grommunio Administrator Documentation

Release 2025.07.03

grommunio GmbH

Jul 03, 2025

Contents

CHAPTER 1

Introduction

This Administrator Manual covers everything regarding installation, operation and maintenance of the grommunio software suite. The intended audience for this documentation is administrators and system operators deploying grommunio.

1.1 About

grommunio delivers a fully-featured communication solution which covers all aspects of the software-defined era. As a modern and modular platform, grommunio helps to simplify all needs of modern communication by providing the following feature set:

- E-Mail
- Calendar
- Contacts
- Tasks
- Notes
- Video meetings
- Chat
- File sync & share
- Web office

1.2 Overview & Concepts

grommunio is shipped as an integrated software appliance for deployment on target systems by combining an embedded, optimized operating system, based on openSUSE. While grommunio is also shipping software repositories for major Linux platforms, the software appliance allows quick deployment on a variety of platforms, including bare metal and virtualized environments.

1.3 Architecture

grommunio's software component stack is modular and consists of the following main components:

Component	Function	Component Group
gromox-delivery, **gromox-delivery-queue**	Local delivery agent that places messages received from Postfix into mail stores	grommunio Groupware
gromox-event	Software bus inter-process communication (IPC) mechanism that allows communication between multiple processes running concurrently on multiple machines.	grommunio Groupware
gromox-http	The mail store (exmdb), and HTTP interface for RPCH, EWS, and optional FastCGI passthrough	grommunio Groupware
gromox-imap	IMAP interface providing industry-leading performance to IMAP clients	grommunio Groupware
gromox-pop3	POP3 interface	grommunio Groupware
gromox-zcore	Bridge process between PHP-MAPI and exmdb	grommunio Groupware
gromox-midb	Message index database, mostly an acceleration mechanism for use by IMAP	grommunio Groupware
grommunio-antispam	grommunio-antispam not only keeps your mail service clean from spam but also provides interfaces for anti-virus scanning and filtering	grommunio Groupware
grommunio-admin-api	A REST interface for automation and which provides the main API for grommunio's administration web interface	grommunio Admin
grommunio-admin-web	grommunio-admin-web is the central administration interface for system, domain and user management	grommunio Admin
grommunio-web	grommunio-web is the user's main web interface delivering a rich user experience to browser-based clients	grommunio web service
grommunio-sync	grommunio-sync provides the main EAS (Exchange ActiveSync) service for native clients, such as iOS, Android and other EAS-capable clients	grommunio web service
grommunio-dav	grommunio-dav provides the main CardDAV and CalDAV service for native clients, such as macOS and other capable clients	grommunio web service
grommunio-files	grommunio-files provides the file sync and share functionality, available to web and native clients	grommunio Files
grommunio-chat	grommunio-chat provides the main enterprise chat functionality, available to web and native clients	grommunio Chat
grommunio-meet	grommunio-meet is the web-based enterprise meeting feature, available to web and native clients	grommunio Meet
grommunio-office	grommunio-office is the web-based document collaboration software suite, available to web clients	grommunio Office
grommunio-archive	grommunio-archive delivers a legally conform archiving solution, available to web clients	grommunio Archive

Other software components used in combination with grommunio:

- MariaDB is the central database for all user metadata which provides the main database for all backend services. No user payload data (e-mails, etc.) are stored in this database.
- Postfix provides world-class functionality and versatility as the de-facto standard MTA which allows even the most advanced mail routing setups.
- nginx is a fast, robust and modern web server acting as the main web server and providing major services via HTTP and RPC to clients.
- SQLite is used for storage of the individual users' mailbox stores.

Some parts of grommunio are shipped as forks of other successful open source software. While grommunio only ships the open variants (with some integration features added), many of these open source vendors deliver enterprise variants of their software components. If the software component is covered by the grommunio subscription, grommunio delivers support for the open source variants of these components as well. Using the enterprise variants of the respective vendors is supported from an integration perspective, yet not for the vendors' product.

At grommunio, the top priority is to deliver a seamless communication and collaboration experience for users and a turnkey installation experience for administrators. These ambitions have led to the inclusion of other software components with additions (such as authentication through a single stack). This way, the integration effort for administrators is kept low, while users benefit from the interaction of multiple software components, delivering a seamless experience.

As a reference, the components of software products forked as part of the grommunio stack are:

- grommunio-files is a fork of Nextcloud with authentication and setup enhancements (https://nextcloud.com/)
- grommunio-meet is a fork of Jitsi with integration enhancements (https://jitsi.org/)
- grommunio-office is a fork of OnlyOffice web with integration enhancements (https://www.onlyoffice.com/)
- grommunio-chat is a fork of an open source chat platform with authentication and integration enhancements
- grommunio-archive is a fork of Piler with authentication and integration enhancements (https://www.mailpiler.org/)
- grommunio-antispam is a fork of rspamd with integration enhancements (https://rspamd.com/)

grommunio maintains and integrates these software solutions with the delivery targets offered by grommunio, such as software appliance and well-packaged software components available for all major Linux distributions. All these components are fully supported by grommunio based on the respective subscription level.

In case an environment or similar installation exists, these components can be integrated on an interface level. Note that grommunio can not support installations not packaged by grommunio. However, if existing enterprise installations are available, the integration of these systems is possible with the correct configuration in place. grommunio subscriptions deliver support for integrations with these enterprise variants or even - based on the interfaces available - alternative solutions.

grommunio delivers a variety of interfaces which allow other solutions to integrate with grommunio. Because of the modular nature of grommunio's software distribution, there is no forced need to use the extra components delivered by grommunio. For turnkey solutions, especially in the SMB market, shipping these components with the simplified integration effort helps administrators to install and operate grommunio as a comprehensive communication platform within just a few minutes.

CHAPTER 2

Quickstart

This chapter covers a short walkthrough which can be used as a check list to install and get grommunio started.

- Download the installation ISO from https://download.grommunio.com/appliance/grommunio.x86_64-latest.install.iso. The installation image is a hybrid installation image which also allows to be transferred to a USB stick with USB imaging tools such as GNU ddrescue or https://rufus.ie.
- Use the installation media from grommunio to install and quickstart the configuration by walking through the following chapters.
- Create or request TLS certificates for secure, encrypted operation of the main services.
- Create the corresponding DNS records (A, MX, TXT and CNAME records).
- Configure the grommunio appliance by running grommunio-setup.

2.1 Minimum requirements

For the installation of grommunio (or using the grommunio Appliance), the following minimal requirements apply:

- Server or virtual machine (VMware, Xen or Hyper-V) with at least:
  - 4 CPU cores
  - 6 GB RAM
- Correctly configured DNS records, at least two, for example:
  - **<FQDN>**, for example **mail.example.com**
  - **autodiscover.example.com**
- A TLS certificate with all included DNS names, alternatively a wildcard certificate for the entire domain. (Let's Encrypt can be configured by grommunio-setup.) If you already own a certificate, it can be re-used provided it is in PEM format, with one file containing the certificate chain and server certificate, as well as a separate key file.

Note: It is strongly recommended to properly set up the corresponding *autodiscover.example.com* DNS entry, otherwise AutoDiscover will not be able to determine the server.

Important: IPv6 is mandatory to be active, since many preconfigurations rely on it. A ”real” IPv6 is not required, the availability of ::1 is sufficient.

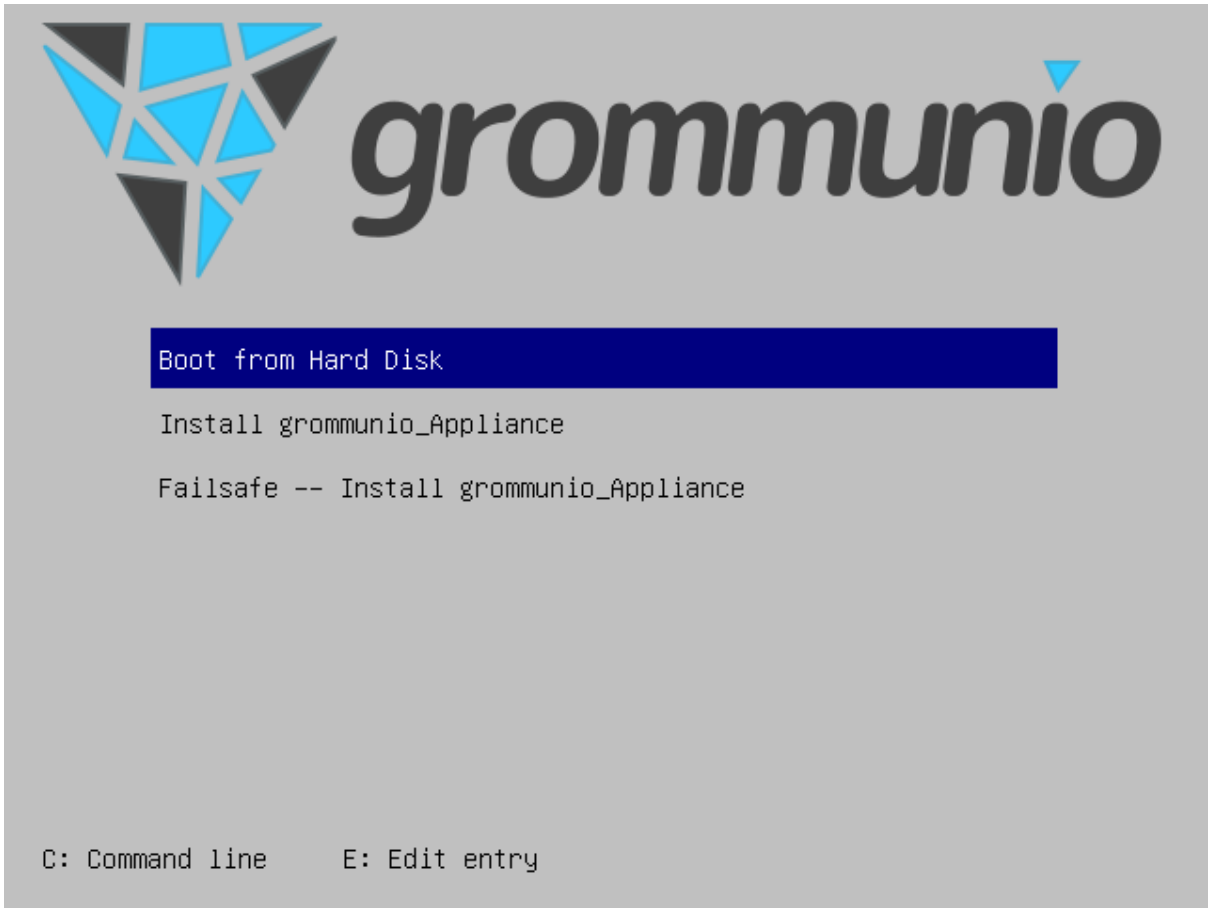
Optional requirements:

- MX DNS records, for incoming mail delivery.
- At the time of certificate generation by Let’s Encrypt, the accessibility of port 80 to all of the defined DNS records is a requirement.

2.2 Installation

1. Download of the bootable x86 image from download.grommunio.com: https://download.grommunio.com/appliance/grommunio.x86_64-latest.install.iso
2. Load the file for installation into the server on which grommunio should be installed on.
3. Run the installer and choose ”Install grommunio_Appliance” from the boot menu to install the appliance.

Important: Note that the installer asks for confirmation to delete and overwrite the installation target!



After the image has been copied to disk, the appliance is ready for boot and upcoming setup.

2.3 Setup

After installation, the appliance displays the grommunio console user interface (CUI). For more detailed instructions of the setup process, refer to grommunio-console-ui-cui.

Important: The initial root password is unset (empty). When asked for password, just press ”Enter”.

To configure grommunio, proceed as follows:

1. Choose **”Change system password”** to set a new root password.
2. Choose **”Network configuration”** to set up networking of the appliance.
3. Choose **”Timezone configuration”** to set up the correct timezone for the appliance.
4. Choose **”Timesync configuration”** to set up the correct timeservers (NTP) for accurate date and time settings.
5. Choose **”grommunio setup wizard”** to guide through subsequent configuration interactively.
6. (Optionally) choose **”Change Admin Web UI password”** to reset the password after setup to your liking.

The ”grommunio setup wizard” invokes *grommunio-setup*, which can be started from the CUI or any other terminal of the appliance.

Note: SSH is enabled by default, therefore grommunio-setup can also be executed from an SSH session. Note that a password must have been set before you can login via SSH.

To navigate within the grommunio setup wizard (grommunio-setup), use the following navigation hints:

- *<TAB>* navigates through dialog elements
- *<ARROW-UP>* or *<ARROW-DOWN>* naviate within form elements (such as when entering subscription details) or menu selections (during database setup)
- *<j>* or *<k>* keys for scrolling longer content-heavy dialogs (as in the finalization dialog)
- *<ESC>* to terminate grommunio-setup at any given stage of the configuration

Additional hotkeys are available at display of grommunio-cui at the bottom of the screen.

grommunio-setup automatically supplies defaults for most dialogs; these can be overridden as desired. For example, grommunio-setup automatically generates passwords which are also available after the installation in the grommunio-setup logfile, */var/log/grommunio-setup.log*.

Important: If the configuration fails for any reason, grommunio-setup can be re-run. However, any re-configuration from scratch is **destructive** and will re-initialize the installation. If you intend to change any system-related parameters, use the grommunio administration interface instead. Any re-run grommunio-setup invocation will warn and ask for confirmation before deleting any data.

Important: The installation process is logged in **/var/log/grommunio-setup.log**. Note that this file has all instance configuration used to configure grommunio-setup. As a subscription owner, you are entitled for support, where, for example, you can send the installation log to grommunio if you need any help. (Password references should be removed.)

Important: It is recommended after successful information to store the installation log in a safe place and delete it from the appliance. Alternatively, the installation log can be stored safely somewhere as reference of any credentials of your installation for later use.

2.3.1 grommunio Admin User

During the process of grommunio-setup, some accounts are automatically generated - such as a database account for user management and also for the initial grommunio administrator (admin).

Important: The admin user of grommunio and the root user of the appliance are separated, non-synced users. The admin user is solely known to the grommunio Administration framework and is (intentionally) not a system user. The credentials of both users are to be kept safe. The root user is the main system administrator while admin is the main grommunio administrator. They can (and should) have different passwords, with the role concept of grommunio it is even recommended not to work with these passwords in production, but instead create less privileged for regular tasks performed.

Note: The password of the primary admin user can be changed anytime by using grommunio-cui or by executing `grommunio-admin passwd --password "ChangeMe"`

2.3.2 Repository configuration

The interactive configuration tool grommunio-setup requests subscription credentials during execution. If you own a valid subscription, enter your subscription details. Without a valid subscription, grommunio-setup activates the community repositories, which are without support and contain non-quality-tested packages. With a valid subscription, your subscription repository is activated and delivers commercial-grade packages for the installation to keep up-to-date with latest features and fixes.

Note: To receive a valid subscription, contact any of our partners or via our established communication channels at https://grommunio.com

2.3.3 Certificates

With grommunio-setup, you are able to choose from multiple choices for certificate installation:

1. **Creation of self-signed certificate**

   Creating your own self-signed certificate is the simplest option - Creating an own self-signed certificate will though show up as untrusted at first connect and needs to be trusted before continuing. This behavior is normal and is because any client that connects has no possibility validation if the certificate has a valid source. This setting is the default and does not require any preparation for certificate generation. grommunio does not recommend this option for production environments, as this option requires any client to first trust the certificate in use. This option is the best for validation and demo installations of grommunio.

2. **Creation of own CA (certificate authority) and certificate**

   Creating your own certificate authority is an extended option which allows you to create self-signed certificates with an own certificate authority. This way, you can (manually) create further certificates under the umbrella of a own central authority with multiple server certificates to

be signed by the same certificate authority generated by yourself. This option is the best for validation and demo installation of larger installations of grommunio with multiple instances.

3. **Import of an existing TLS certificate from files**

   Importing your own certificate allows any type of external certificate pair (PEM-encoded) to be used with your grommunio installation. Note that it is recommended to either use SAN certificates with multiple domains or a wildcard certificate. With your choice of your own TLS certificates, you have the highest flexibility to either use a trusted CA or a publicly signed certificate by an officially trusted certification authority including, but not limited to, Thawte, Digicert, Comodo or others.

4. **Automatic generation of certificates with Let's Encrypt**

   Using this option allows the automatic certificate generation process with the Let's Encrypt certificate authority. Using Let's Encrypt certificates is free of charge, however the terms of service by Let's Encrypt apply, which are referenced during installation. Using this option automatically requests the domains from the selection you made, and automatically starts the validation process. For this automated process to work successfully, Let's Encrypt verifies _all_ defined domain names by creating a challenge on the appliance. For this to work, port 80 (HTTP) needs to be accessible from the Internet during this step of verification (and any subsequent automated renewal) with all the domains pointing to the appliance. This option is recommended for any simple installation and allows the most seamless installation experience if prepared correctly.

Any certificates so generated are placed in $/\mathrm{etc}/\mathrm{grommunio}/\mathrm{ssl}$ and are automatically referenced by any services of the appliance.

2.4 Firewall

For seamless operation, the grommunio appliance opens different ports so that clients can access it. Note that all of the following ports are made available by default:

- 25 (smtp)
- 80 (http)
- 110 (pop3)
- 143 (imap)
- 443 (https)
- 993 (imaps)
- 995 (pop3s)
- 8080 (admin) (disabled per default)
- 8443 (admin https)

Generally, it is recommended to only make available the ports that are required for service access. Note that grommunio's major protocols, RPC over HTTP, MAPI/HTTP, EWS (Exchange Web Services) and EAS (Exchange ActiveSync) are all accessed via port 443 (HTTPS).

When operating with proxies and load balancers, note that for successful operation of proxying RPC, special configuration needs to be in place. The required HTTP transport modes required to operate RPC over proxies are RPC_IN_DATA and RPC_OUT_DATA. Known supported proxy software to support these RPC data channels are: haproxy, squid, nginx and apache.

CHAPTER 3

Guided Installation (grommunio Appliance)

grommunio delivers ready-to-use appliances for:

- bare metal or virtualized environments (ISO)
- container environments (docker)
- specialized, automated virtualization environments (OVA)

and a community image to run grommunio on a raspberry pi.

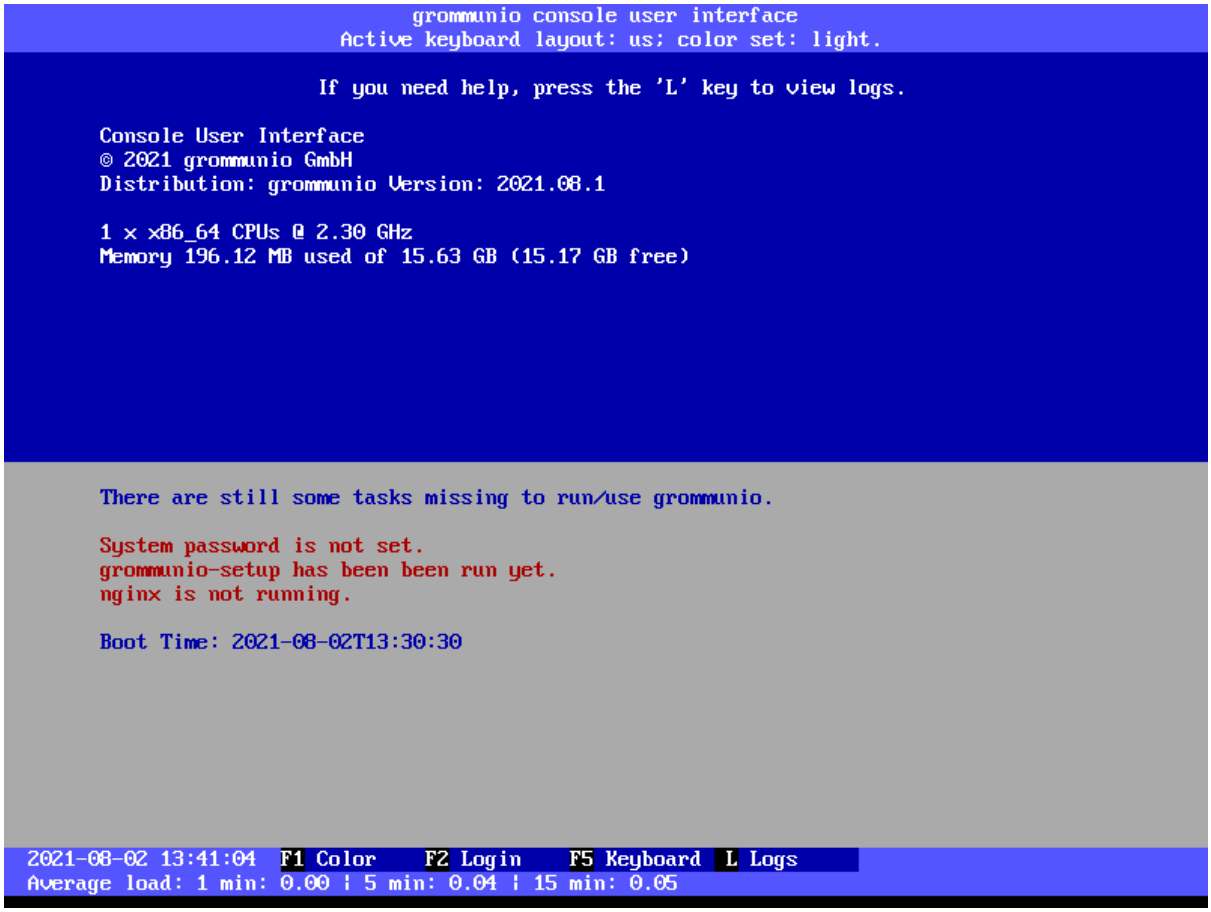
Note: There are multiple ways of automation and deployment for grommunio available. Not all of these methods can be described - If you are looking for a special deployment type, don't hesitate to get in contact with one of grommunio's partners or with grommunio directly. grommunio is available from very small installations to large, hyperscale installations with millions of users.

