# M/MONIT

## User Manual

Version 4.3.4



Tildeslash Ltd.

Copyright © 2025 Tildeslash Ltd. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Tildeslash Ltd., with the following exceptions: Any person is hereby
to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains the Tildeslash copyright notice. No part of this publication may be reproduced or transmitted for commercial purposes, such as selling copies of this publication or for providing paid for support services.

Every effort has been made to ensure that the information in this manual is accurate. Tildeslash is not responsible for printing or clerical errors. Because Tildeslash frequently releases new versions and updates to its applications and Internet sites, images shown in this manual may be slightly different from what you see on your screen.

Portions of the M/Monit Software utilizes or includes third party software. Acknowledgements, licensing terms and disclaimers for such material are distributed with the M/Monit Software, and your use of such material is governed by their respective terms.



# Contents:


**Introducing M/Monit** **8**

M/Monit for Phone and Tablets 9

System requirements 10

Network communication requirements 11

**Installation** **12**

Upgrading 13

Resources 14

**Architecture overview** **16**

**Configuration** **17**

Configuring M/Monit 20

M/Monit configuration files 20

How to change the port number M/Monit listen on? 20

How to setup M/Monit to use SSL/TLS? 21

SSL Certificate Verification for outgoing connections 22

How to setup M/Monit to use MySQL, MariaDB or PostgreSQL? 23

How to increase the login session timeout in M/Monit? 24

How to install the license key? 24

**A First Look at M/Monit** **25**

**Login to M/Monit** **26**

**Dashboard** **27**

Host Status 27

Events in last 24-hours 28

Messages 29

**Status** **30**
Detailed host status 31
Service Details 31
Service actions 32
Host charts 32
**Analytics Reports** **33**
Analytics Query 33
Chart types 34
Chart buttons 36
Chart refresh 37
Scale and zooming 37
Custom selection 38
Panning 38
Tooltip and series highlighting 39
Monit poll-cycle settings and data granularity or dots vs. lines 40
Data gaps 40
Data correlation 41
Uptime Report 42
Services Uptime Report 44
Events Report 45
Event details 46
**Admin** **47**
Hosts 48
Edit host 49
Host Groups 52
Users 53
Overview 53
Create or edit user 54
Appearance 54

Photo 54
Delete a user 54
User self management 55
Access Roles 56
Overview 56
Role types 56
Host Permissions 57
Role Members 57
Role deactivation 57
Alerts 58
Rules 58
Conditions 59
Alert actions 60
Mail servers 62
Jabber servers 63
Alert Service 64
How to prevent Monit from also sending alerts 66
Settings 67
**Appendix A 69**
server.xml 69
Directory and file names 69
<Server> 69
<Service> 69
<Connector> 70
<Engine> 73
<Host> 75
<Context> 77
<Realm> 78
<ErrorLogger> 80
<AccessLogger> 80

<Logger> 82
<SSLClientConfig> 83
<License> 85

**Appendix B** **86**

Access Control 86
Security-Constraint 86
Authentication 90
Login-Config 90
CSRF-protection 94

**Appendix C** **95**

M/Monit behind a proxy 95

**Appendix D** **97**

Automatic TLS certificate renewal 97

| Status | Events | * | Host |
| --- | --- | --- | --- |
| 12 services are available | 324 | | netbsd-x64 |
| 9 services are available | 256 | | alpine315-x64-lxc |
| 10 services are available | 252 | | mysqlreplica1 |
| 9 services are available | 213 | | unifi-lxc |
| 12 services are available | 121 | | mysqlreplica3 |
| 12 services are available | 69 | | mysqlreplica2 |
| 11 services are available | 66 | | debian11-lxc |
| 2 services are available | 47 | | ubuntu-latest-x64 |
| 7 services are available | 46 | | debian10-lxc |
| 2 services are available | 45 | | |
| 2 services are available | 2220 | | freebsd-arm64-qemu |
| 3 services are available | 1643 | | macmini-openbsd.localdomain |
| 1 services are available | 1608 | | alpine-arm64-macmini |
| 3 services are available | 1576 | | freebsd-arm64-macmini |
| 27 services are available | 1419 | | ubuntu-x64 |
| 9 services are available | 1339 | | rpi-arm64 |
| 13 services are available | 828 | | ubuntu1804-arm64-lxc |
| 26 services are available | 535 | | openbsd-arm64-qemu.my.domain |
| 42 services are available | 489 | | alpine315-arm64-lxc |
| 25 services are available | | | openbsd-x64.my.domain |
| are available | | | ubuntu1804-x64-lxc |
| | | | 64-vmware |

# Introducing M/Monit

M/Monit is a system for automatic management and pro-active monitoring of Information Technology Systems. M/Monit can monitor and manage distributed computer systems, conduct automatic maintenance and repair and execute meaningful causal actions in error situations.

M/Monit uses Monit as an agent and can manage and monitor all your hosts and services. M/Monit can start a service if it does not run, restart a service if it does not respond and suspend a service if it uses too much resources.

Monitor system attributes such as CPU, Load, Memory, Disk usage, Files, Directories and Filesystems for changes on all your hosts. Conditional rules can be set and if a value goes outside a defined scope, specific actions can be executed and a notification sent.

Information is collected from the monitored systems and stored in a database. Drill-down and filter functions exist to investigate collected data. Status and events from each monitored system are automatically updated and displayed in charts and tables.

## Benefits

M/Monit is a turn-key solution and requires very little configuration and no setup of third-party components.

Your computer systems will have a higher uptime as M/Monit can handle error conditions automatically, often without the need for human intervention.

M/Monit has a clean, simple and well designed user interface which scales well, if you manage 2 hosts or 1000+ hosts.

Source code with an SDK is available. Parts of the M/Monit system are also released as open-source code.

## Cost-effective

A M/Monit license is a one-time payment and the license does not expire.

The cost is minuscule compared to similar commercial systems and only a fraction of the cost as to the work hours required to setup and configure a comparable open-source system.

## Technology

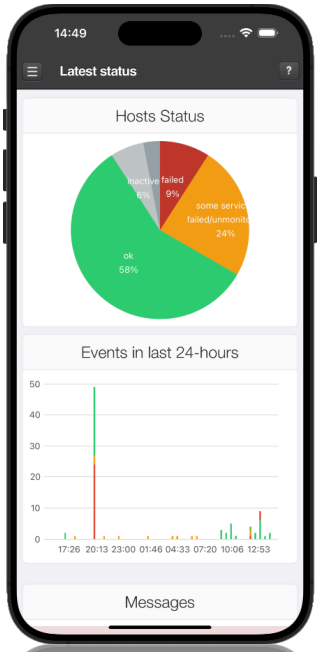
M/Monit is a modern, compact and scalable application server. Thread-pools and a non-blocking, event driven i/o architecture is used to ensure high performance. M/Monit runs on any POSIX system and uses minimal system resources.

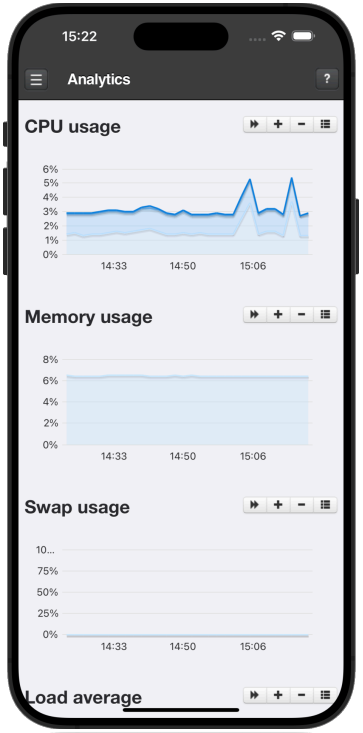
Database access is handled by a connection pool with support for MySQL, MariaDB, PostgreSQL and SQLite.

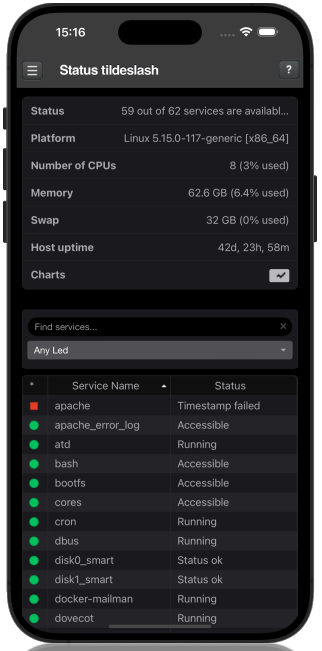
# M/Monit for Phone and Tablets

M/Monit works on iOS with Safari and on Android with the default Chrome browser. Just open the M/Monit URL in your browser. The User Interface will automatically adjust to fit the available screen size, whether you use a phone or a tablet.

On iOS and Android, you can add M/Monit to your home screen as a web app, allowing you to launch M/Monit by tapping its icon. M/Monit will then start in fullscreen mode and appear like a native app. Gestures, such as swiping from the left edge to the right to go back to the previous page, also work.



# System requirements

- M/Monit requires **Monit** as an agent. The Monit software must be version 5.2 or later and installed on all hosts M/Monit should monitor. Prebuilt Monit binaries can be downloaded from https://mmonit.com/monit/download/
- M/Monit runs on any **POSIX** system and is currently tested and available on Linux, FreeBSD, Solaris, MacOS and OpenBSD.
- **Memory** and **Disk** space: A minimum of 10 megabytes of RAM is required, along with approximately 25 MB of free disk space. You may need more RAM depending on the number of monitored hosts and services, how many processor threads the M/Monit server is started with, and the number of active login sessions.
- **CPU** requirements: No special requirements. A low-end system with a single x86_64 CPU should be able to provide enough power to manage hundreds of Monit agents and hundreds of M/Monit web-app users.
- Accurate **time** keeping: M/Monit uses the time of day for reporting and monitoring and it is recommended to investigate if your system clock has the correct time and set time synchronisation facility (NTP) on your system.
- **Random Device**: A random device is needed for creating universal unique and cryptographically strong HTTP and TLS Session identifiers. M/Monit will complain and exit if it cannot find `/dev/random` or `/dev/urandom` on the system.
- M/Monit **depends** only on libc installed on all POSIX systems.
- M/Monit is **distributed** as a tar gzip archive with the following content:
  - The mmonit program
  - Dynamic shared libraries used by mmonit
  - A default SQLite Database
  - The M/Monit Web application
  - Program for upgrading previous versions
  - Source code and API documentation (SDK version)

# Network communication requirements

M/Monit communicate with Monit agents on TCP port 2812. If there is a NAT or PAT (port translation) between M/Monit and Monit, you will need to setup host information in M/Monit so M/Monit can connect to Monit over the network. This can be specified in the admin/hosts page in M/Monit. Otherwise M/Monit will use the host information it receive from Monit when Monit automatically registered itself in M/Monit.

If a connection from M/Monit to Monit agents is not available, M/Monit will not be able to manage services on the Monit Host, but M/Monit can still receive messages from the Monit agents and display status, events and statistics.

M/Monit may need to connect to a SMTP and a Jabber server to send alert notifications. M/Monit may also need to communicate with a database server if it was configured to use MySQL or PostgreSQL.

The first time the mmonit program starts it might connect to mmonit.com on port 443 and ask for a trail license if a license file is not found. This is only done once.

The M/Monit admin page displays a news feed from https://mmonit.com/ Access to this feed is not required and if M/Monit cannot fetch the feed, it is not displayed.

If you need to setup firewall rules for M/Monit, the default set of rules will be:

| Communication | Description |
|---|---|
| Monit agents and M/Monit Browser clients -> M/Monit : 8080/TCP | The M/Monit web-interface |
| M/Monit host -> Monit agents : 2812/TCP | The Monit agents |
| M/Monit host -> MySQL server : 3306/TCP | The MySQL Database server [optional] |
| M/Monit host -> PostgreSQL server : 5432/TCP | The PostgreSQL Database server [optional] |
| M/Monit host -> Jabber server : 5222/TCP or 5223/TCP | The XMPP/Jabber server [optional] |
| M/Monit host -> SMTP server : 25/TCP or 465/TCP or 587/TCP | The SMTP server(s) [optional] |
| M/Monit Browser clients -> https://mmonit.com/ | The M/Monit news feed [optional] |

# Installation




It is very easy to install M/Monit. All you need to do is download the tar gzip package from https://www.mmonit.com/download/ and unpack the tar.gz file. After unpacking, you will have a new directory called mmonit-4.3.4. On macOS, you download an installer instead which will install mmonit in */usr/local/mmonit*.

Now that you have installed M/Monit, it's time to launch it. Simply execute the mmonit program located in the bin directory. Then, point your browser to the host where M/Monit is installed or *localhost* if it is running on the same machine as your browser, for example: http://localhost:8080/ and login as user ***admin*** with password ***swordfish***.

Once started, mmonit will run as a background process. To stop mmonit, use `mmonit stop`. To run mmonit in the foreground and in diagnostic mode, start mmonit with the `-id` options. In diagnostic mode, mmonit will print error and debug informations to the console. Use CTRL+C to stop mmonit in this mode. To see all options for the program, use `mmonit -h`.

You can run mmonit as any UNIX user, although it is recommended (but not necessary) to create a standalone account to run mmonit in production.

# Upgrading

If you already have M/Monit installed you can upgrade to the latest version using the *upgrade* program. This program takes one required option, `-p`, which specifies the path to the previous M/Monit installation. *On **macOS**, running the upgrade program is not necessary, the installer will upgrade M/Monit in /usr/local/mmonit automatically*.

For example, to upgrade M/Monit 3.7.15 installed in */usr/local/* to the latest version: Download M/Monit for your OS and unpack the new version in the same directory. Then stop the previous M/Monit version:

```
/usr/local/mmonit-3.7.15/bin/mmonit stop
```

Run the upgrade program in the new version and specify the path to the previous M/Monit version:

```
/usr/local/mmonit-4.3.4/upgrade/upgrade -p /usr/local/mmonit-3.7.15
```

Finally, start M/Monit 4.3.4:

```
/usr/local/mmonit-4.3.4/bin/mmonit
```

The *upgrade* program will:

- Copy configuration files (including your license) to the new M/Monit installation
- Copy the SQLite database to the new M/Monit installation (if the SQLite backend is enabled)
- Copy uploaded user images to the new M/Monit installation
- Read the database location from the configuration file, `server.xml`, connect to the database, detect old schema version and incrementally upgrade the schema using patch-sets stored in the directory *upgrade/schema*

# Resources

## Website

M/Monit's website has a lot of informations about M/Monit and Monit. It provides up-to-date informations about the application, including new and updated features, development news, and tutorials. The website also sports a wiki with more informations and user submitted content.

## Mailing List

You can subscribe to M/Monit's mailing list to be the first to hear about new releases and important informations about M/Monit. The mailing list is read-only with very low traffic.

## Feedback and Bug Reporting

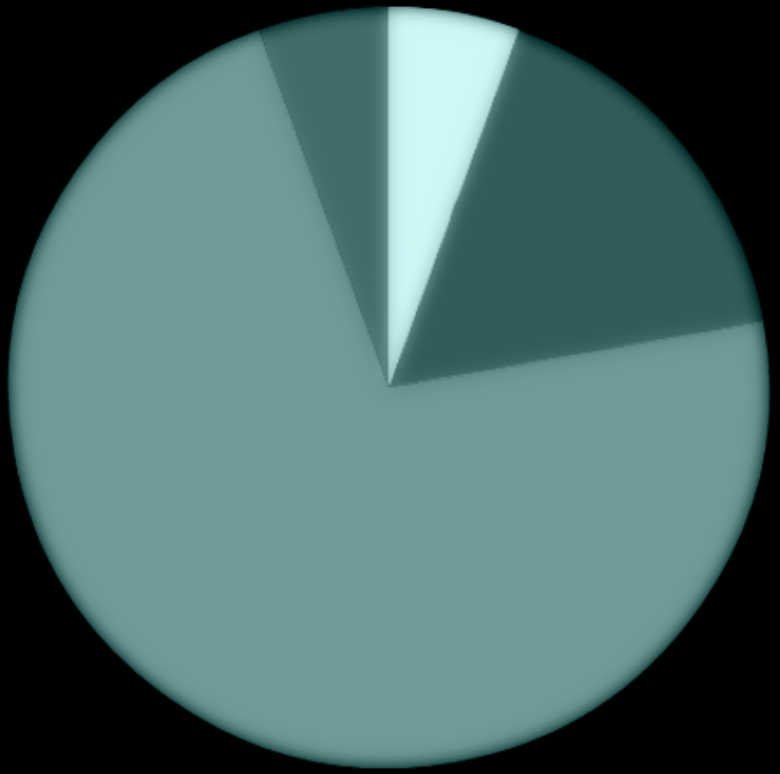
Though we invest a lot of time and hard work making sure Monit and M/Monit are high-quality applications, bugs may still sometimes find their way into the application.

Use the issue report schema at www.mmonit.com/contact/ to let us know about any bugs or feature requests you have.

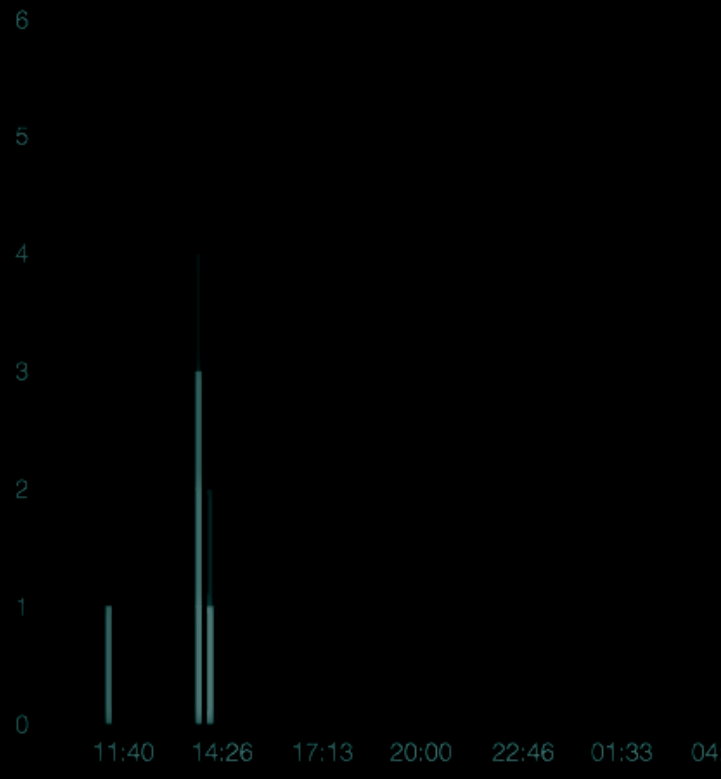
You can also contact us directly at info@mmonit.com

# Latest status

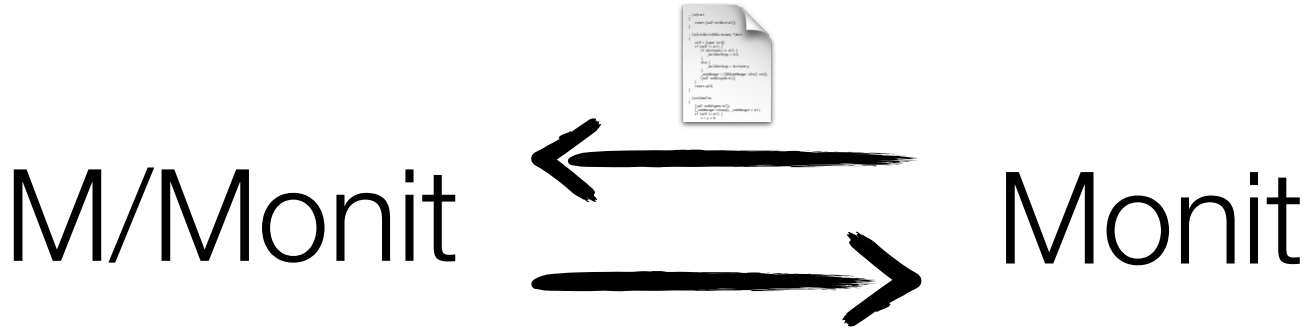
Hosts Status



Events in last 24-hours



# Architecture overview




Monit is a small, powerful monitoring program that runs on each host monitored by M/Monit. With regular intervals, Monit will send a message to M/Monit with the status of the host it is running on. If a service fails or Monit has to perform an action to fix a problem, an event message is sent to M/Monit at once. Both status and event messages are stored in a database. Upon receiving an event message from Monit, M/Monit will consult its rule-set and perform an alert notification if a rule matched.

From M/Monit, you can start, stop and restart services on any of your hosts running Monit.

# Configuration

Before we take a look at the M/Monit web-application let's take a quick look at M/Monit and Monit configuration. As we mentioned earlier, Monit must be configured in order to send event and status information to M/Monit. In the future we plan to add zero-conf capabilities to M/Monit and Monit, but for now you will need to add a few lines to Monit's configuration file. To learn more about Monit and Monit configuration, please consult the documentation and configuration examples at Monit's website.

To setup communication between Monit and M/Monit, add the following statements to the top of each Monit control file, *monitrc*:

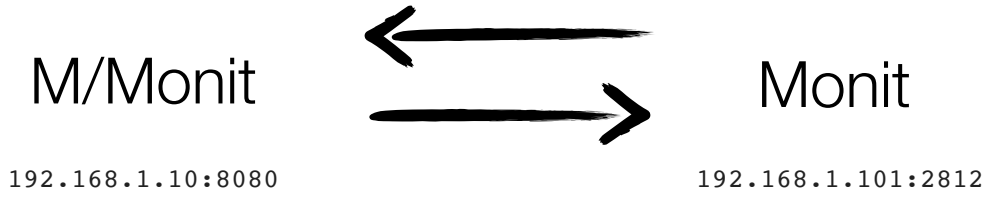
```
1.   set eventqueue basedir <path> [slots <number>]
2.   set mmonit http://<user>:<password>@<host>:<port>/collector
3.   set httpd port 2812
4.       allow localhost
5.       allow <M/Monit-host>
6.       allow username:password
```

The *set eventqueue* statement in line 1 is optional, but recommended. It allows Monit to store event messages if connection to M/Monit should temporarily be unavailable and retry delivery later. This way, no events will be lost. The slots option can be used to set a limit on how many events can be stored so the queue will not grow without limits if M/Monit is not available. The size of a queued message is small (ca. 200 bytes) so the space requirements for, let's say, 1000 queued events is only 200kB.

The *set mmonit* statement in line 2 specify the URL to be used by Monit for sending messages to M/Monit. The M/Monit URL is protected, and a username and a password are required to post messages to M/Monit. You can use the username and password of any valid user in M/Monit. For instance, the default user, "monit" with password "monit". The host and port in the URL, specify respectively the IP address of the machine running M/Monit and the port on which M/Monit is listening.

The *set httpd* statement starting on line 3 allow M/Monit to connect to Monit. Specify the IP-address or host name of the host running M/Monit, this should be the same address as specified in the *set mmonit* statement mentioned above.

Finally, in line 6, specify Basic Authentication username and password M/Monit should use to login to Monit.




Assume we have the machines above where M/Monit runs on 192.168.1.10 and listen on port 8080. Monit runs on 192.168.1.101 and listen on port 2812. You should then add the following to your *monitrc* configuration on 192.168.1.101:

```
set eventqueue basedir /var/monit/ slots 1000
set mmonit http://monit:monit@192.168.1.10:8080/collector
set httpd port 2812
    allow localhost
    allow 192.168.1.10
    allow admin:secret
```

You can also configure M/Monit and Monit to use TLS. In the example below we demonstrate how to setup the two-way communication between M/Monit and Monit to use TLS. The differences from the above example is that the *set mmonit* statement now uses https instead of http and M/Monit's TLS port. In addition, we specify the FQDN (Fully Qualified Domain Name) of the Monit host which match the certificate. We also add *SSL enable*, *pemfile* to the *set httpd* statement to enable TLS in Monit. For more information on [how to setup Monit to use SSL] please see the Monit wiki. How to setup *M/Monit* to use TLS will be addressed shortly.

```
set eventqueue basedir /var/monit/ slots 1000
set mmonit https://monit:monit@192.168.1.10:8443/collector
set httpd port 2812 and use address host1.example.com
    SSL enable
    pemfile /path-to/monit.pem
    allow localhost
    allow 192.168.1.10
    allow admin:secret
```

After you have changed Monit's configuration, you will need to reload or restart Monit. Monit should now start sending messages to M/Monit and automatically register itself, that is, create a host entry for itself in M/Monit's database. At this point, you should be able to see the Monit host in M/Monit's web interface.

