

Install & Admin Guide

for

OpenEMM 21.04

(21.04.000)

AGNITAS AG
www.openemm.org
forum.openemm.org

Author:
Martin Aschoff

Table of Contents

1 Document History and Outlook.....	4
2 Introduction.....	5
2.1 Purpose of OpenEMM.....	5
2.2 General Architecture.....	7
2.3 Open Platform Design.....	7
2.4 Architecture Components.....	7
3 General Requirements.....	9
3.1 Software Stack.....	9
3.1.1 Software Stack Roadmap / End of Life.....	9
3.2 RHEL/CentOS Operating System.....	10
3.3 SLES Operating System.....	11
3.4 MariaDB for RHEL/CentOS.....	12
3.5 MariaDB for SLES.....	13
3.6 MySQL for RHEL/CentOS.....	13
3.7 MySQL for SLES.....	14
3.8 MySQL/MariaDB Configuration.....	14
3.9 DNS Requirements.....	15
4 Server Preparations.....	16
4.1 Firewall Configuration.....	16
4.2 Postfix Deployment.....	16
4.3 Sendmail Deployment.....	19
4.4 Tomcat Deployment.....	20
4.5 Image and PDF Creation Tool wkhtmltox.....	20
4.6 Configuration of Operating System Logging Parameters.....	21
4.7 Miscellaneous.....	22
5 OpenEMM Deployment.....	23
5.1 Download.....	23
5.2 Runtime Deployment.....	23
5.3 OpenEMM Deployment.....	24
5.4 Configuration.....	25
5.5 Startup.....	26

5.6 Test and Production System.....	26
6 OpenEMM Testing.....	28
6.1 OpenEMM Does Not Send Emails.....	28
7 OpenEMM Updates and Upgrades.....	30
7.1 Updating and Upgrading OpenEMM.....	30
7.2 Rollbacks for OpenEMM.....	30
7.3 Do not Skip Major Releases when Upgrading.....	31
7.4 Special Advice for Updating to 21.04.....	32
7.5 Templates and Web Forms.....	33
7.6 Updating Tomcat.....	34
7.7 Preparations before Updating MariaDB.....	34
8 Advanced Configuration.....	36
8.1 Mailloop Configuration.....	36
8.2 Configuration of Default Settings.....	38
8.3 Configuration for MySQL/MariaDB Database.....	39
8.4 Configuration of Webservices.....	40
8.5 Configuration of DKIM Support.....	40
9 OpenEMM Administration.....	42
9.1 Automated Startup.....	42
9.2 Database Backup.....	42
9.3 Generic Database Tuning.....	43
9.4 MySQL/MariaDB Database Tuning.....	43
9.5 Stopping the Sending in Case of Emergency.....	44
9.6 Out of Memory.....	45
9.7 Log Rotation.....	45
9.8 Changing Security-Related Files.....	47
9.9 Switching SMTP Server Type.....	47
10 Apache Native Library.....	49
10.1.1 HTTPS for Tomcat.....	49
11 Glossary.....	51
11.1 Bounce Management.....	51
11.2 DNS.....	51
11.3 FQDN.....	51
11.4 Bounce Management.....	52

11.5 Softbounce Soring.....	52
12 Deployment of Python 3.8 (or later).....	53
13 Third Party Licenses.....	55

1 Document History and Outlook

Version	Date	Change
1.0.2	May 5, 2021	Ne section 8.5 to describe the configuration of DKIM support
1.0.1	April 27, 2021	Section 7.4 added with special info on upgrading to version 21.04
1.0.0	March 26, 2021	Section 3.4+3.5 updated with newly needed libs to install, because OpenEMM now uses MariaDB database driver for MariaDB instead of MySQL driver
	March 26, 2021	Document fork for OpenEMM 21.04

2 Introduction

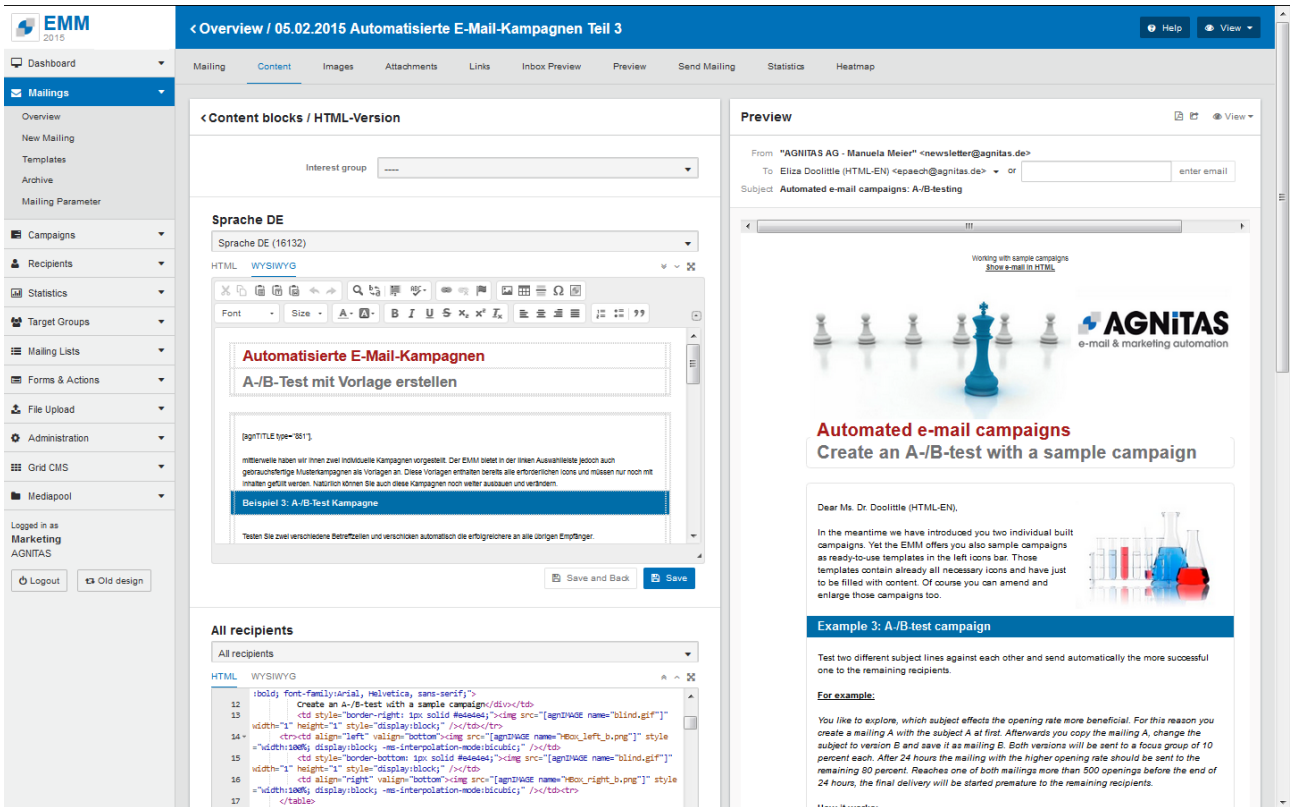
2.1 Purpose of OpenEMM

OpenEMM is a web-based enterprise application for email marketing, email newsletters, service emails (transaction emails and event or time triggered emails), marketing automation and lead management. To summarize it, OpenEMM is a tool for customer relationship management by email.

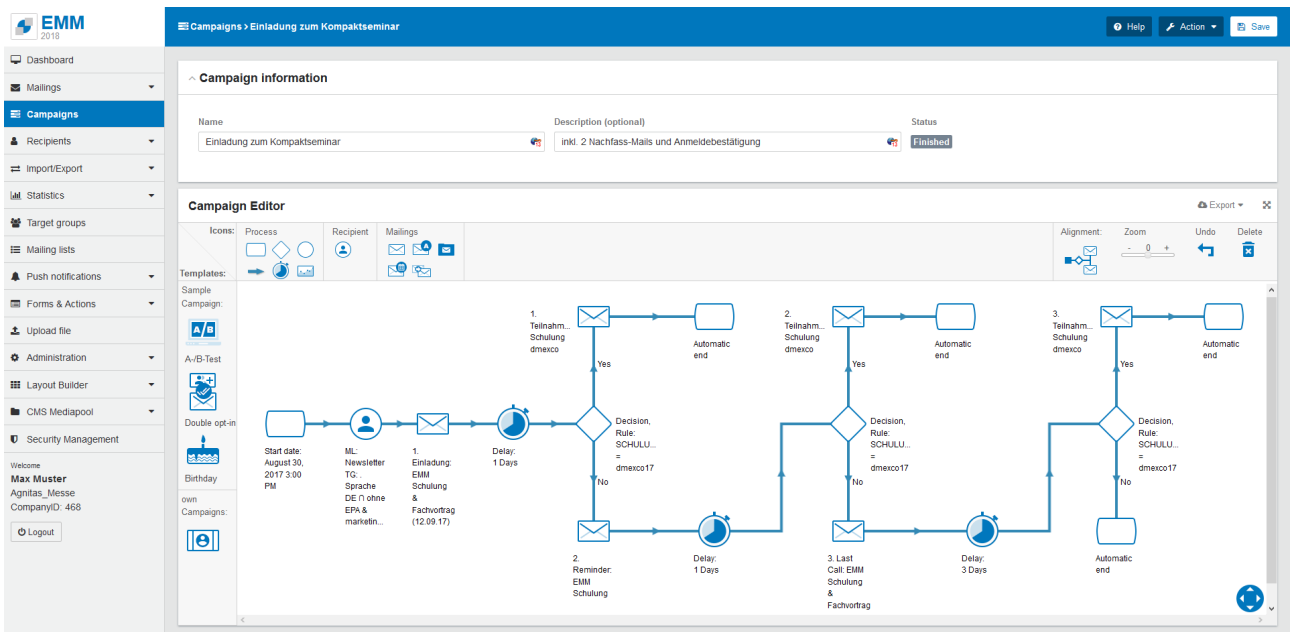
EMM offers tons of features for professional marketing users, among them:

- a console based administration tool for checks, configuration, updates and backups
- a responsive web user interface with great usability and different languages
- a mail template management system
- mailing, template and web forms import and export to load and exchange prepared mailing templates and web forms
- ability to fill mail templates with data from databases and realtime content created on-the-fly
- a graphical workflow manager to create complex campaigns with drag&drop
- individual and (GDPR compliant) anonymous tracking of mail openings, link clicks and deep tracking
- automated bounce management
- graphical realtime statistics with lots of KPIs
- self-defined target groups based on recipient profiles and recipient's behaviour (created visually or with SQL-like syntax)
- a Postfix or Sendmail based mail backend for maximum sending performance with DKIM support
- flexibly configurable data import and export with extensive reporting of results
- sophisticated management of users, user roles and user rights
- an audit-proof searchable and exportable user activity log
- a system status menu with helpful info for OpenEMM administrators
- a scripting feature to enhance the functionality of EMM with triggerable customized actions
- an extensive set of SOAP webservice to manage OpenEMM from remote
- a universal RESTful API to access OpenEMM from remote with HTTP(S) links

The GUI of OpenEMM works with web browsers Google Chrome, Mozilla Firefox and Microsoft Edge. To give you an impression of the web-based user interface of OpenEMM, the following two screenshots illustrate how to build mailings and how to build workflows and campaigns with OpenEMM:



Mailing creation with WYSIWYG editor (upper left) HTML editor (lower left) and preview (right)



Building of workflows and campaigns with OpenEMM's graphical drag&drop editor

2.2 General Architecture

OpenEMM consists of several independent services for scalability. It runs on top of a well proven Open Source software stack without any further dependencies to any commercial software:

- Red Hat Enterprise Linux (**RHEL**), CentOS or Suse Linux Enterprise Server (**SLES**)
- Postfix or Sendmail
- MariaDB or MySQL
- Java OpenJDK
- Apache Tomcat
- Python

2.3 Open Platform Design

A software like OpenEMM must not operate as an island, but it should be capable to be integrated with third party systems like a CRM or CMS software, an e-commerce shop, an ERP platform or a business intelligence software. Therefore, OpenEMM was designed to be a very open and flexible platform with lots of interfaces and extension capabilities.