To deploy grommunio via ISO, you need to make the installation media available to your installation target. The ISO is a generic, bootable installation medium which works in most scenarios. To deploy the ISO with bare metal, the ISO can be imaged to USB drives for simplified installation.

The grommunio Appliance is a general-purpose installation target, which comes with all components required for successful operation of grommunio. It already includes the operating system for simplified management and allows general purpose usage. Every appliance installation is automatically deployed with update servers ready-configured and services prepared for usage. If you are seeking a general-purpose and simple deployment, grommunio Appliance is the right place for you. Simplified update management, backups and full portability allow the appliance to operate for any installation target sizing 1-2000 users with adequate hardware sizing. For larger installations or installations with special deployment needs, such as - but not limited to - geographically split, cluster or hyperscale installations, please refer our partners and/or our support/professional services team. Alternatively, the combined information from the manual installation in this chapter together with the man page sections is sufficient to build the grommunio setup of your needs.

3.1 grommunio Appliance configuration with CUI/setup

The grommunio console user interface (grommunio-cui) provides a console interface which allows the administrator to perform basic tasks to ready the appliance for the admin UI (admin web interface) or admin CLI (admin command line interface), such as network configuration and time synchronization.



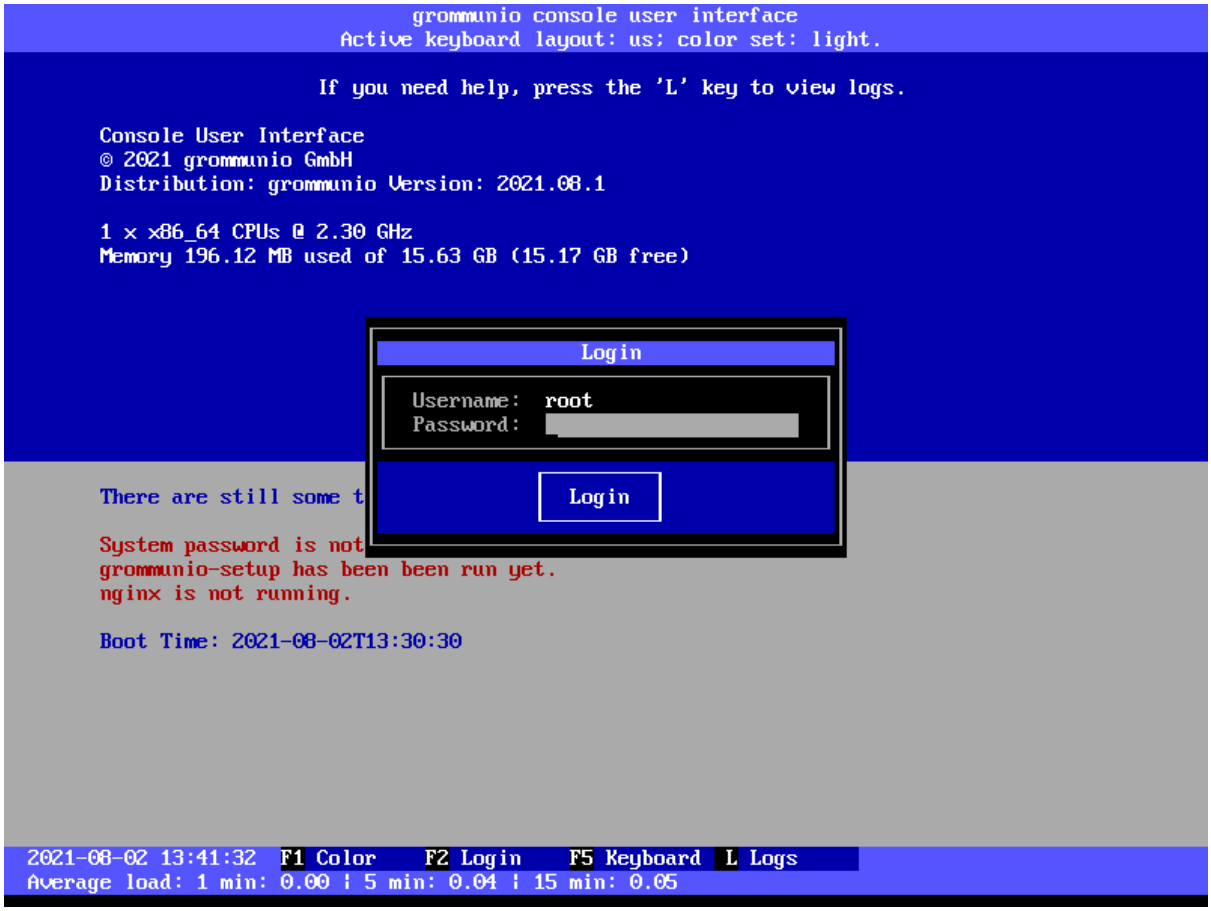
3.2 Main screen

After starting grommunio-cui, you are in the main screen. Upon login, you are able to make system configuration changes.

In the main screen, the following functions are available:

- F1: Switching the color scheme (light vs. dark mode)
- F2: Login to unlock system configuration mode
- F5: Switching of keyboard layout
- L: Open system log viewer

3.3 Login



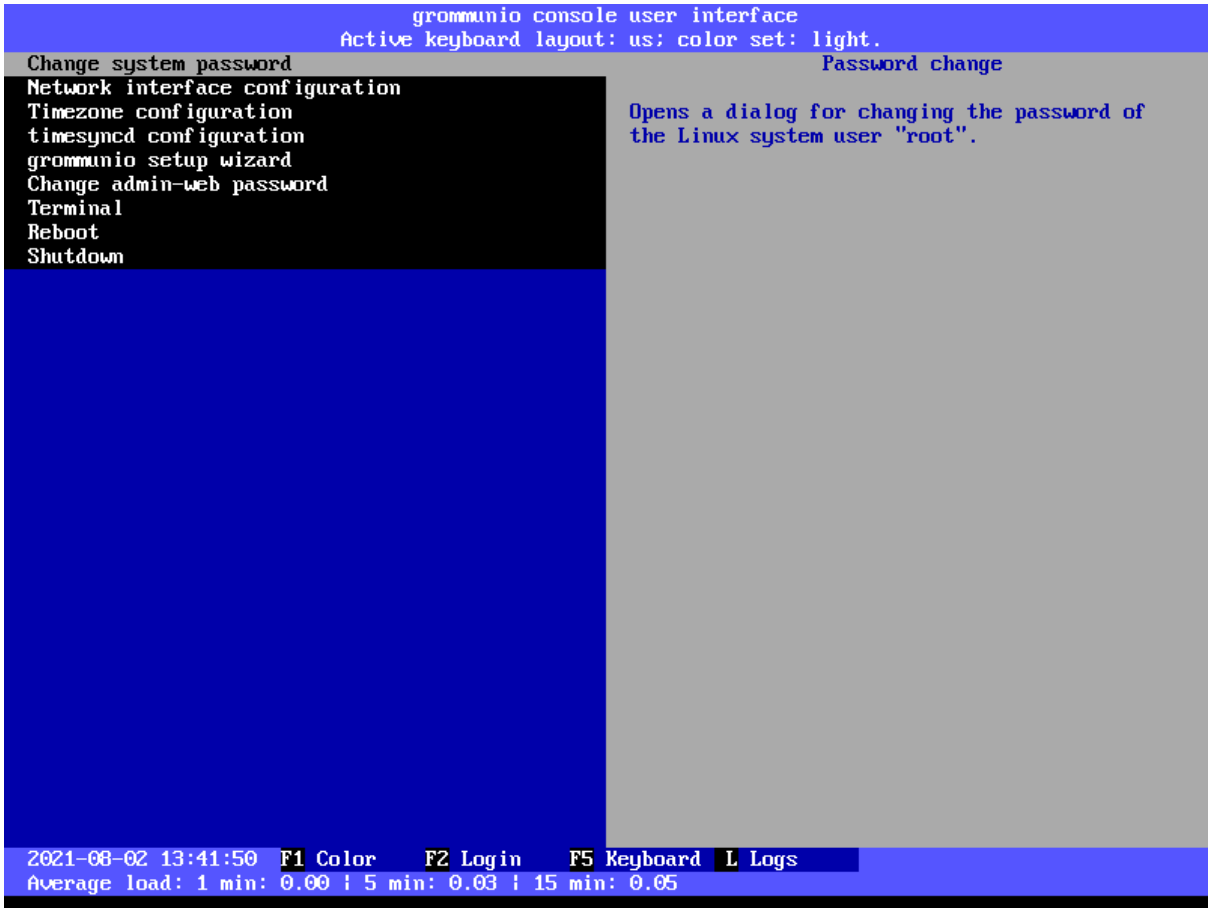
To enter into system configuration mode, press F2 and log in with the system superuser account (root).

Important: The initial root password is unset (empty). When asked for password at first login, just enter an empty password.

3.4 Main configuration screen

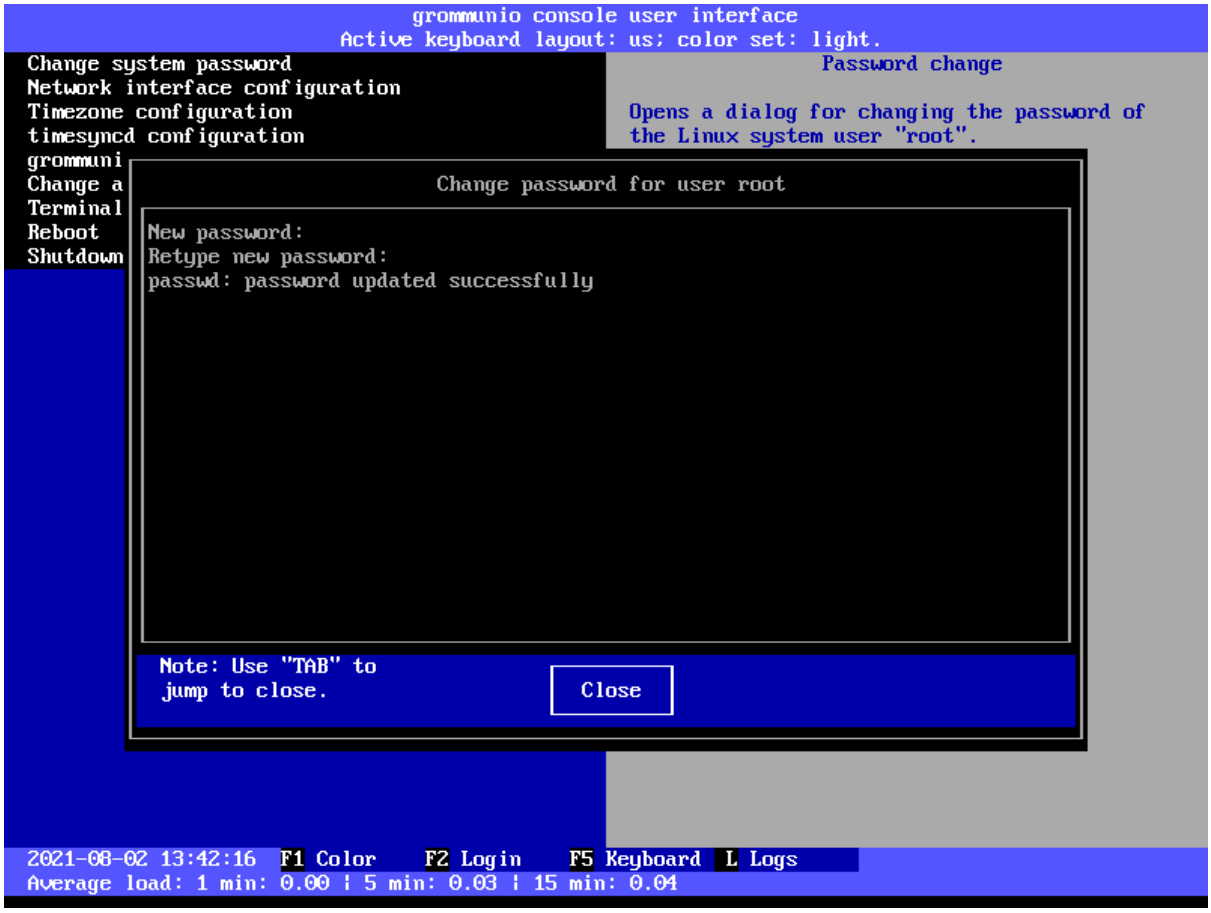
The main menu provides the following functionality available to grommunio-cui:

- Change system password
- Network configuration
- Timezone configuration
- Timesync configuration
- grommunio setup wizard
- Change Admin Web UI password
- Terminal
- Reboot
- Shutdown



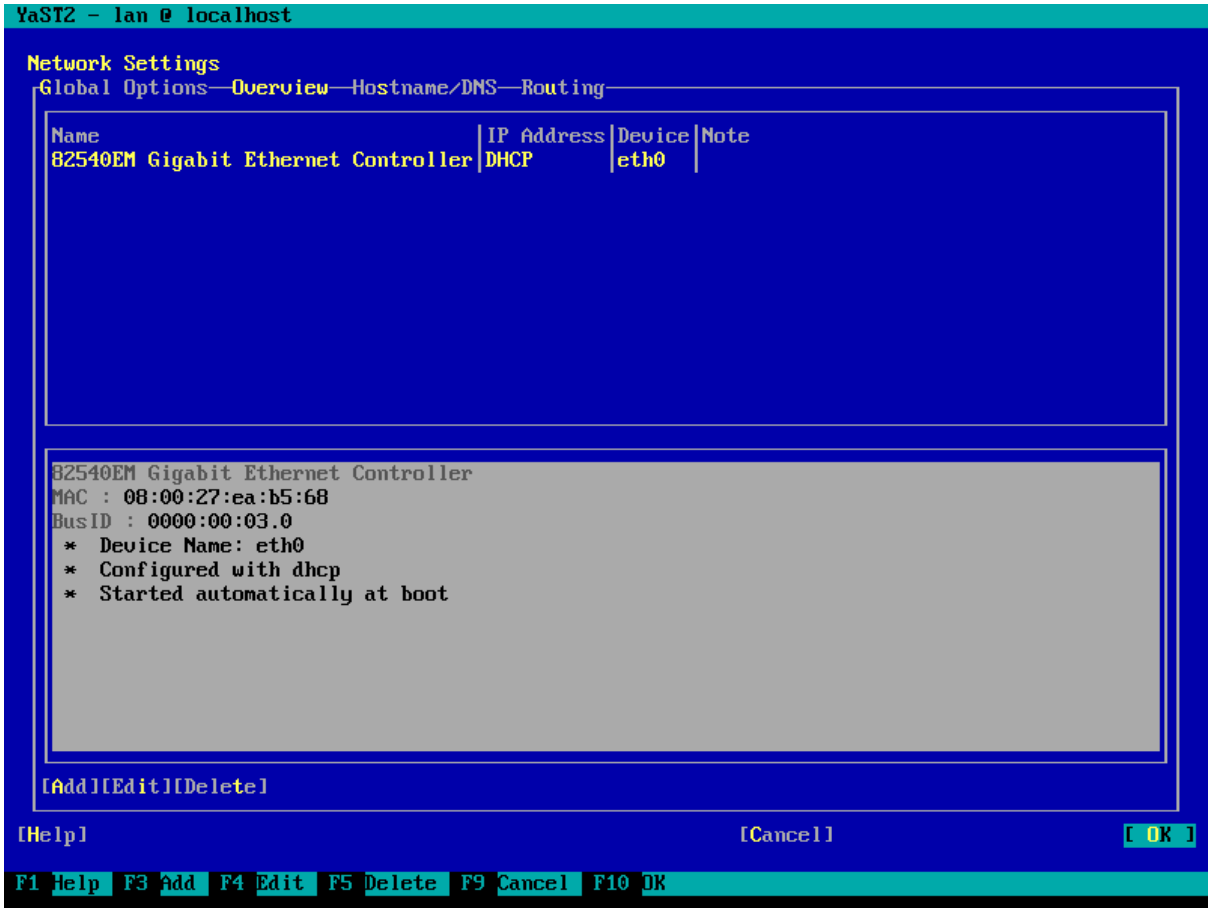
3.5 Change system password

The menu entry Change system password opens a window for setting the superuser (root) account password. Do this directly after installation. Use a secure password. We recommend using a password comprised of four words or more.



3.6 Network configuration

The menu entry Network configuration starts the network configuration utility (yast2 lan), which provides support for all reasonable network configuration settings. For detailed information on how to configure the network by using the yast utility, refer to the online documentation of YaST at https://documentation.suse.com/sles/15-SP3/html/SLES-all/cha-network.html#sec-network-yast



Important: The minimal set of configuration recommended to be changed includes: Hostname, Network Addressing (IP address), DNS (Nameservers), Routing (Default Gateway).

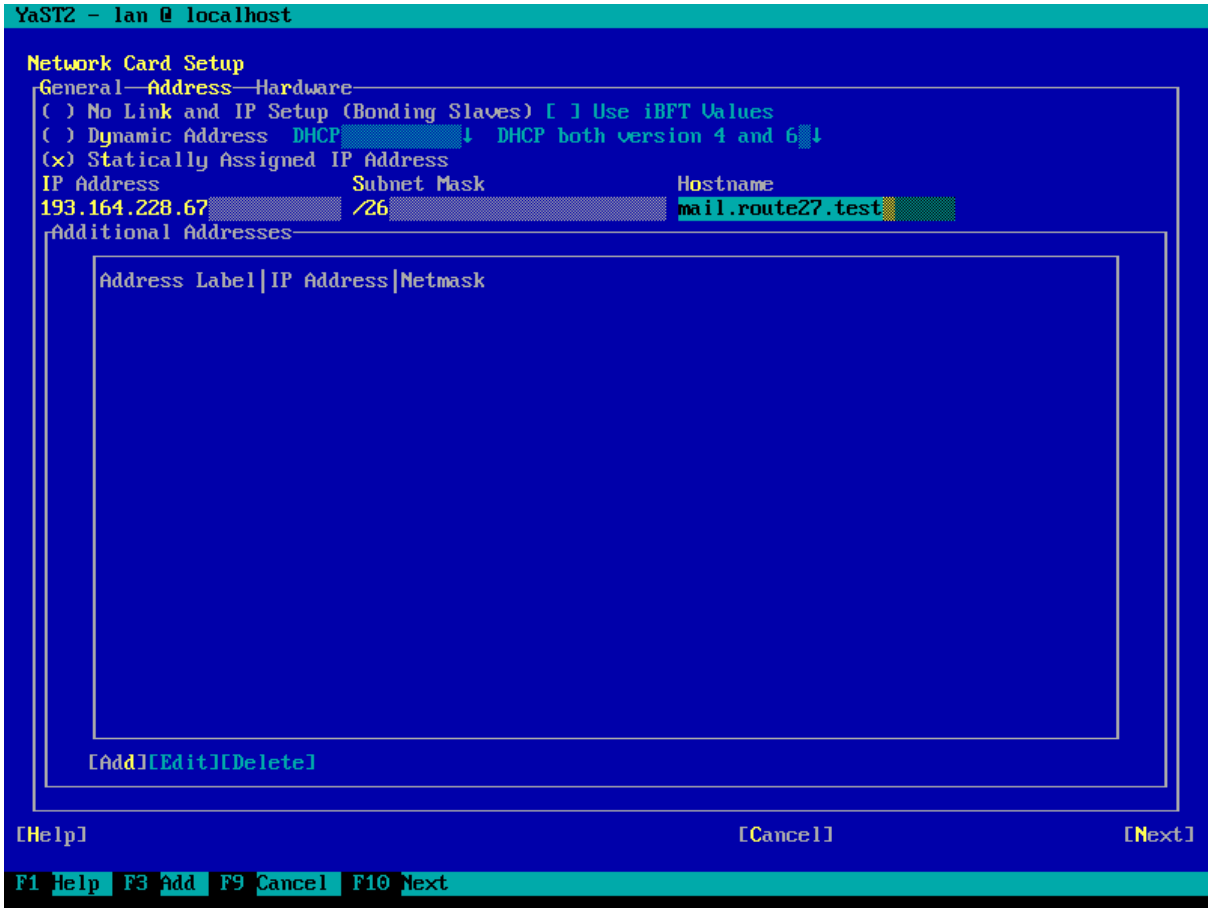
Important: Note that using the domain localhost is not a valid hostname and/or local is not a valid domainname. Make sure to set the hostname and FQDN properly at all setup and installation stages for operating with a valid configuration.