### Monit ID

Each Monit instance is identified by a unique id, stored in the file, $HOME/.monit.id on the host running Monit. $HOME is the home directory of the user running Monit. This file is automatically created at startup by Monit if it does not exist. Care should be taken not to delete or change this file as the ID is used to pair messages from Monit with a host entry in M/Monit.

If you want to place the id-file in another location other than the default, move the id file to its new location and add a *set idfile* statement in *.monitrc* to specify the new location of the Monit id file.

# Configuring M/Monit

Configuring M/Monit is usually not necessary, but you may want to change the **port** number M/Monit use if you already have a server running on port 8080. In addition, you may want to setup M/Monit to use **TLS** or another **database** than the default built-in SQLite database.

## M/Monit configuration files

The configuration file for the M/Monit server can be found in the *conf* directory and is called *server.xml*. A detailed discussion about server.xml and its directives can be found in the appendix. The other configuration file in the *conf* directory is *web.xml*. This file specify mime-mappings for the application and need not be changed. The M/Monit web-application also use a *web.xml* configuration file in `docroot/WEB-INF/` This file should not be changed and comes preconfigured for the application.

## How to change the port number M/Monit listen on?

Change the port attribute in the Connector element if you need M/Monit to listen on a port other than 8080. For example to setup M/Monit to listen on port 8888 on all interfaces:

```
<Connector address="*" port="8888" processors="10" />
```

By default M/Monit binds to all network interfaces and can be reached via any address on the server it is running. If you need M/Monit to only bind to a specific interface, change the address attribute in the Connector element. For example, to setup M/Monit to only bind to 192.168.1.10 use:

```
<Connector address="192.168.1.10" port="8080" processors="10" />
```

## How to setup M/Monit to use SSL/TLS?

If you don't already have a SSL certificate, you can either purchase a SSL certificate or get a free one from https://letsencrypt.org/ or you can create a self-signed certificate yourself using OpenSSL.

Once you have your SSL certificate, copy it to M/Monit's *conf/* directory. In this example we'll use Let's Encrypt, but the approach is the same for any SSL certificate. The two files you need to copy are, *privkey.pem* which contains the private key for the SSL certificate and *fullchain.pem* which contains your SSL certificate and the certificate of the Certificate Authority that signed the server certificate, in that order.

Open *conf/server.xml* in your favourite text editor and set the *secure* attribute in the <Connector> element or uncomment the TLS Connector example:

```
<Connector address="*" port="8443" processors="10" secure="true"/>
```

Scroll down and change the *name* of the <Host> element to match the Common Name (domain name) in the certificate. For example, let's assume you have a certificate for the domain name, *example.com* and the host is registered in DNS with an A-record for the same name. Set the <Host> name to *example.com, the certificate* to *fullchain.pem* and the *certificateKey* to privkey.pem:

```
<Host name="example.com" appBase="."
certificate="conf/fullchain.pem"
certificateKey="conf/privkey.pem" >
```

Also remember to change the *defaultHost* attribute in the <Engine> element to example.com:

```
<Engine name="mmonit" defaultHost="example.com" fileCache="10 MB" />
```

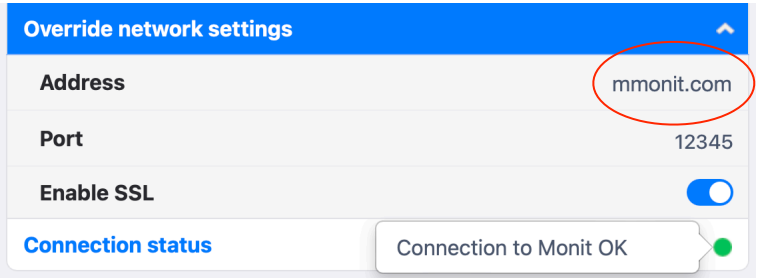
That's it. Restart mmonit and connect securely using https://example.com:8443/ (Not really, as you will have to change example.com to your own domain name, but then it'll work).

You can run M/Monit with both a TLS connector and a non-TLS connector. This is useful if you need M/Monit to listen on more network interfaces. Connections from the outside could connect using TLS, while hosts behind your firewall can connect via internal IP-addresses/interfaces using the non-TLS connector if you wish.

# SSL Certificate Verification for outgoing connections

All outgoing SSL connections made through M/Monit will verify the remote host's SSL certificate. This verification enforce two constraints: the remote certificate cannot be self-signed, and the connection will fail if the hostname in the certificate does not match the address M/Monit connects to. If you rely on a self-signed SSL certificates with Monit or if the certificate's hostname does not match the host, use the <SSLClientConfig> config element in *server.xml* to allow self-signed certificates and/or disable hostname verification.

When communicating with Monit and if you have an SSL certificate for Monit, but the Address in Admin/Host is an *IP address* or does not match the hostname (CN field) in Monit's SSL certificate, you should override the hostname that M/Monit will use, in *admin/hosts*.




Use the DNS name of the host, which should be the same name as the CN field in Monit's SSL certificate. Connection from M/Monit to Monit will then ensure that the certificate hostname matches the given hostname. This is an essential check to prevent man-in-the-middle (MITM) attacks.

Hostname verification is good practice when setting up SSL, but the level of secure data transport guarantee you want to configure is up to you. As mentioned above, the <SSLClientConfig> element can be used to lessen these constraints.

The address mentioned above is reported by Monit. To ensure the address is a FQDN (fully qualified domain name) and not an IP address, you can, alternatively to overriding the network settings, instead specify the hostname (FQDN) in the 'set httpd port 2812 and use address mmonit.com' statement in your monitrc config file. Here we use *mmonit.com* instead of the IP address of the host Monit is running on.

## How to setup M/Monit to use MySQL, MariaDB or PostgreSQL?

M/Monit comes bundled and configured with SQLite as its database system. No extra setup is required. You may use MySQL, MariaDB or PostgreSQL instead if you want. The `migrate_db.sh` script in the *db* directory can be used to move your SQLite data over to MySQL or PostgreSQL later if you started with SQLite.

Setting up M/Monit to use either MySQL or PostgreSQL is a simple two step process:

1. First create the M/Monit database. The database schemas with recipes for creating the database can be found in the *db* directory in the M/Monit home directory.
2. Then configure M/Monit; edit the M/Monit configuration file *server.xml* and replace the default SQLite Realm element with either this one for MySQL or MariaDB:

```
<Realm url="mysql://mmonit:mmonit@127.0.0.1:3306/mmonit"
       minConnections="5"
       maxConnections="25"
       reapConnections="300" />
```

   Or this one for PostgreSQL:

```
<Realm url="postgresql://mmonit:mmonit@127.0.0.1:5432/mmonit"
       minConnections="5"
       maxConnections="25"
       reapConnections="300" />
```

   Change username, password, host and port number in the connection URL as required.

M/Monit is known to work out of the box with PostgreSQL 9.x or later, MySQL 5.x or later and MariaDB 5.x or later

If you have a later database version and are experiencing problems, you can download and build the *libzdb* connection pool library used by M/Monit with your database version. Simply replace the libzdb shared library in *mmonit/lib/libzdb.x* with your own built version. Libzdb is our own open source library and can be downloaded from libzdb's website.

## How to increase the login session timeout in M/Monit?

When you log in to the M/Monit web-application a new session is created. The login session will timeout if there has been no activity for a specific time and users must log in again after a timeout. The session timeout value is specified in the <Context> element in the *sessionTimeout* attribute. The default timeout is 30 minutes. To increase this, for instance to one hour, use *sessionTimeout="1 hour"*.

If you click the "Keep me signed in" control in the login screen, a persistent session is created instead. This session will timeout after 3 months and will be stored in the M/Monit database. All sessions in current use will survive a mmonit server restart.

## How to install the license key?

M/Monit comes with an evaluation license which will expire after 30 days. You can purchase a license online which does not expire. The license key will be sent in an email. Replace the existing license in `conf/license.xml` with your new key and restart M/Monit. If you go to the M/Monit admin page you can see more information about your new license and how to contact us if you need support.

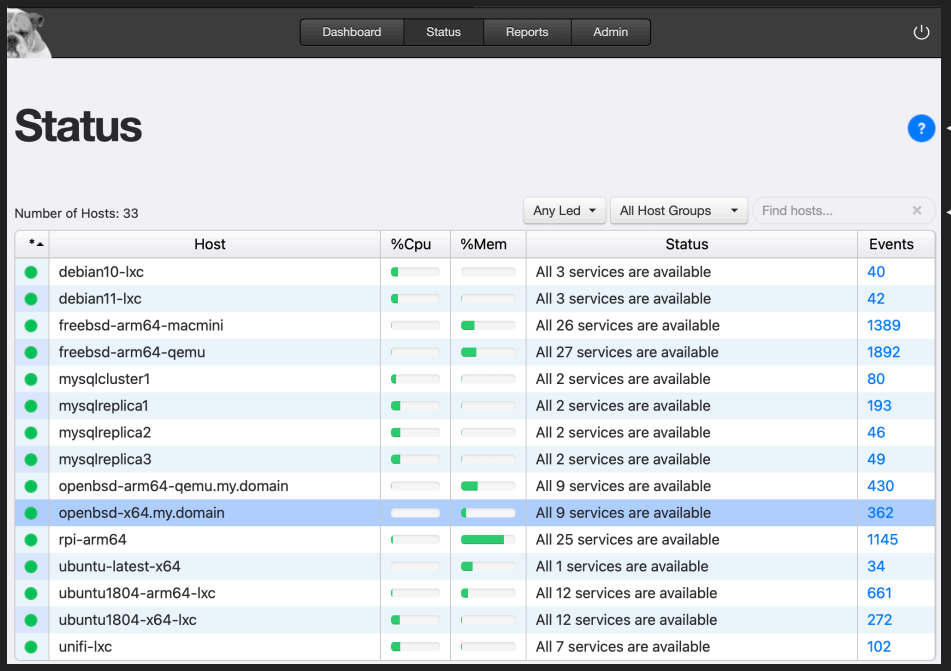


# A First Look at M/Monit

**Dashboard** click the logo to go directly to the dashboard

**Logout** from M/Monit

**Navigation tabs**
Access Dashboard, Status, Reports and Admin




**Help**

**Drill-down menu**
Use the drill-down menu to drill down into data displayed in the table

**Tables** are used many places in M/Monit to display data. Click a column header to sort rows in ascending or descending order. Move the mouse cursor over the table and the active row under the cursor is marked in blue. To see entry details, click the row.

# Login to M/Monit

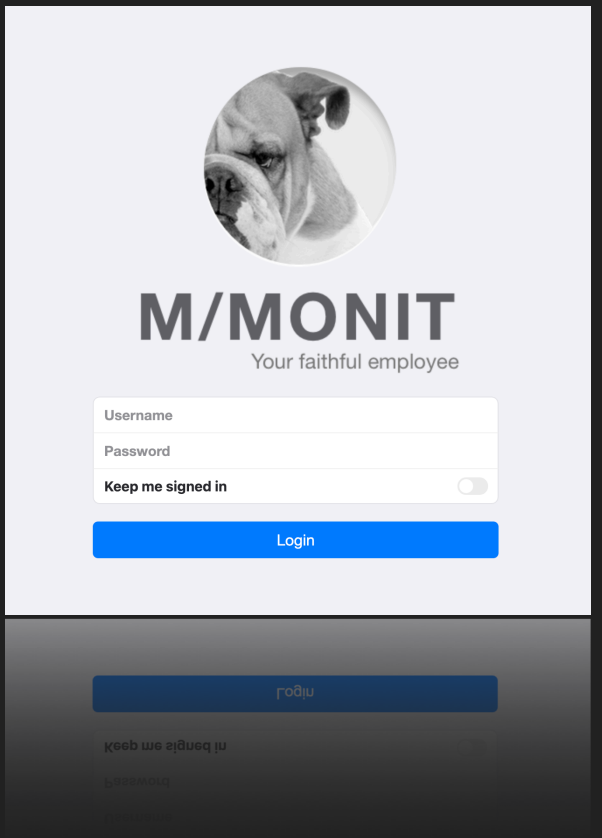
After starting the mmonit program, point your Browser to http://localhost:8080/

M/Monit comes pre-installed with two user accounts you can use the first time to login:

| User | Password |
|---|---|
| admin | swordfish |
| monit | monit |

The admin user is a member of the administrator role and has access to every page and functionality in the web-app. The monit user is a regular user and cannot access the admin pages.

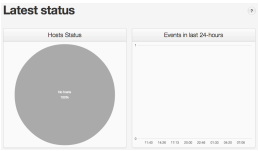
Enter admin and password swordfish and click the blue login button. The *Keep me signed in* switch can be used for persistent login, if checked, a login session is created with timeout set to 3 months and the session is also stored in the database.



# Dashboard




The dashboard is the first page you will see after login. This page provides a quick overview of the status of all hosts monitored by M/Monit and events coming in from the last 24 hours.




The first time you log in to M/Monit, the dashboard will look like the screenshot to the left. Since no Monit hosts have been registered, the pie chart will be grey and the events chart will be empty.

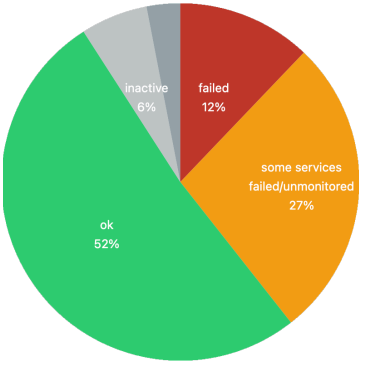
Later, when Monit is setup and start to report in, this page is going to be useful for getting a quick status overview of all your hosts. The charts refresh themselves automatically each 5 seconds.

## Host Status

The Host Status pie chart display the status of all hosts registered in M/Monit and if all hosts and services are online, the pie chart should be all green.

Here are the possible pie chart colors and their meaning:

**Host is offline**
**Host with all services offline**
**Host with some services offline or in unmonitored state**
**Host with all services online**
**Ignored host**
**Inactive host**



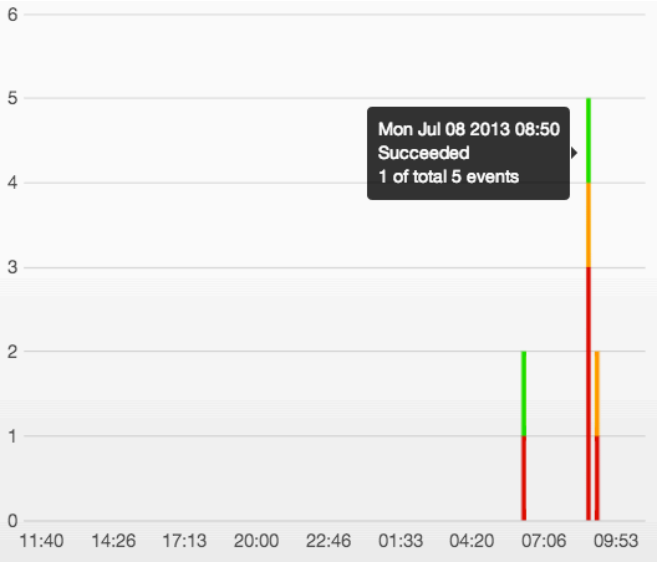
A click anywhere in the pie chart will open the Host Status page. From here you can drill-down on each individual host and see detailed information about the host and its monitored services and also start, stop or restart services.

## Events in last 24-hours

This chart plot events coming in from Monit over the last 24-hours. Events are plotted from right to left. That is, the most recent events are plotted rightmost in the chart. This is a stacked bar chart and each bar represent an aggregate of events in a 10-minute time frame. The chart plots the following colorful events:

| | |
|---|---|
| green | Success, a previous reported error was fixed |
| red | Failure, a service failed |
| orange | Changed, such events are sent when Monit start/stop, when an action is performed, or when a service fire a change event. For instance if a file checksum or a process pid changes. |

If you hoover the mouse pointer over a plot-point, a popup window is displayed with event information as indicated in the picture to the right. Click on a plot-point to open the events log.



## Messages

A list of system messages, active errors and warnings is displayed beneath the charts.

Monit will try to fix a problem when it arises, but issues that have not yet been fixed or cannot be fixed by Monit are listed here. In many ways this list represents your TODO list.

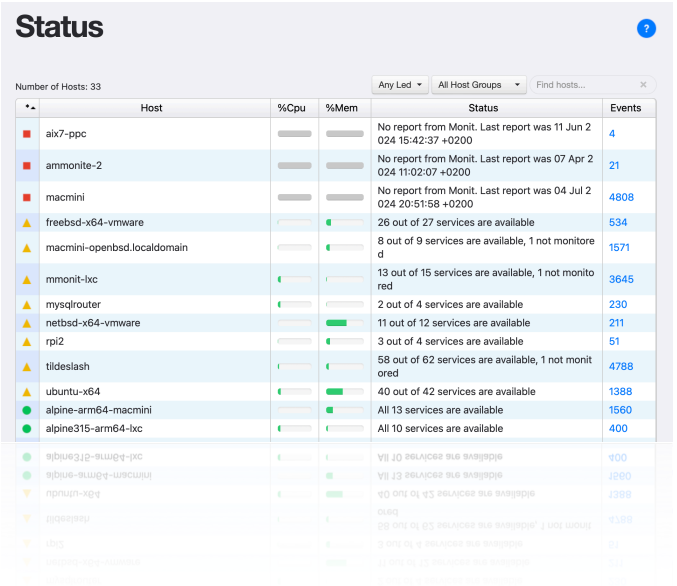
Should an issue be resolved later it is automatically dismissed from this list. You can also dismiss the issues in the list by clicking the **x** in the upper right corner of the box. Dismissal is per user and does not affect the list of other users. You can always see the full list of events (including dismissed and resolved events) in the Event Log.

[Mar 24 2023 16:19:51] Host **netbsd9-x64** reported a problem with **wm0-ipalias**: total upload 200.0 MB matches limit [upload rate > 200 MB in last 2 hour] ×

[Mar 16 2023 13:03:18] Host **tildeslash2n** reported a problem with **vmnet8**: total download packets 900504 matches limit [download packets > 900000 in last 1 hour] ×

[Mar 16 2023 11:37:25] Host **freebsd-x64** reported a problem with **monit**: No report from Monit for 1866310 seconds: the network, the host or Monit is down. Last report was 22 Feb 2023 21:12:15 +0100 ×

# Status




The Status page shows the status of each monitored host based on periodic reports from Monit. The table auto-refresh itself each 5 seconds. Update of new data also depends on how often Monit runs on each host. I.e. the lower the poll-cycle you run Monit with, the closer the host's status will be displayed in "real-time".

A status table row has the following columns:

| | Host | %Cpu | %Mem | Status | Events |
|---|---|---|---|---|---|
| ● | alpine-x64 | | | All 2 services are available | 137 |

The first column displays a LED representing the host's error state:

- ■ Error. Either the host did not report in or all services are down
- ▲ Warning. At least one service is down or in unmonitored mode.
- ● OK. All services are up and running
- ◆ Host is in Ignored or in Inactive state.

The **Host** column displays a descriptive name of the host. The name can be changed in the Admin-Hosts page.

The **%Cpu** and **%Mem** columns displays CPU and Memory usage respectively. The color of the bar depends on the column value: **green** if less than 80%, **orange** if less than 95% and **red** if above 95%, **grey** if no data is available. To display the current value, hover the mouse cursor over the bar.

The **Status** column displays the status of the services in textual form. Finally, the **Events** column displays the number of events related to this host. Click the link in the events column to open the events log and see events for this host.

You can sort the table by clicking a column header. By default the table is sorted on the LED column which is useful to quickly see the hosts with errors or warnings first. To see which hosts are under heavy load, sort the table by %Cpu or %Mem instead.

The drill-down menu above the table can be used to filter the table by LED, by host group or by hosts that match a name or a name fragment. Click a table row to display detailed host status.

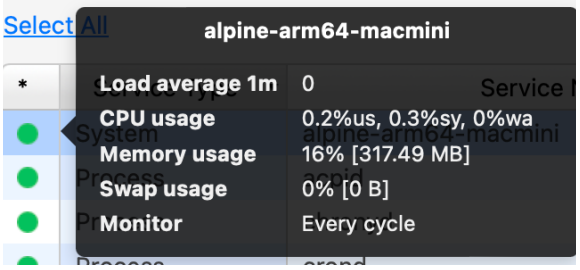
# Detailed host status

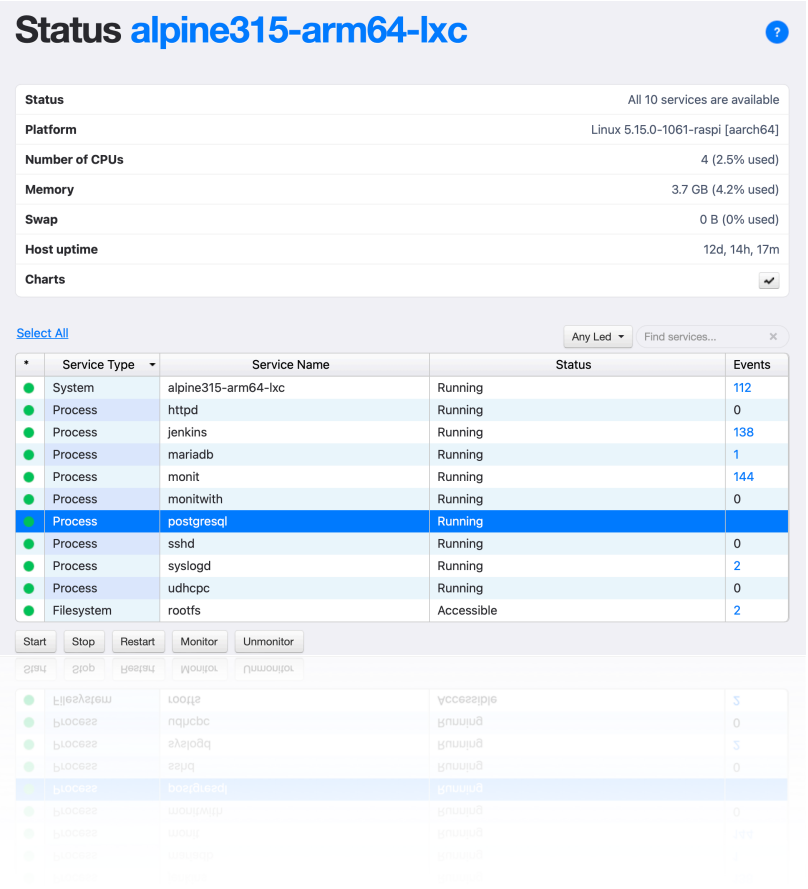
This page displays detailed status of the host and its monitored services.

Table data is based on the last report from Monit and refreshed automatically. The refresh frequency is dependent up on the Monit poll cycle settings and a low poll cycle provides closer to real-time data than a longer poll cycle.

## Service Details

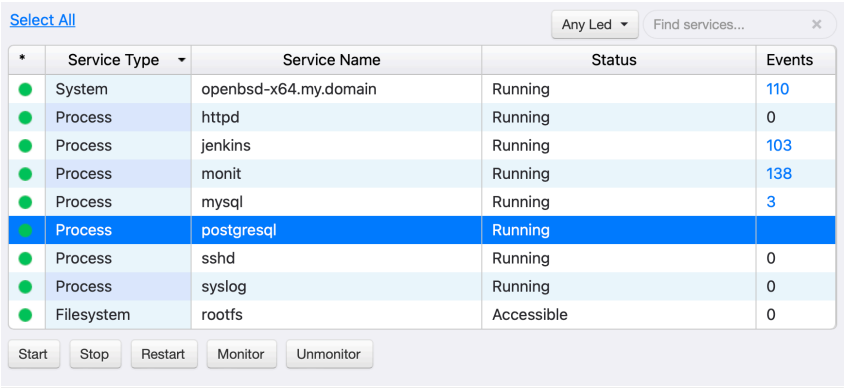
If you hover the mouse cursor over a LED light (or tap the LED on a touchscreen) a callout is displayed with detailed informations about that service:




You can *scroll* the content of the window horizontal, which is useful on a phone where you might not see the full content. If the callout does not show, it means that Monit is either not monitoring this service or the service is being initialised.

## Service actions




Select a service by clicking a table row. Multiple rows can be selected or you can select all rows by clicking the [Select All] link. To unselect rows, press *ESC*.