OpenEMM offers not only a highly customizable graphical interface (GUI) for its users, but also an easy to use URL API für URI tunneling and a rich webservice interface as well as a RESTful API to be used by third-party software.

2.4 Architecture Components

The OpenEMM software is not a monolith, but it is divided into several independent backend and frontend **services** (software). Communication between the various OpenEMM services is done via the database, i.e. the database is the hub of the application.

To gain a better understanding of the purpose of the various backend and frontend services, this is a summary of their main jobs:

- **Backend:** The backend services read mail-related data from the database, generate the individual emails, create mail previews and send the mails out into the Internet. Backend services also register instant bounce messages and collects reponses (like autoresponder and mail replies by recipients) and delayed bounce data to update the database with results from the mail sendings. Backend services send out autoresponder mails defined in the OpenEMM GUI and forward all mail replies not filtered out to a predefined destination (feedback address).
- **GUI** (part of frontend): This service provides the browser-based user interface of OpenEMM.

- **Statistics** (part of frontend): This service generates statistics (tables with numerical values as well as visual charts and diagrams) for the OpenEMM GUI and creates reports in PDF and CSV format for download.
- **Webservices** (part of frontend): This service provides the webservice interface of OpenEMM.

3 General Requirements

This document will guide you through some necessary steps, which are needed to install and configure OpenEMM. It requires a little knowledge of Linux system administration, of MariaDB or MySQL database administration and knowledge of Domain Name Services (DNS) to configure the domain names for various services.

For questions, comments, suggestions for improvement and other contributions to improve this guide, please feel free to use the OpenEMM support forum at forum.openemm.org

3.1 Software Stack

This is the software stack required by OpenEMM **20.10**:

- 64-bit version of **RHEL 7/CentOS 7** or **RHEL 8/CentOS 8** (recommended) or **SLES 15**
- **Java Open JDK 8** or **11** (recommended))
- Apache **Tomcat 9**
- **Python 3.8** or higher
- DBMS: **MariaDB 10.2** to **10.5** (recommended); **MySQL 5.7**
- MTA: **Postfix 2.6** or later (recommended); **Sendmail 8.10** or later
- for image and PDF creation: command line tool **wkhtmltox 0.12.2.1** or later
- for HTTPS support: Apache **Tomcat Native Library**

It may be that OpenEMM also operates on later versions than the ones listed above, but this has not been tested in a production environment, yet.

To install OpenEMM and the required software stack you need a shell access (like *bash*) for your server as user *root*. Unless otherwise noted, you should run all commands as user *root* to make sure you own and you can grant the required permissions.

Instructions that are valid only for RHEL/CentOS, are introduced with a header line **RHEL/CentOS** and are ended with a blank line. Instructions that are valid only for SLES, are introduced with a header line **SLES** and are ended with a blank line, too.

3.1.1 Software Stack Roadmap / End of Life

Here are some additional information on the software stack requirement of future versions of OpenEMM, so that you have enough lead time for your release plannings:

- **Linux:** Support of RHEL 6, which will reach its end of life (EOL) in November 2020, was dropped with OpenEMM **20.10**.
- **Java:** OpenEMM supports LTS releases Java OpenJDK 8 and 11. EOL for OpenJDK 8 will be June 2023 (RHEL) or July 2021 (SLES 15). However, we recommend to switch to Java 11 as soon as possible. OpenEMM **21.10** will be compiled with Java 11 and, therefore, it will need a deployment of Java JDK 11 as runtime environment.

- **DBMS:** Oracle will stop extended support of MySQL version 5.6 by February 2021. Since OpenEMM will not support MySQL 8 in the future (due to compatibility issues), we recommend to switch to a current version of MariaDB in the long run, because Oracle's extended support of MySQL 5.7 will end in 2023.
- **MTA:** If you use Postfix as MTA and send out high volumes of emails, we recommend to use version 3.4 or later, because it offers a significantly increased sending performance (up to 100%). We will drop support of Sendmail with version **21.10** of OpenEMM because Postfix is actively developed further while progress of Sendmail stalls. Also, beginning with version 3.4 performance of Postfix is superior to Sendmail.

3.2 RHEL/CentOS Operating System

Make sure that service *cron* is enabled on all servers and that *SELinux* is disabled on all servers.

Update the operating system to its latest release. This will keep your system in the most stable state and harden it against various intrusion attempts.

Install all required packages. Further dependencies will be resolved and installed automatically by the repository management software.

Install the required packages:

```
# yum update
# yum install gcc make
# yum install xorg-x11-fonts-75dpi zlib fontconfig freetype libX11
libXext libXrender urw-fonts
```

For the backend of OpenEMM 20.10 you have to compile and install Python 3.8 or later. This has to be done after the DBMS is installed and before the database driver is installed and linked with the DBMS libraries (chapter 3.4 to 3.9). See chapter 12 for details on compilation and deployment of Python 3.

The frontend of OpenEMM needs Java. If you want to use Java 8:

```
# yum install java-1.8.0-openjdk-devel
```

If you want to use Java 11:

```
# yum install java-11-openjdk-devel
```

3.3 SLES Operating System

Make sure that service *cron* is enabled on all servers and that *SELinux* is disabled on all servers.

Update the operating system to its latest release. This will keep your system in the most stable state and harden it against various intrusion attempts.

Install all required packages. Further dependencies will be resolved and installed automatically by the repository management software.

Install the required packages:

```
# zypper install gcc
# zypper install zlib fontconfig libfreetype6 libX11-6 libXext6
libXrender1 xorg-x11-fonts
# zypper install zip sudo wget
```

Get a list of all available repositories:

```
# SUSEConnect -list-extensions
```

If you plan to use Java 8, activate the required repository "legacy" (example for SLES 15.1):

```
# SUSEConnect -p sle-module-legacy/15.1/x86_64
```

For OpenEMM 20.10 you have to compile Python 3.8 or later. This has to be done after the DBMS is installed and before the database driver is installed and linked with the DBMS libraries (chapter 3.4 to 3.9). See chapter 12 for details on compilation and deployment of Python 3.

If you want to use Java 8:

```
# zypper install java-1_8_0-openjdk java-1_8_0-openjdk-devel
```

If you want to use Java 11:

```
# zypper install java-11-openjdk java-11-openjdk-devel
```

Install and enable the required logging service:

```
# zypper install rsyslog
# systemctl start rsyslog
# systemctl enable rsyslog
```

Furthermore, directory */usr/sbin* has to be included in the *PATH* variable for all new users. Therefore, add the following line to file *.profile* in directory */etc/skel* before you create the user for OpenEMM:

```
export PATH=$PATH:/usr/sbin
```

3.4 MariaDB for RHEL/CentOS

If you want to use OpenEMM with a MariaDB database, you have to install the database software before you install Python and the Python database driver for MariaDB.

If MySQL is pre-installed on your server, you have to remove it first before installing MariaDB:

```
# systemctl stop mysql
# yum remove mysql*
```

RHEL 7:

Since RHEL/CentOS 7 do **not** offer MariaDB 10.2 or later out of the box, you should include the official MariaDB repository. Use link <https://downloads.mariadb.org/mariadb/repositories> to find the repository suitable for you and follow the instructions by MariaDB.

Install the MariaDB server and client with

```
# yum install MariaDB-server MariaDB-client
```

and install the required MariaDB API and libs with

```
# yum install MariaDB-devel MariaDB-shared
```

RHEL 8:

RHEL/CentOS 8 offer MariaDB 10.3 out of the box. Install the MariaDB server and client with

```
# yum install mariadb-server mariadb
```

and install the required MariaDB API and libs with

```
# yum install mariadb-devel mariadb-connector-c
```

Finally, you need to install Python 3 (see chapter 12) and finalize the configuration with installing the database driver:

```
# python3 -m pip install mariadb
```

To start MariaDB and to make sure it starts every time the server is rebooted:

```
# systemctl enable --now mariadb
```

Tip: After installation of MariaDB, you should consider running script *mysql_secure_installation* to set a password for user root, to remove any anonymous users and to disallow remote logins.

It is not necessary to create a separate database user for OpenEMM, because the OMT (OpenEMM Maintenance Tool) will take care of it (see section *Runtime Deployment* below).

3.5 MariaDB for SLES

SLES supports MariaDB out of the box:

```
# zypper install mariadb mariadb-devel libmariadb-devel
```

Now you need to install Python 3 (see chapter 12) and finalize the configuration with installing the database driver:

```
# python3 -m pip install mariadb
```

To start MariaDB and to make sure it starts every time the server is rebooted:

```
# systemctl start mariadb
```

```
# systemctl enable mariadb
```

Tip: After installation of MariaDB, you should consider running script *mysql_secure_installation* to set a password for user root, to remove any anonymous users and to disallow remote logins.

It is not necessary to create a database user for OpenEMM, because the OMT (OpenEMM Maintenance Tool) will take care of it (see section *Runtime Deployment* below).

3.6 MySQL for RHEL/CentOS

If you want to use OpenEMM with a MySQL database, you have to install the database software before you install Python and the Python database driver for MySQL. If your version of RHEL/CentOS does not offer MySQL 5.7 out of the box, you should include the Software Collections Repository (SCL) for **RHEL 7** with

```
# subscription-manager repos --enable rhel-server-rhsc1-7-rpms
```

or for **CentOS** with

```
# yum install centos-release-scl
```

If an older version of MySQL is pre-installed on any of your servers, you have to remove it first before installing a later MySQL version manually:

```
# systemctl stop mysql
```

```
# yum remove mysql*
```

Now, install MySQL 5.7 with

```
# yum install rh-mysql57-mysql-server mysql-libs mysql-devel
```

As next step you need to install Python 3 (see chapter 12) and finalize the configuration with installing the database driver:

```
# python3 -m pip install mysqlclient
```

To start MySQL and to make sure it starts every time the server is rebooted:

```
# systemctl enable rh-mysql57-mysql  
  
# systemctl start rh-mysql57-mysql
```

Finally, set a symlink to make sure that OpenEMM can find MySQL:

```
# cd /bin  
  
# ln -s /opt/rh/rh-mysql57/root/bin/* .
```

Tip: After installation of MySQL, you should consider running script *mysql_secure_installation* to set a password for user root, to remove any anonymous users and to disallow remote logins.

It is not necessary to create a database user for OpenEMM, because the OMT (OpenEMM Maintenance Tool) will take care of it (see section *Runtime Deployment* below).