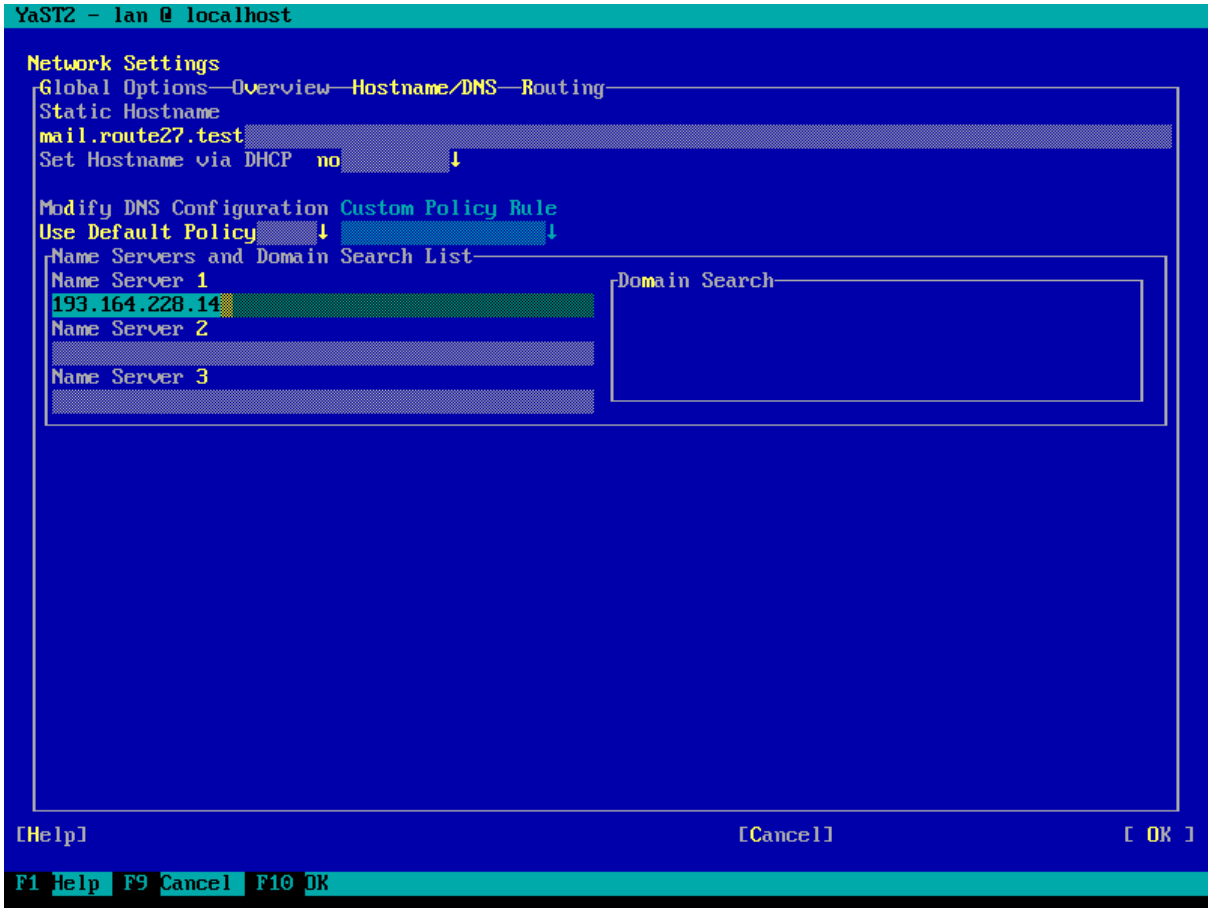
3.6.1 Hostname & FQDN setup

It is a requirement to setup the system hostname and domainname correctly.

Second, for local name resolving of services to work properly, the correct entries should be either available in DNS and/or be set in /etc/hosts.

To do this with the appliance, set the fully qualified domain name (FQDN) in the interface settings (which will be mirrored to /etc/hosts) **and** in the "Hostname/DNS" tab (the static hostname relates to /etc/hostname). This way, any services of the appliance will be able to use the correct addressing based on the domain and host. A correct hostname/DNS setup is mandatory, especially for multi-host setups.

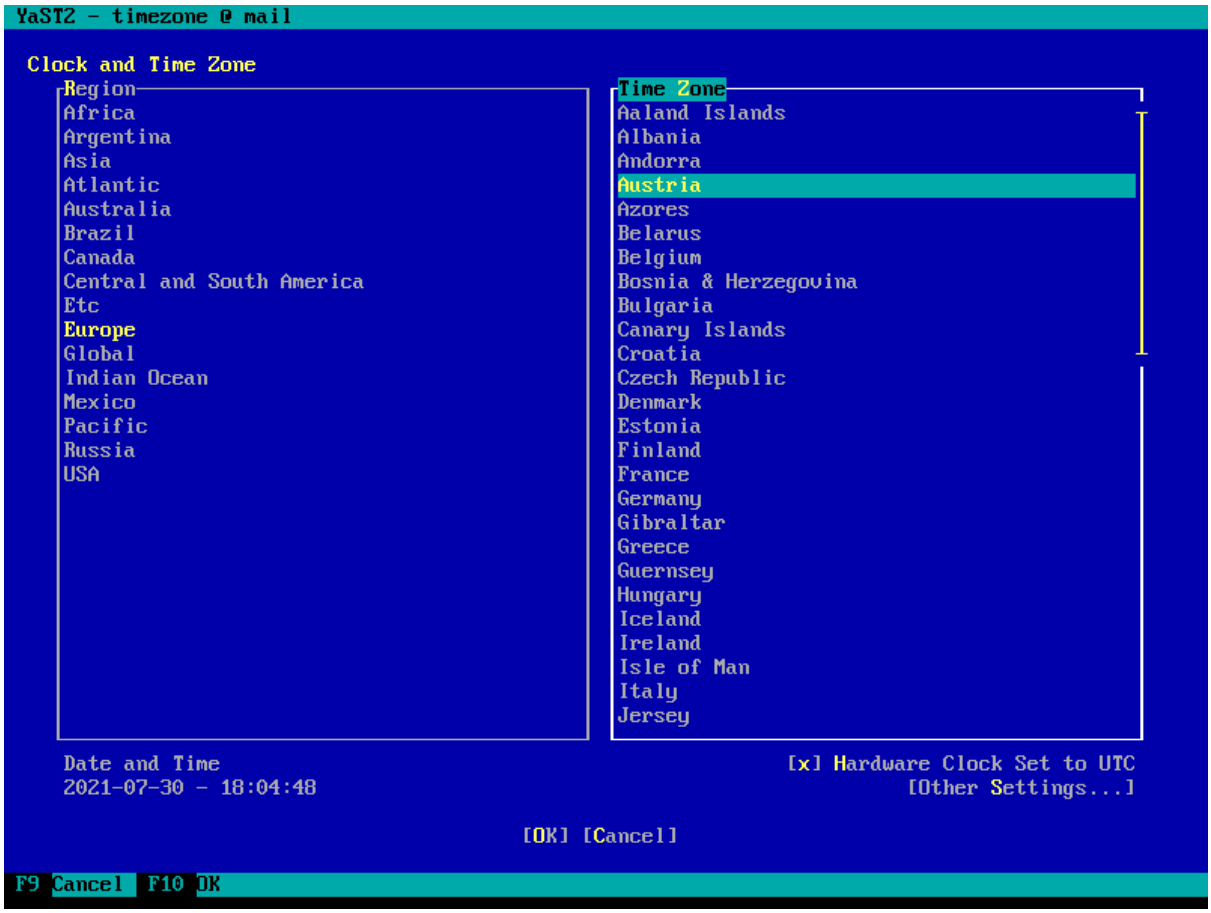

YaST2 - lan @ localhost
Network Card Setup
General Address Hardware
( ) No Link and IP Setup (Bonding Slaves) [ ] Use iBFT Values
( ) Dynamic Address DHCP DHCP both version 4 and 6
(x) Statically Assigned IP Address
IP Address
Subnet Mask
Hostname
193.164.228.67
/26
mail.route27.test
Additional Addresses
Address Label | IP Address | Netmask
[Add] [Edit] [Delete]
[Help]
[Cancel]
[Next]
F1 Help F3 Add F9 Cancel F10 Next


YaST2 - lan @ localhost
Network Settings
Global Options Overview Hostname/DNS Routing
Static Hostname
mail.route27.test
Set Hostname via DHCP no
Modify DNS Configuration Custom Policy Rule
Use Default Policy
Name Servers and Domain Search List
Name Server 1
193.164.228.14
Name Server 2
Name Server 3
Domain Search
[Help]
[Cancel]
[OK]
F1 Help F9 Cancel F10 OK

Important: To verify the settings, the command `hostname` should return the FQDN of the system.

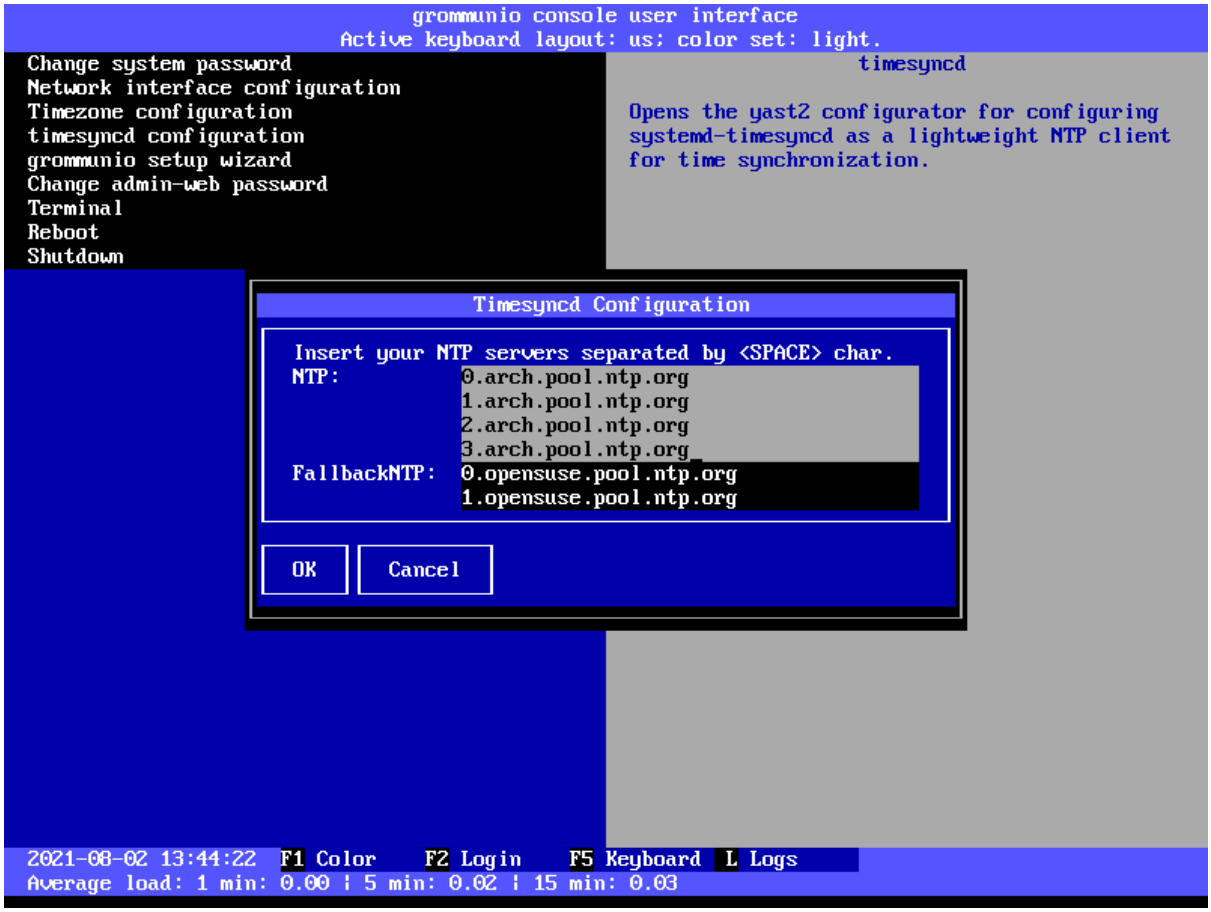
3.7 Timezone configuration

The menu entry Timezone configuration can be used to set the preferred timezone displayed in server logs, etc. It has no practical impact on e-mails, because mail user agents such as grommunio-web translate timestamps to the timezone of the particular device the program is running on anyway.



3.8 Timesync configuration

Timesync configuration is done with a simple interface providing the ability to set the timezone according to your region and timezone of that region. It generally is recommended to keep the setting Hardware Clock Set to UTC, since this provides the recommended timezone-agnostic behavior for services (such as with logs, etc.).



After these basic setup, your grommunio Appliance should:

- be able to connect to the Internet (availability of Updates, etc.)
- have a valid timezone set
- have a valid timeserver configured, with the system time appropriately synchronized

3.9 grommunio setup wizard

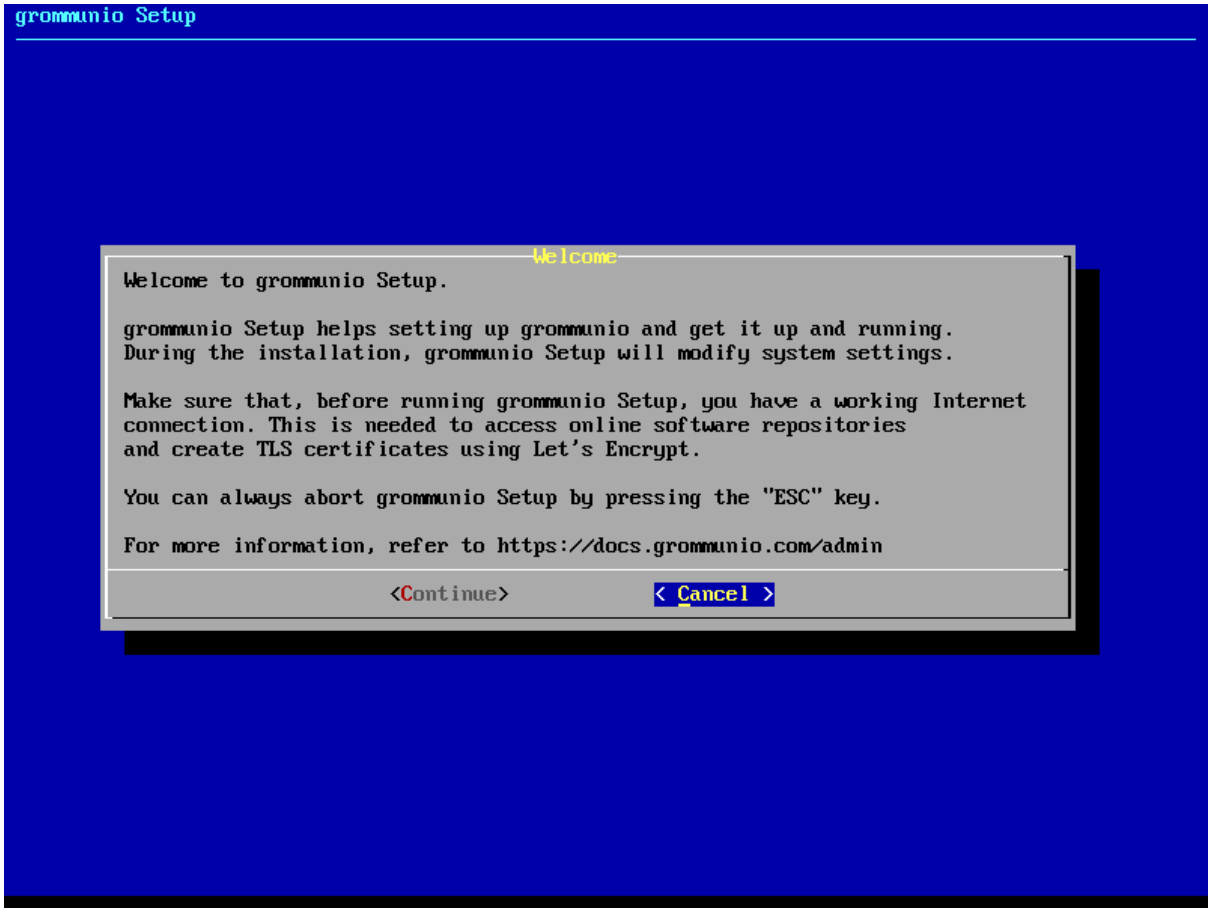
With the previous basic setup steps completed, it is recommended to run the grommunio setup wizard to complete the configuration based on your needs.

The menu entry grommunio setup wizard initiates the grommunio-setup program which walks you through the initial setup of grommunio.

Important: While grommunio-setup can be executed more than once, running through the setup process of grommunio-setup always resets the entire installation. grommunio-setup automatically detects if it has been run already and will warn you that, if you continue, all data stored will be lost.

3.9.1 Welcome screen

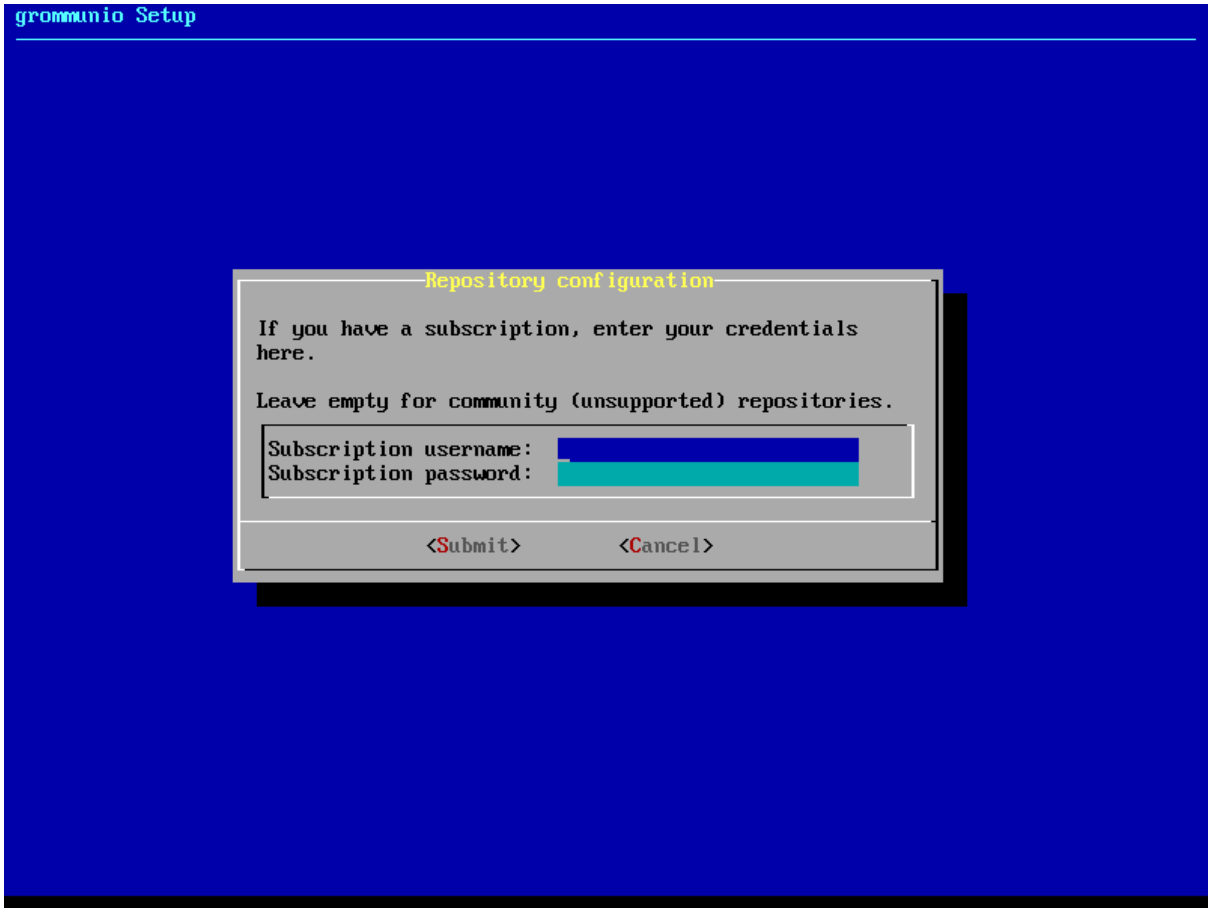
Starting grommunio-setup presents you with a descriptive welcome screen.