Once a row has been selected you can use one of the action buttons to start, stop, restart, monitor or unmonitor services on this host.

*Note: the start, stop and restart actions require that the service has a start and a stop program registered in Monit.*

M/Monit delegates execution of actions to Monit and waits for up to 30 seconds on confirmation from Monit before timing out the action. Actions that have been executed are logged in the Events Report page.

Only users with the admin role or users in a role with the *Service Action* permission are allowed to perform an action on a service. If you are *not* such a user, action buttons are not displayed. Likewise, buttons are not displayed if the host is offline.

*Service action requires a connection to Monit. The information required for connecting to Monit from M/Monit is automatically set by Monit when it registers itself in M/Monit. In some cases, for instance, if there is a NAT or a proxy between Monit and M/Monit it might be necessary to modify the configuration manually in the Admin/Host page.*

## Host charts

Click the chart button to go to the analytics page and display real-time charts for the host.

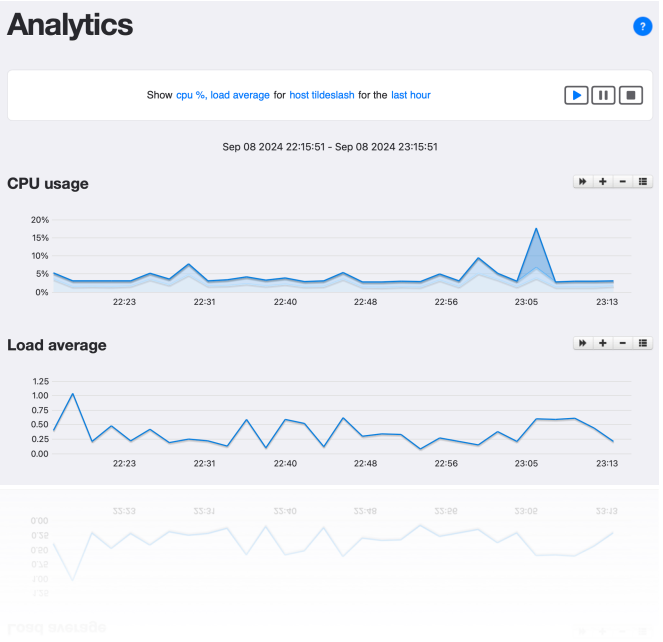
# Analytics Reports

M/Monit collects key performance data from hosts and their monitored services. From this page you can run queries and investigate statistical data.

Charts with data from multiple hosts can be created so you can compare metrics from various hosts in the same chart.

The legend table displays minimum and maximum values in the given range as well as the change rate.

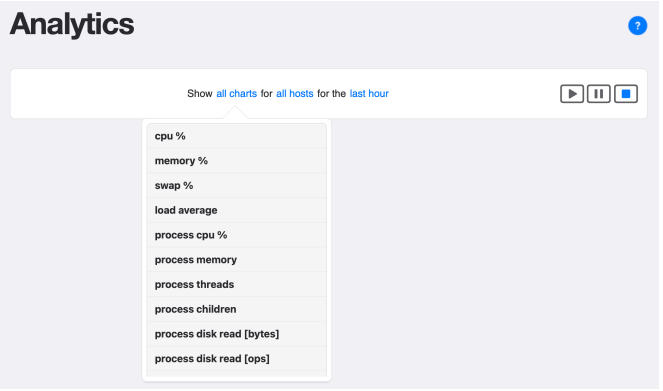
Trend prediction can be used to extrapolate future values based on existing data.




## Analytics Query

Charts are built by selecting chart types, hosts or host groups and a time range. Charts controls:

1. The ▶ button will show charts. The chart will automatically refresh, based on the data sampling rate.
2. The ▮▮ button will stop automatic reload. Press ▶ button to continue.
3. The ■ button will clear charts and reset the filter.




Selected hosts are displayed in the same chart for easy comparison - for example if you select the [*space %*] chart and [*all hosts*], you can see all monitored filesystems in one chart. You can then compare space usage across the whole selection and identify underutilised or saturated filesystems.

The underlying JSON chart data sent from M/Monit to the Browser can become quite large if you monitor many hosts and/or services. If charts starts to become noticeable slow we recommend that you select specific hosts or host groups instead of generating charts for “all hosts”.

## Chart types

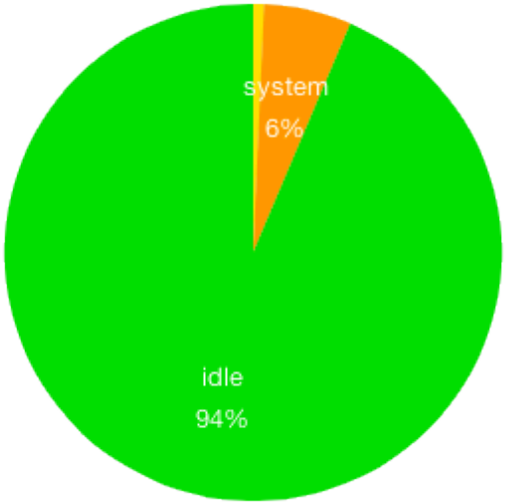
The type of charts that are used depends on the selected time range and on the number of selected hosts. A variety of bar charts, pie charts and line charts are used in different contexts.

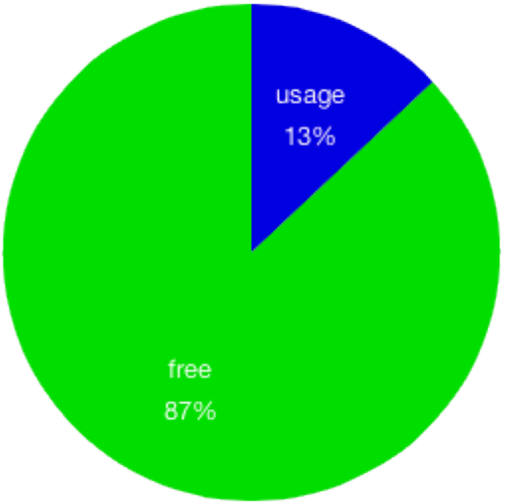
The [last seconds] time range is special and shows the last collected data sample, at maximum one minute old. To see these “real-time” charts, Monit must run with a poll-cycle which is less than one minute, otherwise these charts will not be generated, except if you happen to draw within the minute data was collected.

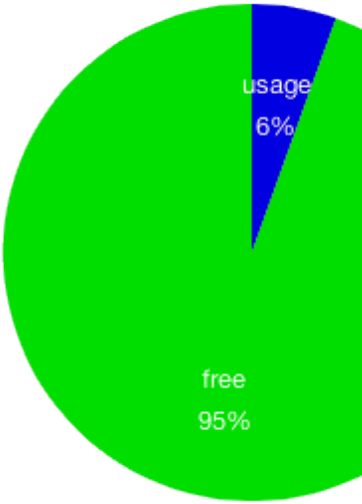
Some charts such as the [cpu %] chart will aggregate data if several hosts are selected and displayed in the same chart:

- If a single host is selected, cpu usage is displayed as a stacked area chart with cpu usage broken down by user%, system% and wait% values
- If more hosts are selected, the chart instead display total cpu usage per host (the sum of user%, system%, wait%)

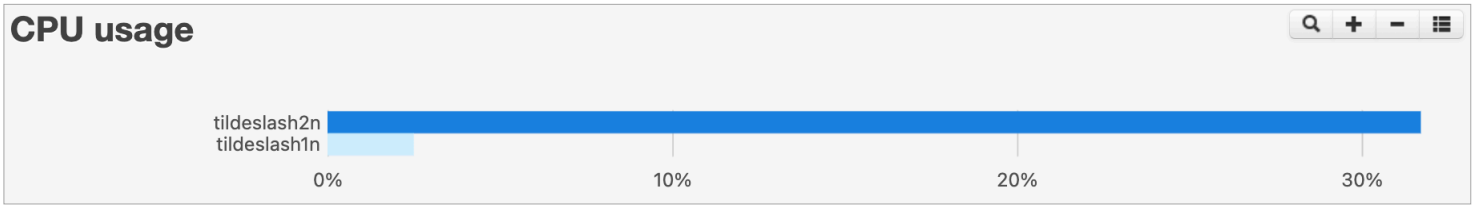
Datasets with zero values are not displayed in the charts.



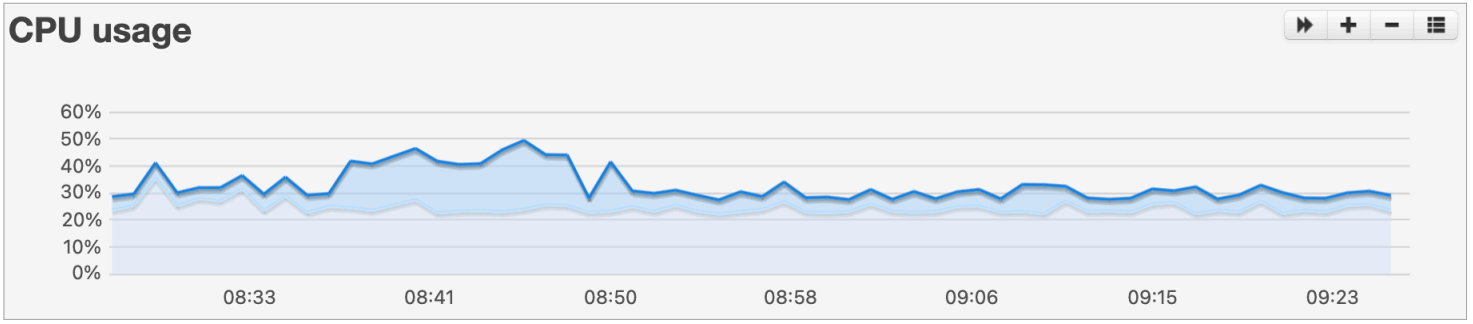



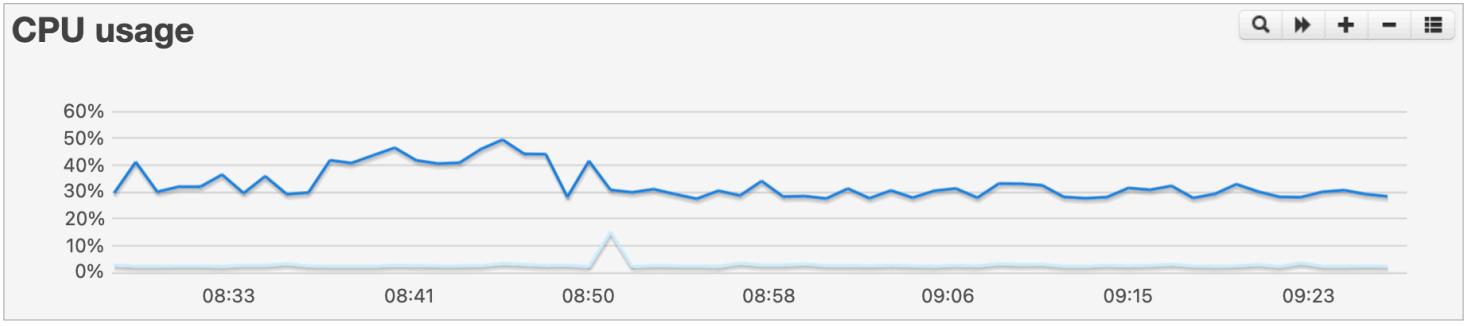
Bar chart example: [last seconds] range for multiple hosts




Stacked area chart example: [last hour] or higher range for a single host




Line chart example: [last hour] or higher range for multiple hosts

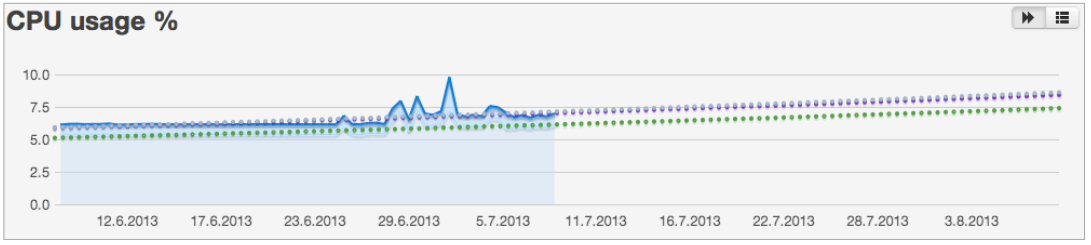


## Chart buttons

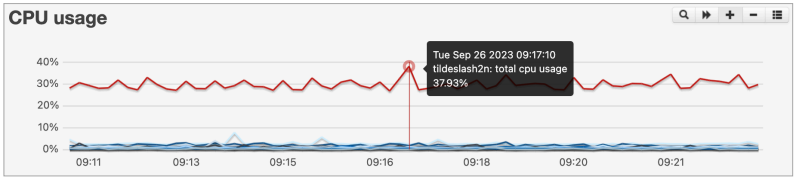
**Search** Use the search button to filter data in a chart and only show entries that match a search term. For example, if you have Apache running on several hosts and want to compare cpu utilization, select the [process cpu %] chart and search for Apache.

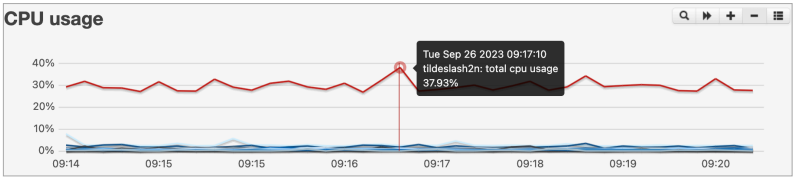



**Trend** Use the trend button to display a trend line and predict the future! The method we use is either a linear or an exponential extrapolation based on a best-fit. A trend line is a stippled line that projects forward in time by extending the selected time range twice. For example, if [month] is selected, trend will show an estimate for next month. Note that predictions may be incorrect if data does not follow a linear or an exponential trend. In general, a longer time range is more likely to produce a better estimate.




**+ and −** for zoom in and out




**Legend** The legend button displays the chart's legend in addition to minimum, maximum and average values and the delta for the given range. The [seconds] range is an exception and only display the current value since a minute is the lowest resolution.
Items can be removed from the chart by clicking the legend's color box. If a trend line is displayed, the legend table will also contain estimated future values.

| | Min | Avg | Max | Change |
|---|---|---|---|---|
| user | 0.06 | 0.27 | 3.32 | +0.04 |
| wait | 0.12 | 0.17 | 0.24 | -0.01 |
| system | 0.05 | 0.1 | 0.34 | -0.02 |

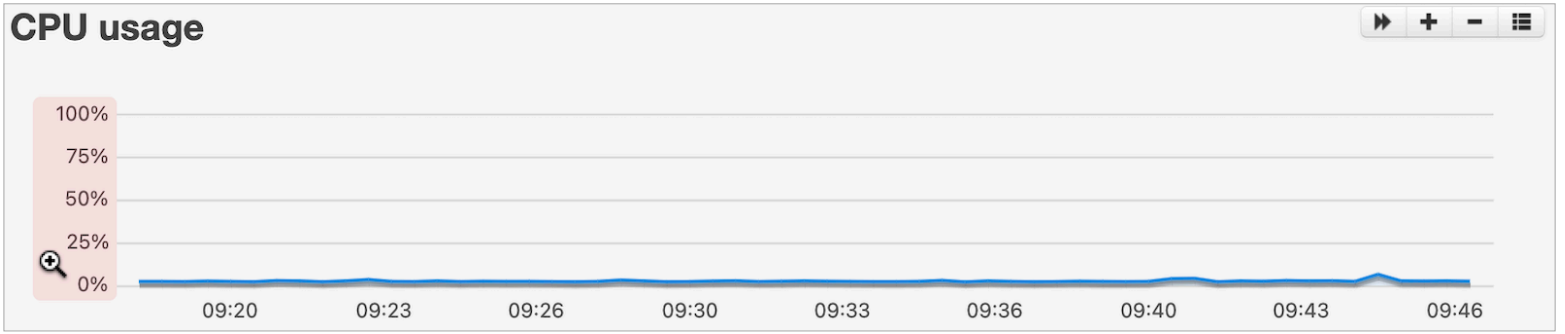
## Chart refresh

The charts refresh automatically. The refresh rate depends on the sampling rate of the selected range (e.g. the “last hour” will refresh once per minute).

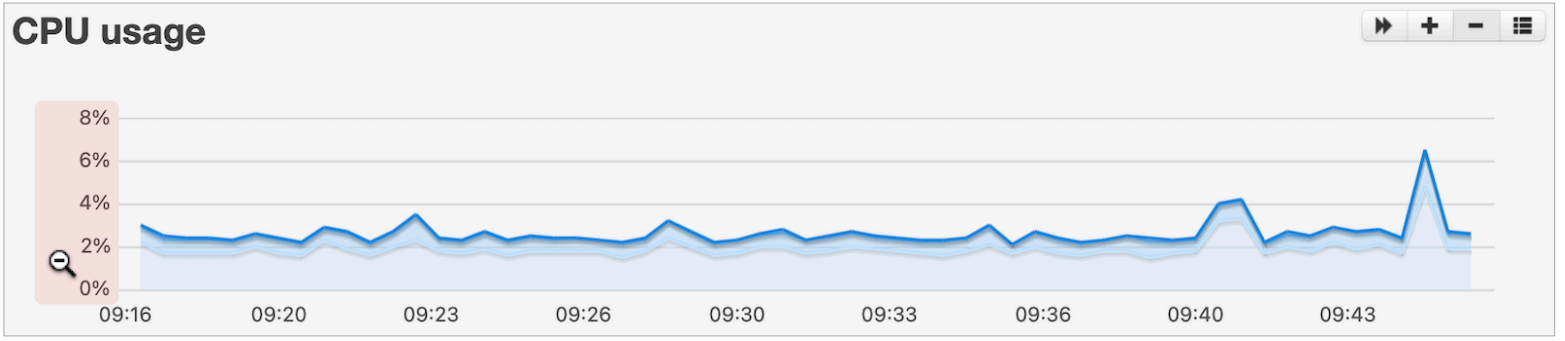
## Scale and zooming

Charts that show percentage, use an Y-axis from 0 to 100. If you hover the cursor over a Y axis, the cursor changes into a magnification glass. Click, and the chart will autoscale the Y-axis to zoom in. Click the Y-axis again to zoom out.
You can also use ✚, ▬ buttons to zoom in and out the whole selection

The following is a chart with the default scale from 0 to 100% for the Y axis:

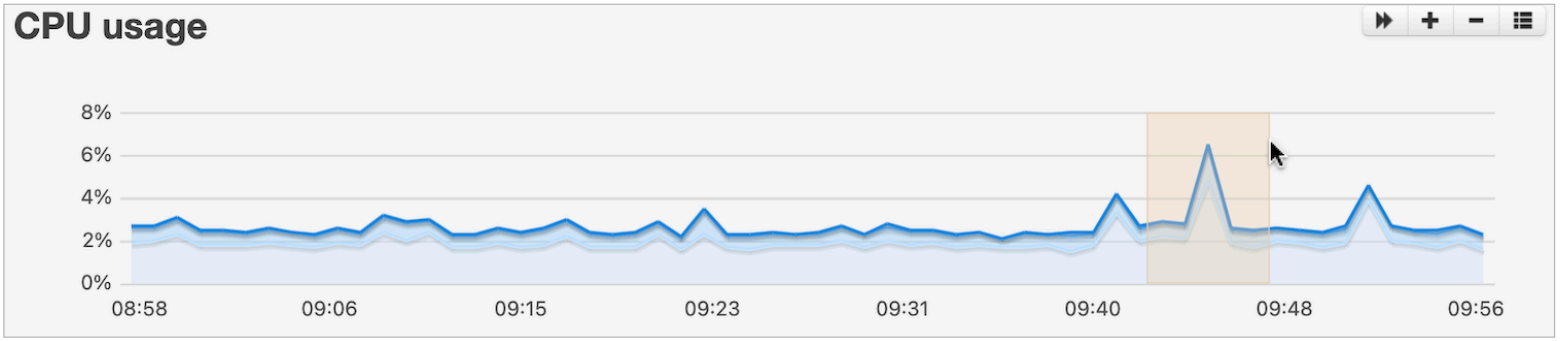



And here is the same chart after clicking the Y-axis and with autoscale from 0 to local maximum%:



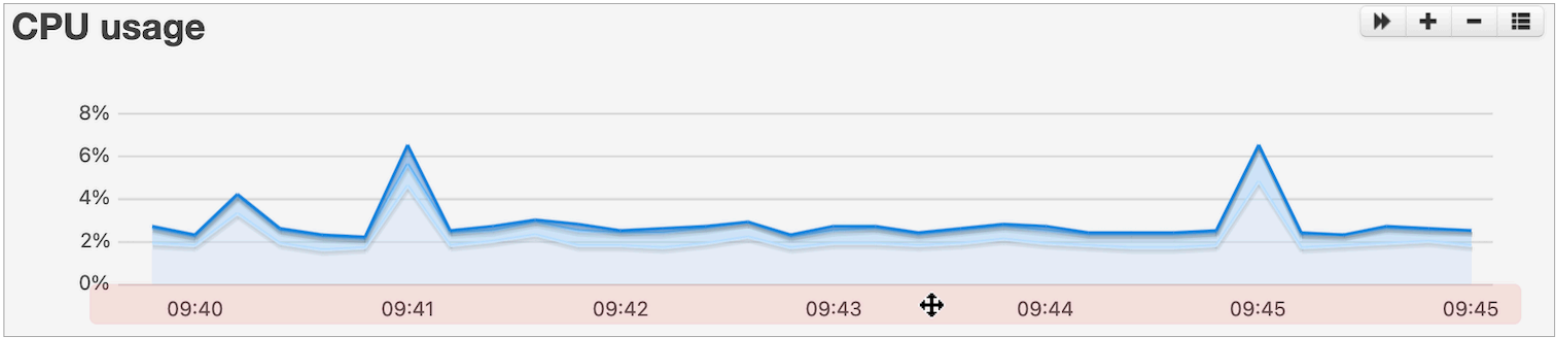
## Custom selection

Click and drag in the chart to zoom and make a custom selection



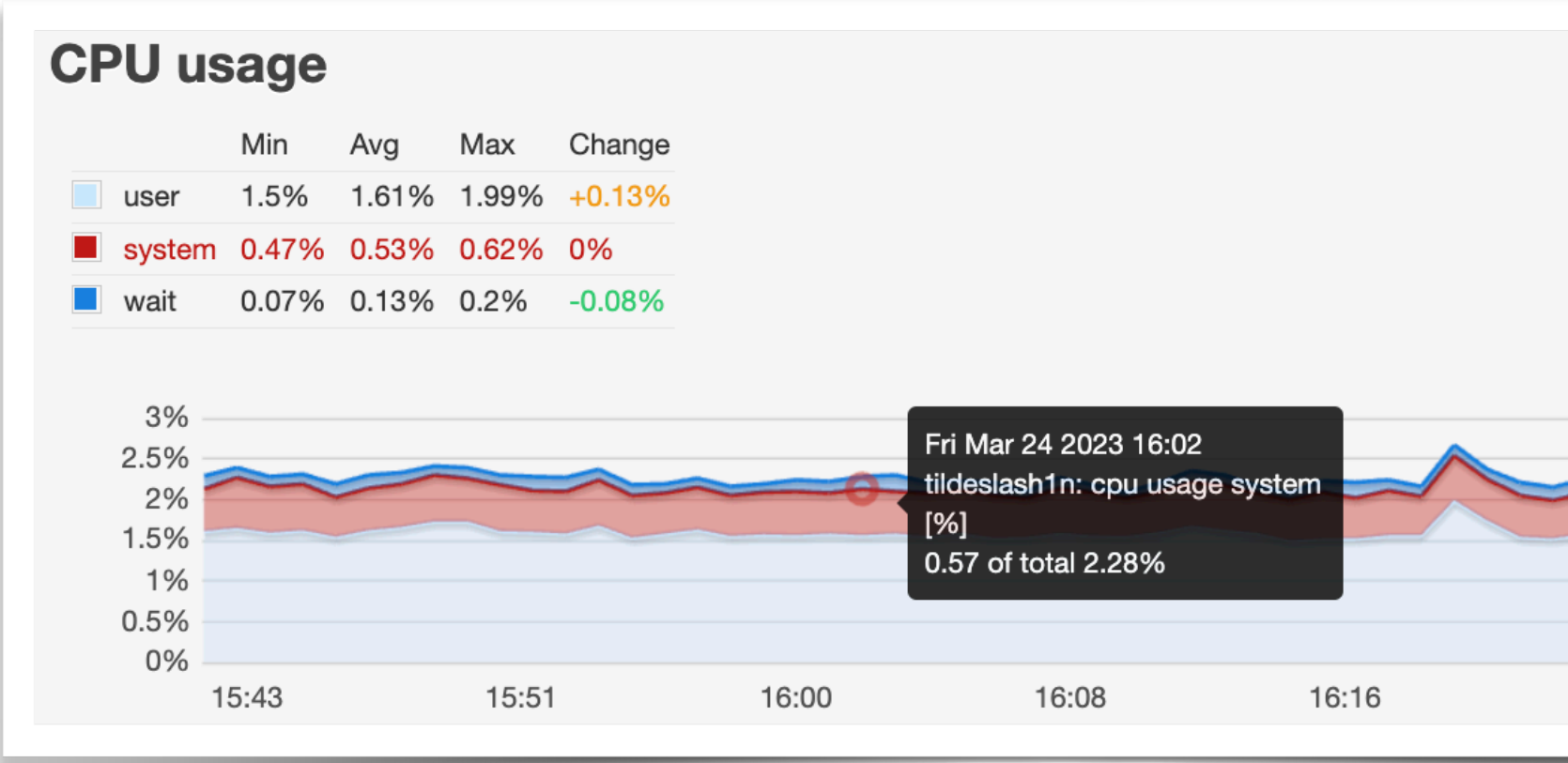

## Panning

To pan the chart, move your cursor over the X-axis and drag it to the left or right. Note: The panning to the right is limited by the current time. (When panning, data is fetched in the background for all charts for the range, click-drag-hold and release when you see charts refresh might make drawing less jittery with many charts and/or hosts)



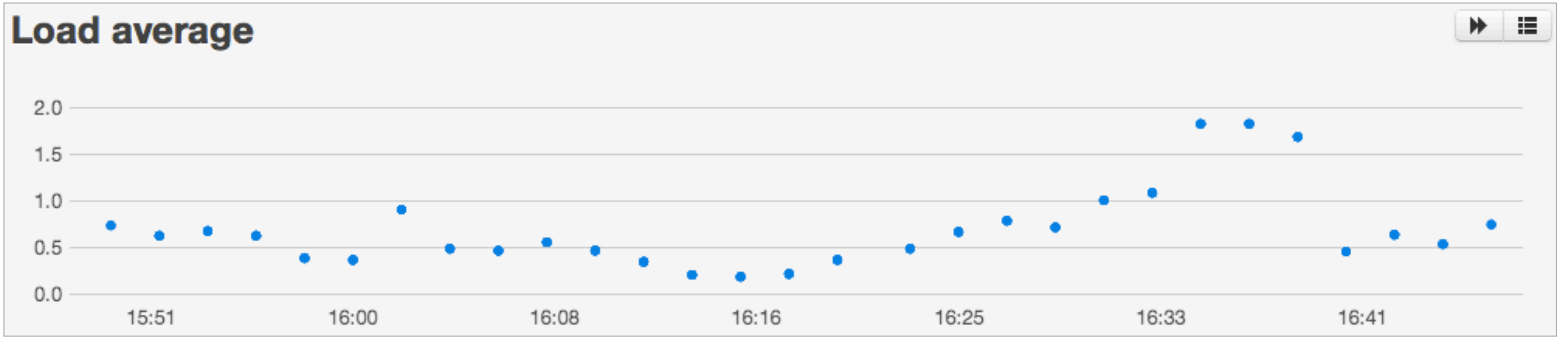
## Tooltip and series highlighting

If you tap or hover over a chart, a tooltip is displayed with a point-in-time value. If the legend table is displayed, the corresponding legend will be highlighted as well.



