support tokens)

3.5 Enabling secret key

- It is **important** that you create a “Secret Key”. This needs to be a set of hard to guess characters, that only you know. ZoneMinder does not create a key for you. It is your responsibility to create it. If you haven’t created one already, please do so by going to `Options->Systems` and populating `AUTH_HASH_SECRET`. Don’t forget to save.
- If you plan on using V2.0 token based security, **it is mandatory to populate this secret key**, as it is used to sign the token. If you don’t, token authentication will fail. V1.0 did not mandate this requirement.

3.6 Getting an API key

To get an API key:

```
curl -XPOST -d "user=yourusername&pass=yourpassword" https://yourserver/zm/api/host/  
login.json
```

If you want to use a stateful connection, so you don’t have to pass auth credentials with each query, you can use the following:

```
curl -XPOST -c cookies.txt -d "user=yourusername&pass=yourpassword&stateful=1" https://  
yourserver/zm/api/host/login.json
```

This returns a payload like this for API v1.0:

```
{
  "credentials": "auth=05f3a50e8f7<deleted>063",
  "append_password": 0,
  "version": "1.33.9",
  "apiversion": "1.0"
}
```

Or for API 2.0:

```
{
  "access_token": "eyJ0eXAiOiJK<deleted>HE",
  "access_token_expires": 3600,
  "refresh_token": "eyJ0eXAiOi<deleted>mPs",
  "refresh_token_expires": 86400,
  "credentials": "auth=05f3a50e8f7<deleted>063", # only if OPT_USE_LEGACY_API_AUTH_
  ↪ is enabled
  "append_password": 0, # only if OPT_USE_LEGACY_API_AUTH is enabled
  "version": "1.33.9",
  "apiversion": "2.0"
}
```

3.7 Using these keys with subsequent requests

Once you have the keys (a.k.a credentials (v1.0, v2.0) or token (v2.0)) you should now supply that key to subsequent API calls like this:

```
# v1.0 or 2.0 based API access (will only work if AUTH_HASH_LOGINS is enabled)

# RECOMMENDED: v2.0 token based
curl -XGET https://yourserver/zm/api/monitors.json?token=<access_token>

# or, for legacy mode:
curl -XGET https://yourserver/zm/api/monitors.json?auth=<hex digits from 'credentials'
  ↪ '>

# or, if you specified -c cookies.txt in the original login request
curl -b cookies.txt -XGET https://yourserver/zm/api/monitors.json
```

Note: If you are using an HTTP GET request, the token/auth needs to be passed as a query parameter in the URL. If you are using an HTTP POST (like when you use the API to modify a monitor, for example), you can choose to pass the token as a data payload instead. The API layer discards data payloads for HTTP GET. Finally, If you don't pass keys, you could also use cookies (not recommended as a general approach).

3.8 Key lifetime (v1.0)

If you are using the old credentials mechanism present in v1.0, then the credentials will time out based on PHP session timeout (if you are using cookies), or the value of AUTH_HASH_TTL (if you are using auth= and have enabled AUTH_HASH_LOGINS) which defaults to 2 hours. Note that there is no way to look at the hash and decipher how

much time is remaining. So it is your responsibility to record the time you got the hash and assume it was generated at the time you got it and re-login before that time expires.

3.9 Key lifetime (v2.0)

In version 2.0, it is easy to know when a key will expire before you use it. You can find that out from the `access_token_expires` and `refresh_token_expires` values (in seconds) after you decode the JWT key (there are JWT decode libraries for every language you want). You should refresh the keys before the timeout occurs, or you will not be able to use the APIs.

3.10 Understanding access/refresh tokens (v2.0)

If you are using V2.0, then you need to know how to use these tokens effectively:

- Access tokens are short lived. ZoneMinder issues access tokens that live for 3600 seconds (1 hour).
- Access tokens should be used for all subsequent API accesses.
- Refresh tokens should ONLY be used to generate new access tokens. For example, if an access token lives for 1 hour, before the hour completes, invoke the `login.json` API above with the refresh token to get a new access token. ZoneMinder issues refresh tokens that live for 24 hours.
- To generate a new refresh token before 24 hours are up, you will need to pass your user login and password to `login.json`

To Summarize:

- Pass your username and password to `login.json` only once in 24 hours to renew your tokens
- Pass your “refresh token” to `login.json` once in two hours (or whatever you have set the value of `AUTH_HASH_TTL` to) to renew your access token
- Use your `access token` for all API invocations.

In fact, V2.0 will reject your request (if it is not to `login.json`) if it comes with a refresh token instead of an access token to discourage usage of this token when it should not be used.

This minimizes the amount of sensitive data that is sent over the wire and the lifetime durations are made so that if they get compromised, you can regenerate or invalidate them (more on this later)

3.11 Understanding key security

- Version 1.0 uses an MD5 hash to generate the credentials. The hash is computed over your secret key (if available), username, password and some time parameters (along with remote IP if enabled). This is not a secure/recommended hashing mechanism. If your auth hash is compromised, an attacker will be able to use your hash till it expires. To avoid this, you could disable the user in ZoneMinder. Furthermore, enabling remote IP (`AUTH_HASH_REMOTE_IP`) requires that you issue future requests from the same IP that generated the tokens. While this may be considered an additional layer for security, this can cause issues with mobile devices.
- Version 2.0 uses a different approach. The hash is a simple base64 encoded form of “claims”, but signed with your secret key. Consider for example, the following access key:

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.
eyJpc3MiOiJab251TWluZGVyIiwiaWF0IjoxNTUzOTQwNzUyLCJleHAiOiJlNTc5NDQzNTIsInVzZXIiOiJhZG1pbiIsInR5cGU
5V0cpw3cFHiSTN5zfGDSrrPyVya1M8_2Anh5u6eN1I
```


If you were to use any [JWT token verifier](#) it can easily decode that token and will show:

```
{
  "iss": "ZoneMinder",
  "iat": 1557940752,
  "exp": 1557944352,
  "user": "admin",
  "type": "access"
}
Invalid Signature
```

Don't be surprised. JWT tokens, by default, are **not meant to be encrypted**. It is just an assertion of a claim. It states that the issuer of this token was ZoneMinder, It was issued at (iat) Wednesday, 2019-05-15 17:19:12 UTC and will expire on (exp) Wednesday, 2019-05-15 18:19:12 UTC. This token claims to be owned by an admin and is an access token. If your token were to be stolen, this information is available to the person who stole it. Note that there are no sensitive details like passwords in this claim.

However, that person will **not** have your secret key as part of this token and therefore, will NOT be able to create a new JWT token to get, say, a refresh token. They will however, be able to use your access token to access resources just like the auth hash above, till the access token expires (2 hrs). To revoke this token, you don't need to disable the user. Go to **Options->API** and tap on "Revoke All Access Tokens". This will invalidate the token immediately (this option will invalidate all tokens for all users, and new ones will need to be generated).

Over time, we will provide you with more fine grained access to these options.

Summarizing good practices:

- Use HTTPS, not HTTP
- If possible, use free services like [LetsEncrypt](#) instead of self-signed certificates (sometimes this is not possible)
- Keep your tokens as private as possible, and use them as recommended above
- If you believe your tokens are compromised, revoke them, but also check if your attacker has compromised more than you think (example, they may also have your username/password or access to your system via other exploits, in which case they can regenerate as many tokens/credentials as they want).

Note: Subsequent sections don't explicitly callout the key addition to APIs. We assume that you will append the correct keys as per our explanation above.

3.12 Examples

(In all examples, replace 'server' with IP or hostname & port where ZoneMinder is running)

3.13 API Version

To retrieve the API version:

```
curl http://server/zm/api/host/getVersion.json
```

3.14 Return a list of all monitors

```
curl http://server/zm/api/monitors.json
```

It is worthwhile to note that starting ZM 1.32.3 and beyond, this API also returns a `Monitor_Status` object per monitor. It looks like this:

```
"Monitor_Status": {
  "MonitorId": "2",
  "Status": "Connected",
  "CaptureFPS": "1.67",
  "AnalysisFPS": "1.67",
  "CaptureBandwidth": "52095"
}
```

If you don't see this in your API, you are running an older version of ZM. This gives you a very convenient way to check monitor status without calling the `daemonCheck` API described later.

3.15 Retrieve monitor 1

```
curl http://server/zm/api/monitors/1.json
```

3.16 Change State of Monitor 1

This API changes monitor 1 to Modect and Enabled

```
curl -XPOST http://server/zm/api/monitors/1.json -d "Monitor[Function]=Modect&
↵Monitor[Enabled]=1"
```

3.17 Get Daemon Status of Monitor 1

```
curl http://server/zm/api/monitors/daemonStatus/id:1/daemon:zmc.json
```

3.18 Add a monitor

This command will add a new http monitor.