3.9.2 Repository setup

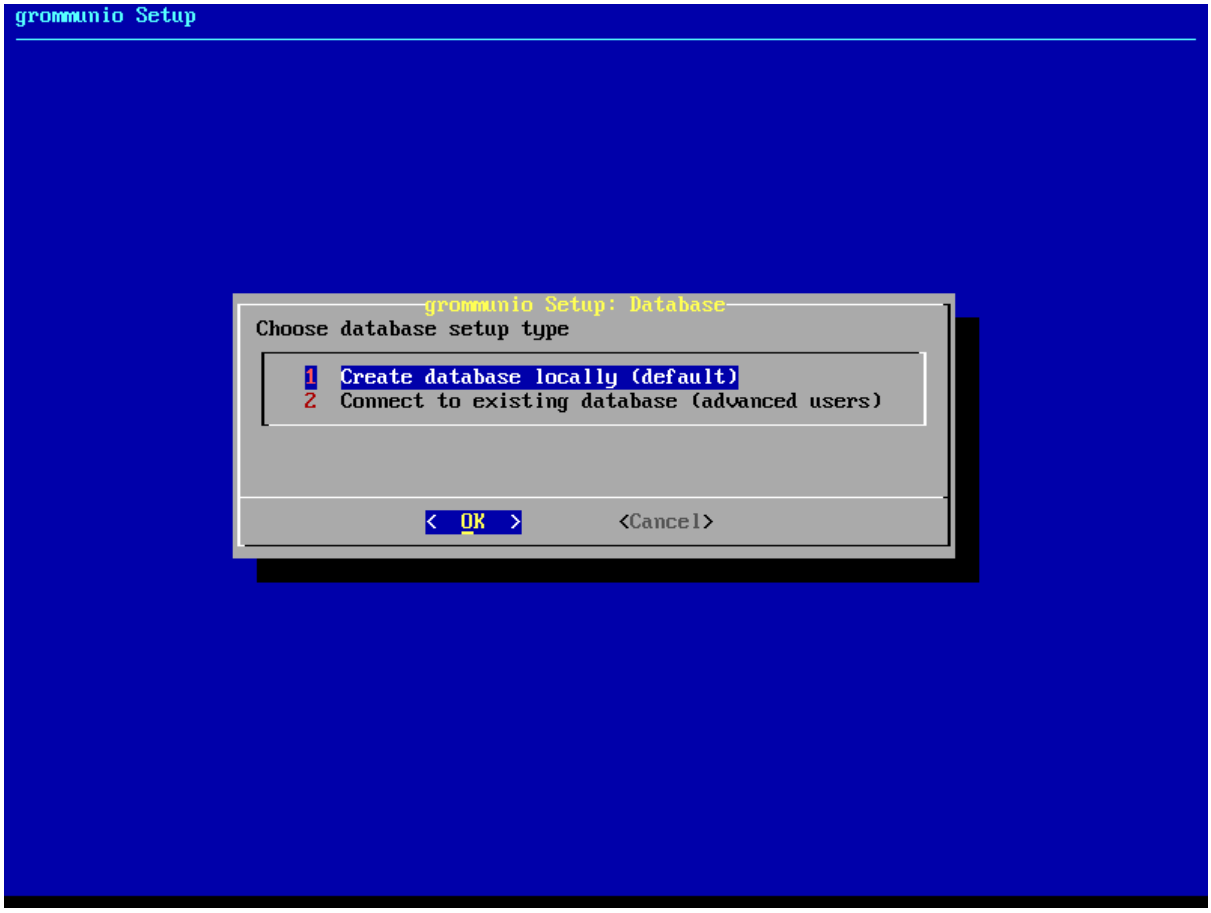
As first step, grommunio-setup requests you to enter subscription details. These subscription details are included in your purchase of the product, alongside with the subscription certificate delivered for installation at a later stage. If left empty, grommunio-setup will automatically include the community repositories.

Note: Community repositories are delivered on a best-effort basis and are not supported. While grommunio welcomes community members to use grommunio, the software distribution available with the subscription repositories include production-relevant benefits. Subscription repositories (available only with a valid subscription) include quality-tested packages, hotfixes and extra features not available with community repositories.



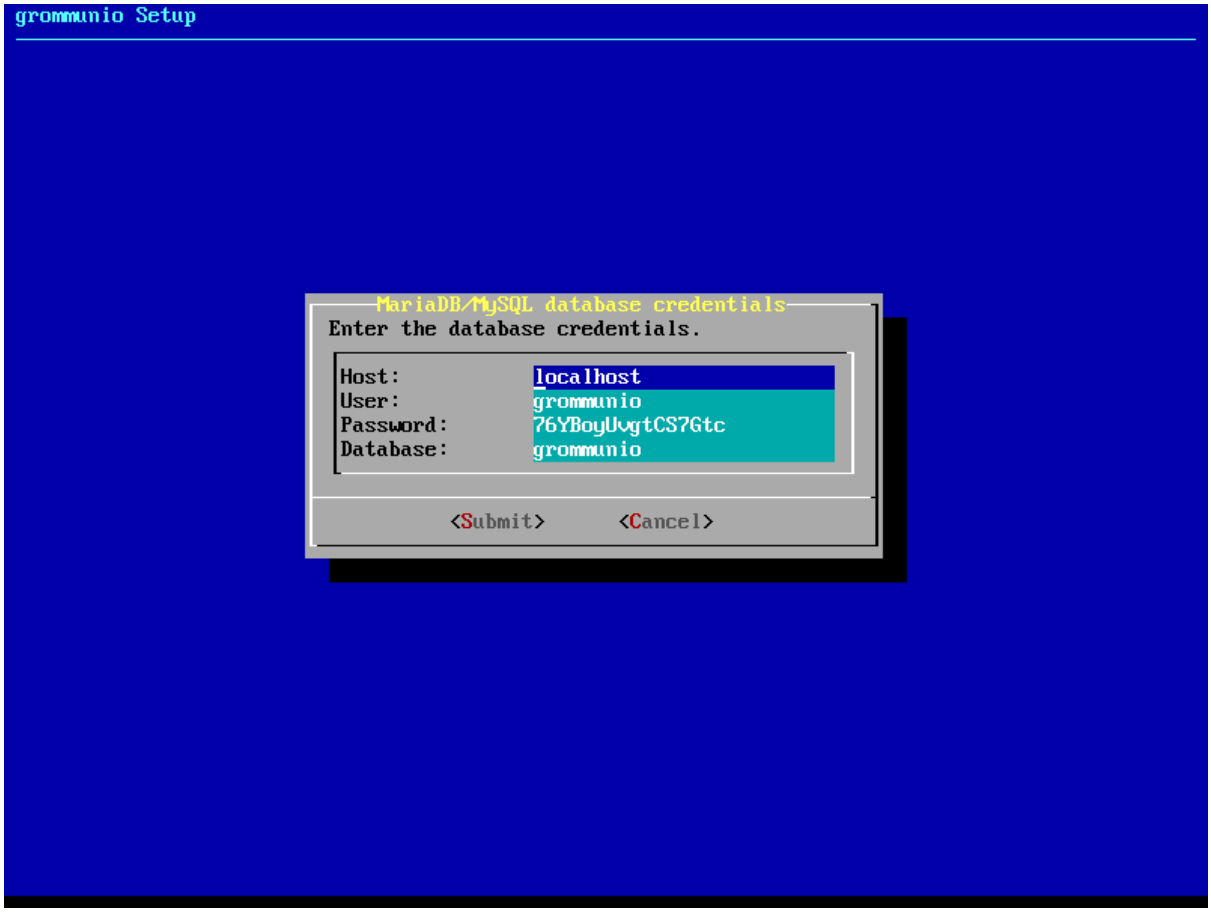
3.9.3 Database variant

In the next stage of grommunio-setup, you are requested to specify which central database type you want to configure. Most installations use the local database installation, where the MySQL-database is initialized and prepared automatically. For larger and/or special setups, e.g. clusters, multi-node and distributed setups, it might be recommended to connect to an already existing database instead.



3.9.4 Database settings

With the choice of ”local database”, the next installation step will automatically provide you with information which is used for initialization of the database. For standard setups, it is recommended to go with the default values. The values for the installation are generated randomly, which protects your installation from unauthorized access.

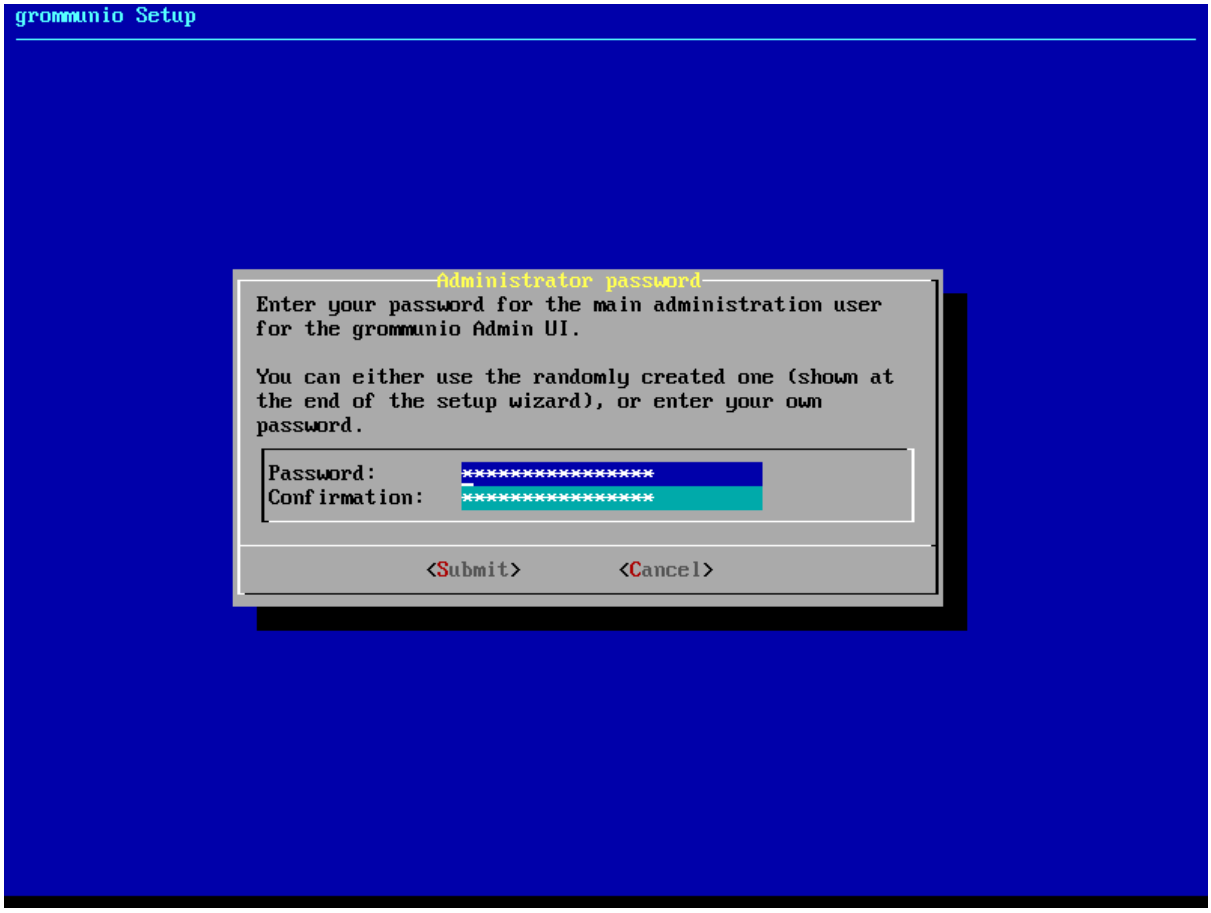


3.9.5 Administration User

After setting up the database, a default administrator password is requested for the login with the grommunio Admin API. The default user (admin) is then initialized with the password entered here. By default, grommunio automatically generates a password and shows it at the end of the setup procedure.

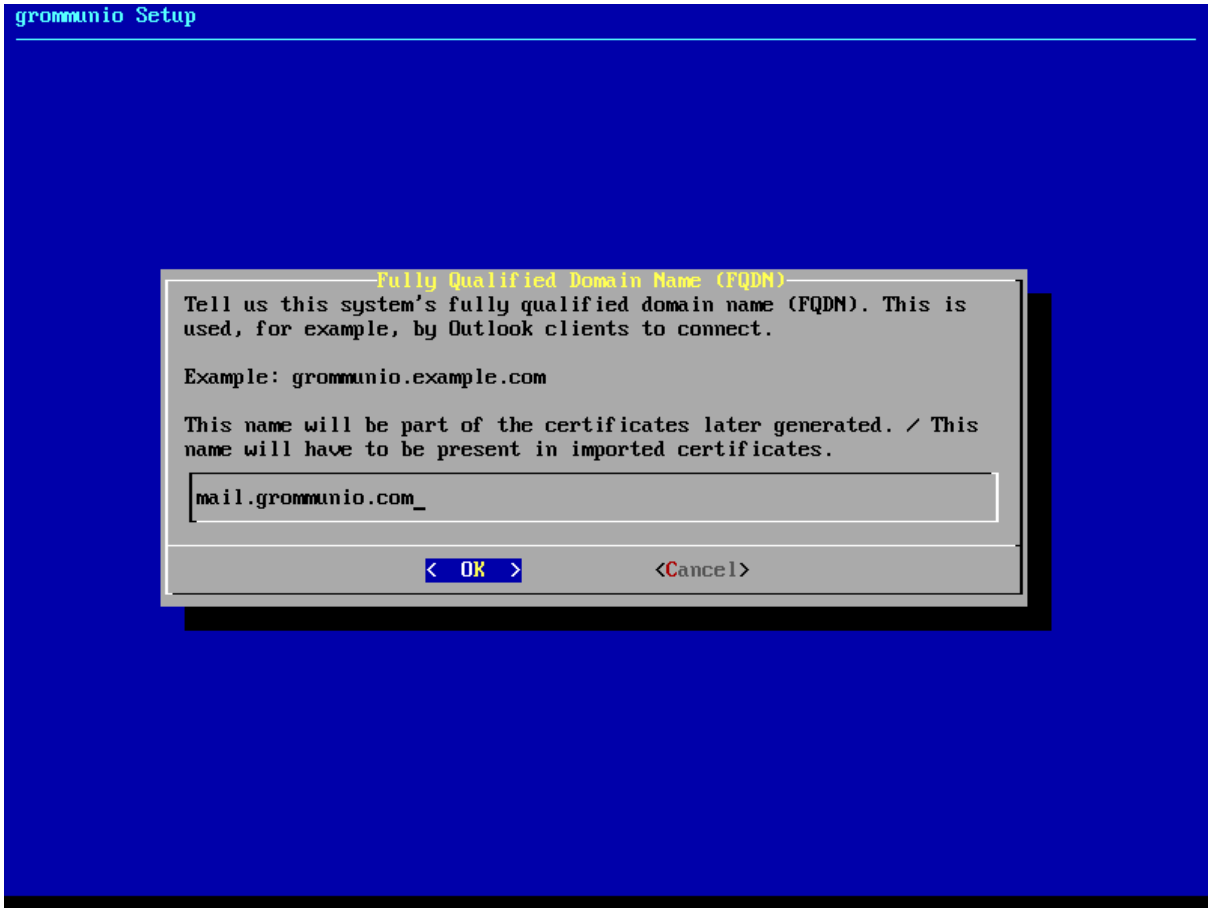
Important: At the end of the setup procedure, the password entered here will be shown in the summary screen after setup. Make sure no unauthorized people are accessing or viewing the system console for retrieval of this major credential.

Note: You can always reset this password at a later stage through grommunio-cui.



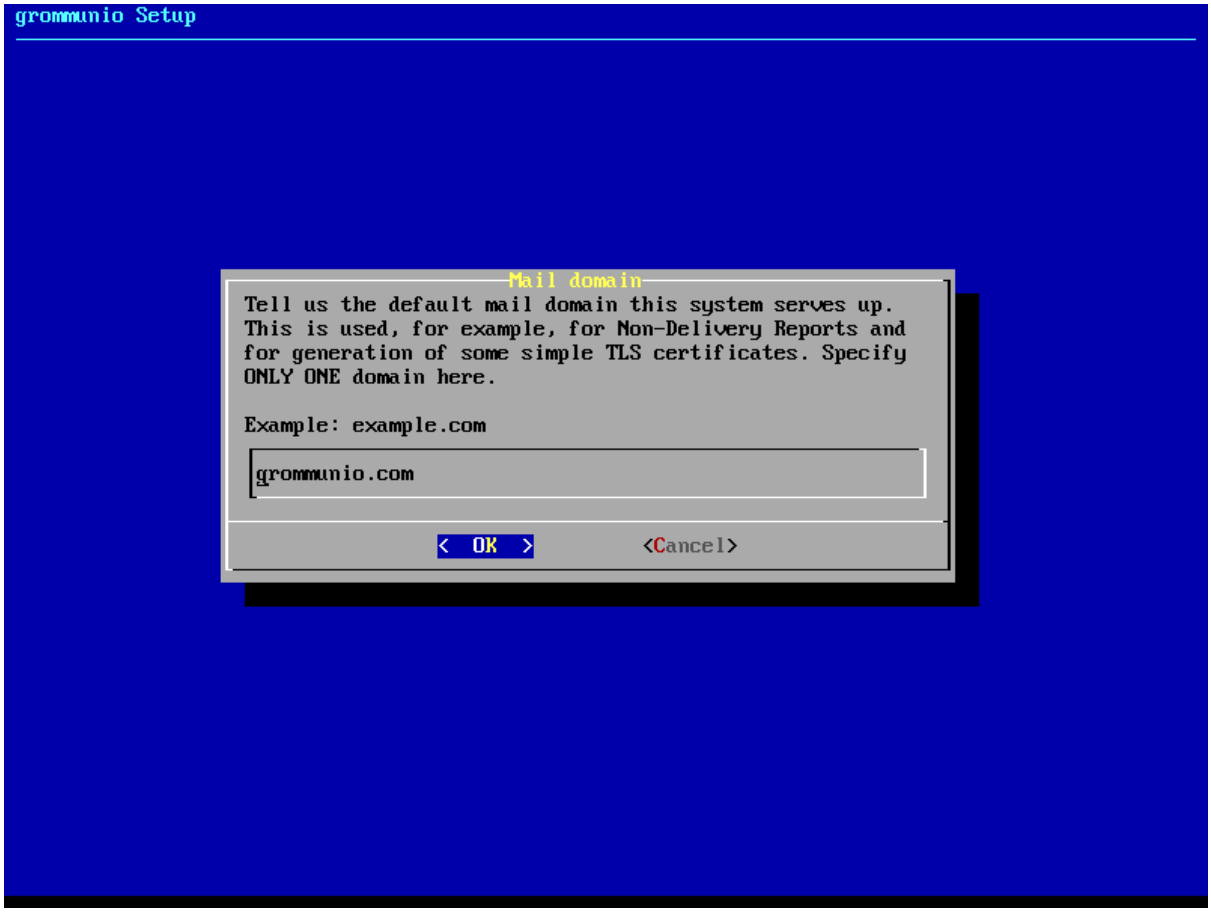
3.9.6 Fully Qualified Domain Name

The next stage of grommunio-setup requests the configuration of the fully qualified domain name (FQDN). The FQDN traditionally consists of the **hostname**, combined with the primary **domain** of the system. The name chosen here is strongly recommended to be part of the certificates generated at a later stage in grommunio-setup.