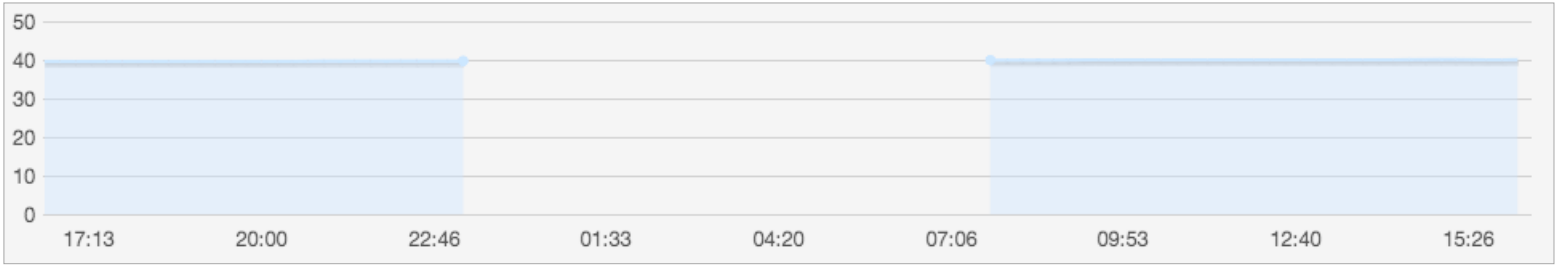
## Monit poll-cycle settings and data granularity or dots vs. lines

Monit send data samples to M/Monit at the end of each poll cycle. If Monit's poll-cycle is longer than 1 minute, charts with [hour] range will show data points instead of a continuous line. The reason for this is that the [hour] charts draw with the lowest resolution or precision, which is 1 minute. To show a continuous line, Monit must have a poll-cycle which is less than 1 minute (50sec max). The next chart resolution is 15 minutes which is used for charts in the [day] range and as long as Monit's poll cycle is less than 15 minutes, charts in this range will display a continuous line. See the table below for chart precision and how data is aggregated.



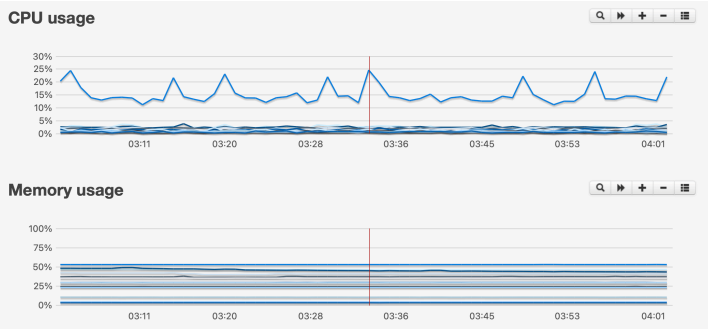

## Data gaps

If Monit stops sending status updates or if M/Monit is stopped, you might see gaps in the charts.  Such gaps represents "no data available".



## Data correlation

M/Monit shows a red vertical line across all charts, to make it easier to correlate various metrics in time.




## Data compression

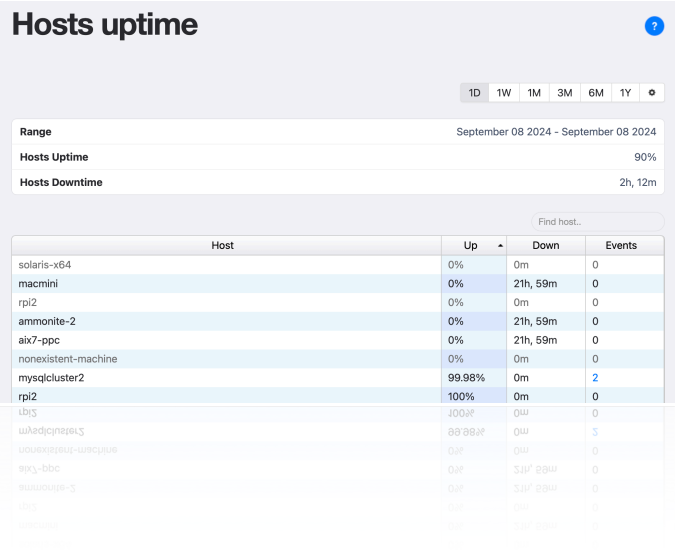
Data is collected and aggregated by M/Monit for best memory and space use and display speed. Historical data is stored for up to three years before being automatically deleted.

The following table show how data is aggregated and the default precision used for the different ranges.

The custom range selection enables users to drill down with precision higher than the default.

| Range | Precision |
|---|---|
| Last Seconds | Absolute value (last data) |
| Last Hour | 2 Minutes |
| Last Day | 1 Hour |
| Last Week | 4 Hours |
| Last Month | 1 Day |
| Last 3 Months | 3 Days |
| Last 6 Months | 1 Week |
| Last Year | 2 Weeks |
| Last Three Years | 1 Month |

# Uptime Report




The overview report show uptime and downtime of all Hosts that are monitored by M/Monit. Hosts with downtime include hosts that are actually down, the network from M/Monit to the host is down or hosts where Monit was not gracefully stopped.

If Monit does not send a status message within a specific time (3 Monit poll cycles, by default) M/Monit will assume the host is down and raise a status failed event. This event is associated with the "monit" service and can be seen in the events log and in the status page as "No report from Monit. Last report was...". When Monit start sending status messages again, M/Monit will raise a status succeed event. This can be seen in the events log as "Monit status report received successfully".

Host uptime is based on both succeeded and failed status events. Resolution is in minutes and the combined downtime for a host must accumulate to at least one minute for the host to appear with downtime in the table.

## Selecting date range



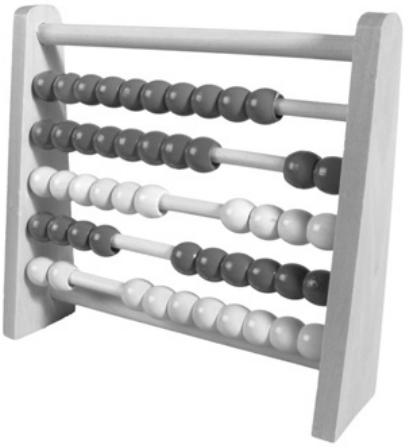

Choosing **1D** sets the range from midnight today until now. **1W** sets the range from 7 days in the past until now, **1M** is one month with blast from the past and so on. You can select custom range by clicking on the **gear** icon.

## Average uptime and downtime

The average uptime and downtime of *all* hosts are displayed in the overview table. Inactive or ignored hosts are not included in the average unless they have downtime.

$$\frac{1}{n} \cdot \sum_{i=1}^{n} u_i$$

## How uptime/downtime is calculated:



1. If a matching failed and succeeded status event is found within the selected date range, the difference in time between the two events is counted as downtime.

2. For open ended failed status events, that is; failed events without a matching succeed event, the downtime is counted as the difference between *now* and the failed status event's timestamp.

3. If a host has not reported within the selected range, the host is assumed to be down if and only if the host is active and its last updated timestamp is older than the selected range's from-date. In this case the difference in time between now and the range's from-date is added to the host's downtime. I.e. the host is down the full range.

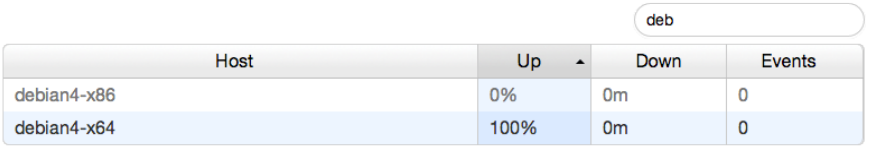
4. Inactive and ignored hosts are not included in the total average numbers unless they have downtime. Their uptime/downtime is set to zero if they have no downtime within the range. You can see inactive and ignored hosts in the table with a grey name.

## Events

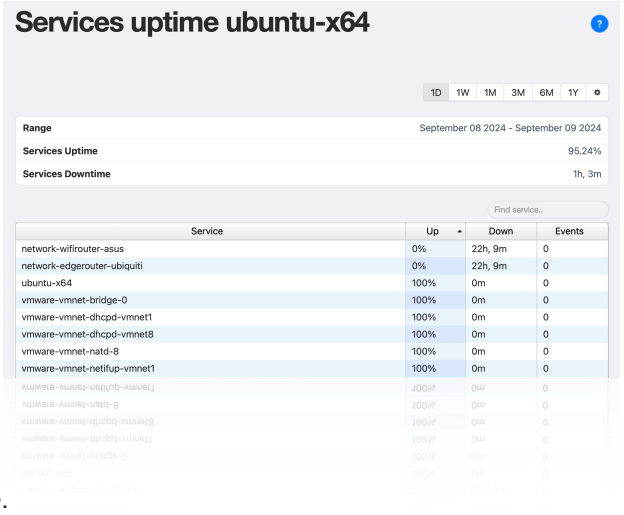
The events column show the number of status events for the host *within* the selected range. Clicking the number brings up the events log with drill-down filters preset to only show events for the host within the selected range and sorted on date in ascending order.

## Host filter

If you monitor more than 15 hosts, a host search field is displayed at the top of the table. The search field can be used to focus on selected hosts only. The table remains filtered even if the date range changes. To reset, remove all text in the search field.



# Services Uptime Report




If you click a host entry in the Host uptime report, the service report is displayed with uptime and downtime for services monitored by Monit on that host.  As with the Host report, the uptime is based on both succeeded and failed events. Resolution is in minutes and the combined downtime for a service must accumulate to at least one minute for the service to appear with downtime in the table.

The uptime and downtime of services are calculated similar to the Host report;

1. If a matching failed and succeeded event is found within the selected range, the difference in time between the two events is counted as service downtime.
2. For open ended failed events, that is; failed events without a matching succeed event, the service downtime is counted as the difference between now and the failed event's timestamp.

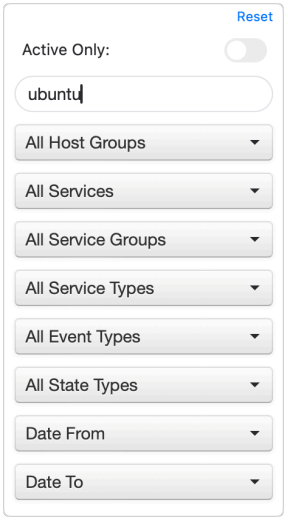
β

This report works around the fact that Monit does not maintain services (failed) state over a Monit restart/reload nor during service un-monitoring. If the host stops sending status to M/Monit, the downtime is added to the Monit service only, as it is unclear whether the host is completely down or if the network connection to M/Monit is down. This may add some inaccuracy to the computed downtime, though in practice it should work well. However, because of these exceptions this report is a beta and the numbers should be seen more as informative than authoritative.

## Greyed out  entries

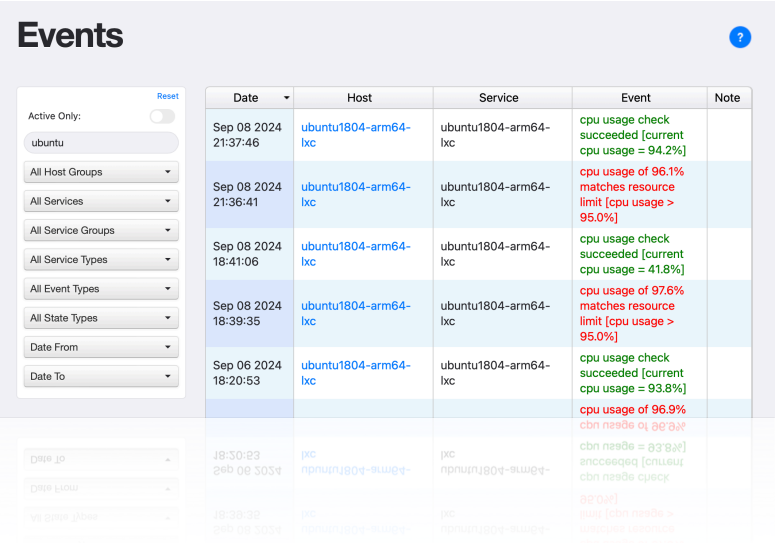
1. If the Host is inactive or ignored all services in the table are grey.
2. Services that are in an unmonitored state are marked as grey. Their uptime/downtime is set to zero unless they have actual downtime within the selected date range. Unmonitored services are not included in the total average numbers unless they have downtime.
3. Services that once existed in your Monit control file (.monitrc) but since have been removed may show up if the selected range is wide enough. These historical services are greyed out.

# Events Report




The Events log can be used to browse all events reported by Monit and stored in M/Monit's database.

The log is initially sorted on date in descending order. I.e. the latest events are listed first. Internally, M/Monit buffer new events in an event queue and flushes new events to the log every 5 seconds.




Use the drill-down menu on the left to filter the log on various criteria. One useful criteria is date. For instance, you can show events for a certain date or within a certain range by using the Date From and Date Until calendar buttons.

Hint: If you click the month in the calendar you can quickly navigate by year or month.

If you enter a Host name in the search field, only events for that host are displayed. Furthermore, only services for that host are displayed in the [All Services] drop-down, and [Service Group] will only display Service Groups, if any, for the Host.

Values set in the drill-down menu are *persistent* across page loads. Use the Reset link to reset all drill-down values.

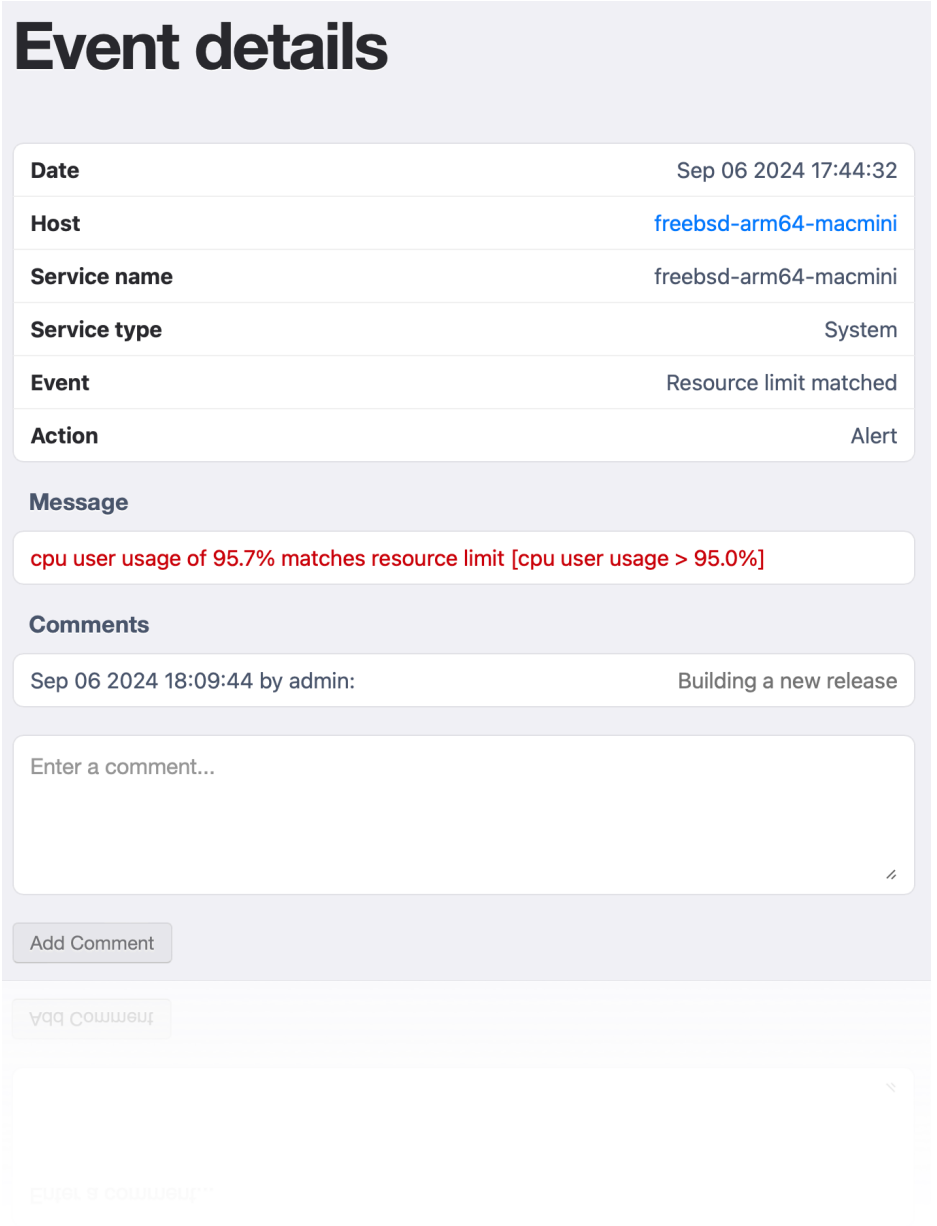
Click an event row to show details for the event.

If the *Active Only* switch is turned on, only active failed events are displayed. That is, failed events that do not have a corresponding succeed event. This list is the same as the *Messages* list displayed in the dashboard, except dismissed events in the dashboard are also shown here. If the table is empty, then all is good, since it means that there are no unsolved failed events in the system.

## Event details

In addition to viewing a few more details you can also add comments. Events with comments are marked in the table with this icon

Comments can be used for tracking events handling, explain the root cause of the problem, consequences, solution and serve as knowledge base.



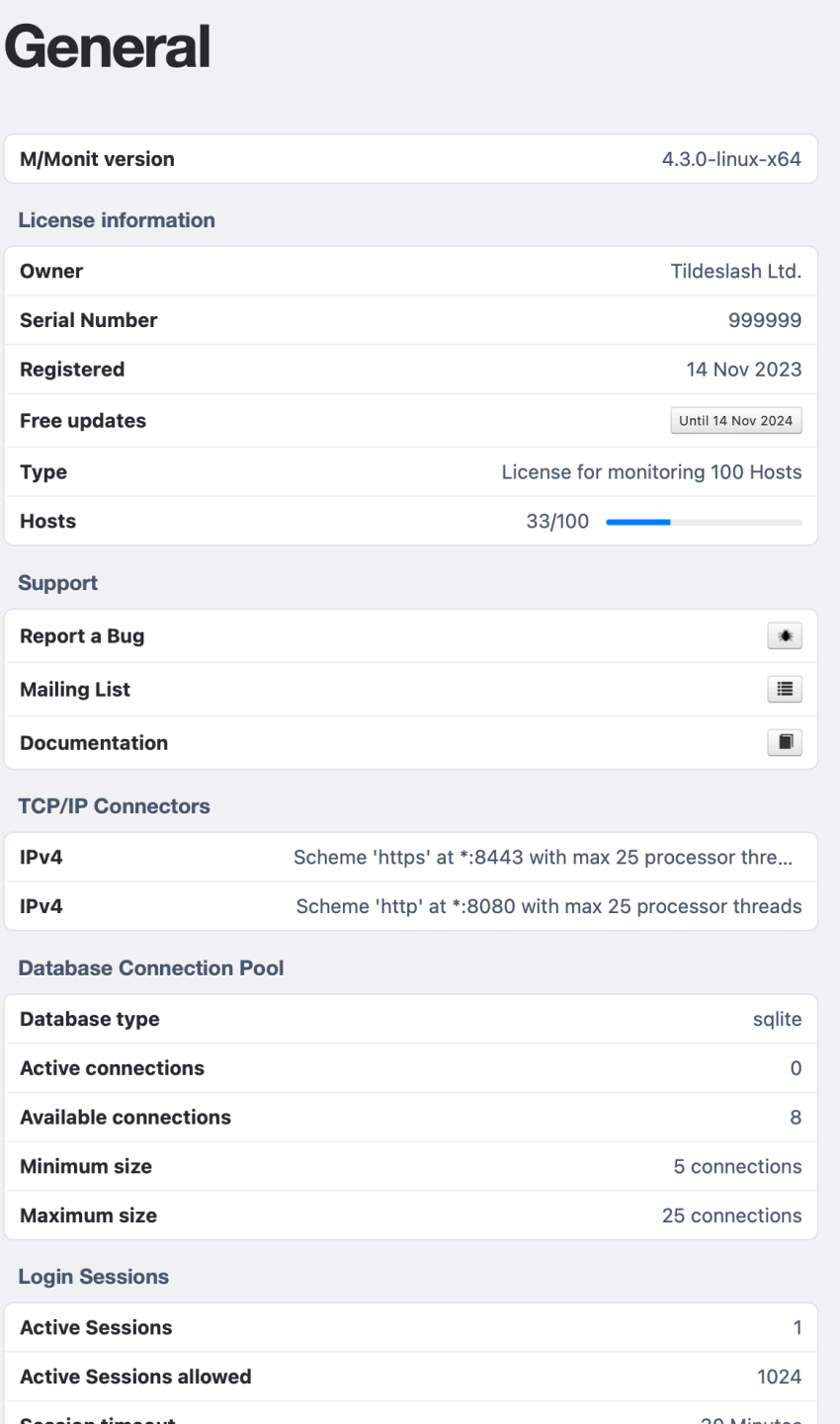
# Admin

Admin functionality is split over several pages, accessible from a submenu.