```
curl -XPOST http://server/zm/api/monitors.json -d "Monitor[Name]=Cliff-Burton\
&Monitor[Function]=Modect\
&Monitor[Protocol]=http\
&Monitor[Method]=simple\
&Monitor[Host]=usr:pass@192.168.11.20\
&Monitor[Port]=80\
&Monitor[Path]=/mjpg/video.mjpg\
&Monitor[Width]=704\
&Monitor[Height]=480\
&Monitor[Colours]=4"
```

3.19 Edit monitor 1

This command will change the 'Name' field of Monitor 1 to 'test1'

```
curl -XPUT http://server/zm/api/monitors/1.json -d "Monitor[Name]=test1"
```

3.20 Delete monitor 1

This command will delete Monitor 1, but will not delete any Events which depend on it.

```
curl -XDELETE http://server/zm/api/monitors/1.json
```

3.21 Arm/Disarm monitors

This command will force an alarm on Monitor 1:

```
curl http://server/zm/api/monitors/alarm/id:1/command:on.json
```

This command will disable the alarm on Monitor 1:

```
curl http://server/zm/api/monitors/alarm/id:1/command:off.json
```

This command will report the status of the alarm Monitor 1:

```
curl http://server/zm/api/monitors/alarm/id:1/command:status.json
```

3.22 Return a list of all events

```
http://server/zm/api/events.json
```

Note that events list can be quite large and this API (as with all other APIs in ZM) uses pagination. Each page returns a specific set of entries. By default this is 25 and ties into WEB_EVENTS_PER_PAGE in the ZM options menu.

So the logic to iterate through all events should be something like this (pseudocode): (unfortunately there is no way to get pageCount without getting the first page)

```
data = http://server/zm/api/events.json?page=1 # this returns the first page
# The json object returned now has a property called data.pagination.pageCount
count = data.pagination.pageCount;
for (i=1, i<count, i++)
{
    data = http://server/zm/api/events.json?page=i;
    doStuff(data);
}
```

3.23 Retrieve event Id 1000

```
curl -XGET http://server/zm/api/events/1000.json
```

3.24 Edit event 1

This command will change the 'Name' field of Event 1 to 'Seek and Destroy'

```
curl -XPUT http://server/zm/api/events/1.json -d "Event[Name]=Seek and Destroy"
```

3.25 Delete event 1

This command will delete Event 1, and any Frames which depend on it.

```
curl -XDELETE http://server/zm/api/events/1.json
```

3.26 Return a list of events for a specific monitor Id =5

```
curl -XGET http://server/zm/api/events/index/MonitorId:5.json
```

Note that the same pagination logic applies if the list is too long

3.27 Return a list of events for a specific monitor within a specific date/time range

```
http://server/zm/api/events/index/MonitorId:5/StartTime >=:2015-05-15 18:43:56/  
↪EndTime <=:2015-05-16 18:43:56.json
```

To try this in CuRL, you need to URL escape the spaces like so:

```
curl -XGET "http://server/zm/api/events/index/MonitorId:5/StartTime%20>=:2015-05-15  
↪%2018:43:56/EndTime%20<=:2015-05-16%2018:43:56.json"
```

3.28 Return a list of events for all monitors within a specified date/time range

```
curl -XGET "http://server/zm/api/events/index/StartTime%20>=:2015-05-15%2018:43:56/  
↪EndTime%20<=:208:43:56.json"
```

3.29 Return event count based on times and conditions

The API also supports a handy mechanism to return a count of events for a period of time.

This returns number of events per monitor that were recorded in the last one hour

```
curl "http://server/zm/api/events/consoleEvents/1%20hour.json"
```

This returns number of events per monitor that were recorded in the last day where there were atleast 10 frames that were alarms”

```
curl "http://server/zm/api/events/consoleEvents/1%20day.json/AlarmFrames >=: 10.json"
```

3.30 Return sorted events

This returns a list of events within a time range and also sorts it by descending order

```
curl -XGET "http://server/zm/api/events/index/StartTime%20>=:2015-05-15%2018:43:56/
↳EndTime%20<=:208:43:56.json?sort=StartTime&direction=desc"
```

3.31 Configuration Apis

The APIs allow you to access all the configuration parameters of ZM that you typically set inside the web console. This returns the full list of configuration parameters:

```
curl -XGET http://server/zm/api/configs.json
```

Each configuration parameter has an Id, Name, Value and other fields. Chances are you are likely only going to focus on these 3.

The edit function of the Configs API is a little quirky at the moment. Its format deviates from the usual edit flow of other APIs. This will be fixed, eventually. For now, to change the “Value” of ZM_X10_HOUSE_CODE from A to B:

```
curl -XPUT http://server/zm/api/configs/edit/ZM_X10_HOUSE_CODE.json -d
↳"Config[Value]=B"
```

To validate changes have been made:

```
curl -XGET http://server/zm/api/configs/view/ZM_X10_HOUSE_CODE.json
```

3.32 Run State Apis

ZM API can be used to start/stop/restart/list states of ZM as well Examples:

```
curl -XGET http://server/zm/api/states.json # returns list of run states
curl -XPOST http://server/zm/api/states/change/restart.json #restarts ZM
curl -XPOST http://server/zm/api/states/change/stop.json #Stops ZM
curl -XPOST http://server/zm/api/states/change/start.json #Starts ZM
```

3.33 Create a Zone

```
curl -XPOST http://server/zm/api/zones.json -d "Zone[Name]=Jason-Newsted\
&Zone[MonitorId]=3\
&Zone[Type]=Active\
&Zone[Units]=Percent\
&Zone[NumCoords]=4\
&Zone[Coords]=0,0 639,0 639,479 0,479\
&Zone[AlarmRGB]=16711680\
&Zone[CheckMethod]=Blobs\
&Zone[MinPixelThreshold]=25\
&Zone[MaxPixelThreshold]=\
&Zone[MinAlarmPixels]=9216\
&Zone[MaxAlarmPixels]=\
&Zone[FilterX]=3\
&Zone[FilterY]=3\
&Zone[MinFilterPixels]=9216\
&Zone[MaxFilterPixels]=230400\
&Zone[MinBlobPixels]=6144\
&Zone[MaxBlobPixels]=\
&Zone[MinBlobs]=1\
&Zone[MaxBlobs]=\
&Zone[OverloadFrames]=0"
```

3.34 PTZ Control Meta-Data APIs

PTZ controls associated with a monitor are stored in the Controls table and not the Monitors table inside ZM. What that means is when you get the details of a Monitor, you will only know if it is controllable (`isControllable:true`) and the control ID. To be able to retrieve PTZ information related to that Control ID, you need to use the controls API

Note that these APIs only retrieve control data related to PTZ. They don't actually move the camera. See the "PTZ on live streams" section to move the camera.

This returns all the control definitions:

```
curl http://server/zm/api/controls.json
```

This returns control definitions for a specific control ID=5

```
curl http://server/zm/api/controls/5.json
```

3.35 Host APIs

ZM APIs have various APIs that help you in determining host (aka ZM) daemon status, load etc. Some examples:

```
curl -XGET http://server/zm/api/host/getLoad.json # returns current load of ZM

# Note that ZM 1.32.3 onwards has the same information in Monitors.json which is more
↳reliable and works for multi-server too.
curl -XGET http://server/zm/api/host/daemonCheck.json # 1 = ZM running 0=not running
```

(continues on next page)

(continued from previous page)