3.7 MySQL for SLES

Go to chapter "MariaDB for SLES" before and install MariaDB instead of MySQL.

3.8 MySQL/MariaDB Configuration

Set the MySQL/MariaDB root password right after installation with

```
# mysqladmin -u root password '<password>'
```

and save it to file *.mysqlpw*:

```
# cd /root  
  
# vi .mysqlpw  
  
# chmod 600 .mysqlpw
```

MySQL and MariaDB offer several modes of operation. OpenEMM requires the following setting in section *[mysqld]* of DBMS configuration file *my.cnf* (usually located in directory */etc*):

```
sql-mode = "STRICT_ALL_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION"
```

Please make sure these three modes (and only these three modes) are set! If the *strict* mode is not selected, if OpenEMM tries to import a string into a database field which is too short to hold the complete string, MySQL/MariaDB would simply cut off the string at the end to make it fit - without any warning! This would harm the integrity of your data.

Preventing *auto-creation* of new database users is a security setting. The ban of *substitutions* makes sure that MySQL/MariaDB uses OpenEMM's choice of database engine InnoDB in any case.

You might want to change further parameters in configuration file *my.cnf* like commenting out *skip-networking* in section *[mysqld]* or setting *bind-address* to the IP address of your database server.

Especially important are parameter *lower_case_table_names*, which should be set to 1 on Linux systems (to enforce lower case table names), and parameter *wait_timeout*, which is set to 28800 by default. This means that MySQL/MariaDB automatically cuts the connection to OpenEMM after 8 hours of inactivity. This leads to an initial connection error when OpenEMM attempts to contact the MySQL/MariaDB database next time. If your OpenEMM installation does not access its MySQL/MariaDB database all the time, you should increase this value to at least one day (86400) or even to a whole week (604800).

Please do not forget to restart MySQL/MariaDB after changing the configuration file. For more advice on how to configure the database, please check out the database documentation.

3.9 DNS Requirements

When setting up the DNS entries for your OpenEMM server, please make sure that your server holds a valid A record and a PTR record which points back to the hostname of your server (see */etc/hosts*) for reverse lookups. This is important because most mailservers that receive emails from your OpenEMM installation will do a reverse DNS lookup in order to check if the FQDN of your server and the PTR record of your server's IP address match. If not, this is an indication of a spambot network and quite often your emails will be rejected.

To be reachable from outside via its FQDN is also important for the GUI service, because this service needs to be able to access itself via the Internet in order to generate and display preview thumbnails, heatmaps and the like.

If you plan to use an SPF entry for the domain which is used for the sending address of your mass mails, make sure to add your server to this SPF record.

4 Server Preparations

Before you are able to install OpenEMM on the server you have to prepare the server first.

Create a group and a user openemm:

```
# groupadd openemm
# useradd -m -g openemm -d /home/openemm -s /bin/bash openemm
# passwd openemm
# su - openemm
```

4.1 Firewall Configuration

Open port 25 and port 8080 in your firewall and add a port forwarding from port 80 to 8080, so you do not have to enter the URL of your OpenEMM server with "":8080" at the end:

```
# firewall-cmd --get-active-zones
```

If your zone is "public" (if not, use the zone name you got with the aforementioned statement):

```
# firewall-cmd --zone=public --add-port=8080/tcp --permanent
# firewall-cmd --zone=public --add-port=25/tcp --permanent
# firewall-cmd --zone=public --add-forward-
port=port=80:proto=tcp:toport=8080 --permanent
# firewall-cmd --reload
```

4.2 Postfix Deployment

RHEL/CentOS:

If you want to use Postfix as SMTP server (MTA), you have to stop and remove Sendmail first (in case it is installed), and you have to install the required packages for Postfix.

```
# systemctl stop sendmail
# yum remove sendmail
# yum install postfix sendmail-milter procmail
```

Further dependencies will be resolved and installed automatically by the repository management software.

Switch the default SMTP server to Postfix with

```
# alternatives --set mta /usr/sbin/sendmail.postfix
```


SLES:

Since SLES does not offer Sendmail, no need to un-install it. Install the required Postfix packages with

```
# zypper install postfix procmail libmilter1_0
```

and create a symlink so that OpenEMM can find the Postfix mail log file:

```
# ln -s /var/log/mail /var/log/maillog
```

After installation of Postfix, you have to change its configuration to unleash all features. To do this, change to the Postfix main configuration directory:

```
# cd /etc/postfix
```

Add some configuration parameters to Postfix' main configuration file *main.cf*:

```
inet_interfaces = all
inet_protocols = all
mailbox_command = /usr/bin/procmail
mailbox_size_limit = 0
message_size_limit = 0
maximal_queue_lifetime = 1d
bounce_queue_lifetime = 1d
smtp_tls_security_level = may
smtp_tls_protocols = !SSLv2, !SSLv3
smtp_tls_ciphers = high
smtp_tls_mandatory_ciphers = $smtp_tls_ciphers
hash_queue_depth = 2
enable_long_queue_ids = yes
relay_domains = /home/openemm/var/run/relay.domains
transport_maps = hash:/home/openemm/var/run/transport.maps
smtpd_milters = unix:/home/openemm/var/run/bav.sock
```

If lines with parameters of the same name already exist in file *main.cf*, comment them with character **#** at the beginning to avoid any warning messages or overwrite them with the new values in case you do not want to keep the original values as backup.

The two files *relay.domains* and *transport.maps* as well as Linux socket *bav.sock* are automatically created at first startup time of the mailloop service. File *relay.domains* specifies your mailloop service domain name, so that responses like auto-replies and bounces sent to an email address with this domain name are accepted by Postfix for relaying. File *transport.maps* defines for the mailloop service domain name the service used for processing (“mailloop”).

Additionally, you have to set parameter *myhostname* in file *main.cf* to the FQDN of your OpenEMM server. Otherwise, mails would be sent with sender domain *localhost.localdomain* instead.

If you want to be able to receive autoresponder, bounce and feedback mails encrypted with the TLS protocol, add

```
smtpd_use_tls = yes
smtpd_tls_loglevel = 2
smtpd_tls_security_level = may
smtpd_tls_received_header = yes
smtpd_tls_session_cache_timeout = 3600s
smtpd_tls_eecdh_grade = strong
smtpd_tls_cert_file = /etc/pki/tls/certs/mail.crt
smtpd_tls_key_file = /etc/pki/tls/private/mail.key
smtpd_tls_CAfile = /etc/pki/tls/certs/cacert.pem
smtpd_tls_CApath = /etc/pki/tls/certs
smtpd_tls_protocols = !SSLv2, !SSLv3
smtpd_tls_ciphers = high
smtp_tls_protocols = $smtpd_tls_protocols
smtp_tls_ciphers = $smtpd_tls_ciphers
tls_random_source = dev:/dev/urandom
tls_random_prng_update_period = 3600s
```

to file *main.cf*. You may have to change the paths to the locations where your TLS keys are located. Certificate *mail.crt* may be a self-signed certificate.

Finally, the configuration parameters for service "mailloop" are defined in configuration file *master.cf*. Add these two lines:

```
mailloop unix - n n - - pipe
    flags=RX user=openemm argv=/usr/bin/procmail /home/openemm/lib/bav.rc
```

Please do **not** omit the two space characters before keyword "flags" to indicate the parser that the line is continued!

Last not least, activate the TLS manager in file *master.cf* by uncommenting (i.e. removing the leading #) line

```
tlsmgr unix - - n 1000? 1 tlsmgr
```

To activate all changes, restart Postfix with

```
# systemctl restart postfix
```

In case you start Postfix separately from OpenEMM and Postfix complains that file *relay.domains* is missing, you can ignore this warning because OpenEMM take care to create this file at startup time in case it is missing.

4.3 Sendmail Deployment

If you want to use Sendmail as SMTP server (MTA), you have to stop and remove Postfix first (in case it is installed), and you have to install the required packages for Sendmail.

```
# systemctl stop postfix
```

```
# yum remove postfix
```

```
# yum install sendmail sendmail-cf sendmail-milter procmail
```

Further dependencies will be resolved and installed automatically by the repository management software.

After that, if necessary, switch the default SMTP server to Sendmail with

```
# alternatives --set mta /usr/sbin/sendmail.sendmail
```

and change Sendmail's configuration: Open file *sendmail.mc*: in directory */etc/mail* and change the old line

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

to the new line

```
dnl DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

This will enable Sendmail to listen on all available network interfaces and allows the processing of asynchronous bounces and auto responder mails. By default, Sendmail listens only on the local interface for connections, but *dnl* comments out this directive.

Additionally, add line

```
INPUT_MAIL_FILTER(`bav', `S=unix:/home/openemm/var/run/bav.sock, F=T')dnl
```

at the end of the file to make Sendmail familiar with mail filter BAV (bounce address validator) so that the bounce management can process asynchronous bounces, too.

If you want to be able to receive autoresponder, bounce and feedback mails encrypted with the TLS protocol, add

```
define(`confCACERT_PATH', `/etc/pki/tls/certs')dnl
```

```
define(`confCACERT', `/etc/pki/tls/certs/cacert.pem')dnl
```

```
define(`confSERVER_CERT', `/etc/pki/tls/certs/mail.crt')dnl
```

```
define(`confSERVER_KEY', `/etc/pki/tls/private/mail.key')dnl
```

to file *sendmail.mc*. You may have to change the paths to the locations where your TLS keys are located. Certificate *mail.crt* may be a self-signed certificate.

The mailloop service of OpenEMM, which processes the asynchronous bounces and auto responder mails needs some more preparations:

```
# cd /etc/mail
# umask 022
# [ -f relay-domains ] || touch relay-domains
# [ -f mailertable ] || touch mailertable
```

These two files *relay-domains* and *mailertable* are automatically filled at first startup time of the mailloop service. File *relay-domains* specifies your mailloop service domain name, so that responses like auto-replies and bounces sent to an email address with this domain name are accepted by Sendmail for relaying. File *mailertable* contains for the mailloop service domain name the internal file path for forwarding the responses to procmail for mail filtering.

To activate your changes to the Sendmail configuration you have to compile the intermediate file *sendmail.mc* to the final configuration file *sendmail.cf* and activate the new configuration with

```
# cd /etc/mail
# make
# systemctl restart sendmail
```

SLES:

Go to the chapter before and install Postfix instead of Sendmail.

4.4 Tomcat Deployment

OpenEMM needs web application engine Tomcat for the frontend services (GUI, statistics and webservices). Tomcat can be installed with OpenEMM Maintenance Tool *OMT.sh*. For details please read section *Runtime Deployment* below.

If you want to operate OpenEMM with the HTTPS protocol, the server key files (**.key*, **.pem*) and server certificate files (**.crt*, *cacerts*) for the TLS configuration (to allow HTTPS connections) must be provided from your side as these files are server and client specific. See chapter 10 for details.

4.5 Image and PDF Creation Tool wkhtmltox

In order to create good looking thumbnail images and PDF documents in the OpenEMM GUI, the OpenEMM server needs the powerful tool *wkhtmltox* to be installed. For RHEL/CentOS, the tool can be installed with OpenEMM Maintenance Tool *OMT.sh*, or you can do it manually with

```
#yum install
https://github.com/wkhtmltopdf/packaging/releases/download/0.12.6-1/
wkhtmltox-0.12.6-1.centos7.x86_64.rpm
```

for CentOS 7 or