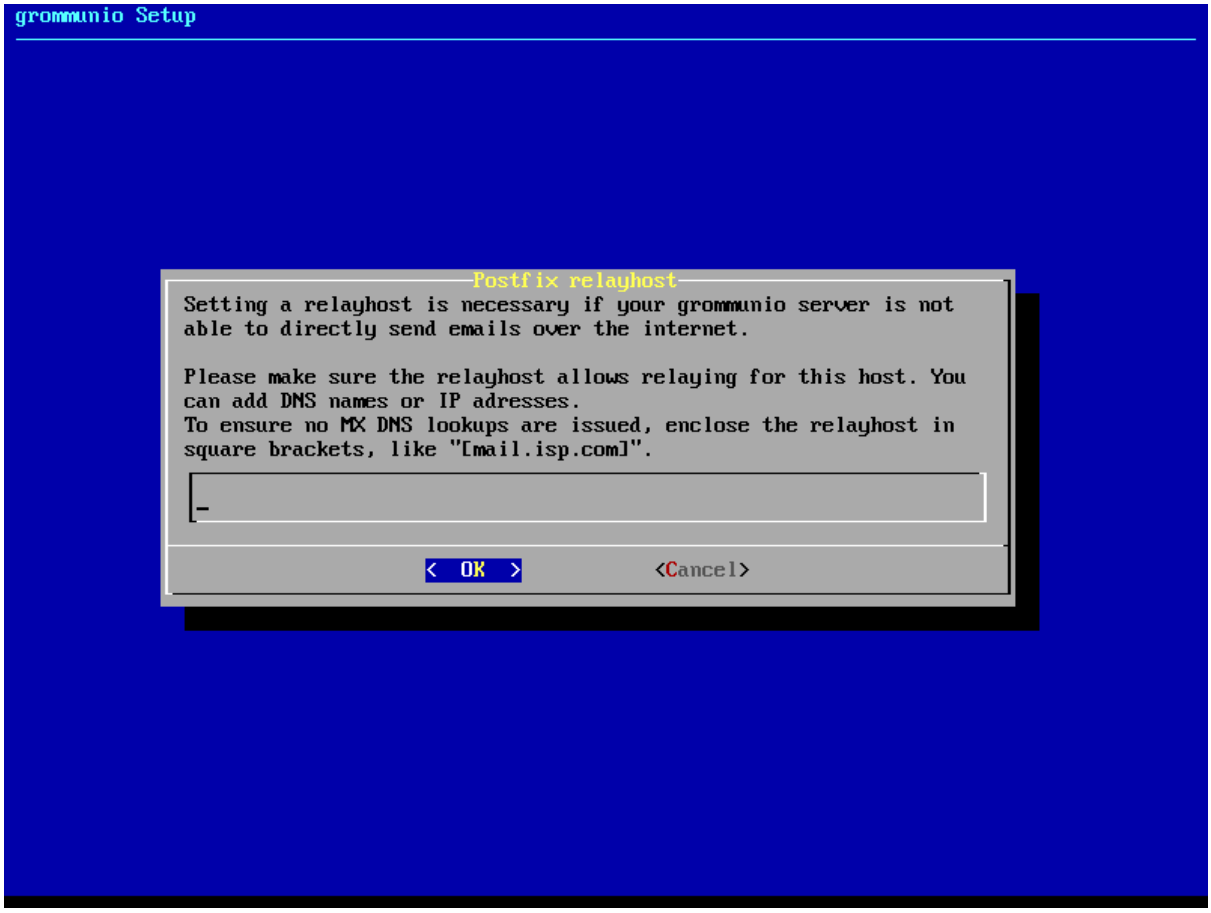
3.9.7 Primary mail domain

By continuing to the next stage, it is requested to provide the primary mail domain. The primary mail domain is important as main system domain for further system configuration.



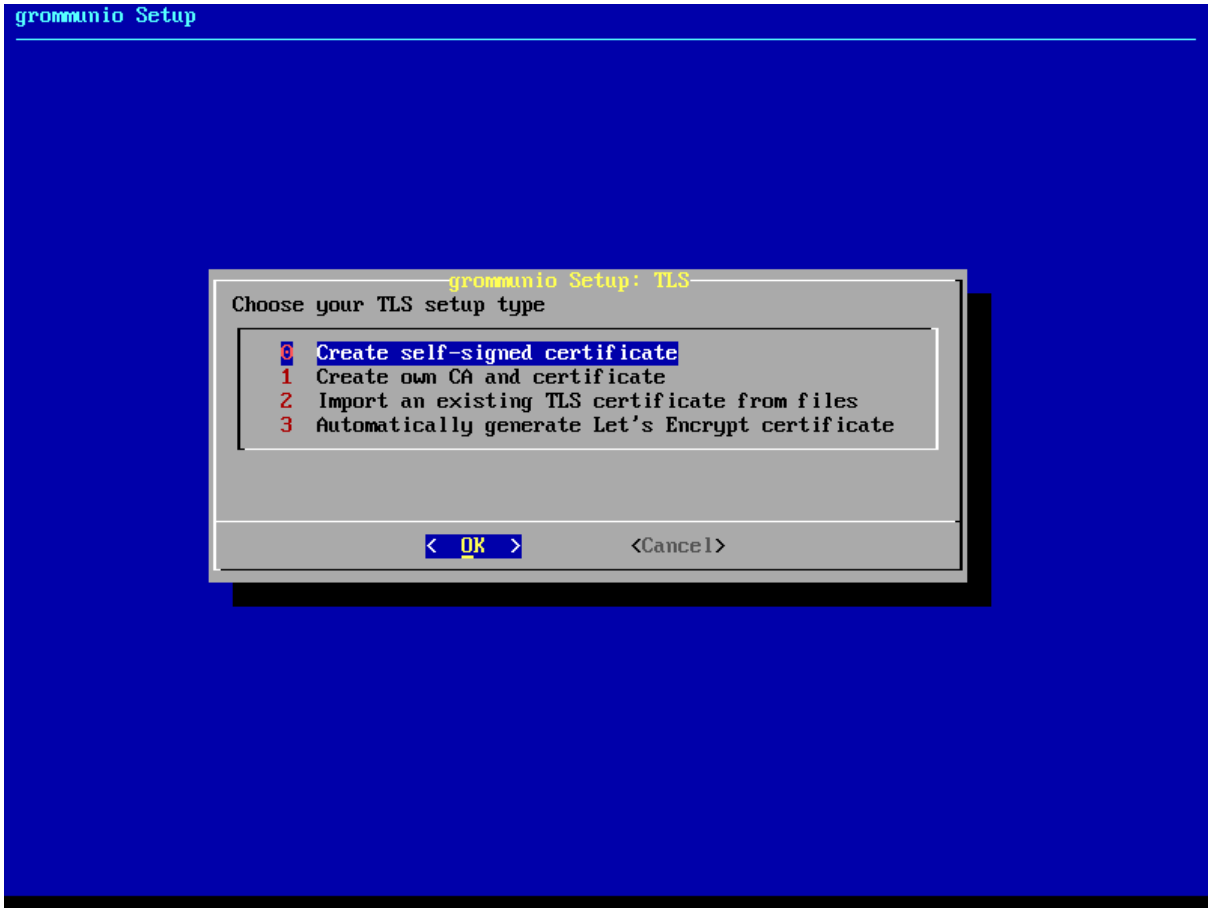
3.9.8 Relayhost configuration

If the installation is not to be directly sending E-Mails (by resolving the recipients' MTAs directly), a relayhost is recommended to be set. This next step allows the configuration of a relayhost which for example can be used for integration with existing firewalls or mail security appliances. If the configured target should be used directly (by requesting the IP address through DNS A records instead of the associated MX records), the relayhost should be enclosed with square brackets, like "[mail.isp.com]".



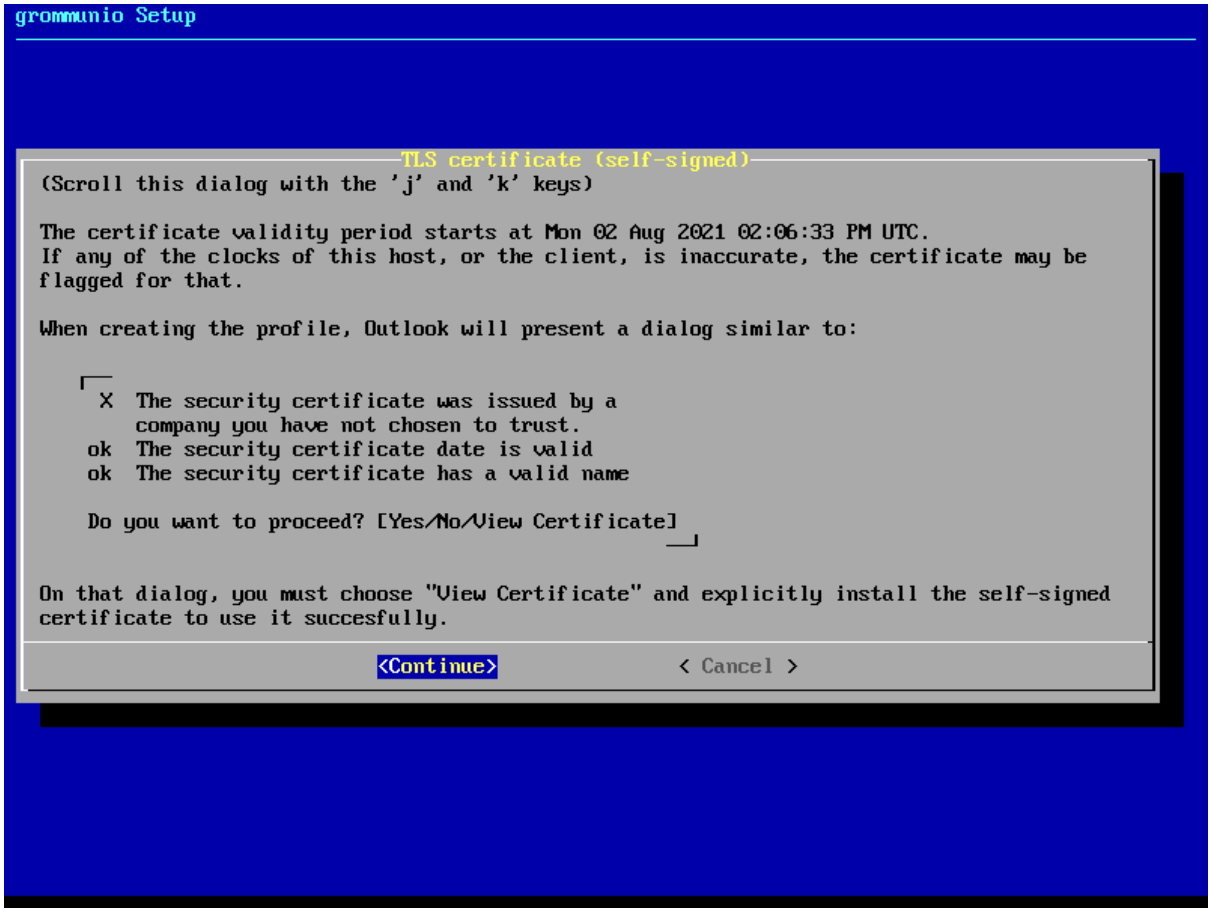
3.9.9 TLS configuration

The next step of configuration with grommunio-setup provides a menu with a choice of the preferred TLS setup with the grommunio installation:



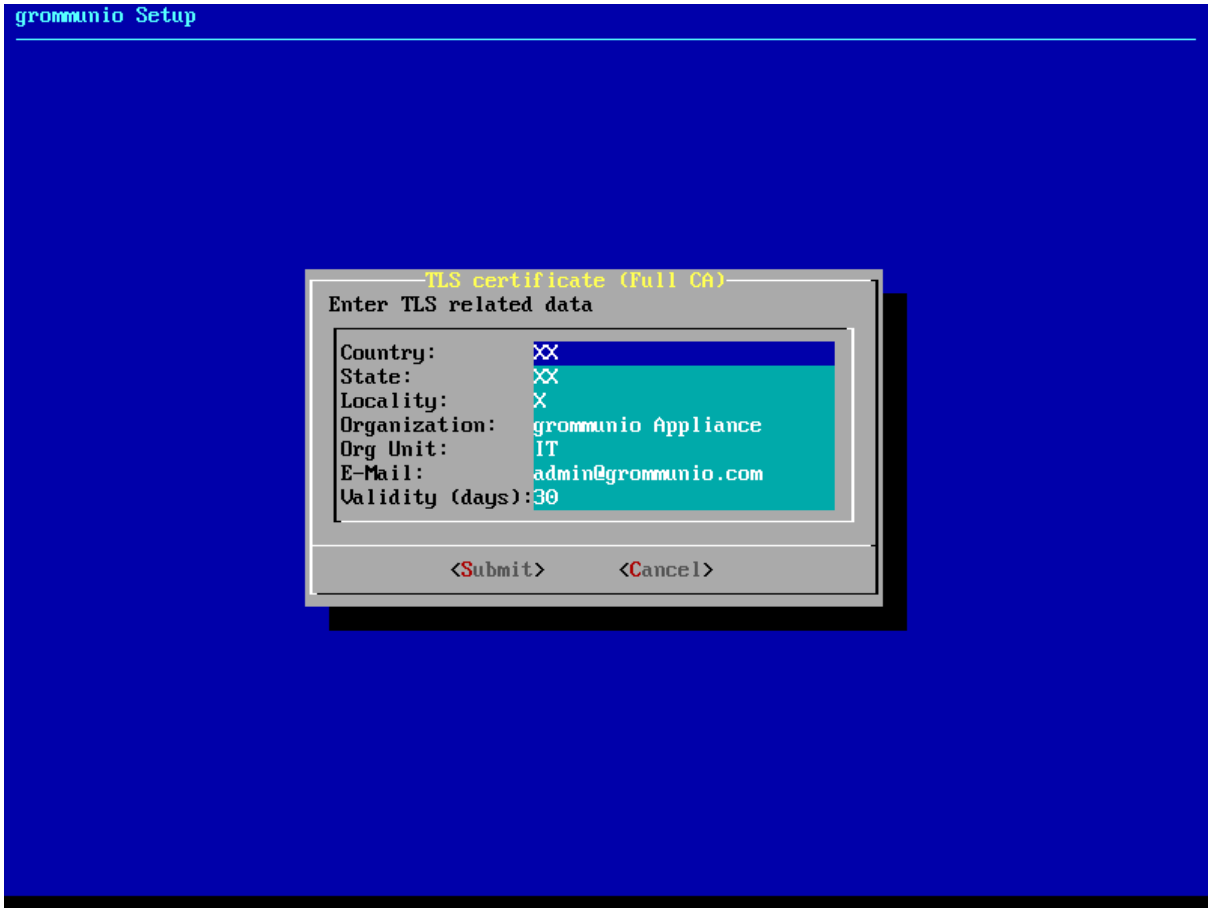
o: **Creation of self-signed certificate**

Creating your own self-signed certificate is the simplest option - Creating an own self-signed certificate will though show up as untrusted at first connect and needs to be trusted before continuing. This behavior is normal and is because any client that connects has no possibility validation if the certificate has a valid source. This setting is the default and does not require any preparation for certificate generation. grommunio does not recommend this option for production environments, as this option requires any client to first trust the certificate in use. This option is the best for validation and demo installations of grommunio.



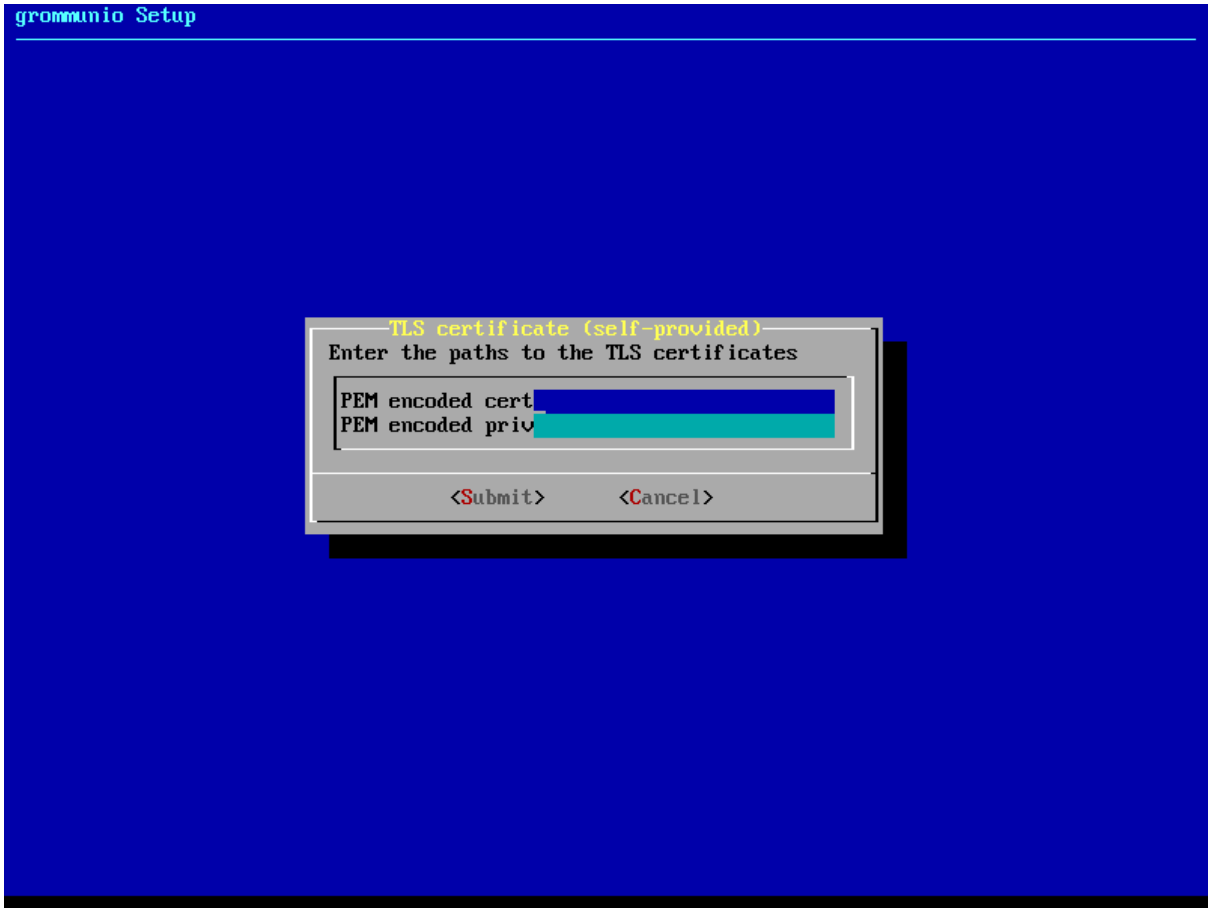
1: **Creation of own CA (certificate authority) and certificate**

Creating your own certificate authority is an extended option which allows you to create self-signed certificates with an own certificate authority. This way, you can (manually) create further certificates under the umbrella of a own central authority with multiple server certificates to be signed by the same certificate authority generated by yourself. This option is the best for validation and demo installation of larger installations of grommunio with multiple instances.



2: **Import of an existing TLS certificate from files**

Importing your own certificate allows any type of external certificate pair (PEM-encoded) to be used with your grommunio installation. Note that it is recommended to either use SAN certificates with multiple domains or a wildcard certificate. With your choice of your own TLS certificates, you have the highest flexibility to either use a trusted CA or a publicly signed certificate by an officially trusted certification authority including, but not limited to, Thawte, Digicert, Comodo or others.



3: **Automatic generation of certificates with Let's Encrypt**

Using this option allows the automatic certificate generation process with the Let's Encrypt certificate authority. Using Let's Encrypt certificates is free of charge, however the terms of service by Let's Encrypt apply, which are referenced during installation. Using this option automatically requests the domains from the selection you made, and automatically starts the validation process. For this automated process to work successfully, Let's Encrypt verifies _all_ defined domain names by creating a challenge on the appliance. For this to work, port 80 (HTTP) needs to be accessible from the Internet during this step of verification (and any subsequent automated renewal) with all the domains pointing to the appliance. This option is recommended for any simple installation and allows the most seamless installation experience if prepared correctly.