```
# The API below uses "du" to calculate disk space. We no longer recommend you use it,
↪if you have many events. Use the Storage APIs instead, described later
curl -XGET http://server/zm/api/host/getDiskPercent.json # returns in GB (not,
↪percentage), disk usage per monitor (that is, space taken to store various event,
↪related information, images etc. per monitor)
```

3.36 Storage and Server APIs

ZoneMinder introduced many new options that allowed you to configure multiserver/multistorage configurations. While a part of this was available in previous versions, a lot of rework was done as part of ZM 1.31 and 1.32. As part of that work, a lot of new and useful APIs were added. Some of these are part of ZM 1.32 and others will be part of ZM 1.32.3 (of course, if you build from master, you can access them right away, or wait till a stable release is out.

This returns storage data for my single server install. If you are using multi-storage, you'll see many such "Storage" entries, one for each storage defined:

```
curl http://server/zm/api/storage.json
```

Returns:

```
{
  "storage": [
    {
      "Storage": {
        "Id": "0",
        "Path": "\/var\/cache\/zoneminder\/events",
        "Name": "Default",
        "Type": "local",
        "Url": null,
        "DiskSpace": "364705447651",
        "Scheme": "Medium",
        "ServerId": null,
        "DoDelete": true
      }
    }
  ]
}
```

"DiskSpace" is the disk used in bytes. While this doesn't return disk space data as rich as `/host/getDiskPercent`, it is much more efficient.

Similarly,

```
curl http://server/zm/api/servers.json
```

Returns:

```
{
  "servers": [
    {
      "Server": {
        "Id": "1",
        "Name": "server1",
        "Hostname": "server1.mydomain.com",
```

(continues on next page)

(continued from previous page)

```

        "State_Id": null,
        "Status": "Running",
        "CpuLoad": "0.9",
        "TotalMem": "6186237952",
        "FreeMem": "156102656",
        "TotalSwap": "536866816",
        "FreeSwap": "525697024",
        "zmstats": false,
        "zmaudit": false,
        "zmtrigger": false
    }
}
]
}

```

This only works if you have a multiserver setup in place. If you don't it will return an empty array.

3.37 Other APIs

This is not a complete list. ZM supports more parameters/APIs. A good way to dive in is to look at the [API code](#) directly.

3.38 Streaming Interface

Developers working on their application often ask if there is an “API” to receive live streams, or recorded event streams. It is possible to stream both live and recorded streams. This isn't strictly an “API” per-se (that is, it is not integrated into the Cake PHP based API layer discussed here) and also why we've used the term “Interface” instead of an “API”.

3.38.1 Live Streams

What you need to know is that if you want to display “live streams”, ZoneMinder sends you streaming JPEG images (MJPEG) which can easily be rendered in a browser using an `img src` tag.

For example:

```



# or



```

will display a live feed from monitor id 1, scaled down by 50% in quality and resized to 640x480px.

- This assumes `/zm/cgi-bin` is your `CGI_BIN` path. Change it to what is correct in your system
- The “auth” token you see above is required if you use ZoneMinder authentication. To understand how to get the auth token, please read the “Login, Logout & API security” section below.

- The “connkey” parameter is essentially a random number which uniquely identifies a stream. If you don’t specify a connkey, ZM will generate its own. It is recommended to generate a connkey because you can then use it to “control” the stream (pause/resume etc.)
- Instead of dealing with the “auth” token, you can also use `&user=username&pass=password` where “username” and “password” are your ZoneMinder username and password respectively. Note that this is not recommended because you are transmitting them in a URL and even if you use HTTPS, they may show up in web server logs.

PTZ on live streams

PTZ commands are pretty cryptic in ZoneMinder. This is not meant to be an exhaustive guide, but just something to whet your appetite:

Lets assume you have a monitor, with ID=6. Let’s further assume you want to pan it left.

You’d need to send a: POST command to `https://yourserver/zm/index.php` with the following data payload in the command (NOT in the URL)

```
view=request&request=control&id=6&control=moveConLeft&xge=30&yge=30
```

Obviously, if you are using authentication, you need to be logged in for this to work.

Like I said, at this stage, this is only meant to get you started. Explore the ZoneMinder code and use “Inspect source” as you use PTZ commands in the ZoneMinder source code. [control_functions.php](#) is a great place to start.

3.38.2 Pre-recorded (past event) streams

Similar to live playback, if you have chosen to store events in JPEG mode, you can play it back using:

```


# or


```

- This assumes `/zm/cgi-bin` is your CGI_BIN path. Change it to what is correct in your system
- This will playback event 293820, starting from frame 1 as an MJPEG stream
- Like before, you can add more parameters like `scale` etc.
- `auth` and `connkey` have the same meaning as before, and yes, you can replace `auth` by `&user=username&pass=password` as before and the same security concerns cited above apply.

If instead, you have chosen to use the MP4 (Video) storage mode for events, you can directly play back the saved video file:

```
<video src="https://yourserver/zm/index.php?view=view_video&eid=294690&token=eW
↪<deleted>" type="video/mp4"></video>