```
# yum install
https://github.com/wkhtmltopdf/packaging/releases/download/0.12.6-1/
wkhtmltox-0.12.6-1.centos8.x86_64.rpm
```

for CentOS 8.

If a later version of *wkhtmltox* is available, you may use the most recent one. But do not use the *wkhtmltox* package of the operating system, because it may be buggy.

SLES:

At press time, no current build of *wkhtmltox* was available for SLES. Therefore, you have to use version 0.12.4, which still has a generic build. You can get *wkhtmltox* at GitHub, and you can install it in */opt* and create a symlink in */usr/local/bin*:

```
# cd /opt
# wget
https://github.com/wkhtmltopdf/wkhtmltopdf/releases/download/0.12.4/
wkhtmltox-0.12.4_linux-generic-amd64.tar.xz
# tar xJpf wkhtmltox-0.12.4_linux-generic-amd64.tar.xz
# rm wkhtmltox-0.12.4_linux-generic-amd64.tar.xz
# chown -R root. wkhtmltox
# ln -s /opt/wkhtmltox/bin/wkhtmlto* /usr/local/bin/
```

4.6 Configuration of Operating System Logging Parameters

With version 7 of RHEL/CentOS a new "feature" was introduced that drops messages from being logged if the server has a high workload. If you use the default configuration values of RHEL/CentOS 7 (i.e. 1,000 entries max. within 30 seconds), and if the OpenEMM server has a high workload due to a high mail output, this can lead to missing entries in the maillog. However, missing entries in the maillog mean that OpenEMM does not know whether mails were delivered successfully, or not and may lead to an incomplete bounce management and incomplete statistics!

RHEL/CentOS 7:

To make sure that even under high workload all messages are logged to the maillog, you should change (or add) the following values of file *journald.conf* in directory */etc/systemd*:

```
RateLimitInterval=10s
RateLimitBurst=10000
```

RHEL/CentOS 8:

Change in file *journal.d.conf* in directory */etc/systemd* lines

```
module(load="imjournal"                # provides access to the systemd journal
        StateFile="imjournal.state")   # File to store the position in the journal
to
module(load="imjournal" StateFile="imjournal.state" ratelimit.interval="10"
        ratelimit.burst="10000")
```

Afterwards, restart the journal daemon with

```
systemctl restart systemd-journald
```

to activate your changes. Additionally, you should add (or change) the following values in file *rsyslog.conf* of directory */etc* after the line with parameter *\$IMJournalStateFile imjournal.state*:

```
$imjournalRateLimitInterval 60
$imjournalRateLimitBurst 60000
```

Afterwards, restart the rsyslog service to activate your changes:

```
systemctl restart rsyslog
```

4.7 Miscellaneous

OpenEMM needs a minimum value of 16384 for kernel parameter *nofile*, which defines the maximum number of open files per process. OpenEMM Maintenance Tool *OMT* will check and change if necessary.

OpenEMM requires read access to the mail log file at */var/log/maillog* and *logrotate* has to be aware of this fact, too. Open file */etc/logrotate.d/syslog* and add the following line after the line *sharedscripts*:

```
# create 0644
```

and run

```
# chmod 644 /var/log/maillog
```

to set the permissions of the current maillog.

5 OpenEMM Deployment

5.1 Download

We recommend to download the deployment & runtime package of OpenEMM from <https://www.agnitas.de/en/download/openemm-binaries/>. This package provides an installer tool for the backend and frontend code and eases installation, updates, administration and maintenance of OpenEMM significantly.

Of course, you can also download the source code of OpenEMM from GitHub at <https://github.com/agnitas-org/openemm> and compile and deploy the software manually yourself. See the build instructions at the end of the OpenEMM wiki page at <https://wiki.openemm.org>.

5.2 Runtime Deployment

Download the OpenEMM runtime tarball (file name: *openemm-runtime-20.10.*.tar.gz*) to */home/openemm* to create the deployment and runtime environment for OpenEMM. Change to user *openemm*, unpack the tarball and start it with

```
# su - openemm
$ tar -xvzpf openemm-runtime-20.10.*.tar.gz
$ OMT.sh
```

While OpenEMM needs Python 3.8 or higher to work, *OMT* itself is fine with any version 3 of, i.e. the *python3* package of the operating system would be sufficient.

At launch time *OMT* checks if kernel parameters in */etc/security/limits.conf* are big enough and, if not, offers to change them. To be able to change this file, you need to start *OMT* as user *root*. After the change of the kernel parameters, you have to reboot the server.

At launch time *OMT* also checks your software stack and offers to install required packages (including Tomcat 9 and Tomcat-Native for HTTPS support) and to set environment variables

- JAVA_HOME (default path: */usr/lib/jvm/java*) for Java
- CATALINA_HOME (default path: */home/openemm/opt/tomcat/* for Tomcat
- Tomcat-Native (default path: */home/openemm/opt/tomcat-native*)
- WKHTMLTOPDF (default path: */usr/bin/wkhtmltopdf*) for PDF tool *wkhtmltox*
- WKHTMLTOIMAGE (default path: */usr/bin/wkhtmltoimage*) for image tool *wkhtmltox*

These parameters will be written to file *setenv.sh* in directory */home/openemm/bin/*.

If one of these tools is not available on your server yet, *OMT* offers to install it for you.

Note: If you want to install Java or tool *wkhtmltox*, you have to restart *OMT* as user *root*:

```
# sudo su -
# /home/openemm/bin/OMT.sh
```

OMT helps you to set up and configure a database connection (file *dbcfg*) including database user and password. In menu *Configuration* → *Change database configuration* you can enter the required properties like database type ("mysql" or "mariadb"), database name ("openemm"), database user (suggestion: "openemm") and your database password. If the database configuration file *dbcfg* does not exist in directory */home/openemm/etc/*, it is also created from scratch. A valid *dbcfg* file looks like this:

```
openemm: dbms=mariadb, name=openemm,
jdbc-connect=jdbc:mariadb://127.0.0.1/openemm?
zeroDateBehavior=convertToNull&useUnicode=true&characterEncoding=UTF-
8, host=127.0.0.1, user=openemm, jdbc-driver=org.mariadb.jdbc.Driver,
password=<password>
```

If the OpenEMM database and its user does not exist, *OMT* offers to set it up for you after you have entered the properties mentioned above. The empty OpenEMM database will be filled later during installation of the OpenEMM code package.

If for whatever reason the creation of the database user for OpenEMM does not work, you can create the user manually with

```
# mysql -u root -p
MariaDB > CREATE USER 'openemm'@'localhost' identified by '<password>';
MariaDB > GRANT ALL PRIVILEGES ON `openemm`.* TO 'openemm'@'localhost' ;
MariaDB > GRANT SELECT ON mysql.proc TO 'openemm'@'localhost';
MariaDB > GRANT RELOAD ON *.* TO 'openemm'@'localhost';
MariaDB > GRANT SUPER ON *.* TO 'openemm'@'localhost';
```

5.3 OpenEMM Deployment

For the deployment of the openemm code tarball you have to start *OMT* as user *root*, because some files of the tarball have to be deployed with root permissions:

```
$ sudo su -
# /home/openemm/bin/OMT.sh
```

Use menu *Install or update package from AGNITAS website* to install the openemm code package. As first step, the menu offers to install an update of the runtime package, if a later version is available. Opposed to upgrading to a new major version of OpenEMM, an update to a new runtime version is always recommended.

After that, versions of Tomcat and the Tomcat Native library are offered, but you can always select "n" for any package, if you do not want to install it or if it already exists. Please be aware that even if you install the Tomcat native library, you have to take care of

its configuraion to be able to use the HTTPS protocol for OpenEMM (see chapter 10 for details).

Answer “no” to all downloads offers until the openemm code package is offered (if you started *OMT* as user *root*). If an upgrade of a major version of OpenEMM is available (like 21.04) this upgrade opportunity is offered first. However, you should not blindly accept it if you are working on a production system (see section *Test and Production System* below).

During Deployment of the openemm code tarball several new directories and symlinks will be created in directory `/home/openemm/`. Also, the OpenEMM database is automatically populated with its initial content.

If you do not want to install or update the code of OpenEMM, you can start *OMT* as user *openemm*:

```
# su - openemm
$ OMT.sh
```

5.4 Configuration

After deployment of the OpenEMM code but before launching it with the restart menu, use menu *Configuration* → *Change basic configuration* to set up and change the configuration of OpenEMM. Note: It is possible to launch OpenEMM without adapting the configuration, but in this case not all features of the software will work properly.

With OpenEMM 20.04 all configuration properties have been migrated from files *emm.properties*, *emm-ws.properties* and *Mailgun.ini* to the OpenEMM database. Therefore, you do not change these files any longer but the corresponding database entries listed in the sub-menu mentioned above. The advantage of the migration from files to database is, that further updates of OpenEMM do not overwrite any individual settings you might have made.

Sub-menu *Change basic configuration* asks you to enter the full URL of your server including the protocol. From this value it deviates the settings for various essential properties.

Sub-menu *Change client/account configuration* offers you to change the redirect domain and the mailloop domain. These value are pre-filled by the basic configuration and you should change those values only if you know what you are doing:

- Set *rdir_domain* to the protocol and FQDN of your server, for example `https://openemm.example.com`. This domain will be used in all measureable links to redirect them through OpenEMM. This property **must** include the appropriate protocol.
- Set *mailloop_domain* to the domain of your sender address. The domain for the mailloop service must be different from the name of your OpenEMM server. It usually is the FQDN which is defined in the MX record for your server, for example `mailing.example.com`, pointing as MX to `openemm.example.com`. In this case, use domain `mailing.example.com` as domain for the sender address of your mailings. (There are other ways for configuration, described in chapter *Mailloop Configuration*.) If you do not configure the mailloop service, OpenEMM can only process instant bounces, i.e. you will not be able to catch all bounces. This property **must not** include any protocol.

Finally, create an initial password for admin user "emm-master" in menu *Security* → *Create new initial 'emm-master' password*. Write down the generated password and use user "emm-master" and this initial password later for your first login in the GUI of OpenEMM. At first login, you will be prompted to change your password to a new one of your choice. In case you forget your password for the admin user at a later time, you can always use this menu of *OMT* to set a new password.

5.5 Startup

Launch OpenEMM with menu *Restart OpenEMM* of *OMT*. Experts may have a look at the Tomcat log during startup with