*Only users in the admin role have access to the admin pages.*

The first page display system, support and contact information as well as configuration from *server.xml*.

If M/Monit is used with an evaluation license, the expiration date of the license is displayed in the Admin Overview page, otherwise, your own license information is displayed, such as the license serial number and the license owner. This information should be provided when requesting support so we can prioritise your request.



# Hosts

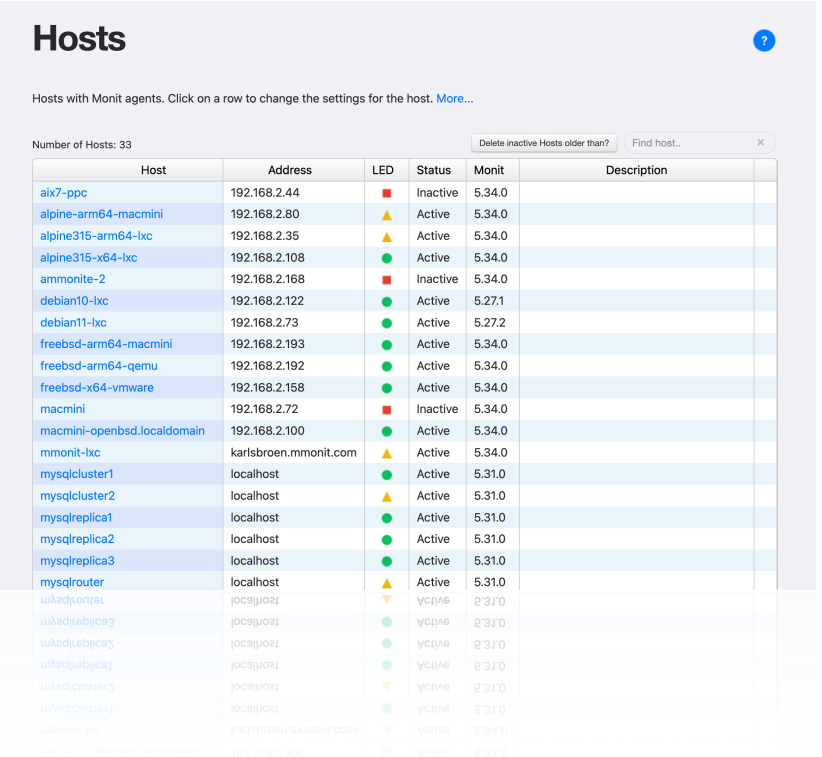
The hosts page lists all hosts registered in and managed by M/Monit.

M/Monit will *automatically* create a new Host entry on the first report from a new Monit host.

To make Monit start reporting to M/Monit, use the set mmonit statement in your Monit configuration file.

After Monit has registered itself in M/Monit you can edit the Host by clicking its Host entry in the table.

The “Find host..” search field above the Host table can be used to quickly filter the table on a host name. You can also use the search field to find hosts you want to delete. If search produces a result, a delete button appears. Click the button to delete all hosts present in the search result.




## Remove inactive Hosts

Inactive hosts are hosts where Monit has been gracefully shutdown or hosts that have been turned-off and now reports a heart-beat failure to M/Monit (because Monit was not gracefully stopped). Such hosts are typical virtual host which you spin-up in the cloud with Monit installed and then take down later.

A *delete inactive hosts* button will appear above the Hosts table if inactive hosts exist in the system. Click the button and select between different intervals for how long ago Monit last reported in to M/Monit. Let's say you select 12 hours. Then, all Hosts that reported into M/Monit more than 12 hours ago will be removed.

## Edit host

The Host Name field is a descriptive name which is used in various places in M/Monit to refer to the Host.

The first time Monit reports in, a new Host entry is created and the Host Name is set to the hostname reported by Monit.
If the hostname later is changed in DNS the host name will automatically be updated in M/Monit after Monit is restarted on the given host.

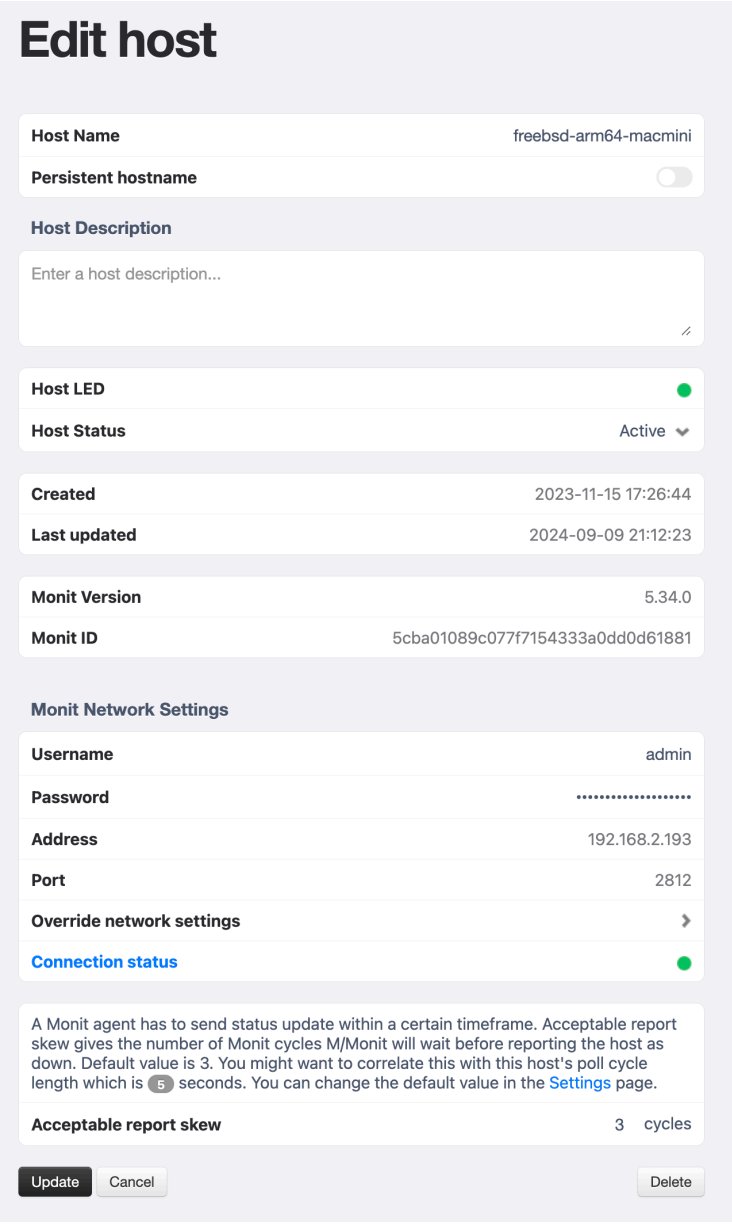
If you change the host name yourself, the *Persistent hostname* switch is turned on and Monit will no longer overwrite the host name automatically. This can be useful if you want to use a descriptive name for the host, such as "database server #1". To reenable automatic host name updates, just turn off the *Persistent hostname* switch.

### Monit ID

Each Monit instance is identified by a unique ID, by default stored in the file `$HOME/.monit.id` on the host running Monit. Where $HOME is the home directory of the user running Monit. This file is automatically created on Monit startup if it does not exist.

Care should be taken not to delete or change this file as the ID is used to pair messages from Monit with a host entry in M/Monit.

If you need to place the id-file in another location other than the default, move the id file to its new location and add a `set idfile` statement in the Monit configuration file, specifying the new location of the Monit id-file.
A new host is automatically created in M/Monit by a status message from Monit if and only if the ID embedded in the message is new and unique, otherwise M/Monit will use the ID to find the host in its database and update its status.



## Host Status

A Host can be in one of three states:

- **Active** When Monit starts, it activates itself automatically in M/Monit by sending a start event message. M/Monit expects periodic status reports from the active hosts. If the host does not send a status report within a certain time frame it is marked as down and an alert is generated by M/Monit.
- **Inactive** If Monit was stopped normally, it automatically deactivates itself by sending a stop event message to M/Monit.
- **Ignored** An Ignored state can be used to manually and temporarily suppress all events from Monit. For example, if you are going to perform maintenance work on the host and you expect Monit to issue lots of alerts, set the Host in the Ignored state to ignore the events sent by Monit for the duration of the work.

## Network settings

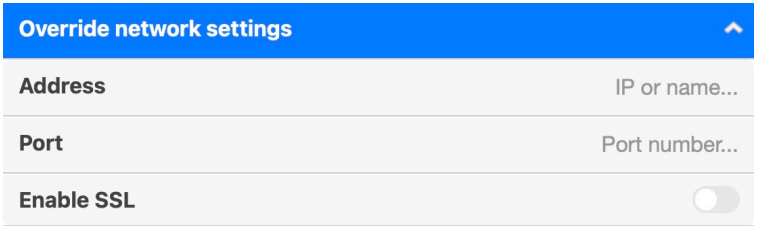
The *Monit Network Settings* fields, specify how M/Monit should connect to Monit. This information is required if you want M/Monit to manage services on the host. That is, if in the host's status details page, a click on an action button should have an effect.

Monit *username* and *password* are credentials used for Monit authentication.

The *address* is either the IP-address or the hostname of the host running Monit. M/Monit will use this address to connect to Monit. The address is read-only and managed by Monit. Dynamic changes (such as DHCP lease) will automatically update the host settings in M/Monit. If Monit is using SSL and the address is an IP-address and not a DNS name (FQDN) then see *Override network settings* below.

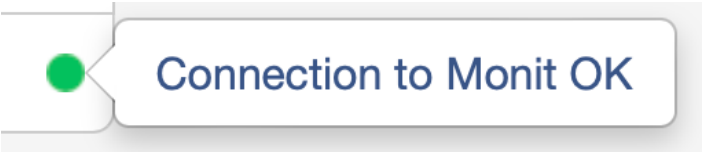
Monit *port* number is the HTTP port the Monit agent is setup with (2812 by default). The port number is also read-only and automatically updated by Monit.

**Override network settings:** If the Monit Host is using SSL or if the Monit Host is behind a NAT or a Proxy and M/Monit must use a different IP-address and port number to reach Monit, click *Override network settings* and specify the outbound IP-address or the Host's fully qualified domain name (FQDN), port number and whether SSL should be used by M/Monit to reach Monit. If SSL is used, enter the Host's DNS name (FQDN) instead of the IP-address, otherwise the SSL certificate Host Name Verification might fail.



Override settings takes precedence and will be used instead of Monit's reported values. If Monit's configuration is changed later, you will need to update the override settings manually.

**Connection status** shows the connection status to Monit, using the current network settings. The check is performed automatically when this page opens, but you can also run the test manually by clicking the *Connection status* link.




## Acceptable report skew



Monit has to send a status update within a certain timeframe.

Acceptable report skew specifies the number of Monit cycles M/Monit will wait before reporting the Host as down. The default value is 3 Monit poll cycles.

If Monit is setup with a short poll cycle, e.g. 5 seconds or if Monit is verifiably up, but for some reasons slow to send reports it can be useful to increase the *skew* value to avoid getting false alerts because Monit did not report in on time.

# Host Groups

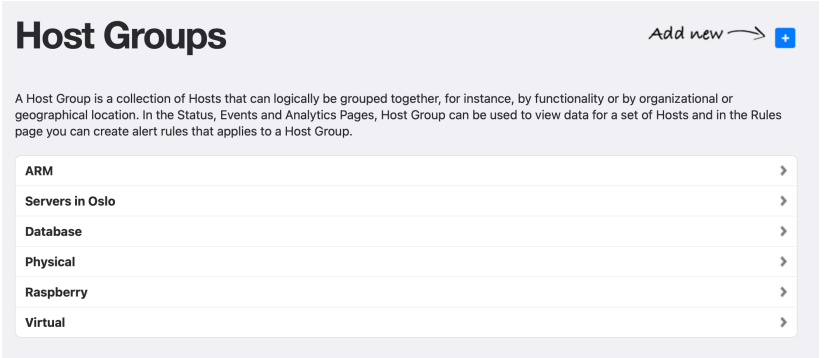
A Host Group is a collection of Hosts that can logically be grouped together, for instance, by its role or by organisational or geographical location.

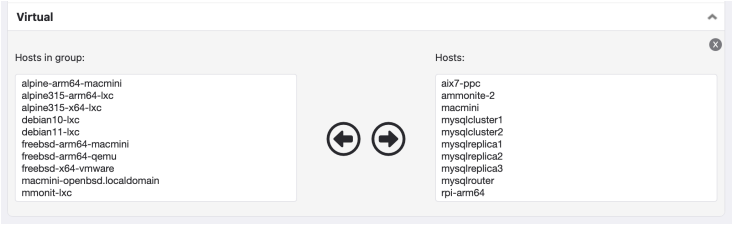
In the Status, Analytics and Events pages you can drill-down on Host Group to view data for a set of Hosts. You can also create alert rules that apply to a Host Group.

To create a new host group, click the blue **+** button. Give the group a name and click **enter**.

Click the row to expand the host group and display controls for *modifying* or *deleting* a group.

To rename a group, simply click its name, the text will switch to an input field. Click **enter** when done.
To delete a group, click the ⊗ icon.




Select hosts from the list to the right and click the left arrow to *add* hosts to the group.

Select hosts from the left list and click the right arrow to *remove* hosts from the group.

To select more than one host in the list, hold down the ALT or ⌘ key when selecting.

# Users

The Users page lists all users allowed to login to M/Monit.

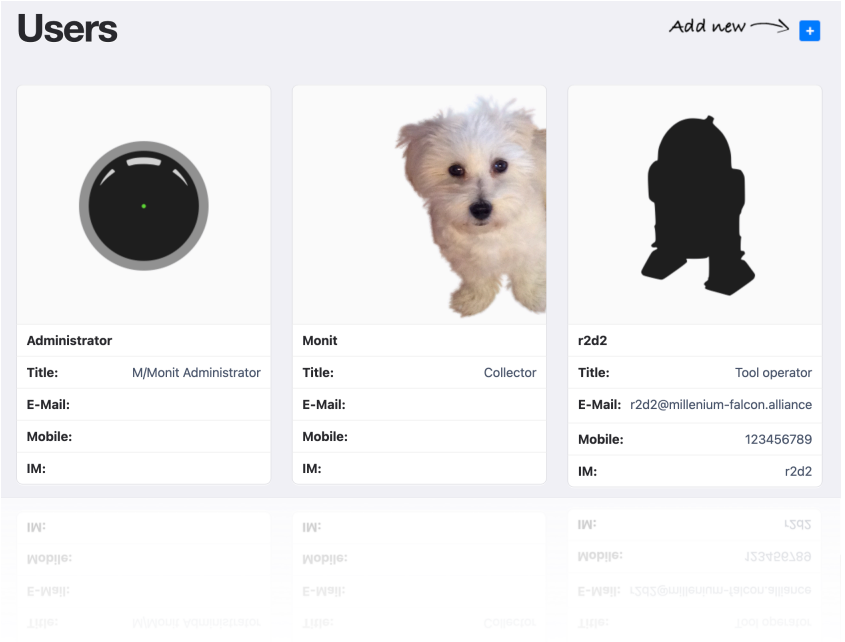
## Overview

To add a new user, click the blue **+** button. To edit an existing user, click the user's image.

M/Monit comes installed with two default accounts:

| Username | Default Password | Role |
|---|---|---|
| admin | swordfish | Administrator |
| monit | monit | Collector |

You can change or remove the default accounts, but remember to keep at least one account with administrator privileges to be able to manage M/Monit and one account for collecting data from Monit.







### Search

On a phone, use the search field at the top of the page to quickly search for a user. Search is on user's full name

## Create or edit user

A Username is required to be able to login to M/Monit. The user name can be changed if needed.

Use Roles to control access to M/Monit. For instance, if the user should have administrator access, assign the admin role. If you want the user to only have access to a select set of hosts, assign a custom role or assign the collector role if this user should be able to post events and status updates to M/Monit.

The other fields are optional, but contact informations can come in handy if more people are using M/Monit.

It is recommended to fill in Email and Jabber information so you will be able to select the user from the Alert page to receive alerts.

## Appearance

The Appearance setting allows you to choose the theme for the application. Select "Auto" to follow your system's Dark or Light mode settings, or manually set the theme to "Light" or "Dark" based on your preference. The selected Appearance is stored in your browser's local storage and will not affect other users.

## Photo

To add or change the user's photo, click the image box. This will open up a dialog from where you can upload an image from your computer. Then, click save. On your phone, click the image to take a photo or upload an existing photo from your library

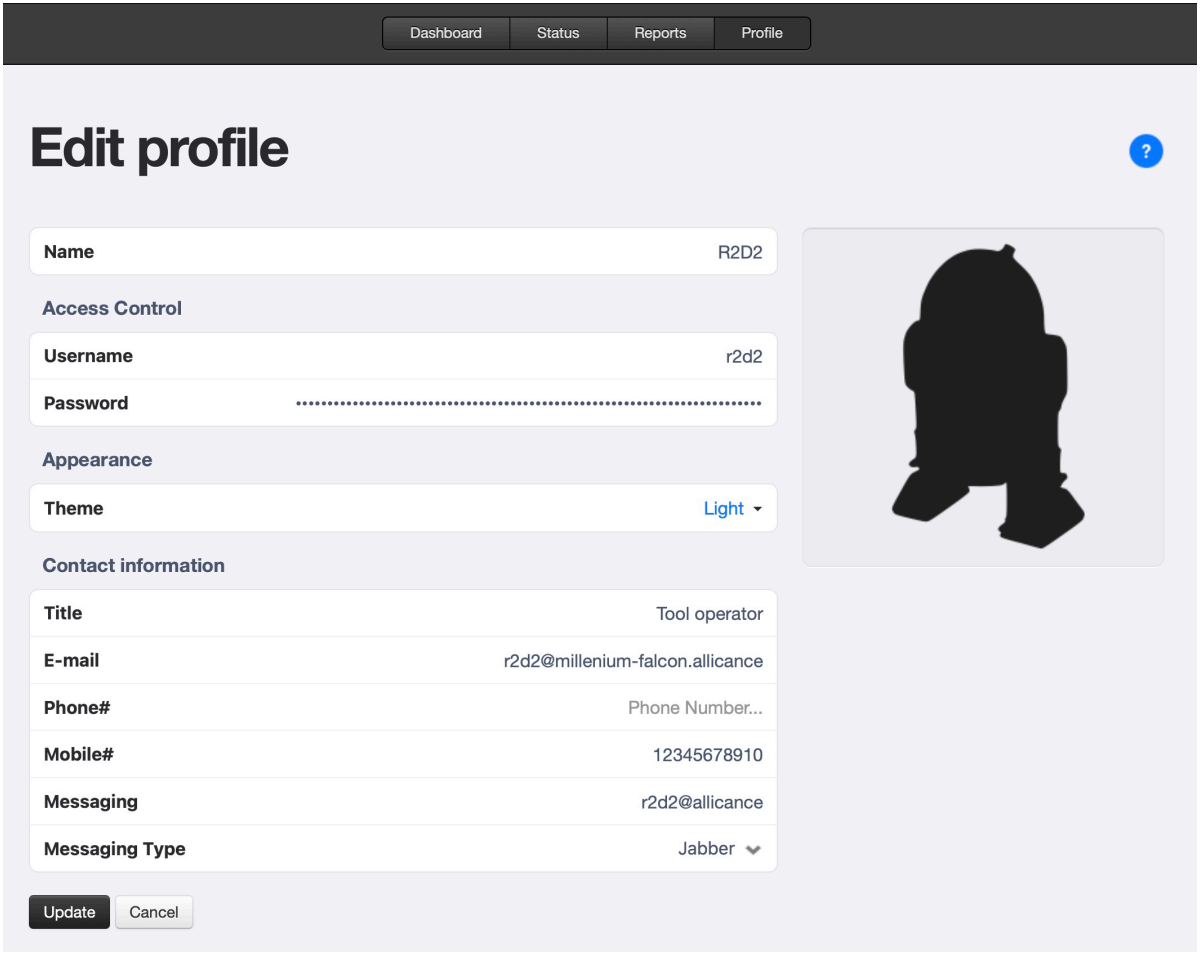
## Delete a user

To delete the user, first click its entry in the overview page and then click the delete button near the bottom of the page.

Remember to keep at least one account with administrator privileges so you can manage M/Monit 😅

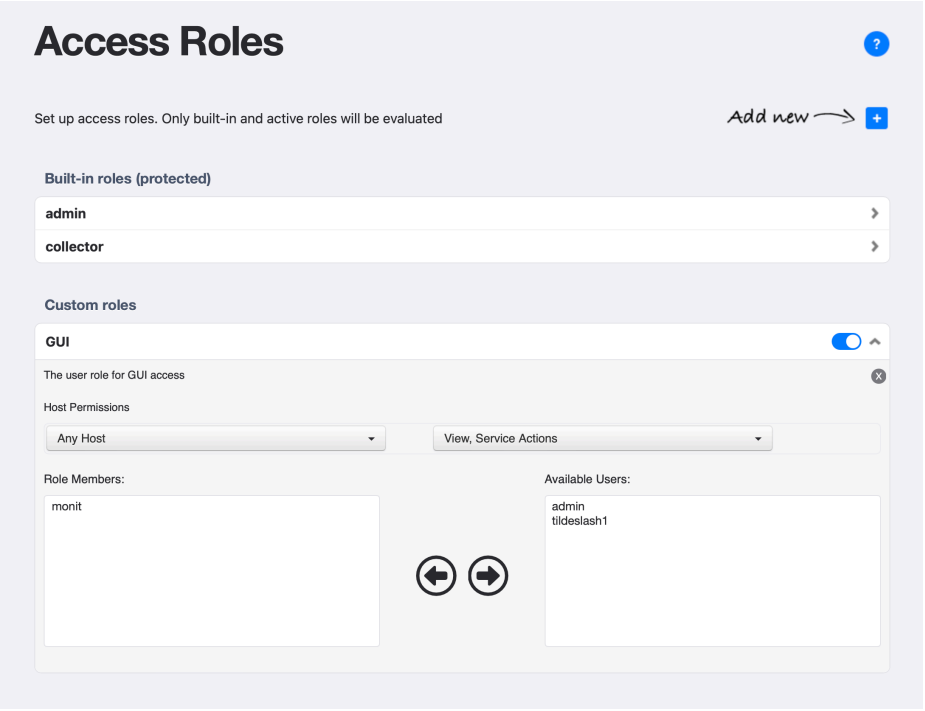
## User self management

Non-admin users cannot access M/Monit's administrator functionality nor any admin pages, but they can edit their own profile, such as update password, contact information and change theme. For these users, the **Admin** pages are replaced with a **Profile** page. Here's how the Profile page looks like for a non-admin user:



# Access Roles

Use this page to create or edit Access Roles. An Access Role is used to limit access and visibility to hosts and host groups to a select set of users.




## Overview

To add a new role, click the blue **+** button.

To edit an existing role, click on the role row to expand it.

## Role types

There are two types of roles:

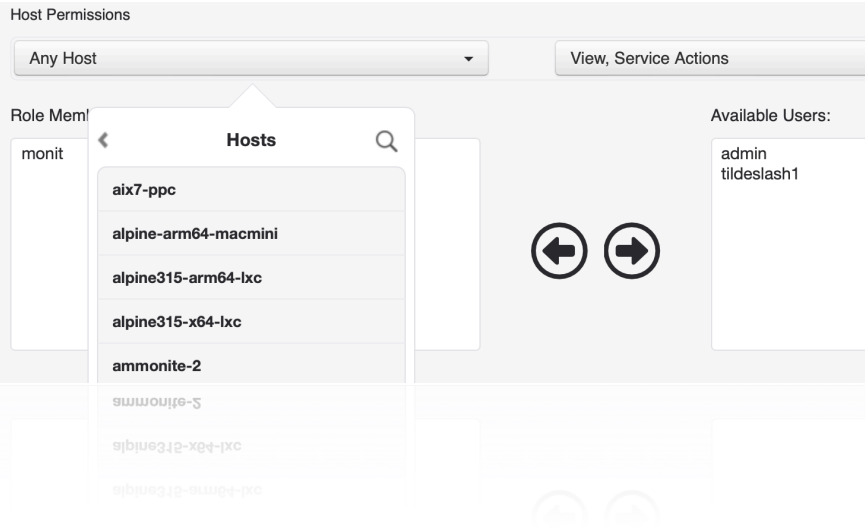
1. **Built-in:**
   There are two built-in roles, an **admin** role for M/Monit administration and a **collector** role for posting status updates and events to M/Monit. You can assign members to the role, but you cannot delete these roles.