# or

<video src="https://yourserver/zm/index.php?view=view_video&eid=294690&auth=33
↪<deleted>" type="video/mp4"></video>
```

This above will play back the video recording for event 294690

3.38.3 What other parameters are supported?

The best way to answer this question is to play with ZoneMinder console. Open a browser, play back live or recorded feed, and do an “Inspect Source” to see what parameters are generated. Change and observe.

3.39 Further Reading

As described earlier, treat this document as an “introduction” to the important parts of the API and streaming interfaces. There are several details that haven’t yet been documented. Till they are, here are some resources:

- zmNinja, the open source mobile app for ZoneMinder is 100% based on ZM APIs. Explore its [source code](#) to see how things work.
- Launch up ZM console in a browser, and do an “Inspect source”. See how images are being rendered. Go to the networks tab of the inspect source console and look at network requests that are made when you pause/play/forward streams.
- If you still can’t find an answer, post your question in the [forums](#) (not the github repo).

Todo: needs to be reviewed - some entries may be old/invalid. I've done one round, but icOn needs to review.

This is the FAQ page. Feel free to contribute any FAQs that you think are missing.

Note: It is always a good idea to refer to the [ZoneMinder forums](#) for tips and tricks. While we try and make sure this FAQ is pruned/adjusted to align with the latest stable release, some of the entries may no longer be accurate (or there may be better suggestions in the forums).

4.1 How can I stop ZoneMinder filling up my disk?

Recent versions of ZoneMinder come with a filter you can use for this purpose already included. The filter is called **PurgeWhenFull** and to find it, choose one of the event counts from the console page, for instance events in the last hour, for one of your monitors. **Note** that this filter is automatically enabled if you do a fresh install of ZoneMinder including creating a new database. If you already have an existing database and are upgrading ZoneMinder, it will retain the settings of the filter (which in earlier releases was disabled by default). So you may want to check if PurgeWhenFull is enabled and if not, enable it.

To enable it, go to Web Console, click on any of your Events of any of your monitors. This will bring up an event listing and a filter window.

In the filter window there is a drop down select box labeled 'Use Filter', that lets you select a saved filter. Select 'PurgeWhenFull' and it will load that filter.

Make any modifications you might want, such as the percentage full you want it to kick in, or how many events to delete at a time (it will repeat the filter as many times as needed to clear the space, but will only delete this many events each time to get there).

Then click on 'Save' which will bring up a new window. Make sure the 'Automatically delete' box is checked and press save to save your filter. This will then run in the background to keep your disk within those limits.

After you've done that, your changes will automatically be loaded into `zmfilter` within a few minutes.

Check the `zmfilter.log` file to make sure it is running as sometimes missing perl modules mean that it never runs but people don't always realize.

Purge By Age To delete events that are older than 7 days, create a new filter with "Date" set to "less than" and a value of "-7 days", sort by "date/time" in "asc" ending order, then enable the checkbox "delete all matches". You can also use a value of week or week and days: "-2 week" or "-2 week 4 day"

Save with 'Run Filter In Background' enabled to have it run automatically. Optional skip archived events: click on the plus sign next to -7 days to add another condition. "and" "archive status" equal to "unarchived only".

Optional slow delete: limit the number of results to a number, say 10 in the filter. If you have a large backlog of events that would be deleted, this can hard spike the CPU usage for a long time. Limiting the number of results to only the first three each time the filter is run spreads out the delete processes over time, dramatically lessening the CPU load.

Warning: We no longer recommend use enable `OPT_FAST_DELETE` or `RUN_AUDIT` anymore, unless you are using an old or low powered system to run Zoneminder. Please consider the remaining tips in this answer to be 'generally deprecated, use only if you must'.

There are two methods for ZM to remove files when they are deleted that can be found in Options under the System tab `ZM_OPT_FAST_DELETE` and `ZM_RUN_AUDIT`.

ZM_OPT_FAST_DELETE:

Normally an event created as the result of an alarm consists of entries in one or more database tables plus the various files associated with it. When deleting events in the browser it can take a long time to remove all of this if you are trying to do a lot of events at once. If you are running on an older or under-powered system, you may want to set this option which means that the browser client only deletes the key entries in the events table, which means the events will no longer appear in the listing, and leaves the `zmaudit` daemon to clear up the rest later. If you do so, disk space will not be freed immediately so you will need to run `zmaudit` more frequently. On modern systems, we recommend that you leave this **off**.

ZM_RUN_AUDIT:

The `zmaudit` daemon exists to check that the saved information in the database and on the file system match and are consistent with each other. If an error occurs or if you are using 'fast deletes' it may be that database records are deleted but files remain. In this case, and similar, `zmaudit` will remove redundant information to synchronize the two data stores. This option controls whether `zmaudit` is run in the background and performs these checks and fixes continuously. This is recommended for most systems however if you have a very large number of events the process of scanning the database and file system may take a long time and impact performance. In this case you may prefer to not have `zmaudit` running unconditionally and schedule occasional checks at other, more convenient, times.

ZM_AUDIT_CHECK_INTERVAL:

The `zmaudit` daemon exists to check that the saved information in the database and on the file system match and are consistent with each other. If an error occurs or if you are using 'fast deletes' it may be that database records are deleted but files remain. In this case, and similar, `zmaudit` will remove redundant information to synchronize the two data stores. The default check interval of 900 seconds (15 minutes) is fine for most systems however if you have a very large number of events the process of scanning the database and file system may take a long time and impact performance. In this case you may prefer to make this interval much larger to reduce the impact on your system. This option determines how often these checks are performed.

4.2 Math for Memory: Making sure you have enough memory to handle your cameras

One of the most common issues for erratic ZoneMinder behavior is you don't have enough memory to handle all your cameras. Many users often configure multiple HD cameras at full resolution and 15FPS or more and then face various issues about processes failing, blank screens and other completely erratic behavior. The core reason for all of this is you either don't have enough memory or horsepower to handle all your cameras. The solution often is to reduce FPS, reduce cameras or bump up your server capabilities.

Here are some guidelines with examples on how you can figure out how much memory you need. With respect to CPU, you should benchmark your server using standard unix tools like top, iotop and others to make sure your CPU load is manageable. ZoneMinder also shows average load on the top right corner of the Web Console for easy access.

In *general* a good estimate of memory required would be:

```
Min Bits of Memory = 20% overhead * (image-width*image-height*image buffer_
↪size*target color space*number of cameras)
```

Where:

- image-width and image-height are the width and height of images that your camera is configured for (in my case, 1280x960). This value is in the Source tab for each monitor
- image buffer size is the # of images ZM will keep in memory (this is used by ZM to make sure it has pre and post images before detecting an alarm - very useful because by the time an alarm is detected, the reason for the alarm may move out of view and a buffer is really useful for this, including for analyzing stats/scores). This value is in the buffers tab for each monitor
- target color space is the color depth - 8bit, 24bit or 32bit. It's again in the source tab of each monitor

The 20% overhead on top of the calculation to account for image/stream overheads (this is an estimate)

The math breakdown for 4 cameras running at 1280x960 capture, 50 frame buffer, 24 bit color space:

```
1280*960 = 1,228,800 (bytes)
1,228,800 * (3 bytes for 24 bit) = 3,686,400 (bytes)
3,686,400 * 50 = 184,320,000 (bytes)
184,320,000 * 4 = 737,280,000 (bytes)
737,280,000 / 1024 = 720,000 (Kilobytes)
720,000 / 1024 = 703.125 (Megabytes)
703.125 / 1024 = 0.686 (Gigabytes)
```

Around 700MB of memory.

So if you have 2GB of memory, you should be all set. Right? **Not, really:**

- This is just the base memory required to capture the streams. Remember ZM is always capturing streams irrespective of whether you are actually recording or not - to make sure its image ring buffer is there with pre images when an alarm kicks in.
- You also need to account for other processes not related to ZM running in your box
- You also need to account for other ZM processes - for example, I noticed the audit daemon takes up a good amount of memory when it runs, DB updates also take up memory
- If you are using H264 encoding, that buffers a lot of frames in memory as well.

So a good rule of thumb is to make sure you have twice the memory as the calculation above (and if you are using the ZM server for other purposes, please factor in those memory requirements as well)

Also remember by default ZM only uses 50% of your available memory unless you change it

As it turns out, ZM uses mapped memory and by default, 50% of your physical memory is what this will grow to. When you reach that limit, ZM breaks down with various errors.

A good way to know how much memory is allocated to ZM for its operation is to do a `df -h`

A sample output on Ubuntu:

```
pp@camerapc:~$ df -h|grep "Filesystem\|shm"
Filesystem                Size      Used Avail Use% Mounted on
tmpfs                     2.6G      923M    1.7G   36% /run/shm
```

The key item here is `tmpfs` → the example above shows we have allocated 1.7G of mapped memory space of which 36% is used which is a healthy number. If you are seeing `Use%` going beyond 70% you should probably increase the mapped memory.

For example, if you want to increase this limit to 70% of your memory, add the following to `/etc/fstab` `tmpfs SHMPATH tmpfs defaults,noexec,nosuid,size=70% 0 0` where `SHMPATH` is the Mounted on path. Here, that would be `/run/shm`. Other systems may be `/dev/shm`.

4.3 I have enabled motion detection but it is not always being triggered when things happen in the camera view

ZoneMinder uses zones to examine images for motion detection. When you create the initial zones you can choose from a number of preset values for sensitivity etc. Whilst these are usually a good starting point they are not always suitable for all situations and you will probably need to tweak the values for your specific circumstances. The meanings of the various settings are described in the documentation ([here](#)). Another user contributed illustrated Zone definition guide can be found here: [An illustrated guide to Zones](#)

However if you believe you have sensible settings configured then there are diagnostic approaches you can use.

4.3.1 Event Statistics

The first technique is to use event statistics. Firstly you should ensure they are switched on in `Options->Logging->RECORD_EVENT_STATS`. This will then cause the raw motion detection statistics for any subsequently generated events to be written to the DB. These can then be accessed by first clicking on the Frames or Alarm Frames values of the event from any event list view in the web gui. Then click on the score value to see the actual values that caused the event. Alternatively the stats can be accessed by clicking on the 'Stats' link when viewing any individual frame. The values displayed there correspond with the values that are used in the zone configuration and give you an idea of what 'real world' values are being generated.

Note that if you are investigating why events 'do not' happen then these will not be saved and so won't be accessible. The best thing to do in that circumstance is to make your zone more sensitive so that it captures all events (perhaps even ones you don't want) so you can get an idea of what values are being generated and then start to adjust back to less sensitive settings if necessary. You should make sure you test your settings under a variety of lighting conditions (e.g. day and night, sunny or dull) to get the best feel for that works and what doesn't.

Using statistics will slow your system down to a small degree and use a little extra disk space in the DB so once you are happy you can switch them off again. However it is perfectly feasible to keep them permanently on if your system is able to cope which will allow you to review your settings periodically.

4.3.2 Diagnostic Images along with FIFO

The second approach is to use diagnostic images which are saved copies of the intermediate images that ZM uses when determining motion detection. These are switched on and off using

Options->Logging->RECORD_DIAG_IMAGES.

Note: In addition to the detailed explanation below, a recently added RECORD_DIAG_IMAGES_FIFO option, also available in Options->Logging can be an invaluable tool to see how your current motion settings are affecting motion detection. The delta stream along with the raw (json output) stream can be invaluable to see the effect in real time. Please refer to the explanation of this feature in [Options - Logging](#)

There are two kinds of diagnostic images which are and are written (and continuously overwritten) to the top level monitor event directory. If an event occurs then the files are additionally copied to the event directory and renamed with the appropriate frame number as a prefix.

The first set are produced by the monitor on the image as a whole. The diag-r.jpg image is the current reference image against which all individual frames are compared and the diag-d.jpg image is the delta image highlighting the difference between the reference image and the last analysed image. In this images identical pixels will be black and the more different a pixel is the whiter it will be. Viewing this image and determining the colour of the pixels is a good way of getting a feel for the pixel differences you might expect (often more than you think).

The second set of diag images are labelled as diag-<zoneid>-<stage>.jpg where zoneid is the id of the zone in question (Smile) and the stage is where in the alarm check process the image is generated from. So if you have several zones you can expect to see multiple files. Also these files are only interested in what is happening in their zone only and will ignore anything else outside of the zone. The stages that each number represents are as follows,

- Alarmed Pixels - This image shows all pixels in the zone that are considered to be alarmed as white pixels and all other pixels as black.
- Filtered Pixels - This is as stage one except that all pixels removed by the filters are now black. The white pixels represent the pixels that are candidates to generate an event.
- Raw Blobs - This image contains all alarmed pixels from stage 2 but aggregated into blobs. Each blob will have a different greyscale value (between 1 and 254) so they can be difficult to spot with the naked eye but using a colour picker or photoshop will make it easier to see what blob is what.
- Filtered Blobs - This image is as stage 3 but under (or over) sized blobs have been removed. This is the final step before determining if an event has occurred, just prior to the number of blobs being counted. Thus this image forms the basis for determining whether an event is generated and outlining on alarmed images is done from the blobs in this image.

Using the above images you should be able to tell at all stages what ZM is doing to determine if an event should happen or not. They are useful diagnostic tools but as is mentioned elsewhere they will massively slow your system down and take up a great deal more space. You should never leave ZM running for any length of time with diagnostic images on.

4.4 Why can't ZoneMinder capture images (either at all or just particularly fast) when I can see my camera just fine in xawtv or similar?

With capture cards ZoneMinder will pull images as fast as it possibly can unless limited by configuration. ZoneMinder (and any similar application) uses the frame grabber interface to copy frames from video memory into user memory. This takes some time, plus if you have several inputs sharing one capture chip it has to switch between inputs between captures which further slows things down.

On average a card that can capture at 25fps per chip PAL for one input will do maybe 6-10fps for two, 1-4fps for three and 1-2 for four. For a 30fps NTSC chip the figures will be correspondingly higher. However sometimes it is necessary to slow down capture even further as after an input switch it may take a short while for the new image to settle before it can be captured without corruption.

When using xawtv etc to view the stream you are not looking at an image captured using the frame grabber but the card's video memory mapped onto your screen. This requires no capture or processing unless you do an explicit capture via the J or ctrl-J keys for instance. Some cards or drivers do not support the frame grabber interface at all so may not work with ZoneMinder even though you can view the stream in xawtv. If you can grab a still using the grab functionality of xawtv then in general your card will work with ZoneMinder.

4.5 Why can't I see streamed images when I can see stills in the zone window etc?

This issue is normally down to one of two causes

- 1) You are using Internet Explorer and are trying to view multi-part jpeg streams. IE does not support these streams directly, unlike most other browsers. You will need to install Cambozola or another multi-part jpeg aware plugin to view them. To do this you will need to obtain the applet from the Downloads page and install the cambozola.jar file in the same directory as the ZoneMinder php files. Then find the ZoneMinder Options->Images page and enable OPT_CAMBOZOLA and enter the web path to the .jar file in PATH_CAMBOZOLA. This will ordinarily just be cambozola.jar. Provided (Options / B/W tabs) WEB_H_CAN_STREAM is set to auto and WEB_H_STREAM_METHOD is set to jpeg then Cambozola should be loaded next time you try and view a stream.

NOTE: If you find that the Cambozola applet loads in IE but the applet just displays the version of Cambozola and the author's name (as opposed to seeing the streaming images), you may need to `chmod (-rwxrwxr-x)` your (`usr/share/zoneminder/`) `cambozola.jar`:

```
sudo chmod 775 cambozola.jar
```

Once I did this, images started to stream for me.

- 2) The other common cause for being unable to view streams is that you have installed the ZoneMinder cgi binaries (zms and nph-zms) in a different directory than your web server is expecting. Make sure that the `-with-cgidir` option you use to the ZoneMinder configure script is the same as the CGI directory configure for your web server. If you are using Apache, which is the most common one, then in your `httpd.conf` file there should be a line like `ScriptAlias /cgi-bin/ "/var/www/cgi-bin/"` where the last directory in the quotes is the one you have specified. If not then change one or the other to match. Be warned that configuring apache can be complex so changing the one passed to the ZoneMinder configure (and then rebuilding and reinstalling) is recommended in the first instance. If you change the apache config you will need to restart apache for the changes to take effect. If you still cannot see stream reliably then try changing `ZM_PATH_ZMS` in your `/etc/zm/config` directory to just use `zms` if `nph-zms` is specified, or vice versa. Also check in your apache error logs.

Lastly, please look for errors created by the `zmc` processes. If `zmc` isn't running, then `zms` will not be able to get an image from it and will exit.

4.6 I have several monitors configured but when I load the Montage view in FireFox why can I only see two? or, Why don't all my cameras display when I use the Montage view in FireFox?

By default FireFox only supports a small number of simultaneous connections. Using the montage view usually requires one persistent connection for each camera plus intermittent connections for other information such as statuses.

You will need to increase the number of allowed connections to use the montage view with more than a small number of cameras. Certain FireFox extensions such as FasterFox may also help to achieve the same result.

To resolve this situation, follow the instructions below:

Enter `about:config` in the address bar

scroll down to `browser.cache.check_doc_frequency` 3 change the 3 to a 1