```
$ tail -f /home/openemm/logs/catalina.out
```

to check for any warnings or error messages. Be patient, because the startup process will take a minute or two.

After OpenEMM has been launched successfully, point your browser to your OpenEMM server and log in with user "emm-master" and the initial password you just created. OpenEMM will ask you to change the initial password to a new one of your choice.

If you want to install the context-sensitive online help feature of OpenEMM (including the user manual with about 500 pages), visit

- <https://www.agnitas.de/en/download/openemm-manual/> (English)
- <https://www.agnitas.de/download/openemm-handbuch/> (German)

to get the download link for the manual package. You can use this link in *OMT's* menu *Install or update package from AGNITAS Cloud* to install the documentation.

5.6 Test and Production System

If OpenEMM is an important software for you (and we hope it is!) we recommend to set up a separate test system alongside your production system. This should be no problem for you, because the complete software stack for OpenEMM is open source and you do not have to pay a dime for the additional instance.

You can create a copy of your OpenEMM production database with menu *Database Maintenance* → *Backup MySQL/MariaDB database* of OpenEMM Installer. On your test system you should only create the empty OpenEMM database with menu *Configuration* → *Change configuration of database connection*. Now you can set up the database copy of your production system with menu *Database Maintenance* → *Restore MySQL/MariaDB database*. But make sure that both systems use the same DBMS version to avoid any trouble.

After that, you only have to adjust the properties containing the server URL in menu *Configuration* → *Change basic configuration*, because they have been copied from your production database as well. Now you are ready to use your test system to try out different

configuration settings, to test database backup and restore or to test an upgrade to a new major OpenEMM version – all that before applying any change to your precious production system.

6 OpenEMM Testing

For testing (and to initialize the system) you should first make sure that MySQL/MariaDB contains the OpenEMM database and the DBMS is already up and running.

If you are not sure about OpenEMM's status use the first menu *Show OpenEMM status of OMT* to check it and use menu *Restart OpenEMM* to restart the software if it does not look right.

To test the correct operation of each service we recommend

- creating a new mailing (testing the GUI of the frontend)
- sending the mailing to an existing email address (testing the backend)
- and sending it to a non-existing email address (testing the bounce management)*
- opening the mail and clicking the links (testing the redirect service of the frontend)
- checking in mailing statistics if openings and clicks were recorded (testing the redirect service and statistics service of the frontend)

*To test the bounce management you have to send the mailing not to admin or test recipients, but to normal recipients, because due to performance reasons mailings to admin and test recipients are sent out instantly without the overhead of a regular mailing. For that reason the bounce management process can not be applied to mailings for admin and test recipients.

6.1 OpenEMM Does Not Send Emails

The most "popular" problem of OpenEMM is, that no emails are sent out. Therefore, we have compiled a checklist with the ten most common reasons why OpenEMM does not send emails:

1. Packages *postfix* and *sendmail-milter* or *sendmail* and *sendmail-cf* are not installed.
2. Postfix or Sendmail has not been started ("systemctl start postfix" or "systemctl start sendmail").
3. Postfix or Sendmail is no longer running ("ps aux | grep postfix/master" or "ps aux | grep sendmail").
4. The wrong MTA is active (Postfix, Sendmail, qmail, Exim, etc.).
5. Port 25 has not been opened.
6. Service *cron* is disabled.
7. SELinux is enabled.
8. Postfix' mailqueue at */var/spool/postfix* or Sendmail's mailqueues at */home/<user>/var/spool/mail* does not exist (Postfix: re-install Postfix, Sendmail: restart Sendmail).
9. For reverse lookups, no PTR record does exist for the IP address of the OpenEMM server (so that mails are blocked as spam by mailbox providers on the receiving end).

10. Postfix' or Sendmail's mail log file in */var/log/maillog* shows errors.

7 OpenEMM Updates and Upgrades

With OpenEMM we make a difference between **updates** (changing to a new minor release, like from 20.10.000.050 to 20.10.000.100) and **upgrades** (changing to a new major release, like from 20.04 to 20.10).

If you plan to do an **upgrade**, we strongly recommend to **update** to the latest minor release of your OpenEMM version, first. Only after that, execute the upgrade to the new major version. This makes sure, that the delta between the two major versions is as little as possible and the old version is well prepared for the upgrade.

If you do not operate a test environment for OpenEMM, but you apply updates and upgrades right to the production environment, **make sure to create a backup with OMT → Database Maintenance → Backup MySQL/MariaDB database before** (even more so, if you upgrade to a beta version of OpenEMM), because you might need the database backup in case of a roll back (see rollback section below for details).

7.1 Updating and Upgrading OpenEMM

If you plan to upgrade OpenEMM 21.04, please check section 7.4 for required updates to your current software stack.

The OpenEMM Wiki at <https://wiki.openemm.org> shows the latest available versions of the runtime and openemm code package at the top.

You can download the latest versions of the packages with menu *Install or update package from AGNITAS Website* of OMT. Please answer “no” to all download offers until the openemm code package is offered.

An update of the openemm code package will also update the schema of the OpenEMM database, if necessary. Please keep in mind that you have to start OMT as user *root*, if you want to update the openemm code package.

Do not forget to restart OpenEMM with menu *Restart OpenEMM* after you have installed a new openemm code package to activate it. But do the restart at a convenient time: Do not restart OpenEMM during a dispatch of a mailing, or right after the dispatch (due to the brief downtime of the redirect service which would cause missed openings and click redirects).

7.2 Rollbacks for OpenEMM

If you are not happy with a version you updated or upgraded to, you can roll back OpenEMM to a former version or roll forward to a later version with menu *Switch OpenEMM version*. This menu lists the active versions of the EMM services available on the current server so that you can switch separately

- the runtime environment (including OMT itself)
- the openemm frontend (consisting of GUI service, statistics service and webservice service)
- the openemm backend (consisting of various services)

- the manual (including all context sensitive help pages)

After you have selected a specific service you get a list of all versions available on the server so that you can choose the version you want to activate.

Warning: If you switch back the openemm frontend or backend not to an older minor version (last 3 digits of the version number), but to an older major version (like going back from 20.10 to 20.04), be advised that the database schema can not be rolled back and, therefore, some feature may not work properly in case it wants to use a database field that does no longer exist in the new database schema.

Therefore, if you want to switch back OpenEMM to an older major version, we recommend to restore the database backup of the corresponding version.

If you need to know which services and versions of OpenEMM were active at what time in the past, menu *Show update history* of OMT provides a list of all available services, their versions and the exact startup times.

Do not forget to restart OpenEMM with menu *Restart OpenEMM* after you have switched the frontend or backend version of OpenEMM.

If you get offered updates only to the latest version of OpenEMM, you have to switch back the runtime version. For instance, a runtime version 20.10 only offers you updates to versions of OpenEMM 20.10. But as soon as you have upgraded to a runtime version 21.04, you get offered upgrades to OpenEMM 21.04 as well.

7.3 Do not Skip Major Releases when Upgrading

When updating OpenEMM, do not skip a major release, i.e. if you use OpenEMM 19.10 and you plan to upgrade to OpenEMM 20.10, **do not skip** the upgrade to version 20.04! You have to insert upgrades to the intermediate major versions in order to make sure that no migration processes for the OpenEMM database are missing.

To make sure that you are able to upgrade to each major release of OpenEMM, update to the runtime version of the next major release and after that download and update the openemm code package. **Do not** update the runtime package to a new major release **without** updating the openemm code package, too!

If you have skipped updating openemm code packages while updating the runtime package, you have to use the switch menu to switch back to an older runtime version (see rollback section above for details).

Note: If you need to switch back to a 20.04 release and you are not offered runtime 20.04.050 or later, please make a copy of your individual file *server.xml* in directory */home/openemm/tomcat/conf* because runtimes before 20.04.050 used to overwrite the existing *server.xml* when switching back.

Even if you do not want to use an intermediate version, you have to restart this version (like 20.04 in the above-mentioned case) once. This makes sure that the startup process in the OpenEMM code is executed and initiates all pending migration tasks. For example, the startup process of OpenEMM 20.04 migrates all configuration properties from configuration file *emm.properties* into the database (among other things). This migration is important, because OpenEMM 20.10 reads its configuration properties from the database.

7.4 Special Advice for Updating to 21.04

Starting with release 21.04, OpenEMM uses the MariaDB database driver instead of the MySQL database driver, if a MariaDB is used for OpenEMM. Therefore, before you upgrade to OpenEMM 21.04, you need to install some additional packages and modules.

RHEL 7/CentOS 7:

If you used the MariaDB packages of RHEL/CentOS (rh-mariadb102) before, you have some work to do! First, you need to include the official MariaDB repository. Use link <https://downloads.mariadb.org/mariadb/repositories> to find the repository suitable for you and follow the instructions by MariaDB.

At first, make a backup of your OpenEMM database – just in case!

After that, you have to uninstall MariaDB and re-install MariaDB from the official MariaDB repository.

Stop OpenEMM and MariaDB:

```
# sudo -u openemm /home/openemm/bin/openemm.sh stop
# systemctl stop rh-mariadb102-mariadb
# systemctl disable rh-mariadb102-mariadb
```

Make sure, that directory `/var/lib/mysql/` is empty and move the MariaDB files there:

```
# ll /var/lib/mysql/
# mv /var/opt/rh/rh-mariadb102/lib/mysql /var/lib/
```

Install the official MariaDB repository:

```
# vim /etc/yum.repos.d/MariaDB.repo
```

This is an example for our settings:

```
# MariaDB 10.5 CentOS repository list - created 2021-04-12 12:48 UTC
# http://downloads.mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.5/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

Now, you can install the new version of MariaDB:

```
# yum install MariaDB-server MariaDB-client MariaDB-shared MariaDB-devel
```

Copy the old configuration file `mc.cnf` in directory `/etc/opt/rh/rh-mariadb102/` to directory `/etc` and check the settings.

```
# cp /etc/opt/rh/rh-mariadb102/my.cnf /etc/my.cnf
# cat /etc/my.cnf
```


To clean up, remove the old MariaDB and start the new one:

```
# yum remove rh-mariadb*
# systemctl start mariadb
# systemctl enable mariadb
# systemctl status mariadb
```

Finally, execute the database upgrade script with

```
# mysql_upgrade -u root -p
```

to update all tables of the database for the new version of MariaDB.

RHEL 8/CentOS 8 (using MariaDB included in the distribution):

```
# yum install mariadb-connector-c mariadb-connector-c-devel
```

SLES 15 (using MariaDB included in the distribution):

```
# zypper install libmariadb-devel
```

All Linux distributions:

Finally, you need the Python 3 module for MariaDB. If you installed Python 3 from a repo as described in chapter 12:

```
# python3 -m pip install mariadb
```

If you installed Python manually as described in chapter 12:

```
# /home/openemm/opt/python3/bin/python3 -m pip install mariadb
```

7.5 Templates and Web Forms

You do not have to start from scratch when producing mailings or creating web forms in OpenEMM. At <https://www.agnitas.de/en/download-center/> you can download templates and web forms which you can import into OpenEMM. **Make sure to replace in web forms any placeholders for a company ID with value "1" if this is not done automatically during import.**

7.6 Updating Tomcat

If you update Tomcat to a version higher than 9.0.30, you have to remove this line from file *server.xml* in directory *home/openemm/conf*:

```
<Connector port="8009" protocol="AJP/1.3" redirectPort="8443"
useBodyEncodingForURI="true" />
```

If you use **HTTPS** (we hope you do!), please do not forget to re-add the APR configuration to Tomcat as mentioned in chapter "Apache Native Library" below.

7.7 Preparations before Updating MariaDB

If you plan to update your version of MariaDB, due to a bug in older versions of MariaDB (MDEV-19292), you may get an error "row size too large" afterwards which prevents you from using your OpenEMM database after the update of MariaDB.

Statement from MariaDB regarding this problem:

"Prior to MariaDB 10.2.26, MariaDB 10.3.17, and MariaDB 10.4.7, MariaDB doesn't properly calculate the row sizes while executing DDL. In these versions, unsafe tables can be created, even if InnoDB strict mode is enabled."