2. **Custom**
   Fully managed by (you) the M/Monit administrator, you can create, modify or delete these roles as you see fit.

A Custom Access Role is setup by choosing a set of hosts or host groups and a set of users who should be members of the role and have access to the hosts.

## Host Permissions

Specifies hosts and host groups the user can view, and sets permissions:




- **Collector access:** Members can update hosts status and post events via M/Monit's collector. To be able to create new hosts, the role also needs the "Any Host" permission. The "Any Host" permission is automatically set if you do not select any hosts.
- **View:** allow members to view role hosts in M/Monit. If this permission is not set, members can still login to M/Monit, but they will not see any hosts nor any data. A user can access his own Profile page and change information.
- **Service Actions:** allow members to perform actions on hosts in the role, such as start, stop or restart a service, in the M/Monit host status details page

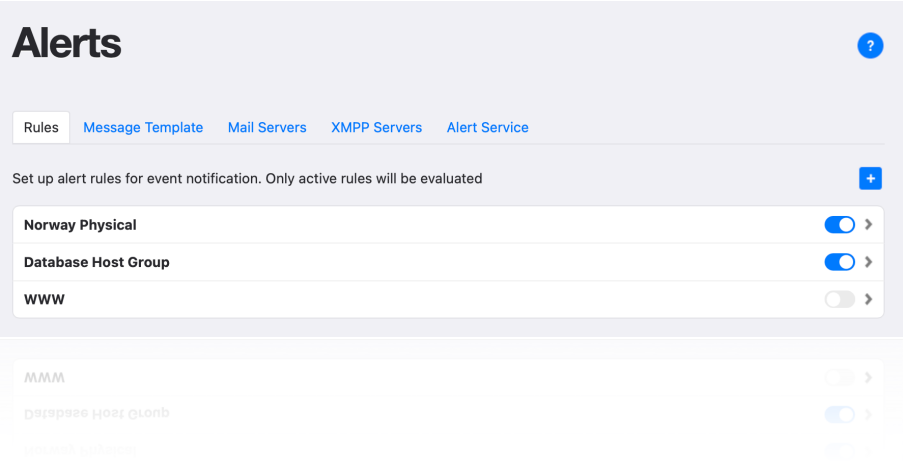
## Role Members

Assign members to the role. You can assign multiple roles to the user, permissions will be given by the *union* of all assigned roles.

## Role deactivation

You can activate and deactivate a role using the switch, next to the role name. It allows one to quickly revoke permissions with one click, without losing the role settings completely.

# Alerts




Alert functionality is spread over five tabs:

- **Rules**, setup alert notifications rules
- **Message settings**, setup the sender's email address, content format and alert aggregation
- **Mail servers**, setup SMTP server(s) for email alerts
- **Jabber servers**, setup an instant messaging server for jabber alerts
- **Alert Service**, M/Monit comes pre-configured with scripts for sending alerts via popular platforms like Slack, Discord and PagerDuty. You can also create your own custom alert scripts to meet specific requirements.

## Rules

When an event message is received from Monit, M/Monit will check the event against a list of user defined rules and if a match is found, the action defined for the rule is executed.

This page lists all rules defined in M/Monit. In the list, a switch is used to indicate if the Rule is active or not (the switch is blue when active). M/Monit will only test incoming events against active rules, *non-active rules are ignored*.

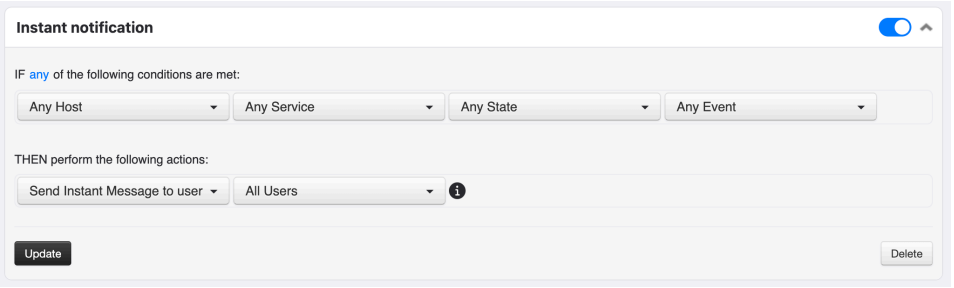
The rule list can be perceived as a sieve, events are sifted through and only events that match a rule, produces an action.

### Create or modify a rule

To add a new rule, click the blue **+** button. When the rule has been setup, click the **Save** button.

To edit an existing rule, click the rule row in the table. To change the name of the rule, click the name and edit it inline. Press Enter or click outside when done. Remember to click the **Update** button when done.

## Conditions




A Rule is specified as an *IF-THEN* statement.

That is, IF a set of conditions are met THEN perform one or more actions.

A rule evaluates to true if condition(s) matches the incoming event.

A condition is a tuple of [Host/Group, Service/Group, Event State and Event Type]. This can be read as: “If the event originated from a certain host and for a certain service and with a specific event state and type, then the condition match”.

Whether the condition triggers or suppresses an action is decided by the *any/none* selector. If set to *any* (the default), then any matching tuple triggers the action. If it is set to *none*, then the tuples serve as an exclusive list and any non-matching event will trigger the action.

For instance, to get notified if a service fails and when the service comes back up again, create one condition row with [“Host”, “Service”, Failed + Succeed, Any Event].

A row of [Any Host, Any Service, Any State, Any Event] is a "catch all" condition and will generate an alert for every event.

### Contextualised drop-down menus

If you select Host first, then the Service sub-menu will only list Services for the selected Host. This makes it easy to setup an alert for a specific service on the host.

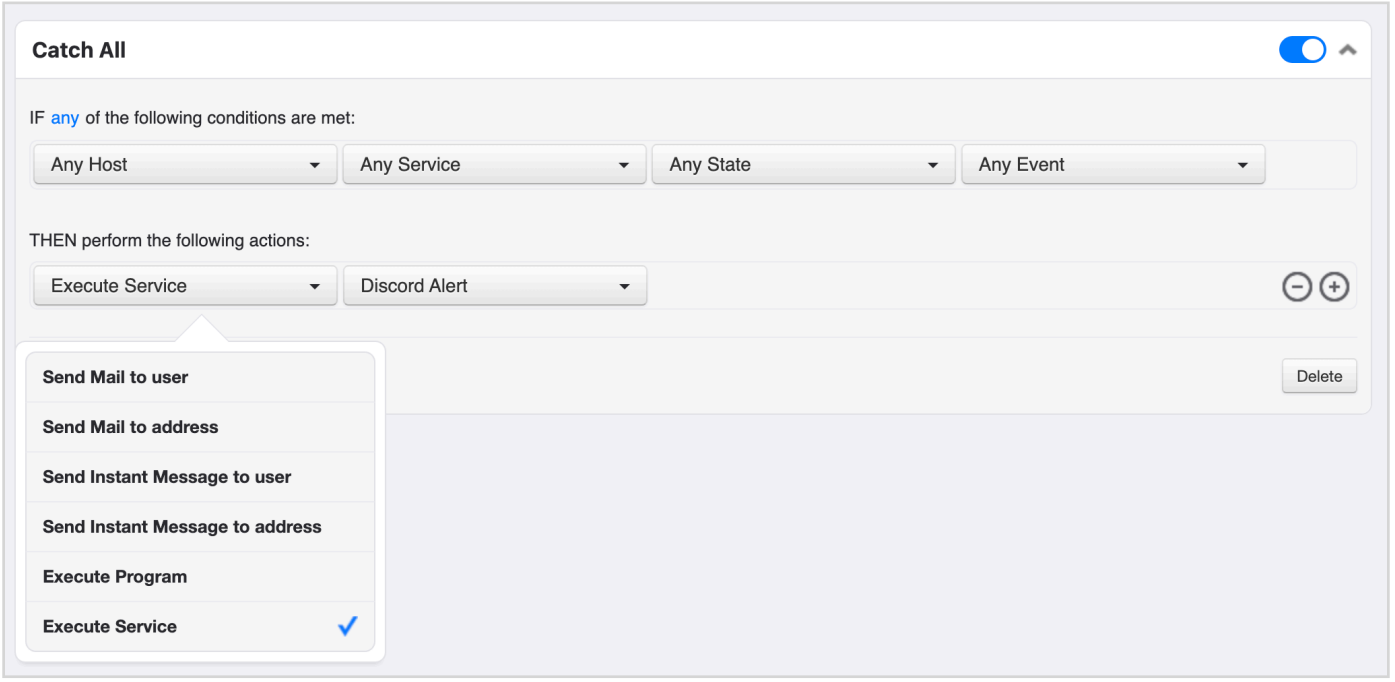
The same is the case for Host Groups. If you select a Host Group in the first drop-down menu, then only Services and Service Groups that are common for all hosts in the group are listed in the Service menu.

If you do not select host, but keep the “Any Host” value, then the Service menu will instead list *all* services registered in M/Monit across all hosts.

If both a Host and a Service is selected, the Events menu will only list event types relevant for the selected service types.

## Alert actions

What M/Monit should do when a rule evaluates to true is specified in one or more **actions**. M/Monit currently supports sending *email* or *jabber* messages, execute a *service* script or a *program* as an action.




- **Email.** The mail server M/Monit should use for sending email is specified in the admin Mail servers page. There are two ways to specify email notification. Either by selecting a user registered in M/Monit or by specifying an email address directly. The [Send mail to user] drop-down target menu contains all users in M/Monit. Disabled users in that list are those who do not have an email.

- **Jabber.** The Jabber server M/Monit should use for sending Jabber messages is specified in the admin XMPP servers page. There are two ways to specify Jabber notification. Either by selecting a user registered in M/Monit or by specifying a Jabber address directly. The [Send instant message to user] drop-down target menu contains all users in M/Monit. Users with no Jabber address are disabled in that menu.

- **Alert Service:** M/Monit comes pre-configured with scripts for sending alerts via popular platforms like Slack, Discord, PagerDuty Microsoft Teams and Telegram. Select [Execute Service] and then one of the Alert Scripts you want to use for alerts. You can also create your own custom alert scripts to meet specific requirements.

  This allows you to easily integrate M/Monit alerts with your preferred communication platforms, ensuring timely notifications for your team.



- **Program.** The Program option allows you to execute a specific binary program or a short script as an alert action. While Alert Services are recommended for complex scripts, the Program option is ideal for simple, direct commands or binary executables. Select [Execute Program] and write the program in the input field. Example uses:

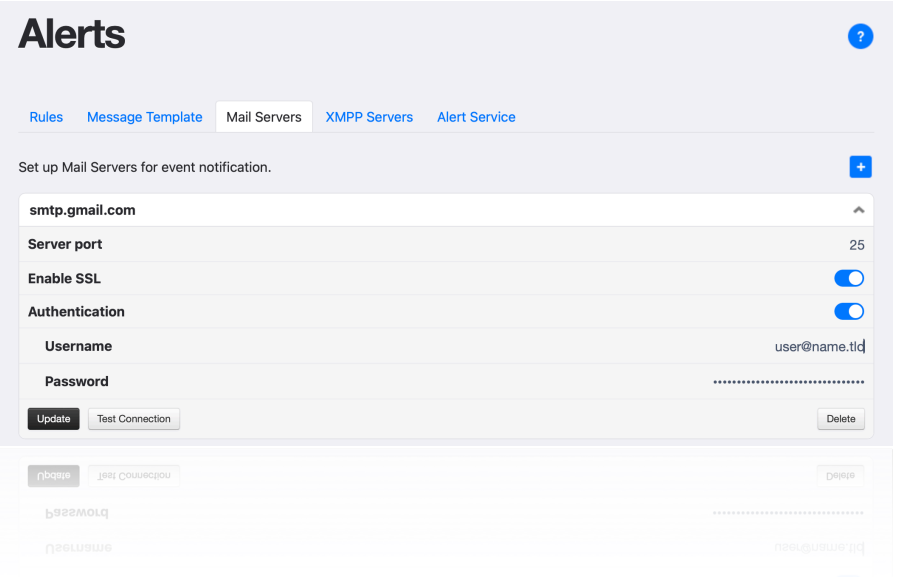
  - Sending an SNMP trap: `/usr/bin/snmptrap -v 2c -c public 192.168.1.100 '' .1.3.6.1.4.1.2021.251.1 s "$MONIT_SERVICE: $MONIT_EVENT"`

  - On macOS (for demonstration): `/usr/bin/say "Alert: $MONIT_SERVICE has an issue"`

  **Note:** For complex alerting needs and reusability, consider using the Alert Service option instead of Program.

## Mail servers

Specify SMTP server(s) M/Monit should use for alert notifications.




To add a new server, click the blue **+** button and fill in all relevant fields. Remember to click the Save button to save the entry. The Save button is disabled until required fields are filled. To edit an existing server, expand its entry and modify the fields. The Update button is disabled unless there are changes to the entry. Click the Delete button to delete the entry.

### SSL

If the Enable SSL option is on, M/Monit will use SMTPS, unless the mail server port number is 25 or 587, in which case STARTTLS is used if required by the SMTP server. Use the DNS hostname of the SMTP server, not an IP-address. M/Monit will perform SSL certificate verification using the hostname when connecting to the server to ensure it is connected to the correct server. See also <SSLClientConfig> in the manual (PDF)

### Test Connection

It is good practice to test the configuration before saving. Click the Test Connection button to have M/Monit test and verify that a connection can be established to the SMTP server and that the mail sender's address (from the Message Template) will be accepted. No message is sent during this test.

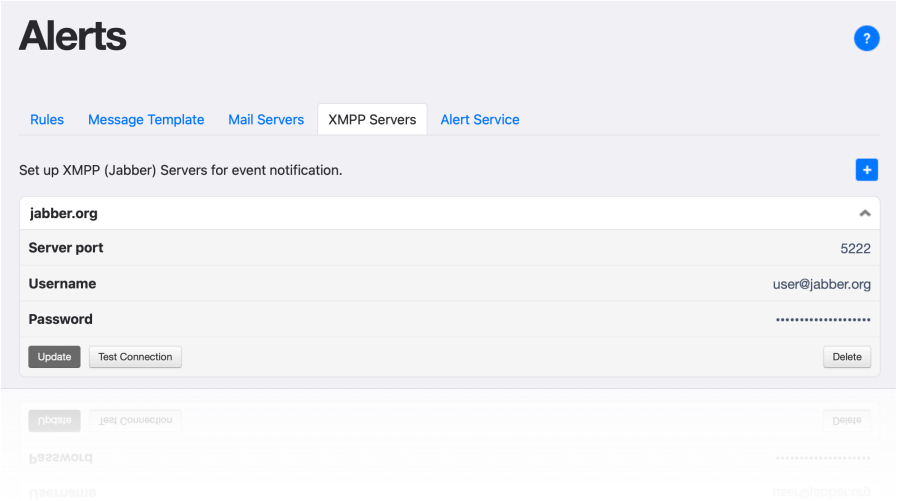
### Message delivery

When sending an alert message, M/Monit will use the first SMTP server in the list and if not available, continue with the next and so on, until the message is sent or all servers failed.

## Jabber servers

Specify Jabber server(s) M/Monit should use for alert notifications.

To add a new server, click the blue **+** button and fill in all relevant fields. Remember to click the Save button to save the entry. The Save button is disabled until required fields are filled. To edit an existing server, expand its entry and modify the fields. The Update button is disabled unless there are changes to the entry. Click the Delete button to delete the entry.




### SSL

If the port number is 5223 M/Monit will connect using SSL/TLS otherwise it will switch to a secure connection and use STARTTLS if required by the server. The default jabber port number is 5222 and if in doubt, use that number. Use the DNS hostname of the Jabber server, not an IP-address. M/Monit will perform SSL certificate verification using the hostname when connecting to the server to ensure it is connected to the correct server. See also <SSLClientConfig> in the manual (PDF)

### Credentials

Username (Jabber ID) and password are required to identify the user who will send the alert message.
Note that the user you specify here should be different from the one you will use in your jabber client. Otherwise the jabber server will think that you are sending messages to yourself and won't deliver the alert message to your client. We recommend using a dedicated Jabber user for M/Monit.

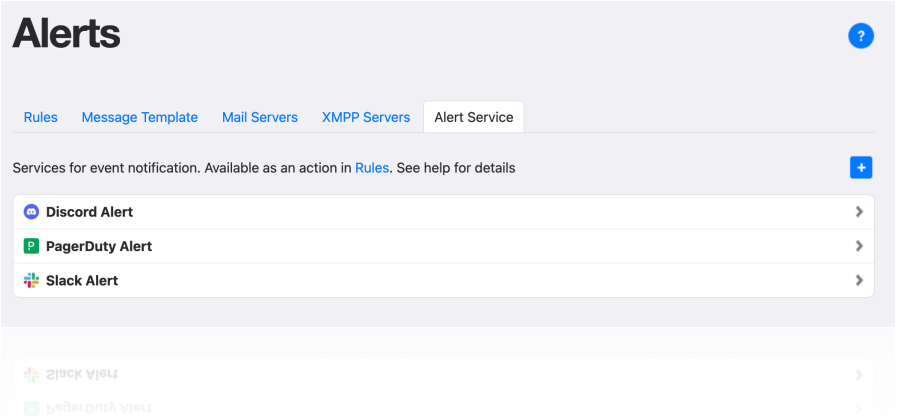
### Test Connection

It is good practice to test the configuration before saving. Click the Test Connection button to have M/Monit test and verify that a connection can be established to the Jabber server and that credentials will be accepted. No message is sent during this test.

### Message delivery

When sending a Jabber alert message, M/Monit will use the first Jabber server in the list. If it is not available, it will continue with the next one, and so on, until the message is sent or all servers fail.

## Alert Service




M/Monit enhances its alerting capabilities through customisable scripts. While pre-configured scripts are available for services like Slack and Discord, you can also create your own alert scripts to meet specific requirements..

Alert scripts are executed in a controlled environment using `/bin/sh -c script`. The execution environment is intentionally minimal:

- The PATH is set to: */bin:/usr/bin:/usr/local/bin:/opt/local/bin*
- All other environment variables are cleared for security reasons

When creating custom alert scripts, you can use shell commands directly or invoke other interpreters like Python. Just remember to use full paths for any commands not in the default PATH, or add necessary directories to the PATH within your script.

A set of environment variables are made available to the script at execution time, describing the event that occurred:

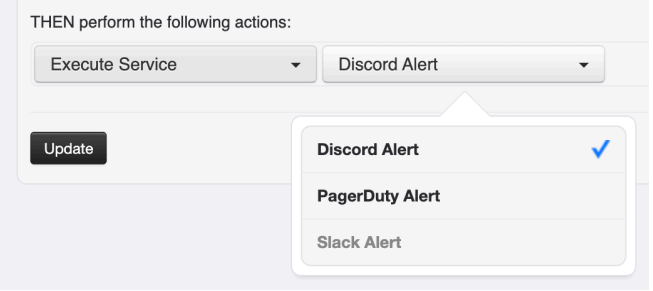
- $MONIT_EVENT: A short string describing the event that occurred
- $MONIT_STATE: Event state, where 0=Success, 1=Failed, 2=Changed, and 3=Initial/Other.
- $MONIT_SERVICE: The name of the service generating the event
- $MONIT_DATE: A date string when the event occurred (local time with UTC offset)
- $MONIT_TIMESTAMP: contains the event timestamp in UTC (seconds)
- $MONIT_HOST: The name of the Monit Host the event originated from
- $MONIT_ACTION: The name of the action which was performed by Monit
- $MONIT_DESCRIPTION: A description of the event condition. I.e. why the event was sent

The value you enter in the URL/Endpoint field can be referred to in the script as $ENDPOINT and the value entered in the API Key/Token field can be referred to as $APIKEY.

M/Monit comes pre-configured with scripts for sending alerts via popular platforms like Slack, Discord and PagerDuty. You can also create your own custom alert scripts to meet specific requirements.

Before using an Alert Service in an Alert Rule, you must first configure it for your environment. Follow these steps:

1. Go to the Alert Service tab to configure the scripts.
2. Set up the necessary credentials and/or endpoints for each service you wish to use.
3. Once configured, return to the Rules page to use the scripts for alerts.
4. Select the action "Execute Service" from the dropdown menu.
5. Choose the specific alert script you want to use (e.g., Slack Alert, Discord Alert or PagerDuty Alert).




This allows you to easily integrate M/Monit alerts with your preferred communication platforms, ensuring timely notifications for your team.

**Tips**: Click the help button in the Alert Service page to see a detailed guide on how to configure the pre-configured services. Also remember to click the [Test Service] button to verify the setup. You should see a test message appear in your channel (e.g. Discord).

## How to prevent Monit from also sending alerts

If Monit has been setup to send alerts and you want M/Monit to take over alert responsibility (recommended) then here is a recipe for how to prevent Monit from also sending alerts so you do not get alert duplicates. This will also simplify Monit's configuration file.

1. Remove any *set mailserver*, *set mail-format* and *set alert* statements from the Monit configuration file
2. Likewise, remove any standalone *set alert* statements with recipients from the Monit configuration file

For instance if you have:

```
set mailserver...
set mail-format...
set alert...

check file monit with path "/usr/local/bin/monit"
      if changed checksum then alert
      alert someone@example.com
```

Change this to:

```
check file monit with path "/usr/local/bin/monit"
      if changed checksum then alert
```

This will cause the alert event to be sent to and processed by M/Monit instead of Monit. Any `exec, start stop` etc will continue to be handled by Monit, but the corresponding alert event is sent to M/Monit for alert processing.

# Settings

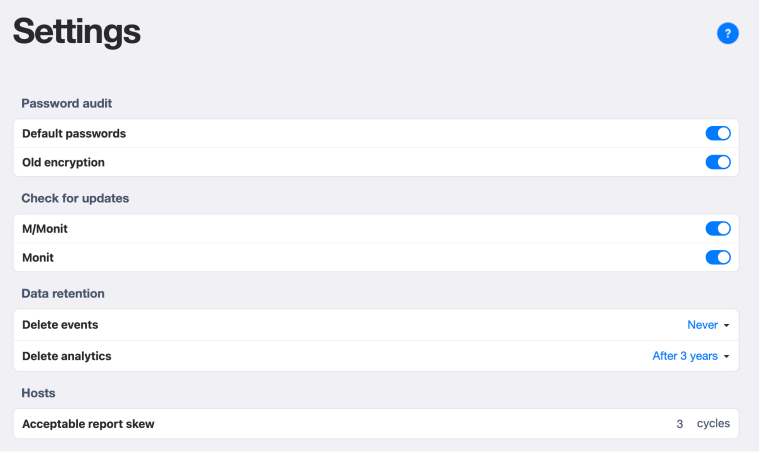
This page provides a few general settings for the application.

The **password audit** section can be used to turn on a subtle UI indication if an account is using one of the well-known default passwords ("swordfish" or “monit") or if an account has password stored as a MD5 hash, used in M/Monit 3. (To update a password from MD5 to the new SHA256 password scheme, simply click Update in admin/users).

You can ask M/Monit to automatically **check for updates**. This will make M/Monit connect to mmonit.com once a day and check for new releases of M/Monit and Monit. If a new version is found, information will be posted in the dashboard.

By default, events are stored in the M/Monit database and never deleted. This means that the events log will grow over time. It can be useful to prune the events log to save space or if drill-down in the events log is starting to become slow due to an abundance of data. The **Delete events** option can be used to delete old events now and automatic purge old events in the future.

**Analytics data** (that is, data used to create charts in M/Monit) is stored and aggregated in the M/Monit database for 3 years, after which it is automatically deleted. If old analytics data is not useful nor relevant, you can use this option to store analytics data for a shorter time and thereby save space and possibly generate charts faster.




A Monit agent has to send status update within a certain timeframe, otherwise M/Monit will generate a heartbeat failure event. The **acceptable report skew** is the number of Monit cycles M/Monit will wait before reporting the host as down. The default value is 3 cycles. The report skew set in this page applies to all new hosts registered in M/Monit. You also can fine-tune this value further in the *Admin/Host* page for a specific host.

Server room at night
Monit poll cycle started
alarm quietened



# Appendix A

# server.xml

The configuration file for M/Monit is *server.xml* and in this chapter we describe the configuration directives used in this file.