```
browser.cache.disk.enable True -> False  
network.http.max-connections-per-server -> put a value of 100  
network.http.max-persistent-connections-per-proxy -> 100 again  
network.http.max-persistent-connections-per-server -> 100 again
```

4.7 I can't see more than 6 monitors in montage on my browser

Browsers such as Chrome and Safari only support up to 6 streams from the same domain. To work around that, take a look at the multi-port configuration discussed in the `MIN_STREAMING_PORT` configuration in *Options - Network*

4.8 Why is ZoneMinder using so much CPU?

The various elements of ZoneMinder can be involved in some pretty intensive activity, especially while analysing images for motion. However generally this should not overwhelm your machine unless it is very old or underpowered.

There are a number of specific reasons why processor loads can be high either by design or by accident. To figure out exactly what is causing it in your circumstances requires a bit of experimentation.

The main causes are.

- Using a video palette other than greyscale or RGB24. This can cause a relatively minor performance hit, though still significant. Although some cameras and cards require using planar palettes ZM currently doesn't support this format internally and each frame is converted to an RGB representation prior to processing. Unless you have compelling reasons for using YUV or reduced RGB type palettes such as hitting USB transfer limits I would experiment to see if RGB24 or greyscale is quicker. Put your monitors into 'Monitor' mode so that only the capture daemons are running and monitor the process load of these (the 'zmc' processes) using top. Try it with various palettes to see if it makes a difference.
- Big image sizes. A image of 640x480 requires at least four times the processing of a 320x240 image. Experiment with different sizes to see what effect it may have. Sometimes a large image is just two interlaced smaller frames so has no real benefit anyway. This is especially true for analog cameras/cards as image height over 320 (NTSC) or 352 PAL) are invariably interlaced.
- Capture frame rates. Unless there's a compelling reason in your case there is often little benefit in running cameras at 25fps when 5-10fps would often get you results just as good. Try changing your monitor settings to limit your cameras to lower frame rates. You can still configure ZM to ignore these limits and capture as fast as possible when motion is detected.
- Run function. Obviously running in Record or Mocord modes or in Modect with lots of events generates a lot of DB and file activity and so CPU and load will increase.
- Basic default detection zones. By default when a camera is added one detection zone is added which covers the whole image with a default set of parameters. If your camera covers a view in which various regions are unlikely to generate a valid alarm (ie the sky) then I would experiment with reducing the zone sizes or adding inactive zones to blank out areas you don't want to monitor. Additionally the actual settings of the zone themselves may not be optimal. When doing motion detection the number of changed pixels above a threshold is examined, then this is filter, then contiguous regions are calculated to see if an alarm is generated. If any maximum or minimum threshold is exceeded according to your zone settings at any time the calculation stops. If your settings always result in the calculations going through to the last stage before being failed then additional CPU time is used

unnecessarily. Make sure your maximum and minimum zone thresholds are set to sensible values and experiment by switching `RECORD_EVENT_STATS` on and seeing what the actual values of alarmed pixels etc are during sample events.

- Optimise your settings. After you've got some settings you're happy with then switching off `RECORD_EVENT_STATS` will prevent the statistics being written to the database which saves some time. Other settings which might make a difference are `ZM_FAST_RGB_DIFFS` and the `JPEG_XXX_QUALITY` ones.

I'm sure there are other things which might make a difference such as what else you have running on the box and memory sizes (make sure there's no swapping going on). Also speed of disk etc will make some difference during event capture and also if you are watching the whole time then you may have a bunch of zms processes running also.

I think the biggest factors are image size, colour depth and capture rate. Having said that I also don't always know why you get certain results from 'top'. For instance if I have a 'zma' daemon running for a monitor that is capturing an image. I've commented out the actual analysis so all it's doing is blending the image with the previous one. In colour mode this takes ~11 milliseconds per frame on my system and the camera is capturing at ~10fps. Using 'top' this reports the process as using ~5% of CPU and permanently in R(un) state. Changing to greyscale mode the blending takes ~4msec (as you would expect as this is roughly a third of 11) but top reports the process as now with 0% CPU and permanently in S(leep) state. So an actual CPU resource usage change of a factor of 3 causes huge differences in reported CPU usage. I have yet to get to the bottom of this but I suspect it's to do with scheduling somewhere along the line and that maybe the greyscale processing will fit into one scheduling time slice whereas the colour one won't but I have no evidence of this yet!

4.9 Why is the timeline view all messed up?

The timeline view is a new view allowing you to see a graph of alarm activity over time and to quickly scan and home in on events of interest. However this feature is highly complex and still in beta. It is based extensively on HTML div tags, sometimes lots of them. Whilst Firefox is able to render this view successfully other browsers, particular Internet Explorer do not seem able to cope and so present a messed up view, either always or when there are a lot of events. Using the timeline view is only recommended when using Firefox, however even then there may be issues.

This function has from time to time been corrupted in the SVN release or in the stable releases, try and reinstall from a fresh download.

4.10 How much Hard Disk Space / Bandwidth do I need for ZM?

Please see [this online excel sheet](#). Note that this is just an estimate

Or go to [this link](#) for the Axis bandwidth calculator. Although this is aimed at Axis cameras it still produces valid results for any kind of IP camera.

As a quick guide I have 4 cameras at 320x240 storing 1 fps except during alarm events. After 1 week 60GB of space in the volume where the events are stored (/var/www/html/zm) has been used.

4.11 When I try and run ZoneMinder I get lots of audit permission errors in the logs and it won't start

Many Linux distributions nowadays are built with security in mind. One of the latest methods of achieving this is via SELinux (Secure Linux) which controls who is able to run what in a more precise way than traditional accounting and file based permissions ([link](#)). If you are seeing entries in your system log like:

```
Jun 11 20:44:02 kernel: audit(1150033442.443:226): avc: denied { read } for pid=5068
comm="uptime" name="utmp" dev=dm-0 ino=16908345 scontext=user_u:system_r:httpd_sys_script_t
tcontext=user_u:object_r:initrc_var_run_t tclass=file
```

then it is likely that your system has SELinux enabled and it is preventing ZoneMinder from performing certain activities. You then have two choices. You can either tune SELinux to permit the required operations or you can disable SELinux entirely which will permit ZoneMinder to run unhindered. Disabling SELinux is usually performed by editing its configuration file (e.g., `/etc/selinux/config`) and then rebooting. However if you run a public server you should read up on the risks associated with disabled Secure Linux before disabling it.

Note that SELinux may cause errors other than those listed above. If you are in any doubt then it can be worth disabling SELinux experimentally to see if it fixes your problem before trying other solutions.

4.12 How do I enable ZoneMinder's security?

In the console, click on `Options->System`. Check the box next to `ZM_OPT_USE_AUTH`. You will immediately be asked to login. The default username is 'admin' and the password is 'admin'.

To Manage Users: In main console, go to `Options->Users`.

You may also consider to use the web server security, for example, `htaccess` files under Apache scope; You may even use this as an additional/redundant security on top of Zoneminders built-in security features. Note that if you choose to enable webserver auth, `zmNinja` may have issues. Please read the [zmNinja FAQ on basic authentication](#) for more information. Also please note that `zmNinja` does not support digest authentication.

4.13 Managing system load (with IP Cameras in mind)

4.13.1 Introduction

Zoneminder is a superb application in every way, but it does a job that needs a lot of horsepower especially when using multiple IP cameras. IP Cams require an extra level of processing to analogue cards as the `jpg` or `mjpeg` images need to be decoded before analysing. This needs grunt. If you have lots of cameras, you need lots of grunt.

Why do ZM need so much grunt? Think what Zoneminder is actually doing. In modect mode ZM is: 1. Fetching a `jpeg` from the camera. (Either in single part or multipart stream) 2. Decoding the `jpeg` image. 3. Comparing the zoned selections to the previous image or images and applying rules. 4. If in alarm state, writing that image to the disk and updating the `mysql` database.

If you're capturing at five frames per second, the above is repeated five times every second, multiplied by the number of cameras. Decoding the images is what takes the real power from the processor and this is the main reason why analogue cameras which present an image ready-decoded in memory take less work.

4.13.2 How do I know if my computer is overloaded?

If your CPU is running at 100% all the time, it's probably overloaded (or running at exact optimisation). If the load is consistently high (over 10.0 for a single processor) then Bad Things happen - like lost frames, unrecorded events etc. Occasional peaks are fine, normal and nothing to worry about.

Zoneminder runs on Linux, Linux measures system load using "load", which is complicated but gives a rough guide on what the computer is doing at any given time. Zoneminder shows Load on the main page (top right) as well as disk space. Typing "uptime" on the command line will give a similar guide, but with three figures to give a fuller measure of what's happening over a period of time but for the best guide to see what's happening, install "htop" - which gives easy to read graphs for load, memory and cpu usage.

A load of 1.0 means the processor has “just enough to do right now”. Also worth noting that a load of 4.0 means exactly the same for a quad processor machine - each number equals a single processor’s workload. A very high load can be fine on a computer that has a stacked workload - such as a machine sending out bulk emails, or working its way through a knotty problem; it’ll just keep churning away until it’s done. However - Zoneminder needs to process information in real time so it can’t afford to stack its jobs, it needs to deal with them right away.

For a better and full explanation of Load: [Please read this](#)

4.13.3 My load is too high, how can I reduce it?

(The previous documentation explained how to use turbo jpeg libraries as an optimization technique. These libraries have long been part of standard linux distros since that article was authored and hence that section has been removed)

Zoneminder is *very* tweakable and it’s possible to tune it to compromise. The following are good things to try, in no particular order;

- If your camera allows you to change image size, think whether you can get away with smaller images. Smaller pics = less load. 320x240 is usually ok for close-up corridor shots.
- Go Black and White. Colour pictures use twice to three times the CPU, memory and disk space but give little benefit to identification.
- Reduce frames per second. Halve the fps, halve the workload. If your camera supports fps throttling (Axis do), try that - saves ZM having to drop frames from a stream. 2-5 fps seems to be widely used.
- Experiment with using jpeg instead of mjpeg. Some users have reported it gives better performance, but YMMV.
- Tweak the zones. Keep them as small and as few as possible. Stick to one zone unless you really need more. Read [this](#) for an easy to understand explanation along with the official Zone guide.
- Schedule. If you are running a linux system at near capacity, you’ll need to think carefully about things like backups and scheduled tasks. updatedb - the process which maintains a file database so that ‘locate’ works quickly, is normally scheduled to run once a day and if on a busy system can create a heavy increase on the load. The same is true for scheduled backups, especially those which compress the files. Re-schedule these tasks to a time when the cpu is less likely to be busy, if possible - and also use the “nice” command to reduce their priority. (crontab and /etc/cron.daily/ are good places to start)
- Reduce clutter on your PC. Don’t run X unless you really need it, the GUI is a huge overhead in both memory and cpu.

More expensive options:

- Increase RAM. If your system is having to use disk swap it will HUGELY impact performance in all areas. Again, htop is a good monitor - but first you need to understand that because Linux is using all the memory, it doesn’t mean it needs it all - linux handles ram very differently to Windows/DOS and caches stuff. htop will show cached ram as a different colour in the memory graph. Also check that you’re actually using a high memory capable kernel - many kernels don’t enable high memory by default.
- Faster CPU. Simple but effective. Zoneminder also works very well with multiple processor systems out of the box (if SMP is enabled in your kernel). The load of different cameras is spread across the processors.
- Try building Zoneminder with processor specific instructions that are optimised to the system it will be running on, also increasing the optimisation level of GCC beyond -O2 will help. This topic is beyond the scope of this document.

Processor specific commands can be found in the [GCC manual](#) along with some more options that may increase performance.

4.13.4 What about disks and bandwidth?

A typical 100mbit LAN will cope with most setups easily. If you're feeding from cameras over smaller or internet links, obviously fps will be much lower.

Disk and Bandwidth calculators are referenced in *How much Hard Disk Space / Bandwidth do I need for ZM?*.

4.13.5 How do I build for X10 support?

You do not need to rebuild ZM for X10 support. You will need to install the perl module and switch on X10 in the options, then restart. Installing the perl module is covered in the README amongst other places but in summary, do:

```
perl -MCPAN -eshell install X10::ActiveHome quit
```

4.14 Extending Zoneminder

4.14.1 How can I get ZM to do different things at different times of day or week?

If you want to configure ZoneMinder to do motion detection during the day and just record at night, for example, you will need to use ZoneMinder 'run states'. A run state is a particular configuration of monitor functions that you want to use at any time.

To save a run state you should first configure your monitors for Modect, Record, Monitor etc as you would want them during one of the times of day. Then click on the running state link at the top of the Console view. This will usually say 'Running' or 'Stopped'. You will then be able to save the current state and give it a name, 'Daytime' for example. Now configure your monitors how you would want them during other times of day and save that, for instance as 'Nighttime'.

Now you can switch between these two states by selecting them from the same dialog you saved them, or from the command line from issue the command `"zmpkg.pl <run state>"`, for example `"zmpkg.pl Daytime"`.

The final step you need to take, is scheduling the time the changes take effect. For this you can use [cron](#). A simple entry to change to the Daylight state at 8am and to the nighttime state at 8pm would be as follows,

```
0 8 * * * root /usr/local/bin/zmpkg.pl Daytime
0 20 * * * root /usr/local/bin/zmpkg.pl Nighttime
```

On Ubuntu 7.04 and possibly others, look in `/usr/bin` not just `/usr/local/bin` for the `zmpkg.pl` file.

Although the example above describes changing states at different times of day, the same principle can equally be applied to days of the week or other more arbitrary periods.

4.14.2 How can I use ZoneMinder to trigger something else when there is an alarm?

ZoneMinder includes a perl API which means you can create a script to interact with the ZM shared memory data and use it in your own scripts to react to ZM alarms or to trigger ZM to generate new alarms. Full details are in the README or by doing `perldoc ZoneMinder` etc.

ZoneMinder provides a sample alarm script called `zmalarm.pl` that you can refer to as a starting point.

4.15 Trouble Shooting

Here are some things that will help you track down whats wrong. This is also how to obtain the info that we need to help you on the forums.

4.15.1 What logs should I check for errors?

ZoneMinder creates its own logs and are usually located in the `/var/log/` directory. Refer to the logging discussion in *Options - Logging* for more details on where logs are stored and how to enable various log levels.

Since ZM is dependent on other components to work, you might not find errors in ZM but in the other components.

```
* /var/log/messages and/or /var/log/syslog
* /var/log/dmesg
* /var/log/httpd/error_log`` (RedHat/Fedora) or ``/var/log/apache2/error_log
* /var/log/mysqld.log`` (Errors here don't happen very often but just in case)
```

If ZM is not functioning, you should always be able to find an error in at least one of these logs. Use the `tail` command to get info from the logs. This can be done like so:

```
tail -f /var/log/messages /var/log/httpd/error_log /var/log/zm/zm*.log
```

This will append any data entered to any of these logs to your console screen (`-f`). To exit, hit `[ctrl -c]`.

4.15.2 How can I trouble shoot the hardware and/or software?

Here are some commands to get information about your hardware. Some commands are distribution dependent. * `[[lspci]] -vv` - Returns lots of detailed info. Check for conflicting interrupts or port assignments. You can sometimes alter interrupts/ ports in bios. Try a different pci slot to get a clue if it is HW conflict (command provided by the `pciutils` package). * `[[scanpci]] -v` - Gives you information from your hardware EPROM * `[[lsusb]] -vv` - Returns lots of detail about USB devices (camand provided by `usbutils` package). * `[[dmesg]]` - Shows you how your hardware initialized (or didn't) on boot-up. You will get the most use of this. * `[[v4l-info]]` - to see how driver is talking to card. look for unusual values. * `[[modinfo btvtv]]` - some `btvtv` driver stats. * `[[zmu]] -m 0 -q -v` - Returns various information regarding a monitor configuration. * `[[ipcs]] ``` -- Provides information on the ipc facilities for which the calling process has read access. * ```[[ipcrm]] ``` -- The `ipcrm` command can be used to remove an IPC object from the kernel. * ```cat /proc/interrupts` - This will dispaly what interrupts your hardware is using.

4.15.3 Why am I getting a 403 access error with my web browser when trying to access `http //localhost/zm?`

The apache web server needs to have the right permissions and configuration to be able to read the Zoneminder files. Check the forums for solution, and edit the apache configuration and change directory permissions to give apache the right to read the Zoneminder files. Depending on your Zoneminder configuration, you would use the `zm` user and group that Zoneminder was built with, such as `wwwuser` and `www`.

4.15.4 Why am I getting broken images when trying to view events?

Zoneminder and the Apache web server need to have the right permissions. Check [this forum topic](#) and similar ones:

4.15.5 I can review events for the current day, but ones from yesterday and beyond error out

If you've checked that the `www-data` user has permissions to the storage folders, perhaps your `php.ini`'s `timezone` setting is incorrect. They `_must_` match for certain playback functions.

If you're using Linux, this can be found using the following command:

```
timedatectl | grep "Time zone"
```

If using FreeBSD, you can use this one-liner:

```
cd /usr/share/zoneinfo/ && find * -type f -exec cmp -s {} /etc/localtime \; -print;
```

Once you know what timezone your system is set to make sure you set the right time zone in ZM (Available in Options->System->TimeZone)

4.15.6 Why is the image from my color camera appearing in black and white?

If you recently upgraded to zoneminder 1.26, there is a per camera option that defaults to black and white and can be mis-set if your upgrade didn't happen right. See this thread: <https://forums.zoneminder.com/viewtopic.php?f=30&t=21344>

This may occur if you have a NTSC analog camera but have configured the source in ZoneMinder as PAL for the Device Format under the source tab. You may also be mislead because `zmu` can report the video port as being PAL when the camera is actually NTSC. Confirm the format of your analog camera by checking it's technical specifications, possibly found with the packaging it came in, on the manufacturers website, or even on the retail website where you purchased the camera. Change the Device Format setting to NTSC and set it to the lowest resolution of 320 x 240. If you have confirmed that the camera itself is NTSC format, but don't get a picture using the NTSC setting, consider increasing the shared memory `kernel.shmall` and `kernel.shmmax` settings in `/etc/sysctl.conf` to a larger value such as 268435456. This is also the reason you should start with the 320x240 resolution, so as to minimize the potential of memory problems which would interfere with your attempts to troubleshoot the device format issue. Once you have obtained a picture in the monitor using the NTSC format, then you can experiment with raising the resolution.

4.15.7 Why do I only see blue screens with a timestamp when monitoring my camera?

If this camera is attached to a capture card, then you may have selected the wrong Device Source or Channel when configuring the monitor in the ZoneMinder console. If you have a capture card with 2 D-sub style inputs(looks like a VGA port) to which you attach a provided splitter that splits off multiple cables, then the splitter may be attached to the wrong port. For example, PV-149 capture cards have two D-sub style ports labeled as DB1 and DB2, and come packaged with a connector for one of these ports that splits into 4 BNC connectors. The initial four video ports are available with the splitter attached to DB1.

4.15.8 Why do I only see black screens with a timestamp when monitoring my camera?

In the monitor windows where you see the black screen with a timestamp, select settings and enter the Brightness, Contrast, Hue, and Color settings reported for the device by `zmu -d <device_path> -q -v`. 32768 may be appropriate values to try for these settings. After saving the settings, select Settings again to confirm they saved successfully.

4.15.9 How do I repair the MySQL Database?

There is two ways to go about this. In most cases you can run from the command prompt -> `mysqlcheck --all-databases --auto-repair -p your_database_password -u your_databse_user`

If that does not work then you will have to make sure that ZoneMinder is stopped then run the following (nothing should be using the database while running this and you will have to adjust for your correct path if it is different):

```
myisamchk --silent --force --fast --update-state -O key_buffer=64M -O
sort_buffer=64M -O read_buffer=1M -O write_buffer=1M /var/lib/mysql/*/*.MYI
```

4.15.10 How do I repair the MySQL Database when the cli fails?

In Ubuntu, the commands listed above do not seem to work. However, actually doing it by hand from within MySQL does. (But that is beyond the scope of this document) But that got me thinking... And phpmyadmin does work. Bring up a terminal. `sudo apt-get install phpmyadmin`

Now go to `http://zoneminder_IP/` and stop the ZM service. Continue to `http://zoneminder_IP/phpmyadmin` and select the zoneminder database. Select and tables marked 'in use' and pick the action 'repare' to fix. Restart the zoneminder service from the web browser. Remove or disable the phpmyadmin tool, as it is not always the most secure thing around, and opens your database wide to any skilled hacker. `sudo apt-get remove phpmyadmin`

4.15.11 I upgraded by distribution and ZM stopped working

Some possibilities (Incomplete list and subject to correction) `[[/usr/local/bin/zmfix: /usr/lib/libmysqlclient.so.15: version `MYSQL_5.0' not found (required by /usr/local/bin/zmfix)]]` :: Solution: Recompile and reinstall Zoneminder. Any time you update a major version that ZoneMinder depends on, you need to recompile ZoneMinder.

4.15.12 Zoneminder doesn't start automatically on boot

Check the list for log entries like "zmfix[766]: ERR [Can't connect to server: Can't connect to local MySQL server through socket '/var/run/mysqld/mysqld.sock' (2)] ". What can happen is that zoneminder is started too quickly after Mysql and tries to contact the database server before it's ready. Zoneminder gets no answer and aborts. August 2010 - Ubuntu upgrades seem to be leaving several systems in this state. One way around this is to add a delay to the zoneminder startup script allowing Mysql to finish starting. "Simply adding 'sleep 15' in the line above 'zmfix -a' in the /etc/init.d/zoneminder file fixed my ZoneMinder startup problems!" - credit to Pada.

4.15.13 Remote Path setup for Panasonic and other Camera

On adding or editing the source you can select the preset link for the parameters for the specified camera . In version 1.23.3 presets for BTTV,Axis,Panasonic,GadSpot,VEO, and BlueNet are available . Selecting the presets ZM fills up the required value for the remote path variable

4.15.14 Why do I get repeated/ mixed/unstable/ blank monitors on bt878-like cards (a.k.a. PICO 2000)

Please have a check at `[[Pico2000]]`;

4.15.15 What causes “Invalid JPEG file structure: two SOI markers” from zmc (1.24.x)

Some settings that used to be global only are now per camera. On the Monitor Source tab, if you are using Remote Protocol “HTTP” and Remote Method “Simple”, try changing Remote Method to “Regexp”.

4.16 Miscellaneous

4.16.1 I see ZoneMinder is licensed under the GPL. What does that allow or restrict me in doing with ZoneMinder?

The ZoneMinder license is described at the end of the documentation and consists of the following section

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

This means that ZoneMinder is licensed under the terms described [here](#). There is a comprehensive FAQ covering the GPL at <https://www.gnu.org/licenses/gpl-faq.html> but in essence you are allowed to redistribute or modify GPL licensed software provided that you release your distribution or modifications freely under the same terms. You are allowed to sell systems based on GPL software. You are not allowed to restrict or reduce the rights of GPL software in your distribution however. Of course if you are just making modifications for your system locally you are not releasing changes so you have no obligations in this case. I recommend reading the GPL FAQ for more in-depth coverage of this issue.

4.16.2 Can I use ZoneMinder as part of my commercial product?

The GPL license allows you produce systems based on GPL software provided your systems also adhere to that license and any modifications you make are also released under the same terms. The GPL does not permit you to include ZoneMinder in proprietary systems (see <https://www.gnu.org/licenses/gpl-faq.html#GPLInProprietarySystem> for details). If you wish to include ZoneMinder in this kind of system then you will need to license ZoneMinder under different terms. This is sometimes possible and you will need to contact me for further details in these circumstances.

4.16.3 I am having issues with zmNinja and/or Event Notification Server

zmNinja and the Event Notification Server are 3rd party solutions. The developer maintains exhaustive [documentation](#) and [FAQs](#). Please direct your questions there.

CHAPTER 5

Contributing

Source hosted at [GitHub](#) Report issues/questions/feature requests on [GitHub Issues](#)

Pull requests are very welcome! If you would like to contribute, please follow the following steps.

- Fork the repo
- Open an issue at our [GitHub Issues Tracker](#). Describe the bug that you've found, or the feature which you're asking for. Jot down the issue number (e.g. 456)
- Create your feature branch (`git checkout -b 456-my-new-feature`)
- Commit your changes (`git commit -m 'Added some feature'`) It is preferred that you 'commit early and often' instead of bunching all changes into a single commit.
- Push your branch to your fork on github (`git push origin 456-my-new-feature`)
- Create new Pull Request
- The team will then review, discuss and hopefully merge your changes.

Welcome to ZoneMinder's documentation. Please navigate to one of the links below.

If you are facing issues that are not covered in the documentation, please feel free to check the [ZoneMinder Forums](#) or join the [ZoneMinder-Chat Slack channel](#) if you prefer real time interaction.

Installation Guide Many distribution repos only hold older versions of ZoneMinder, current versions contain many bug fixes and updated functionality. Instructions here for installing updated packages or compiling from source.

User Guide Guide to setting up ZoneMinder for the first time and detailed guides for using the ZoneMinder front end.

API Information on using the CakePHP based API for interfacing to ZoneMinder

FAQ Frequently Asked Questions

Contributing How to contribute to ZoneMinder. As a community project we always need help, you don't need to be a coder to test or update documentation.

Event Notification Server and Machine Learning hooks Documentation for the 3rd party Event Notification Server and Machine Learning for Object/People/Face detection.

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`