To check if you are affected by this bug, please execute SQL statement

```
SQL> SELECT count(*) FROM information_schema.innodb_sys_tables WHERE name
LIKE 'openemm/%' AND ROW_FORMAT = 'Compact';
```

If the result of this SQL statement is greater than 0, your OpenEMM database uses tables using the format which may cause "unsafe" tables. **In this case, we strongly recommend to convert all database tables to format "dynamic" with statement**

```
ALTER TABLE <table name> ROW_FORMAT = DYNAMIC;
```

before upgrading your version of MariaDB (after you made a backup of your database, of course)!

To simplify this task for you, *OMT* offers an automated conversion in menu *Database Maintenance*, sub-menu *Check MariaDB table format*. Alternatively, here is a little script that does the conversion of all tables (please execute as database user *root*):

```
DELIMITER $$
create procedure tmp_convert_row ()
Begin
    DECLARE done INT DEFAULT FALSE; DECLARE dbtable varchar(100);
    declare tab_curse cursor for select SUBSTRING(name, 9) from
information_schema.innodb_sys_tables where name like 'openemm/%' and
ROW_FORMAT = 'Compact';
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;
    open tab_curse;
```

```
read_loop: LOOP
    fetch tab_cursor into dbtable;
    IF done THEN LEAVE read_loop; END IF;
    SET @SQLText = CONCAT('alter table ', dbtable, '
ROW_FORMAT=DYNAMIC');
    PREPARE stmt FROM @SQLText; EXECUTE stmt; DEALLOCATE PREPARE
stmt;
end loop;
close tab_cursor;
END$$
DELIMITER ;
call tmp_convert_row ();
```

8 Advanced Configuration

This chapter describes some more advanced configuration options. Please be aware that you should backup first any files you modify in case the configuration is broken afterwards.

8.1 Mailloop Configuration

The mailloop service enables OpenEMM to process bounces (and autoresponder mails) which are received hours or even days later by email (asynchronous bounces). You need to define a dedicated mailloop service domain name which is different from the OpenEMM server hostname (set in `/etc/sysconfig/network/`). While you have to set up the A record for the OpenEMM server hostname, you have to set an MX (Mail Exchange) record for the mailloop service domain name, which points to the OpenEMM server hostname for correct mail routing.

For each new bounce filter created in the OpenEMM GUI, OpenEMM creates a new filter address, by default based on the mailloop service domain name. The OpenEMM user should define a mail forwarding for the address(es) used as sender address in its mailings, to direct all incoming response to the filter address(es) for further processing by the mailloop service (see “Alternative A” below).

In our example below the subdomain of the OpenEMM server hostname is *openemm* and the mailloop service subdomain name will be *mailloop*. The (abbreviated) DNS entry for *domain.com* should look like this:

```

                                IN    A      0      <ip address>
openemm                        IN    A      0      <ip address>
mailloop                       IN    A      0      <ip address>
mailloop.domain.com.         IN    MX     10     openemm.domain.com.
```

Replace expression *<ip address>* with the IP address of the OpenEMM server, which also runs the mailloop service.

The 1st line assigns the IP address for *domain.com* and the 2nd line defines the regular hostname. The 3rd line defines the A record and the 4th line defines the MX record for the mailloop service domain name, meaning that host *openemm* accepts emails sent to host *mailloop*.

Validate your setup by using a tool like *dig* or *host*, for example

```

# host -a domain.com
# host -a openemm.domain.com
# host -a mailloop.domain.com
```

When you send emails and want to take advantage of the bounce management for asynchronous bounces, there are two possibilities for the format of the sender address:

Alternative A (the popular one):

Use whatever sender address (or reply-to address) you like. Implement a forward mechanism in the email account of this sender/reply-to address to forward incoming mail sent back to this address to a filter address of OpenEMM. In order to create this filter address, set up a bounce filter in OpenEMM (see user manual). This filter will auto-generate a filter address like aml_1@mailloop.domain.com (*aml_1* = AGNITAS mail loop 1). After processing the incoming mail, the bounce filter will forward the filtered response to a feedback address of your choice (different from the sender address, of course).

The flow of automated responses to your mailings (like autoresponder mails and bounce notifications) works like this:

```
recipient -> sender/reply-to address -> filter address of bounce filter
(auto-generated by OpenEMM) -> feedback address
```

The advantage of this model is that you can choose any sender address you want, but you have to implement an external forward mechanism.

Alternative B (the advanced one):

Use a sender/reply-to address with the mailloop service domain name (for example news@mailloop.domain.com). Since no real email addresses exist for this sender domain name, normally it would not be possible to reply to an email with this sender/reply-to address. To forward responses to a valid email address you have to define a bounce filter.

In the GUI configuration for the bounce filter set field *Filter address* manually to your sender/reply-to address (in this example news@mailloop.domain.com) and this address is then bound to this bounce configuration. Due to performance reasons it may take a few minutes until a newly created entry is known by the system.

Finally, use field *Forward address* to enter the feedback email address of your choice. The flow of automated responses to your mailings (like autoresponder mails and bounce notifications) works like this:

```
recipient -> sender address (= filter address of your bounce filter) ->
feedback address
```

The advantage of this alternative is, that no external forward mechanism is needed.

Alternative C (the most sophisticated one):

There is a third alternative for your filter configuration which allows you to set up a filter address of your own choice without having to implement an external forward mechanism. In this case you create a bounce filter with an individual filter address of your choice. Make sure that in the DNS record of the domain used in your filter address, the MX record points to your server hostname like

```
yourdomain.com. IN MX 10 openemm.domain.com.
```

Now you can use the filter address in configuration field *Envelope e-mail* of any mailing to make sure that any automated response is sent to the OpenEMM server. If you use SPF, make sure that the SPF record of the domain used in your filter address includes your server hostname.

In this case the flow of automated responses to your mailings (like autoresponder mails and bounce notifications) works like this:

recipient -> envelope address (= filter address of bounce filter) ->
feedback address

The advantage of this alternative is, that no external forward mechanism is needed and that sender address and filter address can differ.

8.2 Configuration of Default Settings

Menu *Configuration*, sub-menu *Change configuration in DB* allows you to change certain configuration parameters of the OpenEMM frontend like file paths, server addresses and limiting values. You should have a look at the list of parameters to understand which parameters can be changed.

In the section of parameters with prefix *mailaddress* you should define email addresses for support requests by your users and email addresses for certain notification mails. By default, these email addresses are set to (invalid) sender domain *example.com* and, therefore, would never leave the OpenEMM server.

Some parameters of interest can be modified directly in database table *company_info_tbl*. Parameters with prefix *max* limit certain resources to avoid an overload of the database and parameters with prefix *expire* define, after how many days certain entries are deleted from the database to limit the required storage space.

Table *config_tbl* holds additional parameters valid for the instance of OpenEMM, especially URLs and file paths to components used by OpenEMM as well as system email addresses.

If the OpenEMM database holds more than 10,000 recipients and you open the recipient list you will be greeted with message *The option you selected is too large to be displayed completely. Please limit your selection to reduce the amount of recipients.*

If you want more than 10,000 recipients to be processed for the recipient list (which will take longer to display), set field *max_recipients* in database table *company_tbl* to the value you want:

```
SQL> UPDATE company_tbl SET max_recipients = 100000;
```

To increase security, OpenEMM blocks logins when the same IP address generates a certain number of failed logins. The default value for the maximum number of failed logins is 10 and the default value for the lock out time is 60 seconds. You can change both values in the database with a new entry in table *config_tbl*. Examples for max. 3 retries and 5 minutes lock time:

```
SQL> INSERT INTO config_tbl (class, name, value, creation_date,  
change_date) VALUES ('loginTracking', 'webui.maxFails', '3',  
current_timestamp, current_timestamp);
```

```
SQL> INSERT INTO config_tbl (class, name, value, creation_date,  
change_date) VALUES ('loginTracking', 'webui.ipBlockTimeSeconds', '300',  
current_timestamp, current_timestamp);
```

If you are really into it, have a look at the source code of class *ConfigValue.java* to see, what else you can configure in OpenEMM.

If any change to the database configuration of OpenEMM does not come into effect within 5 minutes, you have to restart OpenEMM.

8.3 Configuration for MySQL/MariaDB Database

Please be aware that the default value of MySQL/MariaDB parameter *max_allowed_packet* may only be 1 or 2 MByte. In this case, you can not load a single data packet (a file for example) bigger than 1 or 2 MByte into the database. This affects the upload of attachments for emails.

To change the maximum size for these files to a bigger value, you need to set the parameter *max_allowed_packet* in section *[mysqld]* of MySQL's or MariaDB's configuration file (usually *my.cnf* in directory */etc* or */etc/opt/mariadb10x*) to something like the following:

```
max_allowed_packet=10M
```

and restart MySQL/MariaDB afterwards, with

```
# systemctl restart mysql
```

Since the transfer of data to the database has some overhead, the value for *max_allowed_packet* should be a little bit higher than the value for *attachment.maxSize* in file *emm.properties*. You can check the current value of *max_allowed_packet* in MySQL/MariaDB with statement

```
SQL> SELECT @@max_allowed_packet;
```

The value of *max_allowed_packet* also limits the maximum size of SQL statements. In OpenEMM this is important for those SQL statements that retrieve statistical data for display in the workflow manager and dashboard calendar. In (the very unlikely) case that the GUI service does not show these numbers and you find a corresponding error message in Tomcat's log *catalina.out*, double the value for parameter *max_allowed_packet* until it works.

If you want to create emails with emojis, the default collation of MySQL/MariaDB can be a problem. Therefore, change the default collation in file *my.cnf* in section *[mysqld]* to

```
character-set-server = utf8mb4
```

```
collation-server = utf8mb4_unicode_ci
```

Please restart MySQL/MariaDB after this change.

8.4 Configuration of Webservices

The webservice interface runs as a separate web application in directory `/home/openemm/webapps/webservices`.

After OpenEMM has been launched you may request the WSDL file for the webservices via URL

```
http://<domain>/2.0/emmservices.wsdl
```

To be able to access the webservices of OpenEMM you have to create a webservice user with a password first. See the user manual for details.

8.5 Configuration of DKIM Support

Starting with version 21.04, OpenEMM supports DKIM. DKIM configuration and administration is done by two shell scripts in directory `/home/openemm/bin`. `dkim-creat` lets you create and implement new DKIM keys. It uses tool `dkim-mgr` for that purpose. `dkim-mgr` is the core tool to maintain DKIM keys.

To create a new DKIM key, launch `dkim-creat` as user `openemm` in your working directory with 3 parameters:

- the domain for which a DKIM key should be created
- the selector to identify the correct DKIM key (like "openemm")
- the bit length of the key (we recommend 2.048 bits, maximum should be 4.096 bits)

`dkim-creat` generates three files in your current directory:

- `<domain>-pub`, containing the public key
- `<domain>-priv`, containing the private key
- `install-<selector>@<domain>.sh`, a little shell script to save your private DKIM key in the OpenEMM database.

Furthermore, `dkim-creat` shows the necessary configuration of the DKIM entry with the public DKIM key in the DNS record of your sender domain.