## Directory and file names

If you specify a file that begin with "/" the server will use that absolute path. If the filename does not begin with "/", the value of the M/Monit home directory is prepended. The M/Monit home value is automatically computed based on the location of the mmonit binary. For instance if the mmonit binary is located in "/usr/local/mmonit/bin/mmonit" the mmonit home directory is "/usr/local/mmonit".

## <Server>

The Server element represents the entire Container and is the single outermost element in the server.xml configuration file. Only one Service elements may be nested inside a Server element

| Attributes | Description |
|---|---|
| N/A | No attributes are defined for this element |

## <Service>

A Service element represents the combination of one or more Connector components that share a single Engine component for processing incoming requests. The only components that may be nested inside a Service element are one or more Connector elements, followed by exactly one Engine element.

| Attributes | Description |
|---|---|
| N/A | No attributes are defined for this element |

## <Connector>

The Connector element represents a Connector component that supports the HTTP/1.1 protocol. It enables the M/Monit Servlet Container to function as a stand-alone web server, in addition to its ability to execute servlets. A particular instance of this component listens for connections on a specific TCP port number on the server. One or more such Connectors can be configured as part of a single Service, each forwarding to the associated Engine to perform request processing and create the response.

At server startup, the Connector will create a pool of servlet request processing threads. The maximum number of threads in the pool is specified by the attribute; *processors*. 10 processor threads are usually more than enough. If you change this, the number of processor threads should match or slightly exceed the number of CPU cores on the system. Increasing the number of processing threads may or may not increase the throughput and speed of the server, in fact it may decrease the performance since more threads means more overhead.

Each incoming servlet request requires a thread for processing. Usually, only CPU bound operations are performed in the servlet thread, while a separate Container thread handle il/o bound operations for all servlet requests and responses. If more simultaneous servlet requests are received than can be handled by the currently available thread pool, requests are queued up inside the Connector, up to the systems maximum available descriptors and when a processor thread becomes available it will immediately start to consume requests from the queue. Operating Systems allows normally anywhere from 128 to 1024 simultaneously open descriptors per process. It is recommended to increase the limits of open file descriptors available to a process before mmonit is started from the console. Use e.g. `ulimit -n 2048` in the console before starting mmonit.

The attribute, *processorTimeout* sets the number of seconds a processor thread will wait for more work before timeout. The Connector increase and reduce the number of processor threads available depending on the work load. The default timeout value is 30 seconds.

The attribute, *connectionTimeout* specify the number of seconds a Connector will wait, after accepting a connection, for the client to send a HTTP request. The default value is 30 seconds.

The attributes, *address* and *port*, specify respectively the network interface M/Monit binds to and the port number M/Monit listen to for incoming connections. Address may be specified as an IP address string, as a host name or you can use '*' to bind to all available interfaces.

The attribute *ipversion* specify the IP-version the Connector should use. If not specified, IP-version 4 will be used. To support both IPv4 and IPv6, specify two Connectors, one setting ipversion to 4 and the other setting ipversion to 6.

By default, a non-TLS HTTP/1.1 Connector is established on port 8080. You may also enable a TLS HTTP/1.1 Connector on port 8443 by uncommenting the second Connector entry in *server.xml*. To make a Connector secure and use TLS, set the secure attribute to true - that's all. The Container will use TLS 1.2 or later by default.

In addition, each virtual Host must specify a certificate file to be used for that specific Host. It is possible to run the Container with both a secure Connector using TLS and a non-secure Connector.

By default, DNS lookup is disabled and the Access Logger will log the IP address instead of the host name. You can enable DNS lookup by setting the enableLookups attribute to "true", but notice that DNS lookups will have an adverse impact on performance if you use an Access Logger.

| Attributes | Description |
|---|---|
| enableLookups | Set to true if you want the AccessLogger to log the Host name from incoming clients connections. Set to false to skip the DNS lookup and return the IP address in String form instead (thereby improving performance). By default, DNS lookups are disabled. |
| redirectPort | If this Connector is supporting non-TLS requests, and a request is received for which a matching <security-constraint> requires TLS transport, the Container will automatically redirect the request to the TLS Connector port number specified here. |
| secure | Define if the Connector should use TLS for incoming client connection. If selected you must also define the SSL certificate in the Host element. |
| ciphers | An optional list of OpenSSL ciphers to use with the Connector when TLS is active. Available ciphers can be found by using this command, `openssl ciphers`. If not set, this default cipher list is used, focusing on the strongest ciphers and using a high security level: ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:ECDHE-RSA-AES256-GCM-SHA384:TLS_AES_256_GCM_SHA384:!aNULL:!eNULL:!LOW:!EXP:!RC4:!3DES:!MD5:!PSK:!SRP:!DSS:!SHA1:!SSLv3:@SECLEVEL=2:@STRENGTH |

| Attributes | Description |
| --- | --- |
| tls | The tls attribute can optionally be used to specify the TLS version number to use with the Connector. This attribute specify a range from the minimum version number to the maximum version number. If not specified, the Connector will use TLS version 1.2 or later. The general format of the tls attribute string is **[min:max]** where min and max are TLS version numbers. For instance `[1.2 : 1.3]` which means support TLS version 1.2 to 1.3. Both min and max are optional. E.g. [TLS 1.2:] means support TLS version 1.2 to the highest version supported by the ssl library. You can also use an exact version string like "TLS version 1.3" or simply "1.3" which means that the Connector will only support TLS version 1.3. Any attempt to connect with a different version will be denied. Note that newer OpenSSL versions (3.x) no longer support TLS version 1.0 nor 1.1 and using these versions will result in an error. |
| connectionTimeout | The number of seconds a Connector will wait, after accepting a connection, for client to send a HTTP request. The default value is 30 seconds. |
| processors | The maximum number of request processing threads that will be used to serve HTTP requests. The Connector create threads as needed and up to the configured number of processors. The default value is 10 processor threads. |
| processorTimeout | Sets the number of seconds a processor thread will wait for more work before timeout. The Connector increase and reduce the number of processor threads available depending on the work load. Default value is 30 seconds. |
| port | The TCP port number on which a Connector will create a server socket and await incoming connections. Your operating system will allow only one server application to listen to a particular port number on a particular IP address. The default port is 8080. |
| ipversion | Specify the IP-version to be used by the Connector. Legal values are either 4 or 6. Unless specified, IP-version 4 will be used. |
| contentLengthLimit | The maximum content length allowed for a Request. I.e. the maximum allowed value of the http Content-Length header. If content is larger than this number, the server will return a 413 *Request entity to large* error code. Default value is 50 MB. Value in bytes. |
| contentLengthBufferLimit | The size threshold limit in bytes after which uploaded content from a Request will be written to disk instead of in-memory. Default value is 32 KB. Increasing this limit can speed up processing if the server should handle frequent file uploads. Value in bytes. |

| Attributes | Description |
| --- | --- |
| proxyScheme | If this Connector is being used in a proxy configuration, configure this attribute to set the proxy server scheme we should pretend request to this Connector is using. For example, you would set this attribute to *https* if the proxy server is using https. The default value is *http*. |
| proxyName | If this Connector is being used in a proxy configuration, configure this attribute to specify the server name to be returned for calls to request.getServerName(). See Proxy Support for more information. |
| proxyPort | If this Connector is being used in a proxy configuration, configure this attribute to specify the server port to be returned for calls to request.getServerPort(). See Proxy Support below for more information. |

## <Engine>

The Engine element represents the entire request processing machinery associated with a particular Service. It receives and processes all requests from one or more Connectors, and returns the completed response to the Connector for ultimate transmission back to the client. Exactly one Engine element MUST be nested inside a Service element.

The Host defined in the *defaultHost* attribute is used to process Requests directed at Virtual Hosts not configured in this configuration file. The default Host will also handle HTTP/1.0 based requests without a Host header.

The *filecache* attribute is used to set the file cache size for the Engine. The Engine cache static files in memory, to speed up transmission. This is particular useful for secure TLS transmissions. The cache size attribute is set in megabytes (MB) and the default size is 10MB. If the size is set to a value less than 1MB it is ignored and the maximum cache size set to 1MB. You can disabled the cache by setting filecache to 0. Disabling the file cache is strongly advised against for production systems.

You can nest one or more Host elements inside the Engine element, each representing a different virtual host associated with this server. At least one Host is required, and one of the nested Hosts MUST have a name that matches the name specified for the defaultHost attribute, mentioned above.

You can also nest at most one instance of the following utility components inside an Engine element:

- ErrorLogger - Configure an error logger that is used by the Server to dump error and warning messages.
- Realm - Set the security realm database used to authenticate individual users and used by the M/Monit application.

| Attributes | Description |
| --- | --- |
| name | The Engine name. Merely for documentation purpose. |
| defaultHost | The default host name, which identifies the Host that will process requests directed to host names on this server, but which are not configured in this configuration file. This name MUST match the name attributes of one of the Host elements nested immediately inside. |
| filecache | The file cache size for the Engine. Normally, the Engine cache "small" static files in memory to speed up transmission and reserve the sendfile() OS "zero-copy" mechanism for transmitting larger files. In some situations the "zero-copy" mechanism cannot be used, especially when files are transmitted over TLS. In this case larger files are also cached to speed up transmission. The cache size attribute is set in MB and the default size is 10Mb. |

## <Host>

The Host element represents a virtual host, which is an association of a network name for a server (such as "www.example.com" with the particular server on which M/Monit is running. This name must be registered in the Domain Name Service (DNS) server that manages the Internet domain you belong to - contact your Network Administrator for more information.

In many cases, System Administrators will wish to associate more than one network name (such as www.example.com and example.com) with the same virtual host. This can be accomplished by using the alias attribute. You may add as many Host aliases as you like, but note that a Host alias must also be a valid DNS name.

The appBase attribute defines the application root directory for the Host. This directory may contain web applications to be deployed on this virtual host.

The certificate attribute is used to specify the SSL certificate file for the Host. Certificates must be in the PEM format and the file must contain the following entries in this order: The Host certificate private key, the Host certificate and finally, unless this is a self-signed certificate, the certificate of the authority that signed the Host certificate.

If you don't already have a SSL certificate you can create a self-signed certificate yourself using this OpenSSL command:

```
openssl req -new -newkey rsa:2048 -x509 -days 365 -nodes -out mmonit.pem -keyout mmonit.pem
```

Note that a self-signed certificate will generate a warning in your browser, saying that the certificate is not trusted. This does not mean that the connection won't be secure, just that the browser cannot validate the identity of the certificate authority. A self-signed certificate can be useful for testing, but in practice we recommend that you get a commercial or a free SSL certificate via, for instance, https://letsencrypt.org/ The above OpenSSL command will put the private key in the mmonit.pem file together with the certificate. This means that the certificateKey attribute is not needed, only the certificate, as *certificate="conf/mmonit.pem"* in <Host>

You can nest one or more Context elements inside the Host element, each representing a different web application associated with the virtual host.

You can also nest at most one instance of the following utility components by nesting a corresponding element inside your Host element:

- *AccessLogger* When you run a web server, one of the output files normally generated is an access log, which generates one line of information for each request processed by the server, in a standard format.
- *Logger* A Logger shared by all Contexts related to this virtual host. The Logger will process all log messages for a Host, plus messages from Contexts and Servlets associated with the Host.
- *SSLClientConfig* Configure outbound TLS connections for all web-applications (Context) in a Host element

| Attributes | Description |
|---|---|
| appBase | The Application Base directory for this virtual host. This is the pathname of a directory that may contain web applications to be deployed on this virtual host. You may specify an absolute pathname for this directory, or a pathname that is relative (to the mmonit home directory). |
| name | The network name of this virtual host, as registered in your Domain Name Service server. One of the Hosts nested within an Engine MUST have a name that matches the defaultHost setting for that Engine. See the Host Name Alias attribute below for information on how to assign more than one network name to the same virtual host. |
| alias | An alias for the host. That is; a DNS C record. You can add more than one alias attribute, as long as the Host alias represents a real DNS record. |
| certificate | Specify the SSL certificate file for the Host. Entries must be in the PEM format and must contain the following entries in this order: the Host certificate and the certificate of the authority that signed the Host certificate. If you use Let's Encrypt, use `fullchain.pem` here. |
| certificateKey | Specifies the host certificate private key file. The entry must be in the PEM format. Alternatively, you can concat the private key into the start of the certificate file above and not use this attribute. If you use Let's Encrypt, use `privkey.pem` here. |
| address | *Deprecated*. No longer used. M/Monit supports TLS SNI so you can have more virtual hosts on the same server with their own SSL certificates, without the need to hardcode the IP-address in M/Monit's configuration. |

## <Context>

The Context element represents a web application, which is run within a particular virtual host. A web application is a collection of servlets, html documents, images and other resources put in a directory structure with a standard layout.

The web application used to process each HTTP request is selected by the Container based on matching the longest possible prefix of the Request URL against the context path of each defined Context. Once selected, that Context will select an appropriate servlet to process the incoming request, according to the servlet mappings defined in the web application deployment descriptor file (which MUST be located at /WEB-INF/web.xml within the web app's directory hierarchy).

You may define as many Context elements as you wish, nested within a Host element. Each such Context MUST have a unique context path, which is defined by the path attribute. In addition, you can define a Context with a context path equal to a zero-length string. This Context becomes the default web application for this virtual host, and is used to process all requests that do not match any other Context's context path. If such a context is not defined in this configuration file, the Servlet Container will create a default context with ROOT as the docBase directory.

Each Context utilize a Session Manager to manage HTTP sessions for the Context. The maxActiveSessions attribute specify the maximum number of sessions that will be created for the Context. The manager expire idle sessions after *sessionTimeout* seconds. It is possible to turn off sessions for a Context by setting the attribute maxActiveSessions to 0 or to a negative value. Note that sessions are not enabled unless maxActiveSessions is defined and set to a positive value.

| Attributes | Description |
|---|---|
| docBase | The Document Base (also known as the Context Root) directory for this web application. You may specify an absolute pathname for this directory or a pathname that is relative to the appBase directory of the owning Host. |
| path | The context path of this web application, which is matched against the beginning of each request URL to select the appropriate web application for processing. All of the context paths within a particular Host must be unique. If you specify a context path of an empty string (""), you are defining the default web application for this Host, which will process all requests not assigned to other Contexts. *M/Monit is setup as the default application and must be assigned to “”.* |

| Attributes | Description |
|---|---|
| reloadable | Set to true if you want the Servlet Container to monitor classes in WEB-INF/classes/ for changes, and automatically reload the web application if a change is detected. This feature is very useful during application development, but it requires runtime overhead and is not recommended for use on deployed production applications. |
| sessionTimeout | The session timeout in seconds for this web application. The default value is 900 seconds (15 minutes) |
| saveSessions | Set to true if you want to enable persistent Sessions. I.e. Sessions are stored in the Realm database and retained during application restart. The default value is false. |
| maxActiveSessions | The maximum number of active sessions that will be created by this Context. The default is 4096. If this attribute is not used or the value is 0 or less the Context will not support sessions. |

## <Realm>

The Realm element specify the database to be used by M/Monit and the underlying security realm to authenticate individual users and store HTTP sessions. If the Realm database is not defined, M/Monit will not start and abort its operation.

Connections from M/Monit to the Realm database is maintained by a database Connection Pool. The url attribute specify the connection to the database server on a standard URL format. The format of the connection URL is defined as:

```
database://[user:password@][host][:port]/database[?[property1=value1][&property2=value2]...]
```

The property names, user and password are always recognized and specify how to login to the database. Other properties depends on the database server in question. User name and password can alternatively be specified in the authentication part of the URL. If port number is omitted, the default port number for the database server is used.

The optional attributes, *minConnections* and *maxConnections*, specify respectively, the minimum number of concurrent available connection and the maximum number of database connections that can be created. The pool will dynamically increase and reduce the number of active Connections in the pool between minConnections and maxConnections depending on the load. If not set, minConnections is 5 and maxConnections is 20.

The attribute *reapConnections* specify if the Connection Pool should run a reaper thread, which will close and remove unused Connections from the Pool. The value is sweep time in seconds. I.e. the reaper thread will sleep for reapConnections seconds,wake up, clean up the pool if necessary and go back to sleep. If this attribute is not set, the Connection Pool will not start with a reaper thread. It is highly recommended to activate a reaper thread for the connection pool so stale and excess connections automatically are removed and closed.

The location of the default SQLite Realm database may be given as an absolute or relative path. If relative, the absolute path is computed relative to M/Monit home. As mentioned above, instead of SQLite you may use MySQL or PostgreSQL as demonstrated in the commented out Realms in *server.xml*.

A realm operates according to the following rules:

- When a user attempts to access a protected resource for the first time, M/Monit will call the authenticate() method of this Realm. Thus, any changes you have made to the database directly (new users, changed passwords or roles, etc.) will be immediately reflected.
- Once a user has been authenticated, the user (and his or her associated roles) are cached within the server for the duration of the user's login. For FORM-based authentication, that means until the session times out or is invalidated; for BASIC authentication, that means until the user closes their browser.
- The information in the user database is controlled by the M/Monit admin/users/ page.

Only one Realm element should be defined per Server and the element is defined within an Engine element.

| Attributes | Description |
|---|---|
| URL | The database to be used by M/Monit and the underlying security realm to authenticate individual users and store HTTP sessions. Connections to the database is maintained by a Connection Pool. |
| minConnections | The minimum and initial number of concurrent available connections to the Realm database. The default value is 5 connections. |
| maxConnections | The maximum number of connections to the Realm database. |
| reapConnections | Start the Realm Database Connection Pool with a reaper thread which will close and remove unused Connections from the Pool. Value in seconds, inside the range [1..86400] sec |

## <ErrorLogger>

An ErrorLogger is used for logging debug and error messages (including stack tracebacks). The ErrorLogger does not buffer data but writes directly to the log file. Each entry in the log file is prefixed with a timestamp. The ErrorLogger component is optionally and if it is not defined the Server will write error messages to stderr.

Only one ErrorLogger element should be defined per Server and the element is defined within an Engine element.

| Attributes | Description |
|---|---|
| directory | Absolute or relative pathname of a directory in which log files created by this Logger will be placed. If a relative path is specified, it is interpreted as relative to M/Monit home. |
| fileName | The name of the log file the ErrorLogger will write to. The filename must not be prefixed with a path. |
| rotate | The rotate attribute may be used to rotate the log file (without having to restart the mmonit server). The value is either "day", "week" or "month". If the value is "day" then on the first logged message after midnight each day, the current log file will be closed and renamed with a postfix date and a new log file is opened with the file name given in the fileName attribute. If the value is "week" then on the first logged message after midnight each Saturday the log file is rotated. Likewise if the value is "month" then on the first logged message in a new month the log file is rotated. The current log file, that is, the file the server writes to is always the file given in the fileName attribute. |

## <AccessLogger>

The AccessLogger create log files in the same format as those created by standard web servers. These logs can later be analyzed by log analysis tools to track page hit counts, user session activity, and so on. An AccessLogger is associated with a virtual host and will record ALL requests processed by that Host.

The AccessLogger flush to the log file every 10 seconds or writes every 32Kb of log entries if this is reached sooner.

The log format used by an AccessLogger is the Common Log Format plus entries for Referer and User-Agent. An entry in the log file may look like:

```
64.87.72.95 - admin [18/Oct/2013:01:10:21 +0200] "GET /status/ HTTP/1.1" 200 3707 "http://localhost/" -
```

The first field (64.87.72.95) is the hostname or IP address of the connecting machine. The second is a username from an ident lookup. If no ident lookup was performed the '-' character is used. The third is the auth-username if authentication was performed. The fourth is the timestamp for the request. The fifth the HTTP request sent to the server. The sixth field is the HTTP status code returned in the response. The seventh field is the response-size, that is, the size of the response entity, not including HTTP headers. If no entity was returned in the response the value is zero. The last two fields contains HTTP headers sent by the Browser. The fields are respectively the HTTP Referrer and the User-Agent. Note that the Browser may opt not to send any these HTTP headers in which case the last two fields will have the value "-".

| Attributes | Description |
| --- | --- |
| directory | Absolute or relative pathname of a directory in which log files created by this Logger will be placed. If a relative path is specified, it is interpreted as relative to M/Monit home. |
| fileName | The name of the log file the AccessLogger will write to. The filename must NOT be prefixed with a path. |
| pattern | This attribute is reserved for future use and will be used to select a particular log format. |
| rotate | The rotate attribute may be used to rotate the log file (without having to restart the mmonit server). The value is either "day", "week" or "month". If the value is "day" then on the first logged message after midnight each day, the current log file will be closed and renamed with a postfix date and a new log file is opened with the file name given in the fileName attribute. If the value is "week" then on the first logged message after midnight each Saturday the log file is rotated. Likewise if the value is "month" then on the first logged message in a new month the log file is rotated. The current log file, that is, the file the server writes to is always the file given in the fileName attribute. |

## <Logger>

A Logger is associated with a Host, and Servlets registered within the Host Context can use a Logger for writing log messages to a log file. M/Monit uses this log file to write application specific data.

| Attributes | Description |
|---|---|
| directory | Absolute or relative pathname of a directory in which log files created by this Logger will be placed. If a relative path is specified, it is interpreted as relative to mmonit home. |
| fileName | The name of the log file the Logger will write to. The filename must NOT be prefixed with a path. |
| timestamp | Set to true to cause all logged messages to be prefixed with a timestamp (the default). Set to false to skip stamping. |
| rotate | The rotate attribute may be used to rotate the log file (without having to restart the mmonit server). The value is either "day", "week" or "month". If the value is "day" then on the first logged message after midnight each day, the current log file will be closed and renamed with a postfix date and a new log file is opened with the file name given in the fileName attribute. If the value is "week" then on the first logged message after midnight each Saturday the log file is rotated. Likewise if the value is "month" then on the first logged message in a new month the log file is rotated. The current log file, that is, the file the server writes to is always the file given in the fileName attribute. |

## <SSLClientConfig>

The SSLClientConfig element is a component for configuring outbound TLS (Transport Layer Security) connections within the context of a Virtual Host. This element is optional; if undefined, outbound TLS defaults to recommended settings.

The primary purpose of the SSLClientConfig element is to fine-tune the security aspects of outbound TLS connections initiated by your web applications. It allows for detailed control over various parameters such as certificate validation, cipher suite selection, protocol version enforcement, hostname verification, and more. These configurations help in enforcing security best practices, ensuring data integrity, and maintaining confidentiality during data transmission.

This element is nested inside a <Host> element, applying its configurations to all outbound TLS connections initiated by the web applications associated with that virtual host. The table below describe each element that can be nested inside SSLClientConfig. All elements are optional and if omitted, defaults to recommended values.

| Attributes | Description |
|---|---|
| SelfSignedCertificate | This setting allows you to control the acceptance of self-signed SSL certificates. By default, connections to servers with self-signed certificates are disallowed to maintain a high level of trust and security. *Default*: false (disallow). Omitting this element also defaults to disallow. |
| CipherSuites | Specifies the list of cipher suites that are permitted for use in outbound TLS connections. This setting is particularly important for ensuring that only strong cipher suites are used, thereby preventing potential vulnerabilities associated with weak or compromised ciphers. The format follows the one used in OpenSSL. Omitting this element defaults to a suites that are good enough. |
| Protocols | Defines which versions of the TLS protocol should be enabled for outbound connections. The format of the versions string is the same as the *tls* attribute in <Connector> above. *Default*: [1.2:]. Omitting this element defaults to support both TLS version 1.2 and 1.3. |
| HostnameVerification | Ensures that the server's certificate hostname matches the requested hostname. This is an essential check to prevent Man-In-The-Middle (MITM) attacks. Hostname verification is enabled by *default* to maintain the integrity and authenticity of the server your applications connect to. Omitting this element *defaults* to enabled. |

| Attributes | Description |
| --- | --- |
| ClientCertificate | Facilitates mutual TLS authentication by specifying the path to the client certificate and private key. This setting is used in scenarios where both the client and the server authenticate each other, adding an extra layer of security to the connection. Omitting this element *defaults* to ClientCertificate not being used. |
| CACertificatePath | Allows you to specify a custom path for root CA certificates used to verify server certificates. This can be useful when connecting to servers with certificates issued by private or non-standard CAs, or when additional trust anchors are required. Omitting this element *defaults* to CACertificatePath not being used and the default CA certificate store of the SSL library will be used instead. |
| HandshakeTimeout | Defines the maximum allowed duration for completing the TLS handshake process. This timeout setting helps in preventing hanging connections and ensuring that your applications remain responsive, even in the face of network or server delays. *Default* is 3 seconds. Adjust based on network conditions and server responsiveness. Note that the TLS handshake is blocking and the thread doing the connection might block for timeout milliseconds. Using a value of more than 5 seconds is not recommended. Omitting this element also defaults to 3 seconds. |

Here is an example of how the <SSLClientConfig> element might be configured within your *server.xml*:

```
<SSLClientConfig>
  <SelfSignedCertificate allow="false" />
  <CipherSuites>
    <!-- Cipher suite configuration omitted for brevity -->
  </CipherSuites>
  <Protocols versions="[TLSv1.2:TLSv1.3]" />
  <HostnameVerification enable="true" />
  <ClientCertificate path="/path/to/cert" keyPath="/path/to/key" />
  <CACertificatePath path="/path/to/ca/certs" />
  <HandshakeTimeout value="3 seconds" />
</SSLClientConfig>
```

## <License>

M/Monit is a licensed product and comes with an evaluation license which will expire. If you have purchased a full license, replace the *license.xml* file in *conf/* with your new *license.xml* file and *restart* M/Monit.

| Attributes | Description |
| --- | --- |
| file | The location of the M/Monit's license file. The file is typically placed in M/Monit's conf/ directory. |

# Appendix B

# Access Control

M/Monit supports a subset of the Web Application deployment descriptor as it is defined in the Java Servlet Specification, version 2.4. A Web Application deployment descriptor (web.xml) is used to define security constraints and other Web Application parameters. The following web.xml descriptors are used in M/Monit:

| |
|---|
| docroot/WEB-INF/web.xml |
| docroot/collector/WEB-INF/web.xml |

These files need not be changed and come preconfigured. This appendix explains the configuration directives used to protect the M/Monit web application resources.  A protected resource may be a static file, servlets, Server Pages or a collection of files. To access a protected resource a user must authenticate and supply credentials that match the security constraint for the protected resource. The security mechanism is built into M/Monit and protecting a resource is a question of configuration.