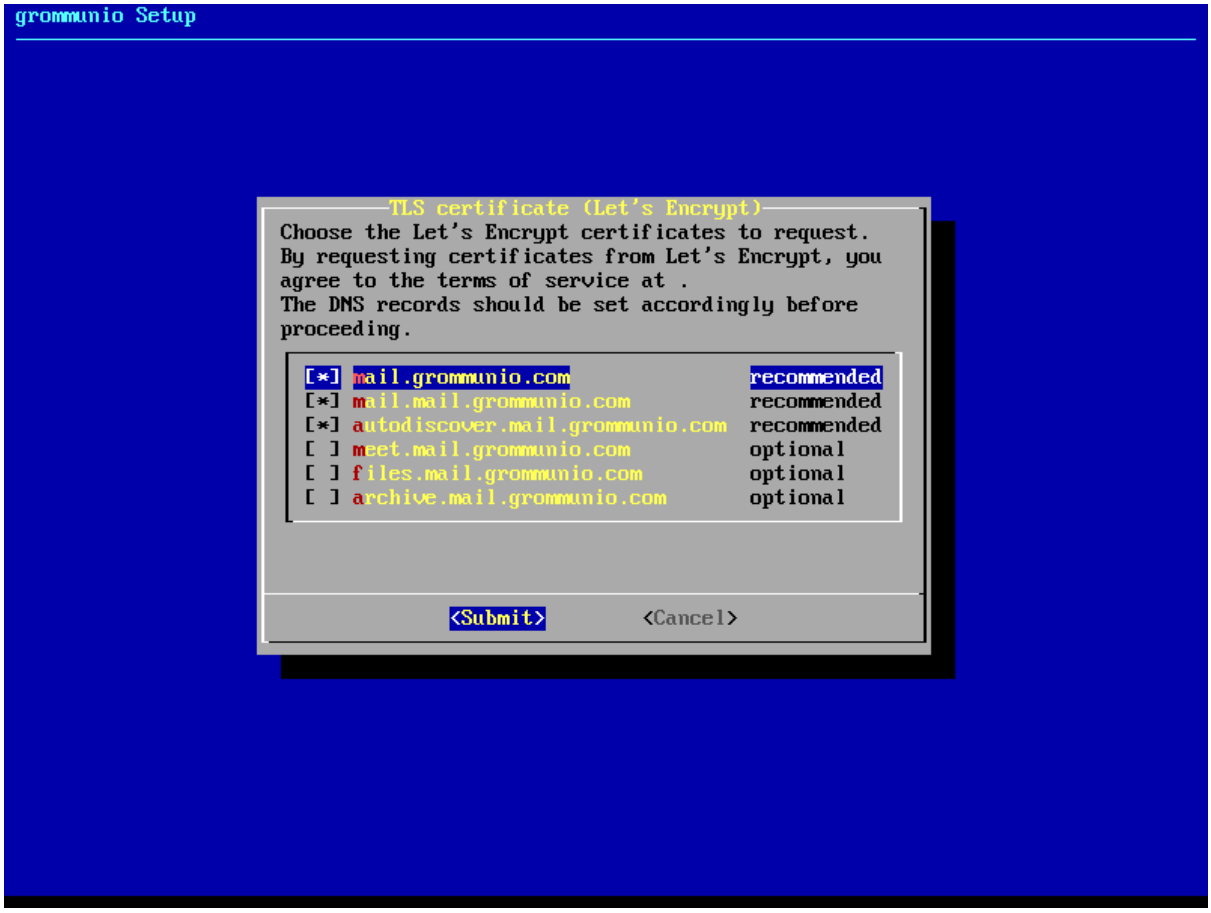
3.a: **Generation of certificates with Let's Encrypt for Multi-Domains**

For adding more domains to your Let's Encrypt certificate you can use the following command:

```
certbot certonly -n --standalone --agree-tos \
--preferred-challenges http \
--cert-name="<domain1>" \
-d "<domain1>" \
-d "<domain2>" \
-d "<domain3>" \
-d "<domain4>" \
-d "<domain5>" \
-m "me@domain1.com" \
--pre-hook "service nginx stop" \
--deploy-hook /usr/share/grommunio-setup/grommunio-certbot-renew-hook \
--post-hook "service nginx start"
```

While --cert-name="<domain1>" stands for the original domain and -d "<domain2>" to -d

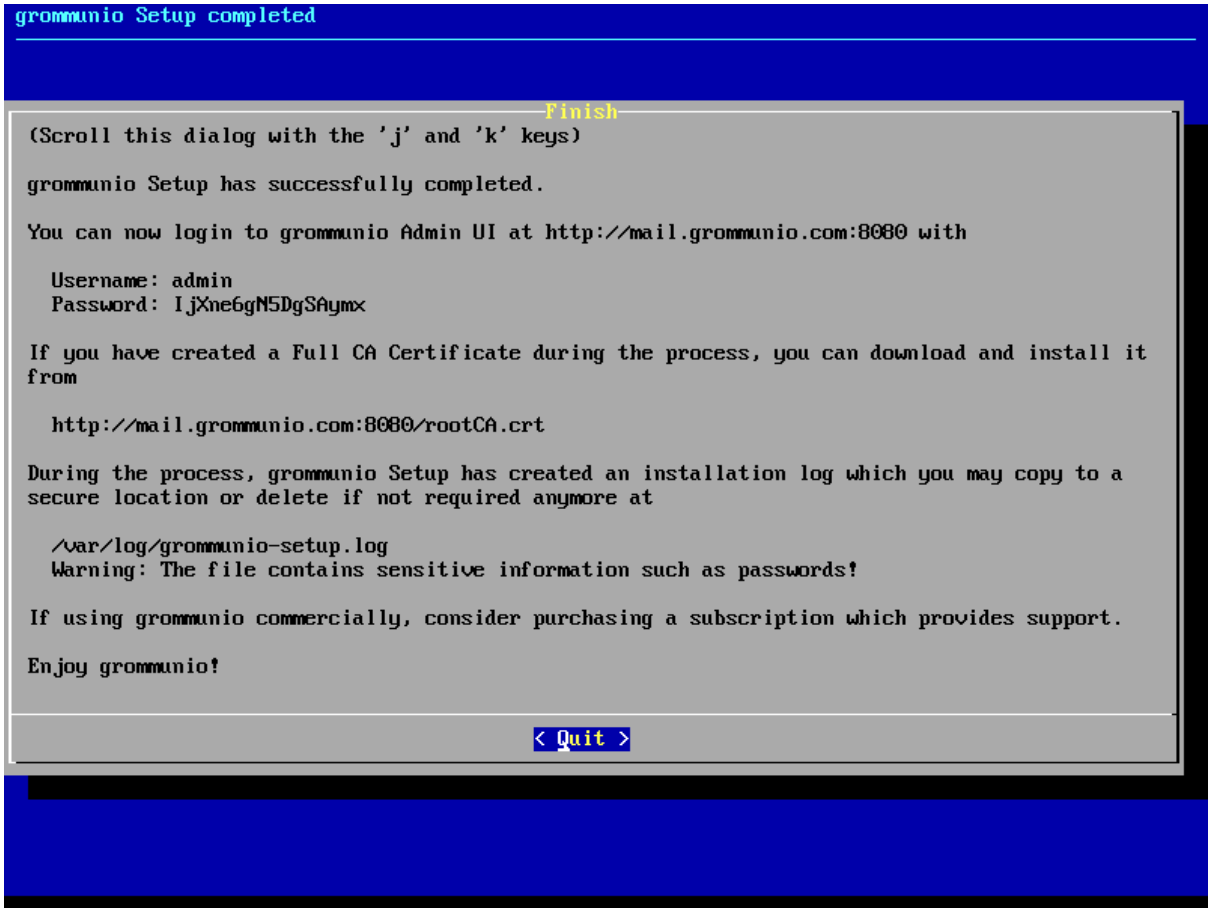
”<domain5>” are the multi domains to add to the LE certificate. The -m ”me@domain1.com” is your email address while the --pre-hook ”service nginx stop” stops nginx before the certificate modification, the --deploy-hook /usr/share/grommunio-setup/grommunio-certbot-renew-hook makes the changes and the --post-hook ”service nginx start” starts nginx after the modification.



Any certificates so generated are placed in /etc/grommunio/ssl and are automatically referenced by any services of the appliance.

3.9.10 Setup finalization

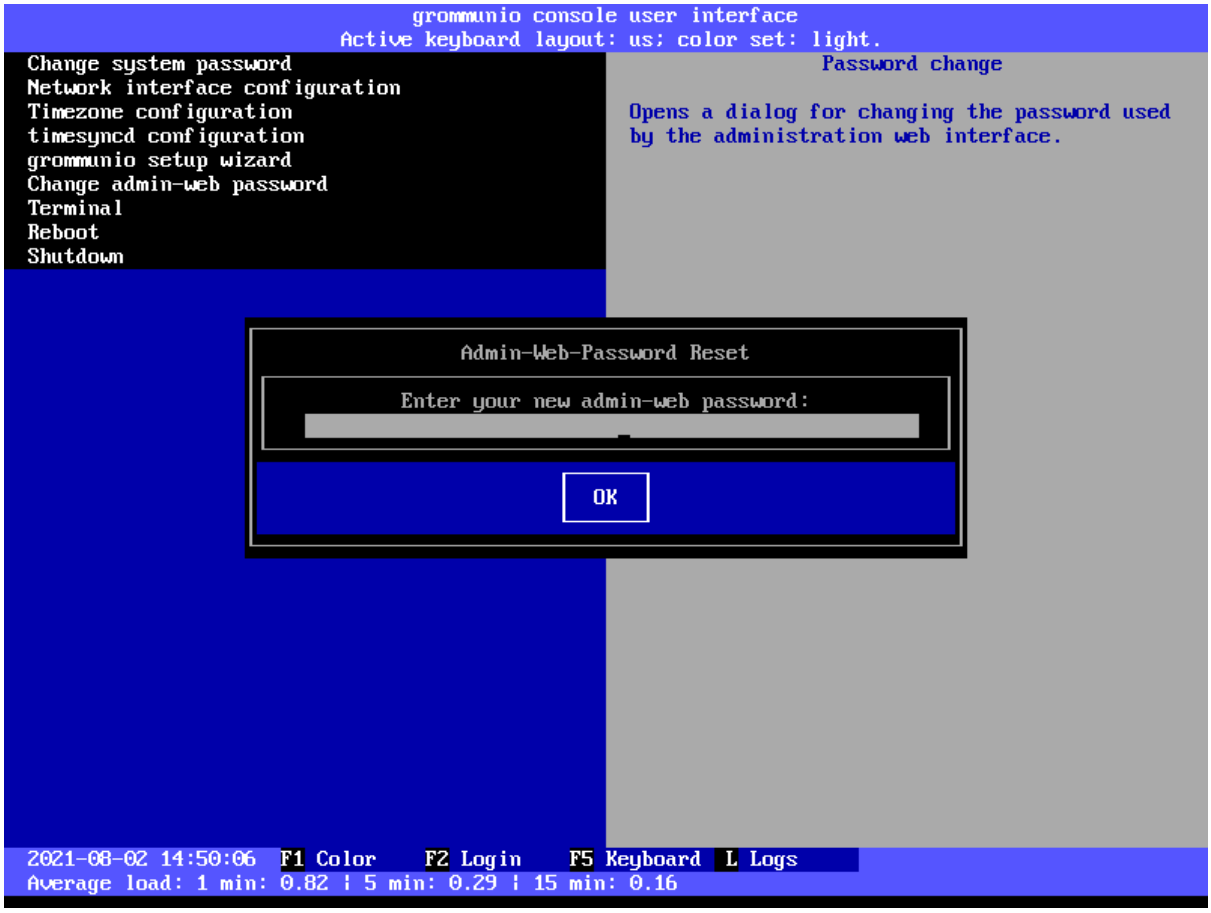
After all above steps of grommunio-setup have been completed, the final dialog shows the summarized information of the installation is shown as reference.



Important: All installation/setup relevant information is stored at /var/log/grommunio-setup.log. This file includes the passwords used for initialization which you may copy to a secure location or delete if not required anymore.

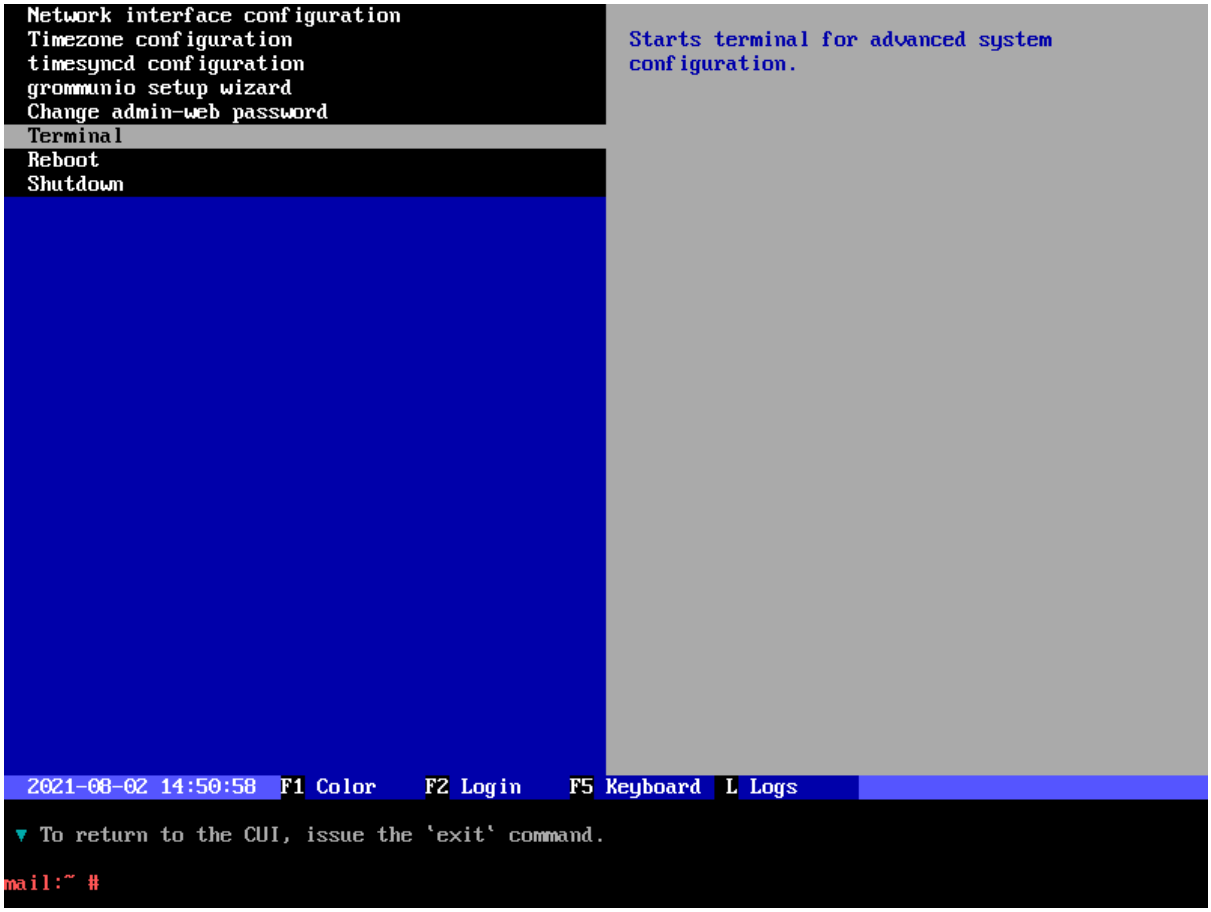
3.10 Admin web password reset

The menu entry Admin web password reset changes the password of the main administration user (admin). For administrators which want to execute this option without running grommunio-cui first, this can be done anytime by executing the command grommunio-admin passwd.



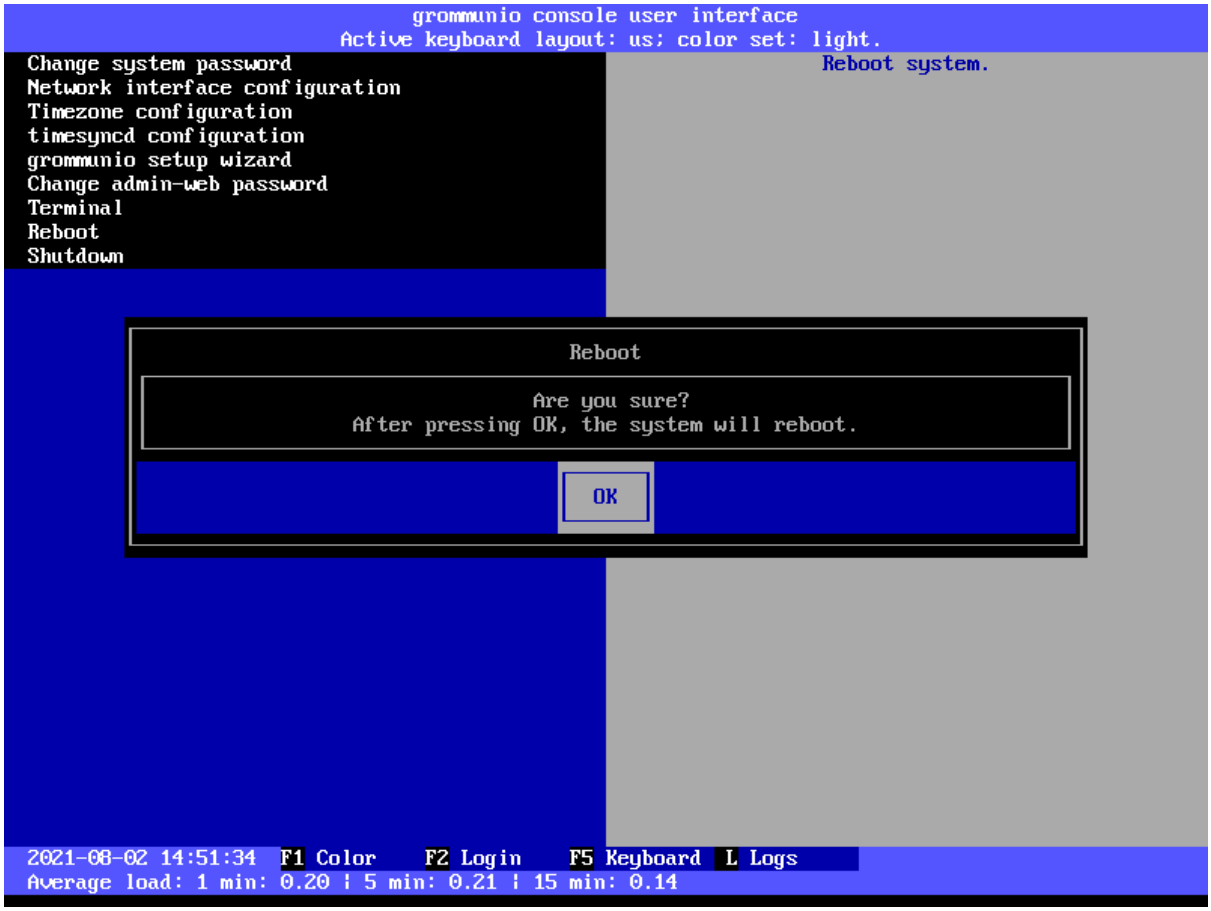
3.11 Terminal

The option Terminal enables a class shell with the ability to exit back to grommunio-cui by issuing the exit command at any given time. This option should be used with care and only by experienced administrators.



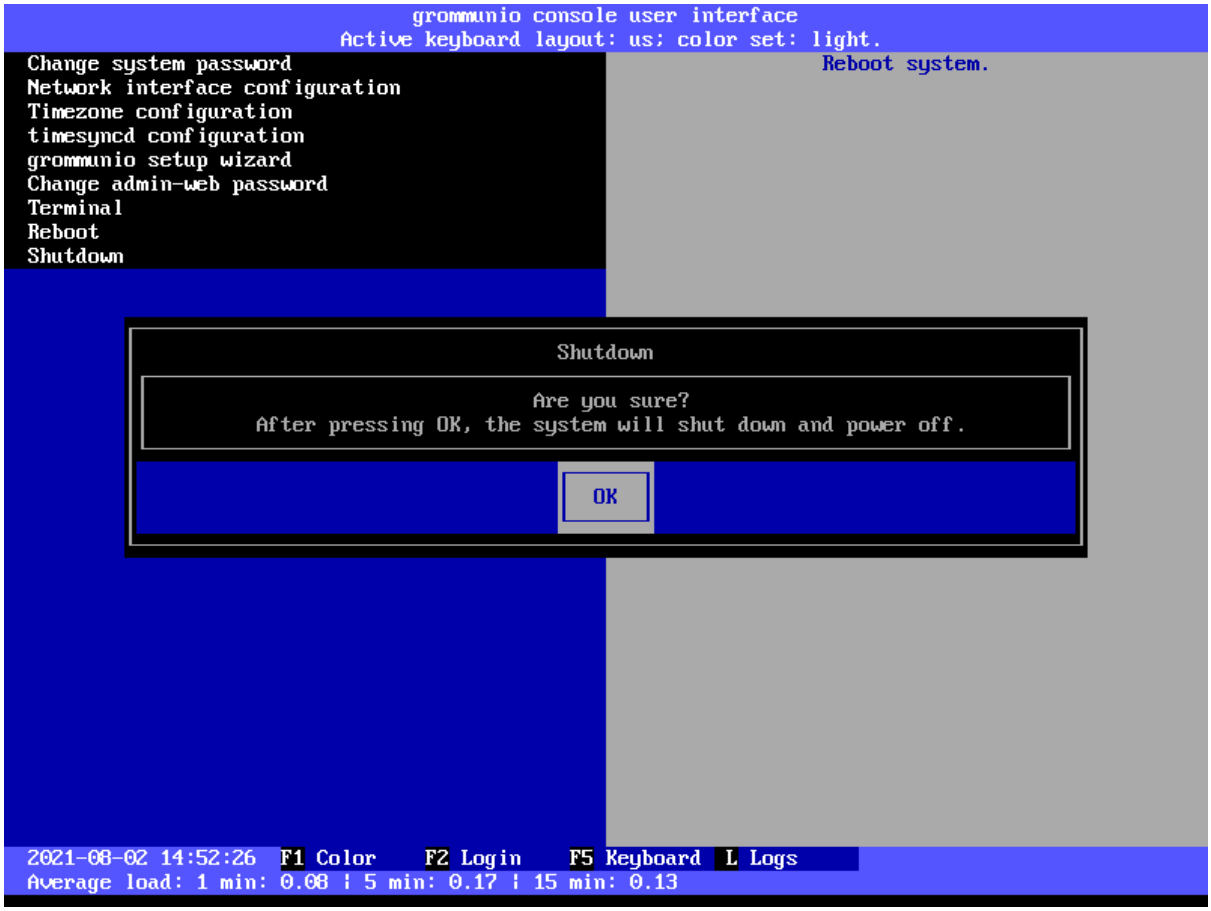
Important: Note that the Terminal executed here provides full administrative rights (root access) to the Appliance. With this level of permissions it is recommended to proceed with extreme caution.

3.12 Reboot



The option Reboot reboots the entire grommunio Appliance. Note that during the reboot the services provides will not be available.

3.13 Shutdown



The option Shutdown shuts down the entire grommunio Appliance. Note that until the Appliance has been made available again by starting it again, the services will not be available.

CHAPTER 4

Manual Installation (Custom Integration)

While the grommunio Appliance delivers a comprehensive solution for the majority of installations, some special needs might require a different approach. For these cases, the grommunio base system and core (groupware) can be installed manually with guidance from this chapter.