This is an example for executing `dkim-creat`:

```
$ dkim-creat sample.com selector 2048
```

```
Creating private/public key pair
```

```
selector@sample.com.priv/selector@sample.com.pub for sample.com (selector selector) with 2048 bit
```

```
Installation sample for your DNS:
```

```
selector._domainkey.sample.com. IN TXT "v=DKIM1;
```

```
p=MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtWA2bIcq3n95h7hixhTDDSt2Bjc  
y209N700AHm/811SY/
```

```
PbkY7n5x5LELGvMuuOfg7QVChTM5dslDmJ2EHG8TSptZRuOPCw1fW2X3J8zkJK74wLSSVUoKW
```



```
TYwefp7+WyfoY+XxLzc40DWS1kxYNSvO9KaKxctMYYiKruAtdpnDBuK/  
tEbTJZBBWH55vwTmXhYbX6L2ZsKIyKjueGfx7RTm3GrVAWRTpBo9krBbM0PL4dP8h3yySr9Hv  
/WwXk19qeaC0oksVKChiXpc8CCt1gdLWQ5FeqoKnc6LkMkl4th0gPY/  
voJB6EQA1BC5ZNbpYxgma1UEv1bG37iKV7hJ+LhQIDAQAB"  
Creating install-selector@sample.com.sh .. add done.
```

As soon as the public DKIM key is filed in the DKIM entry of the DNS record for your sender domain, execute script *install-**<selector>**@**<domain>**.sh* to save the private DKIM key into the OpenEMM database.

9 OpenEMM Administration

9.1 Automated Startup

If you want OpenEMM to automatically launch at server reboot, you can use a systemd unit file for that purpose. Create a new file `openemm.service` in directory `/etc/systemd/system/` with the following content (please note that the *After* properties belong into one line):

```
[Unit]

Description=OpenEMM startup script

After=var-run.mount network.target local-fs.target time-sync.target
sendmail.service postfix.service

[Service]

User=openemm
Group=openemm

Type=oneshot

RemainAfterExit=true

ExecStart=/home/openemm/bin/openemm.sh start
ExecStop=/home/openemm/bin/openemm.sh stop

TimeoutSec=300

StandardOutput=journal+console

[Install]

WantedBy=multi-user.target
```

After deploying this file, reload the systemd-daemon and enable the openemm service with

```
# systemctl daemon-reload
# systemctl enable openemm
```

At next server reboot, OpenEMM will be started automatically.

9.2 Database Backup

For MySQL/MariaDB there exist plenty of strategies for database backups and tons of books and Internet resources on that subject. However, if you run only a medium

MySQL/MariaDB database with a few GByte of data and if you can live with an interruption of services of very few minutes, you may simply use tool *mysqldump*:

```
# mysqldump -aCceQx --hex-blob --routines --triggers -u root -p -r
openemm.sql openemm
```

Executed at the command line, this statement copies a database dump in a very robust format into text file *openemm.sql*. The database dump can be imported back into an empty database *openemm* simply with

```
# mysql -u root -p openemm < openemm.sql
```

Menu *Database Maintenance* of *OMT* offers a backup and restore of the OpenEMM database based on these commands.

If you run a bigger MySQL/MariaDB database which should not be stopped during backup time, we recommend the use of the tool *Percona XtraBackup*.

9.3 Generic Database Tuning

80% of all application performance problems are really database performance problems. If you run a big OpenEMM installation and you are not satisfied with the application's performance, here are some database tuning tips you should try.

If certain tenants of your OpenEMM database hold a long list of recipients, you may speed up database operations like calculating statistics significantly with a combined index on several fields of table *customer_1_binding_tbl*.

We recommend the following two indices in case they do not exist yet:

```
SQL> CREATE INDEX custbind1$mldid_user_cuid$idx ON customer_1_binding_tbl
(mailinglist_id, user_status, customer_id);
```

```
SQL> CREATE INDEX custbind1$user_mldid_cuid$idx ON customer_1_binding_tbl
(user_status, user_type, mailinglist_id, customer_id);
```

If you use any other profile field than *email* for duplicate checks, you should put an index on this field in *customer_1_tbl*:

```
SQL> CREATE INDEX cust1$<fieldname>$idx ON customer_1_tbl (<fieldname>) ;
```

Replace placeholder *<fieldname>* with the name of the database field you use for duplicate checks.

9.4 MySQL/MariaDB Database Tuning

Since version 5.5 of MySQL, InnoDB has been the default engine of MySQL (and later MariaDB). While InnoDB supports row locking and real transactions for better crash protection (opposed to MyISAM), the internal data structure is more complex than MyISAM's, which leads to larger table sizes, slower writes, slower full table scans and

slower handling of BLOBs and CLOBs. Also, backup and recovery via *mysqldump/mysql* is slower.

Because InnoDB is much more sensitive to configuration parameters than MyISAM, you should at least add properties *innodb_buffer_pool_size* and *innodb_log_file_size* in section *[mysqld]* of MySQL's or MariaDB's configuration file (usually *my.cnf* in directory */etc* or */etc/opt/rh/rh-mariadb10x*), because the internal default values of 128 MByte and 5 MByte are much too small for bigger databases with lots of InnoDB tables.

As a rule of thumb: *innodb_buffer_pool_size* should be set to 50% of the RAM of your server and *innodb_log_file_size* should be set to $\frac{1}{4}$ of the size of *innodb_buffer_pool_size*, but not higher than 256 MByte to limit recovery time after a database crash.

To prevent the InnoDB engine from saving all table data into system tablespace file *ibdata1* in directory */var/lib/mysql* you may add property

```
innodb_file_per_table=1
```

in section *[mysqld]*.

If you would like to get some more recommendations on how to optimize MySQL's or MariaDB's configuration file, we suggest to run the old but trusted *tuning-primer* script available at <http://www.day32.com/MySQL/tuning-primer.sh>

9.5 Stopping the Sending in Case of Emergency

A mail dispatch can be stopped in the GUI of OpenEMM. But if you want to dig deeper and stop the sending of a certain mailing at the command line level, it is important to understand the process of its generation, distribution and dispatch:

Meta mail packages are generated by backend. These files are located in directory */home/openemm/var/spool/META*

and one part of the file name is the ID of the mailing. If files with the ID of the mailing to be stopped, are removed, they are no longer distributed by the backend for dispatch to its mailer service. However, to avoid internal conflicts, you have to stop process *pickdist* before deleting the files with

```
# /home/openemm/bin/pickdist.sh stop
```

and you have to re-start *pickdist* afterwards with

```
# /home/openemm/bin/pickdist.sh start
```

If the backend has already distributed meta mail packages for processing by its mailer service, you will find the packages which have not been processed yet or which are in process right now in directory

```
/home/openemm/var/spool/mail/incoming
```

If a file is already in process, deleting it does not help. You may check if a file is in process with

```
# ps aux | grep xmlback
```

If you see process *xmlback* with a file name as argument, this file is already in process, i.e. process *xmlback* generates the final email files out of the meta mail package.

Otherwise you may delete the file(s). In this case you should stop process *pickdist* before deleting the files with

```
# /home/openemm/bin/pickdist.sh stop
```

and you have to re-start *pickdist* afterwards with

```
# /home/openemm/bin/pickdist.sh start
```

If a meta mail package is in process by the mailer service, there is no easy way to delete the email files generated by these packages. Depending on the MTA you use (Postfix or Sendmail) you may stop the MTA processes, check the mail queues and manually delete the files of all emails belonging to the mailing to be stopped.

Please be aware that the statistics for a certain mailing in the GUI of OpenEMM may not be correct if you manually delete meta mail packages of this mailing or email files of this mailing from the MTA queues.

9.6 Out of Memory

If you work with big lists and experience an error message like

```
Java.lang.OutOfMemoryError: Java heap space
```

you have to allocate more memory to the Java Virtual machine (JVM). You can increase the minimum and maximum memory in file *emm.sh.additional.properties* (which overwrites settings of *emm.sh*) in directory */home/console/bin* or */home/rdir/bin* by increasing the values of parameters **-Xms** for minimum and **-Xmx** for maximum memory in variable *JAVA_OPTS_EXTERNAL*. If you have allocated all memory available and the error remains, you should increase your server RAM to at least 2 GByte (better: 4 GByte) and modify the parameters accordingly.

9.7 Log Rotation

To prevent the Tomcat log from filling up the hard disk of the OpenEMM server, you may install a log rotation to get rid of old log files. Create file *tomcat-openemm* in directory */etc/logrotate.d* with this suggested content:

```
/home/openemm/logs/catalina.out {  
    copytruncate  
  
    daily  
  
    rotate 7  
  
    compress
```

```

dateext
size 10k
missingok
sharedscripts
postrotate

#####
# zip files older than 180 min and delete access_logs older than 90 days
#####
find /home/openemm/logs/access -name "*.log" -mmin +180 -exec gzip -9 {} \;
find /home/openemm/logs/access -name "*.log.gz" -mtime +90 -exec rm {} \;
find /home/openemm/logs -name "*.gz" -mtime +10 -exec rm {} \;
find /home/openemm/logs -name "*manager*.log" -mmin +180 -exec rm {} \;
find /home/openemm/logs -name "localhost*.log" -mmin +180 -exec rm {} \;

#####
# delete files older than 5 days in logs/webapps/
#####
find /home/openemm/logs/webapps -type f -mtime +5 -exec rm {} \;

#####
# Delete old ARCHIVES > 10 days
#####
find /home/openemm/var/spool/ARCHIVE -type d -mtime +10 2>/dev/null | xargs
-r rm -fr

#####
# Delete old logfiles from backend
#####
find /home/openemm/var/log -name "*.log" -mtime +30 -exec rm {} \;
endscript
}

```

For redirect servers you should replace value *10k* in line 6 to *1M* and replace *console* with *rdir* in all paths. You also may shorten the various retention times to your individual needs.

9.8 Changing Security-Related Files

OpenEMM passwords are saved in the database not only encrypted but salted as well. The salt file with the file extension *salt* is located in directory */home/openemm/conf/keys*.

If you want to change the salt file, please do it before you start operating OpenEMM, because otherwise all saved passwords will not work any longer. For generating a new salt just save a string of letters, digits and other characters of the ASCII character set (decimal values 33 to 126) with a maximum length of 32 characters in a simple text file and name the new file like the old salt file. You may change the file name in configuration file *emm.properties* of the corresponding service.

To prevent access of the statistics service by a random server, the statistics service is accessible via HTTPS protocol only, uses a private key and grants access only to those servers providing the corresponding public key. This public key is provided by the GUI service. The private key in file *birt_private.pem* is located in directory */home/openemm/conf/keys/*. The corresponding public key in file *birt_public.pem* is located in the same directory, since in OpenEMM GUI and statistics service operate on the same server.

If you think that the keys are not safe (enough) for your purpose, you may replace them and restart OpenEMM.

9.9 Switching SMTP Server Type

If you want to switch the type of SMTP server used by OpenEMM (Postfix or Sendmail) after OpenEMM has gone productive, you should wait patiently until all pending mailings have been delivered and stop OpenEMM with

```
# openemm.sh stop
```

As second step you have to stop the current SMTP server:

```
# systemctl stop sendmail
```

or

```
# systemctl stop postfix
```

Next step should be to clean the mail queues of the current SMTP server to avoid sending out old mails in case you will switch back later. To clean Sendmail's mail queues execute

```
# rm -r /home/mailout/*QUEUE*/*
```

To clean Postfix' mail queues, execute

```
# postsuper -d ALL
```

Before you switch to the new default SMTP server, make sure that you have installed the required packages and that you have made the required configuration modifications explained in chapter "Postfix Deployment" or "Sendmail Deployment".