## Security-Constraint

Security constraints are used to protect access to web resources. A constraint specify the parts of a web application that are protected and the security roles allowed to access the protected area. A constraint can also be used to limit access to clients from certain IP addresses and require that TLS should be used.

A security constraint is specified in a web application's deployment descriptor with the security-constraint element and with the following sub-elements: *web-resource-collection, auth-constraint, ip-constraint* and *user-data-constraint*.

## <web-resource-collection>

The `<web-resource-collection>` element define the areas of a web application that are protected. URL-patterns are used to specify all the URLs that are protected.

Several url-patterns may be specified inside a web-resource-collection. Each pattern must conform to one of the following types:

1. Explicit path. This pattern is used to protect one resource, for instance /secret/index.html or /secret/ (To protect sub-directories, see the next type).
2. Path prefix, such as /secret/articles/* or /catalog/* These url-patterns begins with a / and ends with a /* and protect all resources beginning with that prefix.
3. Extension, such as *.html, *.csp or *.zip. These patterns begins with a * and protect every file that ends with that extension.

```
<security-constraint>
    <web-resource-collection>
        <!-- Define the context-relative URL(s) to be protected -->
        <url-pattern>/secret/*</url-pattern>
    </web-resource-collection>
</security-constraint >
```

The following example specify a web resource collection where all pages in the *secret* directory are protected from access. Access can also be denied for certain HTTP request methods, only. In the following example the HTTP POST method is protected while access with other HTTP methods are allowed.

```
<security-constraint>
    <web-resource-collection>
        <url-pattern>/*</url-pattern>
        <!-- If you list http methods, only those methods are protected -->
        <http-method>POST</http-method>
    </web-resource-collection>
</security-constraint >
```

## <auth-constraint>

Users belongs to security roles and access control is performed against roles rather than users. The `<auth-constraint>` element specify the user roles that are permitted access to a resource collection. The special reserved role name `*` is used to indicate all roles.

The following example define an auth-constraint where only users with the role admin are allowed access.

```
<security-constraint>
    <web-resource-collection>
        <url-pattern>/admin/*</url-pattern>
    </web-resource-collection>
    <auth-constraint>
        <role-name>admin</role-name>
    </auth-constraint>
</security-constraint>
```

## <ip-constraint>

The `<ip-constraint>` element is used to specify that only clients with a certain ip address can access the resource collection. The special * character can be used as a wild card in an ip-address string.

The following example allow access for localhost and clients from 80.69.226.* Clients from other ip-addresses will be denied access.

```
<security-constraint>
    <web-resource-collection>
        <url-pattern>/*</url-pattern>
    </web-resource-collection>
    <ip-constraint>
        <ip-address>127.0.0.1</ip-address>
        <ip-address>80.69.226.*</ip-address>
    </ip-constraint>
</security-constraint>
```

## <user-data-constraint>

The `<user-data-constraint>` element define how data should be protected during transfer between client and server, in the `<transport-guarantee>` sub-element. Legal values for this element are NONE, INTEGRAL or CONFIDENTIAL. The values, INTEGRAL and CONFIDENTIAL, require data to be transported over a secure connection, while NONE indicate no special requirements. In practice, specifying a transport-guarantee of INTEGRAL or CONFIDENTIAL means that data will be transported over a TLS connection. CONFIDENTIAL indicate that strong encryption should be used.

The following example configure a web resource collection where access to the directory /confidential/ must be done over TSL (HTTPS).

```
<security-constraint>
    <web-resource-collection>
        <url-pattern>/confidential/*</url-pattern>
    </web-resource-collection>
    <user-data-constraint>
        <transport-guarantee>CONFIDENTIAL</transport-guarantee>
    </user-data-constraint>
</security-constraint>
```

If a secure <connector> is setup for M/Monit, then access to the directory /confidential/ over http will automatically be redirected to https. If no secure <connector> is setup for M/Monit, trying to access /confidential/ over http will return a forbidden error page.

## Authentication

Authentication is the process by which clients provide credentials that are verified by the server. If the credentials are accepted the client is granted access. Generally, credentials are username and passwords but can also be client certificates (X509). Authentication with username and password is conducted against a security realm database defined in *server.xml* in the <Realm> element. The user database is maintained in M/Monit via admin/users.

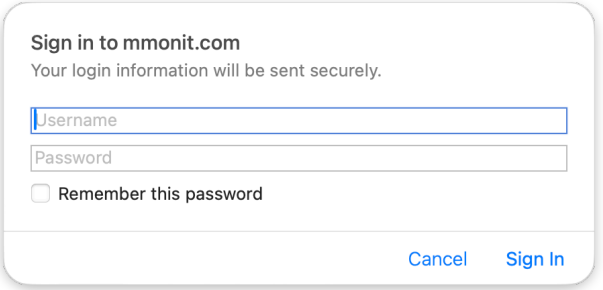
## Login-Config

The <login-config> element is used to configure how authentication should be performed. There can be only one login-config element per web application. The following authentication methods are supported: HTTP Basic Authentication, Form Based Authentication and HTTPS Client Authentication (currently not supported in M/Monit).

The <login-config> element consists of the following sub-elements: *auth-method*, *realm-name* and *form-login-config*. The auth-method element specify the authentication method. Legal values for this element are one of BASIC, FORM or CLIENT-CERT. The realm-name element indicate the realm name to be used for Basic Authentication and finally the form-login-config element specifies the login and error page to be used with the form login authentication method.

### HTTP Basic Authentication

Basic Authentication, described in RFC 2617, is a simple challenge-response authentication mechanism defined at the HTTP protocol level. Using Basic Authentication, the server will challenge a client when it tries to access a protected web area by sending a 401 (Unauthorized) response to the client. Upon receiving this response the client browser will open a dialog where the user can enter a username and a password, such as the one shown here.



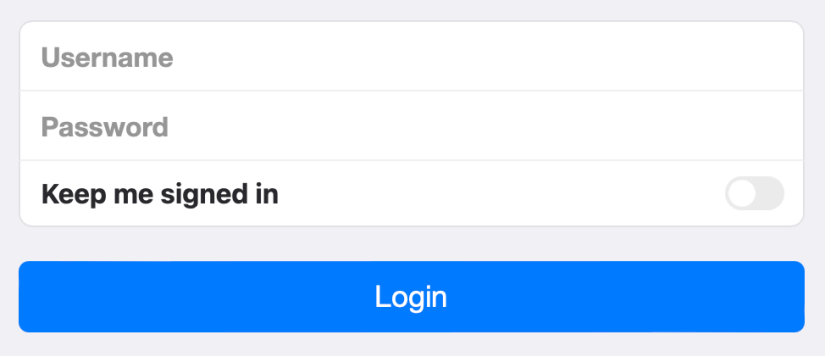

The browser sends the username and password (base64 encoded) to the server in an HTTP Authorization header. The server authenticate the submitted credentials against the security realm database and if accepted sends the requested page back to the client in a standard response. (If the credentials are not accepted the server sends another 401 (Unauthorized) response message, prompting the browser to display the login dialog again.)
Once authenticated, a browser will continue to send the Authorization header with every subsequent request. Thus, the concept of logout does not exist with Basic Authentication and the only way to stop the browser from sending the information is to restart the browser.

The following example configure a login-config element to use Basic Authentication. Notice that the realm-name is displayed in the browser's login dialog. A browser may also use the realm name internally to decide which credentials to send.

```
<login-config>
    <auth-method>BASIC</auth-method>
    <realm-name>Example Protected Area</realm-name>
</login-config>
```

## Form Based Authentication

The advantage of using Form Based Authentication over Basic Authentication is that a custom login page can be used to ask users for a username and a password. Another advantage is that users can logout from a web area protected by form based authentication.

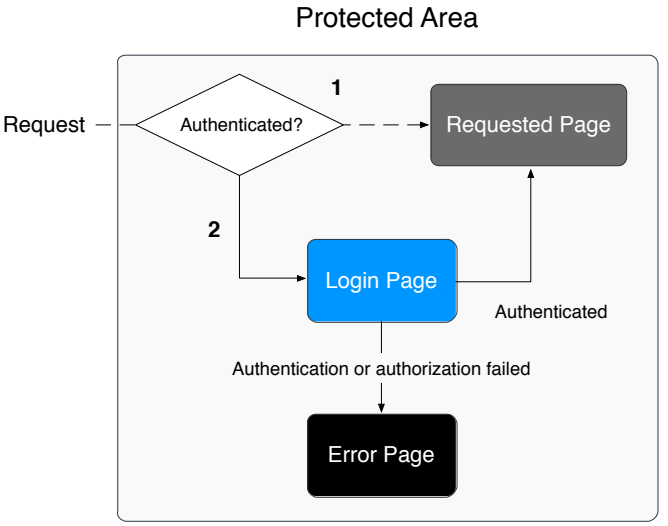



The M/Monit login page is an example of a page which uses Form Based Authentication.

The only requirements for a login page is that it must contain fields for entering a username and a password. These fields must be named *z_username* and *z_password*, respectively and the form action must be *z_security_check*. This example show how the login form could be coded in HTML:

```
<form method="POST" action="z_security_check">
    <input type="text" name="z_username">
    <input type="password" name="z_password">
    <input type="submit" value="Login">
</form>
```

The following diagram illustrate the authentication process:




1. Upon access to a protected area the M/Monit server determines if the client has been previously authenticated, if this is the case, the requested page is sent back to the client.

2. If the client has not been authenticated, the M/Monit server stores the original request URL and displays a login page. The client posts a login form back to the M/Monit server and the server attempts to authenticate the user credentials embedded in the form. If authentication fails the server returns an error page. If authentication succeeds, the server checks if the authenticated user belongs to a security role that is authorized to access the requested page. If the user is authorized the server redirects the request to the original stored request URL. If the user is not authorized to access the requested page the M/Monit server will send a 403 Forbidden response back to the client.

Form Based Authentication utilize HTTP session and clients must support cookies. If the session timeout or is invalidated, the user is logged out and subsequent requests require the user to re-authenticate.

To logout a user, a servlet can invalidate the session by calling the HttpSession_invalidate() method or a ServerPage can be used to invalidate the session. This last alternative is used in M/Monit, and the ServerPage we use is docroot/login/logout.csp. The following example configure a login-config element to use Form Based Authentication:

```
<login-config>
    <auth-method>FORM</auth-method>
    <form-login-config>
        <form-login-page>
            /login.html
        </form-login-page>
        <form-error-page>
            /error.html
        </form-error-page>
    </form-login-config>
</login-config>
```

The *form-login-page* and *form-error-page* elements must contain a relative URL value starting with '/' and the path must be located within the context document root. The URL path may refer to a servlet, a HTML page or a ServerPage.

### HTTPS Client Authentication

This authentication method is a strong authentication mechanism and employ HTTPS client certificates to verify the authenticity of a connecting client. Clients must posses a Public Key Certificate known by the server. The client must connect to the server over TLS and the server ask the client to send its certificate. Upon receiving the certificate the server compares the certificate to a library of existing certificates. Access is granted if the client certificate is in the library.

The following example configure a login-config element to use HTTPS client certificates Authentication:

```
<login-config>
    <auth-method>CLIENT-CERT</auth-method>
</login-config>
```

*This authentication method is currently not supported in M/Monit, but we are considering adding it together with LDAP authentication in a later version.*

### Example

For an example of a web.xml document using a security constraint and a Form Based login method, see M/Monit's own deployment descriptor in *docroot/WEB-INF/web.xml* and for Basic Authentication, see *docroot/collector/WEB-INF/web.xml.*

## CSRF-protection

Cross-Site Request Forgery (CSRF) is a type of attack that occurs when a malicious Web site, email, blog, instant message, or program causes a user's Web browser to perform an unwanted action on a trusted site for which the user is currently authenticated.

M/Monit protect against CSRF attacks by using a scheme called Double Submit Cookies. In short, this means that M/Monit requires that the authenticated session cookie is submitted with the request as usual, but that the value is also repeated in either a request parameter or in a HTTP header, of which an attacker cannot replicate as they cannot read the cookie. The M/Monit web-application is automatically configured to perform CSRF-protection.

If you use M/Monit's HTTP-API from your own program, CSRF might not be a concern. If this is the case, use the `z_csrf_protection` parameter during login and set it to `off` in which case CSRF-protection will be turned off. Otherwise, you must read the CSRF-Token from M/Monit's session API and submit the token in a HTTP header or as a request parameter with each POST request to /admin/

### Configuration

CSRF-protection is configured in `web.xml` in the *<web-resource-collection>* element. Protected URLs should be wrapped in a *<csrf-protect>* element. If you list http-methods, only those methods are CSRF-protected, otherwise all http methods will be protected. In general, you will not need to protect GET requests unless they have side-effects, such as performing an action other than fetching a resource. Here is an example of a security constraint element which uses CSRF-protection and where only the POST method is protected.

```
<security-constraint>
    <web-resource-collection>
          <csrf-protect>
                <url-pattern>/admin/*</url-pattern>
          </csrf-protect>
          <!-- If you list http methods, only those methods are protected -->
          <http-method>POST</http-method>
    </web-resource-collection>
</security-constraint >
```

# Appendix C

## M/Monit behind a proxy

M/Monit can be used from behind a proxy server. Here is an example on how to configure the proxy in front of M/Monit. In this example M/Monit listens on http://192.168.1.10:8080 and we configure a HTTP proxy so M/Monit is accessible via the sub-path /mmonit/ That is, http://proxy.address/**mmonit**/

### Apache with mod_proxy

```
<Location /mmonit/>
     Order deny,allow
     Allow from all
     ProxyPassReverseCookiePath / /mmonit/
</Location>
ProxyPass /mmonit/ http://192.168.1.10:8080/ disablereuse=on
ProxyPassReverse /mmonit/ http://192.168.1.10:8080/
```

### Nginx proxy

```
location /mmonit/ {
    rewrite ^/mmonit/(.*) /$1 break;
    proxy_pass  http://192.168.1.10:8080;
    proxy_redirect http://192.168.1.10:8080 /mmonit;
    proxy_cookie_path / /mmonit/
}
```

The above configuration tells Apache or Nginx to forward URLs of the form http://proxy.address/**mmonit**/* to M/Monit running on 192.168.1.10 and listening on port 8080. This is all the configuration needed to proxy M/Monit via a *sub-path*. M/Monit configuration need not be changed.

Another option is to use a *sub-domain* rather than a *sub-path*. In this case, access to M/Monit will go via an URL that looks like, https://**mmonit**.proxy.address/ In this example, the HTTP proxy server will also be using *https,* but M/Monit will continue to use http.

This setup requires that the proxyScheme attribute in M/Monit's Connector is set to *https* even though the Connector itself is using http. The reason is that redirects from M/Monit must use the https scheme to match the proxy server's scheme.

```
<Connector port="8080" proxyScheme="https" ... />
```

Apache configuration for https://**mmonit**.proxy.address rather than https://proxy.address/**mmonit**/ will be something like:

```
<VirtualHost *:443>
    ServerName mmonit.proxy.address:443
    ProxyPass / http://192.168.1.10:8080/
    ProxyPassReverse / http://192.168.1.10:8080/
    TLS & other config..
</VirtualHost>
```

You may also prefer the hostname and port number to reflect that specified in the original request, not the one on which M/Monit itself is listening. This can be achieved by using the proxyName and proxyPort attributes in the <Connector> element. Please note that these proxy attributes should *not* be used when setting up a *sub-path* proxy. In practice, when using a sub-domain, you should configure the <Connector> element in *server.xml*, with the following proxy settings:

```
<Connector port="8080" proxyScheme="https" proxyName="proxy.address" proxyPort="443" ... />
```

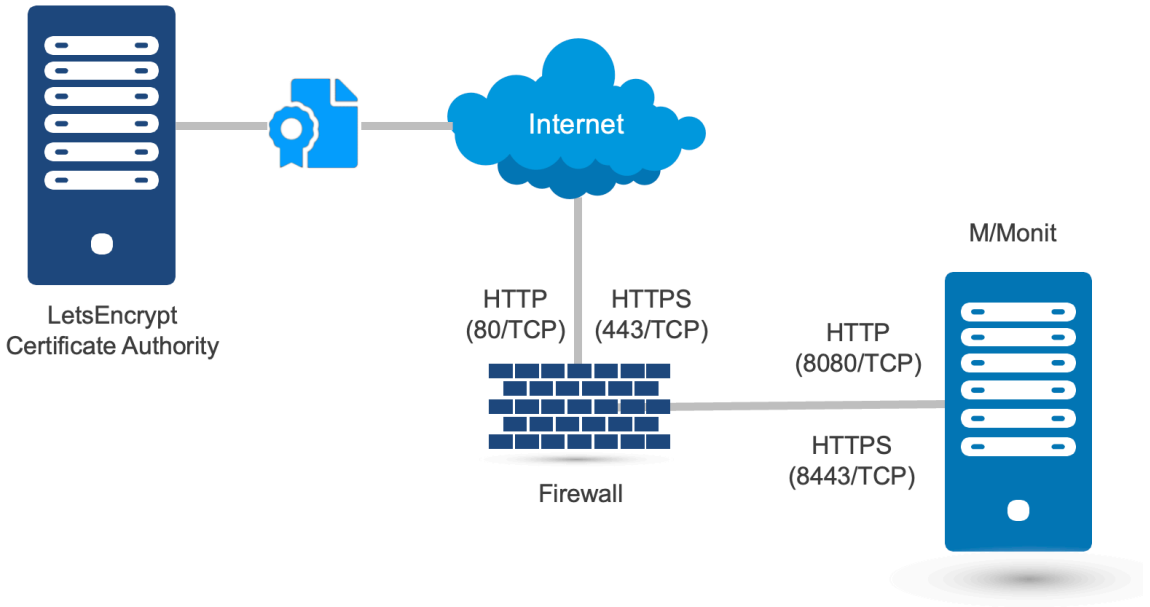
This will cause M/Monit to pretend that all proxied requests were directed to *https://mmonit.proxy.address* on port 443. This is particularly useful when M/Monit performs redirects from the login page or other pages inside its web-app. Otherwise these redirects will contain the hostname and port number of M/Monit. It is valid to omit the proxyName attribute from the <Connector> element. If you do so, the value used in URLs emitted by M/Monit will be the host name on which M/Monit is running. In the example above, it would be localhost.

# Appendix D

## Automatic TLS certificate renewal

You can use a free TLS certificate from a Certificate Authority like LetsEncrypt with M/Monit. LetsEncrypt uses the ACME protocol for automatic certificate requests and renewal, where you have to demonstrate the control over the domain (using either DNS or the web server document root).

In this article, we'll describe how to use the *certbot* utility to request and renew certificates automatically, when M/Monit is accessible from the internet. Note: You can run M/Monit on an isolated host, with different ports and can setup firewall and port forwarding accordingly.



We assume the following:

1. M/Monit is running as a non-privileged user “mmonit” on ports 8080 and 8443 and installed in /home/mmonit/current

2. The M/Monit process is controlled via monit (running as root user), using the following configuration:

```
check process mmonit with pidfile /home/mmonit/current/logs/mmonit.pid
        start program = "/home/mmonit/current/bin/mmonit -d" as uid "mmonit" and gid "mmonit"
        stop program = "/home/mmonit/current/bin/mmonit stop" as uid "mmonit" and gid “mmonit"
```

3. We want to allow access to the plain HTTP port (80) from the internet only during the certificate request and renewal (optional).

4. The user, under which M/Monit is running (“mmonit”) can control the M/Monit process, and allow access to port 8080 on local firewall via “sudo”. Sudo configuration:

```
mmonit  ALL=(root) NOPASSWD: /usr/local/bin/monit restart mmonit
mmonit  ALL=(root) NOPASSWD: /usr/sbin/ufw insert 1 allow from any to any port 8080
mmonit  ALL=(root) NOPASSWD: /usr/sbin/ufw delete allow from any to any port 8080
```

**Initial setup:**

1. Install *certbot* on the machine where M/Monit is running (available as a package for most platforms)

2. A plain HTTP connector is necessary in the M/Monit configuration (conf/server.xml). Add a corresponding **port forwarding**, between the public port 80/tcp and internal 8080/tcp, so M/Monit will be accessible via *http://mmonit.mydomain.com*. Note: do not add a secure connector for 8443 yet:

```
<Connector address="*" port="8080" processors=“10"/>
```

3. Start M/Monit

4. Request the certificate - execute certbot utility as a user, under which M/Monit is running (e.g. “mmonit”):

```
$ certbot certonly \
    -n \
    --http-01-address "M/Monit’s public IP address" \
    -d "M/Monit’s public host name (e.g. mmonit.mydomain.com)“ \
    --work-dir /home/mmonit/letsencrypt \
    --logs-dir /home/mmonit/letsencrypt \
    --config-dir /home/mmonit/letsencrypt \
    --agree-tos \
    --email "your email address" \
    --pre-hook "sudo ufw insert 1 allow from any to any port 8080" \
    --post-hook "sudo ufw delete allow from any to any port 8080; sudo monit restart mmonit" \
    --webroot-path /home/mmonit/current/docroot/ \
    --webroot
```

Notes:

- You can control M/Monit in a different way, for instance with systemd, just replace the restart command in the example with the corresponding command (e.g. “**systemsctl start mmonit**”)

- The configuration of access to the port 8080 via local firewall in pre-hook and post-hook is optional. Replace the *ufw* command with the corresponding firewall utility for your platform.

5. Add the SSL/TLS connector to M/Monit configuration (conf/server.xml):

```
<Connector address="*" port="8080" processors="10"/>
<Connector address="*" port="8443" processors="10" secure="true"/>
<Host name="mmonit.mydomain.com" appBase="."
 certificate="/home/mmonit/letsencrypt/live/mmonit.mydomain.com/fullchain.pem"
 certificateKey="/home/mmonit/letsencrypt/live/mmonit.mydomain.com/privkey.pem" >
…
<Engine name="mmonit" defaultHost="mmonit.mydomain.com" fileCache="10 MB" />
```

6. Restart M/Monit

**Certificate renewal:**

Setup cron to run once a week and check if certificate renewal is necessary (the *certbot* will renew the certificate by default 30 days before the expiration). Here is an example crontab configuration for the "mmonit" user:

```
0 1 * * 0 certbot renew --cert-name mmonit.mydomain.com --work-dir /home/mmonit/letsencrypt --
logs-dir /home/mmonit/letsencrypt --config-dir /home/mmonit/letsencrypt
```

Note: The script will open access to port 8080/tcp during the renewal, so the LetsEncrypt ACME can verify the domain, and close it when the renewal is finished. It will restart M/Monit automatically when the certificate renewal is finished.