Note: grommunio is a comprehensive communication and collaboration solution with many services and components. With this modularity, grommunio is extremely versatile and allows various installation types which all of them can't be covered in detail. This section is intentionally as generic as possible.

Important: Please note that this section is targeted at adept administrators who are experienced in advanced linux administration and configuration.

This chapter assumes a basic system is running already. *Basic* in this regard means:

- a system service manager of some kind should be running (systemd, sysvinit, etc.)
- the system should be in its typical multi-user state (in terms of systemd, *multi-user.target* should have at least been started; in terms of sysvinit, init level 3 or 5)
- should have an interactive shell for you to use
- should not be ephemeral and not lose its state when turned off

4.1 Establish networking

[Text-based screenshot of networkctl being issued from a command shell.]

```
localhost:~ # networkctl
IDX LINK  TYPE     OPERATIONAL SETUP
  1 lo    loopback carrier     unmanaged
  2 host0 ether    routable    configured

2 links listed.
```

(continues on next page)

(continued from previous page)

```
localhost:~ # networkctl status host0
* 2: host0
                     Link File: n/a
                  Network File: /etc/systemd/network/host0.network
                          Type: ether
                         State: routable (configured)
                  Online state: online
                    HW Address: aa:b2:5f:b1:9d:46
                           MTU: 1500 (min: 68, max: 65535)
                         QDisc: noqueue
    IPv6 Address Generation Mode: none
           Queue Length (Tx/Rx): 32/32
               Auto negotiation: no
                         Speed: 10Gbps
                        Duplex: full
                          Port: tp
                       Address: 192.0.2.196
                                2001:db8:10b:45d8::f27
                       Gateway: 192.0.2.1
                                2001:db8:10b:45d8::1
             Activation Policy: up
           Required For Online: yes

Mar 31 23:47:13 localhost systemd-networkd[22]: host0: Link UP
Mar 31 23:47:13 localhost systemd-networkd[22]: host0: Gained carrier
```

For this setup, we enabled systemd-networkd and the network configuration put in place apriori. This section is a reminder to hook up the host to Internet, as it will be needed to get the package repositories later. The particular method of network configuration varies wildly between operating systems, and not every system is using systemd-networkd. Consult the documentation relevant for your environment to get online.

[Text-based screenshot of iproute2 being issued from a command shell.]

```
localhost:~ # ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: host0@if17: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group␣
↪default qlen 1000
    link/ether aa:b2:5f:b1:9d:46 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.0.2.196/24 scope global host0
       valid_lft forever preferred_lft forever
    inet6 2001:db8:10b:45d8::f27/64 scope global
       valid_lft forever preferred_lft forever
```

IPv6 is mandatory on the host itself. If you have ::1 assigned, that is sufficient.

We advise that a packet filter (i.e., a firewall) should be installed and configured by default to disallow every service. More details will be presented throughout the sections going forward. However, the following ports need to be open for grommunio to function properly:

- VPN, SSH and/or port 8443 (AWEB) for the admin
- smtp/25 for server-to-server mail passing
- https/443 for end-user interactions
- imaps, pop3s for end-user interactions if desired

4.2 Declare hostname identity

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
localhost:~ # echo mail.example.net >/etc/hostname
localhost:~ # hostname mail.example.net
localhost:~ # exec bash --login
mail:~ #
```

If you have not set a hostname yet, do so, especially if some default setting has left you with localhost as the hostname. You cannot reasonably reach localhost from another machine without unnecessary pains.

We used example.net for the domain part of later e-mail addresses (e.g. someuser@example.net), and this machine that Grommunio will be installed has a hostname of mail.example.net. Arbitrary names can be chosen as long as they are meaningful for their intended network.

Note: The hostname(5) manual page provided in Linux/systemd systems says that the name in /etc/hostname should be a single label (no dots). If you choose to do this, and if the single label does not constitute a fully-qualified name already, you must set the host_id directive in /etc/gromox/http.cfg to the fully-qualified name. (AutoDiscover responses contain references to the server. Other services like zcore, imap, etc. do not depend on FQDNs.)

4.3 Package manager setup

Pre-build packages are available for different platforms on https://download.grommunio.com. Different operating systems may use the same archive format (RPM, DEB, etc.), or the same repository metadata formats (such as rpm-md, apt). However, do not use a repository which does not *exactly match* your system nor do not attempt to convert between formats. This action might lead to unnecessary problems.

4.3.1 zypp

openSUSE uses yum-style .repo files for declaring repositories. For openSUSE Leap 15.4, you can create a file /etc/zypp/repos.d/grommunio.repo and populate it as below:

```
[grommunio]
enabled=1
autorefresh=1
baseurl=https://download.grommunio.com/community/openSUSE_Leap_15.4
type=rpm-md
keeppackages=0
```

Retrieve the GPG key and import it into the RPM database to trust it. Then, optionally, download the repository metadata (if not, it will be done the next time you install anything).

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # curl https://download.grommunio.com/RPM-GPG-KEY-grommunio >gr.key
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3175  100  3175    0     0  18021      0 --:--:-- --:--:-- --:--:-- 18039
mail:~ # rpm --import gr.key
```

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # zypper ref grommunio
Retrieving repository 'grommunio' metadata ... [done]
Building repository 'grommunio' cache ... [done]
Specified repositories have been refreshed.
```

4.3.2 dnf

RHEL uses .repo files as well, though in another directory. The file to edit would be /etc/yum.repos.d/grommunio.repo, with contents:

```
[grommunio]
name=grommunio for Enterprise Linux 9
baseurl=https://download.grommunio.com/community/EL9/
enabled=1
gpgcheck=1
gpgkey=https://download.grommunio.com/RPM-GPG-KEY-grommunio
```

Accept the GPG key during the first package installation or update when proceeding with dnf or yum commands.

Note: Our packages depend on packages in the CodeReady Linux Builder and the EPEL repository. To enable them, run dnf install epel-release followed by crb enable.

4.3.3 apt

For Debian-based systems, the repository information needs to be added. Create a new file in /etc/apt/sources.list.d/, e.g. grommunio.sources:

```
Types: deb
URIs: https://download.grommunio.com/community/Debian_12
Suites: Debian_12
Components: main
Signed-By: /usr/share/keyrings/download.grommunio.com.gpg
```

```
# wget -qO - https://download.grommunio.com/RPM-GPG-KEY-grommunio | gpg --dearmor --output /usr/
↪share/keyrings/download.grommunio.com.gpg
```

(This equally works for *Ubuntu_22.04*, for example. For the specific case of Ubuntu installations however, the Ubuntu universe repository is *also* required, so be sure to enable it. For Debian, the base distribution is sufficient.)

Then import the GPG key and update the repositories.

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
# curl https://download.grommunio.com/RPM-GPG-KEY-grommunio >/etc/apt/trusted.gpg.d/download.
↪grommunio.com.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3175  100  3175    0     0  50396      0 --:--:-- --:--:-- --:--:-- 50396
```

Note: The apt-key command is deprecated and should no longer be used. For more information, see the apt-key(8) manpage.

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
# apt-get update
Hit:1 http://gb.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://gb.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://gb.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:4 https://download.grommunio.com/community/Ubuntu_22.04 Ubuntu_22.04 InRelease [4,692 B]
Hit:5 http://security.ubuntu.com/ubuntu jammy-security InRelease
Get:6 https://download.grommunio.com/community/Ubuntu_22.04 Ubuntu_22.04/main amd64 Packages [7,
↪072 B]
Get:7 https://download.grommunio.com/community/Ubuntu_22.04 Ubuntu_22.04/main i386 Packages [4,
↪637 B]
Fetched 16.4 kB in 1s (23.8 kB/s)
Reading package lists... Done
```

4.4 TLS certificates

For obtaining a certificate, refer to external documentation.

- Self-signed certificate: https://stackoverflow.com/a/10176685
- Let's Encrypt certificate: https://certbot.eff.org/instructions

The certificate's key must be passwordless as interactive prompting is not implemented.

If you plan on using multiple subdomains for your deployment, e.g. meet.example.net for *grommunio-meet* and mail.example.net for *grommunio-web*, the generation a certificate with a *subjectAltName* (SAN) field, or even a wildcard certificate, may be desirable over individual certificates. Not all network protocols have something like Server Name Indication (SNI), and even fewer system services support multiple individual certificates even if they serve multiple IP addresses.

Autodiscover clients, as part of their routine, attempt to resolve and use autodiscover.example.net in addition to example.net. Because it tries multiple names, a SAN entry for the autodiscover. subdomain is not strictly necessary. See MS-OXDISCO §3.1.5 for further details.

The following services need access to the certificate(s):

- gromox
- nginx
- postfix (optional)

Some processes read TLS certificates and their keyfiles *after* switching to an unprivileged user identity. For this reason, certificate files may need to be enhanced with a filesystem ACL or duplicate copies be made with suitable ownership.

4.5 nginx

nginx is used as a frontend to handle HTTP requests. RPC/HTTP requests are proxied to Gromox, Administration API (AAPI for short) requests are proxied to an uwsgi instance, and requests to the chat API are proxied to a Mattermost instance.

An alternative HTTP server may be used if you feel comfortable in configuring *all* of it, however this guide will only focus on nginx from here on.

Obtain the nginx package for your operating system, and have the service started both on next boot and immediately.

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # zypper in nginx nginx-module-vts
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following 26 NEW packages are going to be installed:
  fontconfig libX11-6 libX11-data libXau6 libXpm4 libaom3 libavif13 libdav1d5
  libdb-4_8 libexslt0 libfontconfig1 libfreetype6 libgd3 libgdbm6
  ilbgdbm_compat4 libjbig2 libjpeg8 libpng16-16 librav1e0 libtiff5 libwebp7
  libxcb1 libxslt1 nginx nginx-module-vts perl

26 new packages to install.
Overall download size: 15.2 MiB. Already cached: 0 B. After the operation,
additional 68.4 MiB will be used.
Continue? [y/n/v/...? shows all options] (y):
```

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
(22/26) Installing: libXpm4-3.5.13-1.8.x86_64 ... [done]
(23/26) Installing: libfontconfig1-2.13.1-2.12.x86_64 ... [done]
(24/26) Installing: libgd3-2.3.3-2.2.x86_64 ... [done]
(25/26) Installing: nginx-1.21.5-1.1.x86_64 ... [done]
Additional rpmoutput:
/usr/bin/systemd-sysusers --replace=/usr/lib/sysusers.d/nginx.conf -
Creating group nginx with gid 477.
Creating user nginx (User for nginx) with uid 477 and gid 477.
(26/26) Installing: nginx-module-vts-0.1.116-1.1.x86_64 ... [done]
mail:~ # systemctl enable --now nginx
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/
↪nginx.service
```

In this screenshot, we also requested the installation of the nginx VTS module, which AAPI uses for reporting traffic statistics. VTS is **not** available for all platforms, and you can omit it.

Being the main entrypoint for everything, the nginx HTTPS network service must be configured in the packet filter to be publicly accessible. In other words, open port 443.

By *default*, Debian-based distributions ship default web server configs which are in conflict with grommunio. It is recommended to remove the default web service entry, generally located at /etc/nginx/sites-enabled/default. By removing this link, the webserver default website is disabled.

Configuration snippets should solely be edited under /etc/. Files in /usr belong to the vendor/the distribution and are subject to (silent) changes when an update is processed by the package manager.

4.6 nginx support package

The grommunio-common package contains the initial configuration fragments for nginx. Install it.

```
zypper in grommunio-common
```

The nginx default configuration as shipped by Linux distributions (file /etc/nginx/nginx.conf) contains a line include conf.d/*. The support package places a file to /etc/nginx/conf.d/grommunio.conf, such that the nginx-related grommunio configuration gets automatically loaded on the next nginx (re-)start.

The actual fragment files for nginx are located under /usr/share/grommunio-common for packaging policy reasons; they are not meant to be modified. However, they have further include directives pointing back to /etc to facilitate overriding specific aspects.

/usr/share/grommunio-common/nginx/locations.d/autodiscover.conf for example contains the fragment that tells nginx to recognize the /Autodiscover space and forward such requests to gromox-http on port 10443 (see later section).

4.7 TLS for nginx

Create /etc/grommunio-common/nginx/ssl_certificate.conf and populate with the certificate directives, exchanging paths as appropriate:

```
ssl_certificate zzz.pem;
ssl_certificate_key zzz.key;
```

(The exact chain of includes is /etc/nginx/nginx.conf > /etc/nginx/conf.d/grommunio.conf > /usr/share/grommunio-common/nginx.conf > /etc/grommunio-common/nginx/ssl_certificate.conf.)

The port 80 and 443 listen declarations are provided by /usr/share/grommunio-common/nginx.conf.

nginx's configuration can be tested and shown, respectively:

```
nginx -t
nginx -T
```

4.8 MariaDB

MariaDB/MySQL is used to store the user database and other auxiliary configuration parameters. If you plan on setting up a Gromox cluster, this database needs to be globally available to all nodes that will host Gromox services.

A preexisting MariaDB server may be used. All the standard tools and procedures that the database community has developed around SQL are applicable, in terms of e.g. configuration, backup/restore, and replication.

Assuming that you are going for a new SQL server instance, source the MariaDB packages from your operating system, and have the service started both on next boot and immediately.

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # zypper in mariadb mariadb-client
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following 15 NEW packages are going to be installed:
  libJudy1 libaio1 libedit0 libltdl7 liblzo2-2 libmariadb3 libodbc2
  libpython3_8-1_0 libwrap0 mariadb mariadb-client mariadb-errormessages
  python38-base python38-mysqlclient

15 new packages to install.
Overall download size: 33.3 MiB. Already cached: 0 B. After the operation,
additional 160.7 MiB will be used.
Continue? [y/n/v/...? shows all options] (y):
```

```
(13/15) Installing: python38-base-3.8.12-3.2.x86_64 ... [done]
(14/15) Installing: pytnon38-mysqlclient-2.0.3-2.2.x86_64 ... [done]
(15/15) Installing: mariadb-10.6.5-4.1.x86_64 ... [done]
mail:~ # systemctl enable --now mariadb
Created symlinks /etc/systemd/system/mysql.service → /usr/lib/systemd/system/mariadb.service.
Created symlink /etc/systemd/system/multi-user.target.wants/mariadb.service → /usr/lib/systemd/system/
↪mariadb.service
```

After the installation, create a blank database and user identity for accessing it.

[Terminal screenshot of an interactive mysql session.]

```
mail:~ # mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4
Server version: 10.6.5-MariDB MariaDB package

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE `grommunio`;
Query OK, 1 row affected (0.001 sec)

MariaDB [(none)]> GRANT ALL ON `grommunio`.* TO 'grommunio'@'localhost' IDENTIFIED BY
↪'freddledgruntbuggly';
Query OK, 0 rows affected (0.004 sec)

MariaDB [(none)]>
```

```
CREATE DATABASE `grommunio`;
GRANT ALL ON `grommunio`.* TO 'grommunio'@'localhost' IDENTIFIED BY 'freddledgruntbuggly';
```

The MariaDB network service is not meant to be open to the public Internet. Within your private network, it may need to be opened if (and only if) you plan on using it in a multi-host Grommunio setup, or when your plans about database replication demand it.

In certain versions, such as MySQL 8 (on e.g. Ubuntu 20.04), the GRANT statement no longer implicitly creates users and one must use CREATE USER instead. Furthermore, authentication with MariaDB/older MySQL clients may fail due to what appears to be a hashing method change; the remedy is an extra parameter for CREATE USER or ALTER USER.

4.9 Gromox in general

Gromox is the central groupware server component of grommunio. It provides the services for Outlook RPC, IMAP/POP3, an LDA for ingestion, and a PHP module for Z-MAPI.

The pre-built package is available in the Grommunio repositories. This guide is subsequently based on such a pre-built Gromox. Experts wishing to build from source and who have general knowledge on how to do so are referred to the Gromox installation documentation on specific aspects of the build procedure.

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # zypper in gromox
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following 26 NEW packages are going to be installed:
  gromox libHX32 libbfio1 libcdata1 libcerror1 libcfile1 libclocale1 libcnotify1
  libcpath1 libcsplit1 libcthreads1 libfcache1 libfdata1 libfmapi1 libgumbo1
  ilbjsoncpp25 libpff1 libuna1 php8 php8-cli php8-mysql php8-pdo php8-soap
  system-user-gromox system-user-wwwrun timezone

26 new packages to install
Overall download size: 5.8 MiB. Already cached: 0 B. After the operation,
additional 19.3 MiB will be used.
Continue? [y/n/v/...? shows all options] (y):
```

Gromox runs a number of processes and network services. None of them are meant to be open to the public Internet, because nginx is already that important point of ingress. The Gromox exmdb

service (port 5000/tcp by default) needs to be reachable from other Gromox nodes in a multi-host grommunio setup for internal forwarding to a mailbox's home server.

Daemon executables are located in /usr/libexec/gromox, they have short names like http, zcore, etc. The manpage carries the same name, so you would use man http to call up the corresponding manpage. The configuration files read by default follow the same scheme, e.g. /etc/gromox/http.cfg. Process information utilities such as ps(1) may show the full path of the executable or just http, depending on how these diagnostic utilities are used. The systemd unit name, though, is gromox-http.service.

All log output goes to stderr. When run from systemd, this is automatically redirected to the journal.

4.10 Gromox user database

The connection parameters for MariaDB need to be conveyed to Gromox with the file /etc/gromox/mysql_adaptor.cfg, whose contents could look like this:

```
mysql_username=grommunio
mysql_password=freddledgruntbuggly
mysql_dbname=grommunio
schema_upgrade=host:mail.example.net
```

Make sure to set restrictive permissions on this file (cf. section *Permissions*).

The data stored in MariaDB is shared among all mailbox nodes in a clustered setup. Table schema (DDL) changes are necessary at times, but at most one node in such a cluster should perform these changes to avoid running the risk of corruption. The hostname after host: specifies which machine will be considered authoritative, if any. The schema_upgrade=host:... line should be consistent across all mailbox nodes. It is possible to completely omit schema_upgrade, at which point no updates will be done automatically.

After the database parameters have been set in the configuration file, the initial tables can be created by issuing the gromox-dbop command:

[Text-based screenshot of shell prompts (not part of the command) and commands to issue.]

```
mail:~ # gromox-dbop -C
Creating admin_roles
Creating associations
Creating configs
Creating domains
Creating forwards
Creating groups
Creating hierarchy
Creating members
Creating mlists
Creating options
Creating orgs
Creating specifieds
Creating users
Creating aliases
Creating user_properties
Creating admin_role_permission_relation
Creating admin_user_role_relation
Creating classes
Creating fetchmail
Creating secondary_store_hints
Creating user_devices
Creating user_device_history
Creating task_queue
mail:~ #
```

If automatic schema upgrades are disabled, manual updates can be performed later with:

```
gromox-dbop -U
```

4.11 gromox-event/timer

gromox-event is a notification daemon for an interprocess channel between gromox-imap/gromox-midb. No configuration is needed.

gromox-timer is an at(1)/atd(8)-like daemon for delayed delivery. No configuration is needed.

```
systemctl enable --now gromox-event gromox-timer
```

4.12 gromox-http

Because nginx was set up earlier as a frontend to listen on ports 80 and 443, gromox-http needs to be moved "out of the way" (its built-in defaults are also 80/443). In addition, the daemon needs to be told the paths to the TLS certificates. A manual page is provided with all the configuration directives and can be called up with `man 8gx http`. For now, these directives for `/etc/gromox/http.cfg` should suffice:

```
listen_port=10080
listen_ssl_port=10443
http_support_ssl=yes
http_certificate_path=zzz.pem
http_private_key_path=zzz.key
```

Run the service.

```
systemctl enable --now gromox-http
```

Perform a connection test. The expected result of requesting the / URI will be a 404 status code. (It could serve a static HTML file, but the default config has no such file, and / is not mapped to anywhere.)

```
curl -kv https://localhost:10443/
```

Expected output:

```
> GET / HTTP/1.1
> Host: localhost:10443
...
< HTTP/1.1 404 Not Found
...
```

Gromox's default config however has a mapping for `/web` (to `/usr/share/grommunio-web`). If you have the `grommunio-web` package already installed, requests to this subdirectory will succeed. You can test the following URLs (port 10443 for gromox-http directly, 443 for nginx, respectively) with curl from the server command-line, and it should serve a static file:

```
curl -kv https://localhost:10443/web/version
curl -kv https://localhost:443/web/version
# firefox https://mail.example.net/web/version
```

Using a browser from a separate desktop machine is also possible provided port 10443 was made accessible. (Normally, 10443 need not be exposed to any other hosts.) The result for localhost:10443 and localhost:443 should be the same. Expected output:

```
< HTTP/1.1 200 OK
< Date: Tue, 29 Mar 2024 23:08:33 GMT
< Content-Type: text/plain
< Content-Length: 26
< Accept-Ranges: bytes
< Last-Modified: Tue, 29 Mar 2024 07:09:12 GMT
< ETag: "19165e1100000000-1a000000-98b0426200000000"
<
User-agent: *
Disallow: /
```

4.13 gromox-midb & zcore

gromox-midb is the IMAP Message Index Database, and gromox-zcore the bridge process for PHP-MAPI. No further configuration needed.