You have to switch the default SMTP server to either Postfix with

```
# alternatives --set mta /usr/sbin/sendmail.postfix
```

or to Sendmail with

```
# alternatives --set mta /usr/sbin/sendmail.sendmail
```

After that you may launch the new SMTP server with

```
# systemctl start sendmail
```

or

```
# systemctl start postfix
```

and finally restart OpenEMM:

```
# openemm.sh start
```

You may check the type of SMTP server being active on a server running a mailer service with

```
# su - -c 'source scripts/config.sh; echo $MTA' openemm
```


10 Apache Native Library

10.1.1 HTTPS for Tomcat

If you are using OpenEMM without HTTPS support, *OMT* shows a warning like this

```
Check APR installation: FAILED
```

each time you start the frontend. We urge you to use the HTTPS protocol for production environments so that your installation is GDPR compliant!

To use OpenEMM with the HTTPS protocol you should use the APR based native library for Tomcat (APR = Apache Portable Runtime). *OMT* offers you to install the library in its download menu. However, if your environment does not permit, you can install the native library manually.

RHEL/CentOS:

At first, install the following required packages:

```
# yum install openssl-devel apr-devel
```

SLES:

At first, install the following required packages:

```
# zypper install -y libopenssl-devel libapr1-devel
```

You can download the source tarball from <http://archive.apache.org/dist/tomcat/tomcat-connectors/native> to the directory defined in \$CATALINA_HOME (default: */home/openemm/opt/tomcat*). Select the source code of the latest stable version of 1.2.

Configure the source tarball and compile the code with

```
# cd /home/openemm/opt
# tar -xvzf tomcat-native-1.2.24-src.tar.gz
# cd tomcat-native-1.2.24-src/native/
# ./configure --with-apr=/usr/bin/apr-1-config --with-ssl=yes
--prefix=/home/openemm/opt/tomcat-native-1.2.24
--with-java-home=/usr/lib/jvm/java
# make
# make install
```

(We use version 1.2.24 in this example.)

Now, the APR native library is installed in directory */home/openemm/opt/tomcat-native-1.2.24*. To make sure that the OpenEMM startup script *openemm.sh* finds this library in the right place, you have to define a version-independent symlink with

```
# ln -s /home/openemm/opt/tomcat-native-1.2.24
/home/openemm/opt/tomcat/tomcat-native
```

Whether you installed the native library manually or with the help of *OMT*, in any case there is some configuration work left. If you do not do it, you will be able to access OpenEMM with the HTTP protocol only (which is not recommended for production environments).

Configuration of the native library is done by changing the connector type and its properties for Tomcat 8.5 or 9 in tag *Connector* of Tomcat's server configuration file *server.xml* in directory */home/openemm/tomcat/conf*.

```
<Connector
    protocol="org.apache.coyote.http11.Http11AprProtocol"
    port="8443"
    enableLookups="false"
    disableUploadTimeout="true"
    acceptCount="100"
    maxThreads="1000"
    scheme="https"
    secure="true"
    SSLEnabled="true"
    useBodyEncodingForURI="true"
    server="<server_name>"
    <SSLHostConfig
        disableCompression="true"
        honorCipherOrder="true"
        ciphers="ECDHE-ECDSA-AES256-GCM-SHA384 ECDHE-RSA-AES256-GCM-
SHA384 ECDHE-ECDSA-CHACHA20-POLY1305 ECDHE-RSA-CHACHA20-POLY1305 ECDHE-ECDSA-
AES128-GCM-SHA256 ECDHE-RSA-AES128-GCM-SHA256 ECDHE-ECDSA-AES256-SHA384 ECDHE-
RSA-AES256-SHA384 ECDHE-ECDSA-AES128-SHA256 ECDHE-RSA-AES128-SHA256"
        protocols="all"
        caCertificateFile="<certificate_chain_file>"
        <Certificate
            certificateFile="<certificate_file>"
            certificateKeyFile="<certificate_key>" />
    </SSLHostConfig>
</Connector>
```

Replace placeholder `<server_name>` with the FQDN of your server. Replace `<certificate_chain_file>` with the path and name of your *cacert* file. Replace `<certificate_file>` with the path and name of your *crt* file. And replace `<certificate_key>` with the path and name of your *key* file.

11 Glossary

11.1 Bounce Management

EMM's automated bounce management provides you with the capability to keep your mailing lists clean and up-to-date automatically. A bounce message is an error message, which is sent from a mail server on the recipient's side to the sender if an email is not deliverable. Bounce management administers emails which are undeliverable temporarily (soft bounce) or permanently (hard bounce). It also filters error messages and autoresponder mails.

11.2 DNS

DNS is the abbreviation for Domain Name System. This system forwards requests directed to a FQDN to a certain IP address. Each entry in the DNS maps the IP address of a server to a human readable address. Example: In place of the IP address 83.169.23.100, which points to the AGNITAS website, you may use the DNS address *www.agnitas.com*, which is much more convenient (for a human).

11.3 FQDN

A Fully Qualified Domain Name (FQDN) links to an IP address of a server. The FQDN may be composed of letters and numbers and by using this option nobody has to remember the difficult number sequence (IP). A FQDN is divided in three levels:

- The affix of the domain is the Top Level Domain (TLD). Example: *com*, *org* or *de*
- The domain name will be inserted in front of the TLD. Example: *agnitas*
- The FQDN starts with the *hostname*. For webpages this is very often *www*

Example: The FQDN *www.yourdomain.com* is composed of

- *www* = hostname
- *yourdomain* = domain name
- *com* = TLD

As you can see, the FQDN consists of the hostname, the domain name and the top level domain separated by dots. The combination of domain name and TLD is commonly referred as *domain*. The FQDN can be expanded by a *subdomain* (like *miami*). The

subdomain will be inserted between the hostname and the domain. Example:
www.miami.yourdomain.com .

11.4 Bounce Management

OpenEMM does not treat all hardbounce messages reported by remote mail servers as hardbounce. In fact, some messages are only treated as softbounces, although their bounce codes starting with 5 would indicate a hardbounce.

The reason for this kind of ignorant behaviour is intentional, because some mail servers are not properly configured regarding the generation of hardbounce messages and mistakenly report permanent delivery errors - some even by intention to pretend that certain email addresses do not exist. If OpenEMM would handle those fake hardbounce messages as real hardbounces email addresses of existing recipients would be disabled. As result, we only try to accept bounces as hardbounces which are really proved to be hardbounces.

If you want to modify the bounce management of OpenEMM, file *bav.rule* in directory */home/openemm/lib* is the right place. This file lists in section *[hard]* bounce messages which are recognized as hardbounces, and section *[soft]* lists bounce messages recognized as softbounces. The messages are formatted as regular expressions to allow, among others, the use of wildcards. You may add your own set of messages here.

By the way, if a hardbounce message is recognized as a softbounce even if it is a real hardbounce, this is not a problem. Because a real hardbounce is reported for each mailing again and is counted as a softbounce each time, it will be finally caught by the softbounce scoring of OpenEMM and converted to a hardbounce in the end.

11.5 Softbounce Scoring

If an email address generates lots of softbounces (temporary delivery problems) this is actually an indication that the email address is undeliverable permanently (hardbounce). OpenEMM provides softbounce scoring to identify those email addresses and to convert them to hardbounces.

The rules for converting a softbounce to a hardbounce work like this:

1. Select all email addresses in the softbounce table which generated more than 40 softbounces and where the time-lag between the first and last bounce is longer than 30 days.
2. If no mail opening or link click was registered within the last 30 days for an email address which matches the before-mentioned conditions, this address is flagged as a hardbounce.
3. If at least one opening or click was registered within the last 30 days, this address is removed from the softbounce table, i.e. its bounce count is reset to zero.

12 Deployment of Python 3.8 (or later)

Do not install the Python package provided by your operating system. Since the version of Python 3 provided by RHEL/CentOS or SLES is too old for OpenEMM 20.10, you have to compile, build and install version 3.8 (or later) by yourself.

RHEL/CentOS:

If you use RHEL or CentOS, there is a shortcut, because you can use a special repository for Python 3.8 so that there is no need to compile and build is yourself.

First, install some required development packages:

```
# dnf install gcc gcc-c++ libgcrypt-devel libxml2-devel openssl-devel
```

Activate the repo with

```
# dnf module enable python38
```

Install Python 3.8, PIP and all requires packages:

```
# dnf install python38 python38-devel python38-pip python38-requests
```

Install some more Python modules:

```
# pip3.8 install py3dns xlrd xlwt xlutils paramiko pypsf dnspython dkimpy  
pycrypto requests httpie setproctitle inotify aiodns aiohttp aiohttp-  
xmlrpc aiosmtpd
```

To be on the safe side, you should also execute

```
# alternatives --set python3 /usr/bin/python3.8
```

to make sure that "python3" means Python3.8 - and that's it!

RHEL/CentOS:

If you do not want to use the before-mentioned repo for Python 3.8, install the following required packages:

```
# yum install wget gcc gcc-c++ bzip2-devel
```

```
# yum install gdbm-devel libgcrypt-devel libffi-devel libxml2-devel  
ncurses-devel
```

```
# yum install openssl-devel readline-devel sqlite-devel zlib-devel xz xz-  
devel
```

SLES:

At first, install the following required packages:

```
# zypper install -y gawk gcc gcc-c++ gdbm-devel libbz2-devel libdb-4_8-  
devel libffi-devel libxml2-devel libnsl-devel libopenssl-devel libuuid-
```

```
devel make ncurses-devel readline-devel sqlite3-devel tar wget xz-devel  
zlib-devel
```

Download the compressed tarball of Python 3.8.x (or later) from the official Python website python.org to a working directory of your choice like `/root/python3`.

To create and deploy your customized version of Python, execute the following commands as user `root` (with Python 3.8.2 as example):

```
# mkdir /home/openemm/opt/Python-3.8.2  
# rm -f /home/openemm/opt/python3  
# ln -s Python-3.8.2 /home/openemm/opt/python3  
# tar -xaf Python-3.8.2.tar.xz  
# cd Python-3.8.2  
# ./configure --prefix=/home/openemm/opt/Python-3.8.2  
# make  
# make test  
# make install  
# export PATH="/home/openemm/opt/python3/bin:$PATH"  
# which python3
```

The last command should result in output

```
/home/openemm/opt/python3/bin/python3
```

Install Python's package manager *PIP*:

```
# python3 -m pip install --upgrade pip
```

Finally, install some Python modules required by EMM Inhouse:

```
# python3 -m pip install py3dns  
# python3 -m pip install xlrd xlwt xlutils  
# python3 -m pip install paramiko pypdf dnspython dkimpy  
# python3 -m pip install pycrypto  
# python3 -m pip install requests  
# python3 -m pip install httpie  
# python3 -m pip install setproctitle  
# python3 -m pip install inotify  
# python3 -m pip install aiodns aiohttp aiohttp-xmlrpc aiosmtpd
```

13 Third Party Licenses

OpenEMM relies on several great open source frameworks, libraries and tools. In order to give due credit to those fine projects, you will find the individual licenses of the open source projects used by OpenEMM in subdirectory */home/openemm/webapps/emm/licences*.