```
systemctl enable --now gromox-midb gromox-zcore
```

4.14 gromox-imap & pop3

Similar to http.cfg, configure the TLS certificate paths for the IMAP/POP3 daemons. Skip this section if you do not intend to run these protocols.

IMAP/POP3 can run in unencrypted mode, but only for developers. Hence, imap_force_starttls is set here. In /etc/gromox/imap.cfg, declare:

```
listen_ssl_port=993
imap_support_starttls=true
imap_certificate_path=zzz.pem
imap_private_key_path=zzz.key
imap_force_starttls=true
```

In /etc/gromox/pop3.cfg:

```
listen_ssl_port=995
pop3_support_stls=true
pop3_certificate_path=zzz.pem
pop3_private_key_path=zzz.key
pop3_force_stls=true
```

Enable and start services you wish to utilize. Adjust your packet filter configuration for these new ports as needed.

```
systemctl enable --now gromox-imap gromox-pop3
```

Manual testing can be performed with a utility like *telnet*, *socat*, and *curl* can issue more complex IMAP/POP3 protocol command chains.

```
curl -kv imaps://localhost/
curl -kv pop3s://localhost/
```

Expected output for IMAP:

```
*   Trying ::1:993...
...
< * OK mail.example.net service ready
```

(continues on next page)

(continued from previous page)

```
> A001 CAPABILITY
< * CAPABILITY IMAP4rev1 XLIST SPECIAL-USE UNSELECT UIDPLUS IDLE AUTH=LOGIN␣
↪STARTTLS
< A001 OK CAPABILITY completed
…
```

Expected output for POP3:

```
*   Trying ::1:995...
* TCP_NODELAY set
* Connected to localhost (::1) port 995 (#0)
…
< +OK mail.example.net pop service ready
> CAPA
< +OK capability list follows
< STLS
< TOP
< USER
< PIPELINING
< UIDL
< TOP
< .
> LIST
< -ERR login first
```

4.15 PHP-FPM

The installation of the gromox package already pulls in php-fpm as a dependency.

For completeness, verify that PHP knows about the MAPI module.

```
echo -en '<?php phpinfo(); ?>' | php | grep mapi
```

Verify that the gromox pool file was placed.

```
ls -al /etc/php8/fpm/php-fpm.d/gromox.conf
```

Then enable/start php-fpm:

```
systemctl enable --now php-fpm
```

For completeness, verify that the socket in the pool file was created:

```
ls -al /run/gromox/php-fpm.sock
```

Try to elicit a response from the Autodiscover code, via gromox-http (10443) and/or nginx (443). (/usr/share/grommunio-common/nginx/locations.d/autodiscover.conf defines the handler for the /Autodiscover URI path, to pass all requests to gromox-http on port 10443. gromox-http forwards this to php-fpm. This way, Autodiscover also works in test setups without a frontend like nginx.)

```
curl -kv https://localhost:10443/Autodiscover/Autodiscover.xml
curl -kv https://localhost:443/Autodiscover/Autodiscover.xml
# firefox https://mail.example.net/Autodiscover/Autodiscover.xml
```

Expected result of this operation:

```
> GET /Autodiscover/Autodiscover.xml HTTP/1.1
> Host: localhost:10443
```

(continues on next page)

(continued from previous page)

```
...
< HTTP/1.1 200 Success
< Date: Tue, 29 Mar 2024 23:54:16 GMT
< Transfer-Encoding: chunked
< Content-type: text/html; charset=UTF-8
<
E-2000: invalid request method, must be POST!
```

4.16 Administration API (AAPI)

Install the grommunio-admin-api package. This package contains a command-line interface, and an application server implemented using uwsgi.

```
zypper in grommunio-admin-api
```

Edit /etc/grommunio-admin-api/conf.d/database.yaml to make AAPI aware of the MariaDB configuration:

```
DB:
  host: 'localhost'
  user: 'grommunio'
  pass: 'freddledgruntbuggly'
  database: 'grommunio'
```

Set the password for the AAPI admin. This shell command can also be used later to recover from a lost password situation.

```
grommunio-admin passwd
```

grommunio Admin Web supports the exposure of the available features to be seen in the upper left corner. Since grommunio can be installed in a distributed way, this setting can be configured in /etc/grommunio-admin-common/config.json.

```
{
        "mailWebAddress": "https://mail.example.com/web",
        "chatWebAddress": "https://mail.example.com/chat",
        "videoWebAddress": "https://mail.example.com/meet",
        "fileWebAddress": "https://mail.example.com/files",
        "archiveWebAddress": "https://mail.example.com/archive"
}
```

This configuration file needs to be made available to nginx, ideally in the pluggable location of /etc/grommunio-admin-common/nginx.d/web-config.conf.

```
location /config.json {
  alias /etc/grommunio-admin-common/config.json;
}
```

The main user of the uwsgi server is the Administrator Web interface (AWEB), so do enable/start the service now.

```
systemctl enable --now grommunio-admin-api
```

4.16.1 Permissions

The pre-built packages create the following identities:

- Group gromox, which is used for objects in the information store (/var/lib/gromox)
- Group gromoxcf, which is used for configuration files (/etc/gromox)
- Gromox service user: gromox of group gromox, with supplementary group gromoxcf
- AAPI service user: grommunio of group grommunio, with supplementary groups gromox and gromoxcf

The intention is that Gromox and AAPI services can interact with the information store and configuration files.

The directory /var/lib/gromox and all contents shall be owned by user gromox or grommunio. The group owner shall be gromox with read-write permission. Others should not have any access whatsoever.

```
drwxrwx--- 5 gromox gromox 62 Feb 13 23:15 /var/lib/gromox
```

The directory /etc/gromox and all contents are supposed to be owned by either user root or grommunio, be owned by group gromoxcf read-only, and be otherwise inaccessible. Gromox has no need to update config files at all, just read them. AAPI needs to write there (which it can via the grommunio user ownership). Any other users ought not to be able to access this directory as it contains credentials for MySQL and LDAP.

```
drwxr-x--- 2 grommunio gromoxcf 125 Feb 20 21:47 /etc/gromox
```

4.17 nginx support package for AAPI/AWEB

The installation of grommunio-admin-api or grommunio-admin-web also pulls in grommunio-admin-common, which places a number of nginx fragments into the filesystem similar to the earlier grommunio-common.

The package adds nginx configuration fragments to make it listen on port 8080 unencrypted. You can edit /etc/nginx/conf.d/grommunio-admin.conf and disable the inclusion of /usr/share/grommunio-admin-common/nginx.conf and/or enable encrypted access by uncommenting /usr/share/grommunio-admin-common/nginx-ssl.conf. The latter will make nginx listen on port 8443.

Create /etc/grommunio-admin-common/nginx-ssl.conf as a file, or as a symlink to /etc/grommunio-common/nginx/ssl_certificate.conf to the existing TLS directives.

```
ln -s /etc/grommunio-common/nginx/ssl_certificate.conf /etc/grommunio-admin-common/nginx-ssl.conf
```

Reload/restart nginx as needed. Adjust your packet filter configuration for the new ports as needed.

The fragment files installed a route for the /api/v1 URI space to be forwarded to the uwsgi process. It is now possible to make requests to the AAPI endpoints, and we can test for that with curl or even firefox.

```
curl -kv https://localhost:8443/api/v1/login
# firefox https://mail.example.net:8443/api/v1/login
```

The expected result is a JSON response.

```
…
< HTTP/1.1 405 METHOD NOT ALLOWED
…
{"message":"Method 'GET' not allowed on this endpoint"}
```

An authenticated request can also be made:

```
curl -kv https://localhost:8443/api/v1/login -d 'user=admin&pass=freddledgruntbuggly'
```

Expected output:

```
{"grommunioAuthJwt":"eyJ0..."}
```

4.18 Administration Web Interface (AWEB)

AWEB is a package containing a HTML/JavaScript frontend and which will make use of AAPI's endpoints via REST.

```
zypper in grommunio-admin-web
```

Since this package contains just static files, the login page is now ready. Visit https://mail.example.net:8443/ and log in with the credentials you have previously assigned (username: admin, password: as you did).

The details on how to use AWEB (sometimes also referred to as AUI) are provided on the Grommunio documentation website.

4.18.1 Known issues

The systemd service list in the dashboard (subsection "Performance", box container in the left third) has action buttons to trigger systemctl enable/disable/start/stop/restart. Despite the placement of the file /usr/share/polkit-1/rules.d/pkit-10-gromox.rules, AAPI is unable to issue systemctl commands, and a red error box with text Interactive authentication required will appear.

4.18.2 Create domain & user

Create the example.net domain, and a user using AWEB. Afterwards, one can test the login/use in various ways. For example, to run the Autodiscover procedure from the command-line:

```
PASS=abcdef /usr/libexec/gromox/autodiscover -e boop@example.net
```

Expected output:

```
<?xml version="1.0" encoding="utf-8"?>
<Autodiscover xmlns="http://schemas.microsoft.com/exchange/autodiscover/responseschema/2006">
<Response xmlns=...
```

You can connect with Outlook if necessary.

To be able to log into IMAP/POP3, the user must have this feature explicitly enabled. This can be changed using AWEB by going to the *Domains* > *example.net* > *Users* tab on the left-hand side navigation pane. Once enabled,

```
curl -kv imaps://localhost/ -u boop@example.net:abcdef
```

Expected output:

```
...
> A001 CAPABILITY
< * CAPABILITY IMAP4rev1 XLIST SPECIAL-USE UNSELECT UIDPLUS IDLE AUTH=LOGIN␣
↪STARTTLS
```

(continues on next page)

(continued from previous page)

```
< A001 OK CAPABILITY completed
> A002 AUTHENTICATE LOGIN
< + VXNlciBOYW1lAA==
> Ym9ua0Byb3V0ZTM4LnRlc3Q=
< + UGFzc3dvcmQA
> YWJjZGVm
< A002 OK logged in
> A003 LIST "" *
< * LIST (\HasNoChildren) "/" {5}
* LIST (\HasNoChildren) "/" {5}
< INBOX
...
```

4.19 grommunio-web

Install grommunio-web. Verify that you can load the login page and login:

```
curl -kv https://localhost:443/web/
# firefox https://mail.example.net/web/
```

4.20 Loopback mail

The *gromox-delivery-queue* and *gromox-delivery* services comprise the Local Delivery Agent. This LDA supports a bit of SMTP to facilitate it being used in a filter-free loopback scenario. That is, one can send mail from example.net to example.net (only), with no SMTP to the outside.

(A mail composed and submitted with grommunio-web will ultimately be emitted by the *gromox-zcore* process, which sends it to *localhost:25*. Alternatively, when using Outlook, the *gromox-http* process emits the mail to *localhost:25*. And on port 25, one can either run the LDA, or indeed a full MTA like Postfix.)

On some systems which exuberantly start services (hi Debian), you may need to disable an existing MTA first before being able to perform this test. (Alternatively, you can skip right to the "Postfix" section below.)

```
systemctl stop postfix
systemctl enable --now gromox-delivery gromox-delivery-queue
```

4.21 Postfix

Because *gromox-delivery-queue* listens on port 25 by default, it needs to be moved out the way when putting a full MTA in its place. Edit /etc/gromox/smtp.cfg and declare:

```
listen_port = 24
```

Within the Postfix configuration, we will be making use of the *mysql* lookup plugin, so do install that alongside Postfix itself:

```
zypper in postfix postfix-mysql
```

Set up a few Postfix directives:

```
postconf -e virtual_mailbox_domains=mysql:/etc/postfix/mbxdom.cf
postconf -e virtual_mailbox_maps=mysql:/etc/postfix/mbxmaps.cf
postconf -e virtual_alias_maps=mysql:/etc/postfix/alias.cf,mysql:/etc/postfix/mbxmaps.cf
postconf -e virtual_transport="smtp:[localhost]:24"
```

Filenames for these additional configuration fragments, mbxdom.cf, mbxmaps.cf and alias.cf, can be freely chosen. Add the MariaDB parameters to the alias resolution fragment that (here) goes into /etc/postfix/alias.cf:

```
user = grommunio
password = freddledgruntbuggly
hosts = localhost
dbname = grommunio
query = SELECT mainname FROM aliases WHERE aliasname='%s'
```

Then, add the MariaDB connection parameters to the domain resolution fragment that (here) goes into /etc/postfix/mbxdom.cf:

```
user = grommunio
password = freddledgruntbuggly
hosts = localhost
dbname = grommunio
query = SELECT 1 FROM domains WHERE domain_status=0 AND domainname='%s'
```

Furthermore, add the MariaDB parameters to the mailbox resolution fragment, here in /etc/postfix/mbxmaps.cf:

```
user = grommunio
password = freddledgruntbuggly
hosts = localhost
dbname = grommunio
query = SELECT username FROM users WHERE username='%s'
```

If you plan to use reject_authenticated_sender_login_mismatch for SMTP submission, e.g. in the Postfix directive smtpd_sender_restrictions or smtpd_recipient_restrictions to ensure that the SASL login name is an owner for the e-mail sender address, set another Postfix directive:

```
postconf -e smtpd_sender_login_maps=mysql:/etc/postfix/mbxmaps.cf,mysql:/etc/postfix/alias.cf
```

Finally, enable/restart the services so they can take their new places:

```
systemctl enable --now gromox-delivery gromox-delivery-queue postfix
systemctl restart gromox-delivery-queue postfix
```

4.22 Other services

This chapter only covers the core component. Optional components, such as Chat, Meet, Files, Office and/or Archive, will be published in their own chapter at a later date.

CHAPTER 5

Container Installation

While for the majority of installations the grommunio Appliance delivers a comprehensive solution for most installation targets, some special needs might not be possible to satisfy. For these cases, the grommunio base system and core (groupware) can be installed on container management systems such as Kubernetes and Docker Compose with guidance from this chapter.

This chapter assumes a basic system is running already. **Basic** in this regard means:

- should have an interactive shell for you to use
- should not be ephemeral and not lose its state when turned off

We will build a container for the Grommunio Core suite.

- Automatic configuration of various services
- Grommunio Core (gromox-http, gromox-antispam, gromox-event, gromox-midb, gromox-postfix,gromox-timer, gromox-zcore, gromox-imap, gromox-pop3, gromox-delivery, gromox-delivery-queue, gromox-admin, nginx, redis and php-fpm)
- Configurable via config files and environment variables.

Note: Future versions will configure all variables via the environment

- This Container uses a openSUSE Linux base and includes s6 overlay enabled for PID 1 init capabilities.

Note: This is a complex piece of software that tries to get you up and running with sane defaults, you will need to switch eventually over to manually configuring the configuration file when depending on your use case.

Important: Do not use our defaults in production environments! Please adapt them according to your usage requirements.

5.1 Prerequisites and Assumptions

This image assumes that you have an external MySQL/MariaDB container. A useful prerequisite is to read the grommunio documentation.

5.2 Installation

Automated builds of the image are available on Docker Hub and is the recommended method of installation.

5.2.1 Quick Start

- The quickest way to get started is using docker-compose or kubernetes.

  See the examples folder for a working docker-compose example and kubernetes example that can be modified (and **should be**) for development or production use.
- Set various *environment variables* to understand the capabilities of this image.
- Map *persistent storage* for access to configuration and data files for backup.

5.3 Configuration

5.3.1 Persistent Storage

The following directories are used for configuration and can be mapped for persistent storage.

5.3.2 Environment Variables

Below is the complete list of available options that can be used to customize your installation.

They will be added/updated as soon as components implement environment variable based deployments natively.

5.3.2.1 General Options

5.3.2.2 Database Options

5.4 Shell Access

For debugging and maintenance purposes you may want access the containers shell.

CHAPTER 6

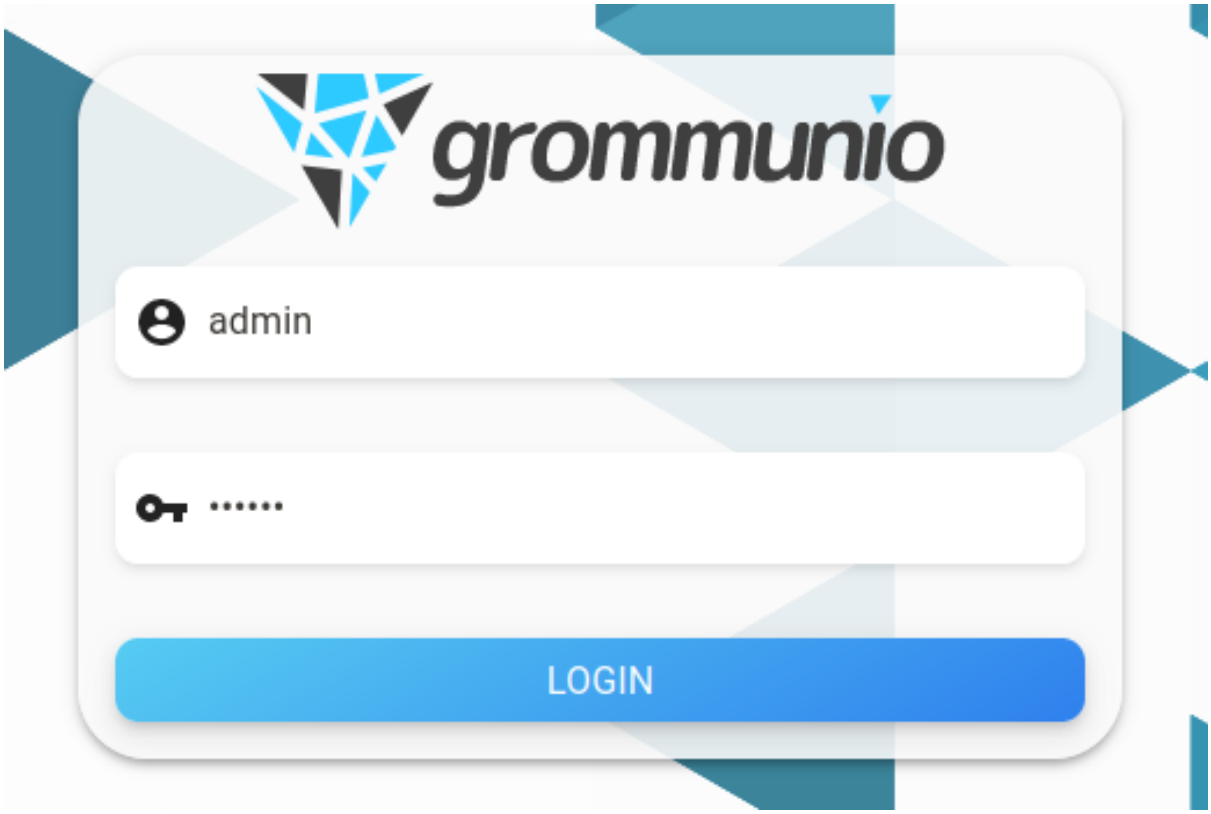
Administration

This chapter includes details on how to administrate components of grommunio with the available toolset.

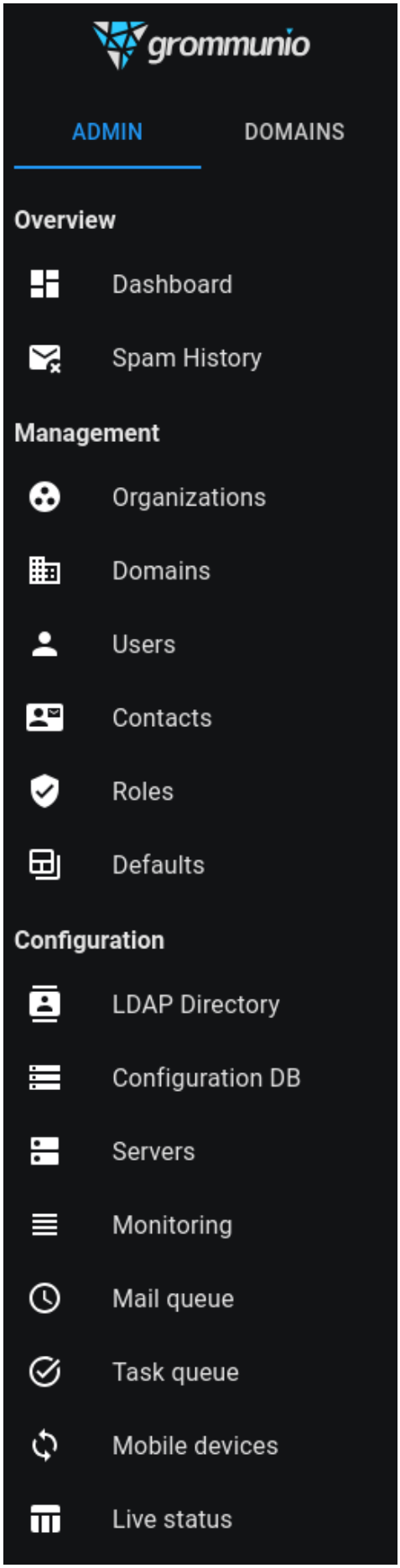
6.1 grommunio admin UI (AUI)

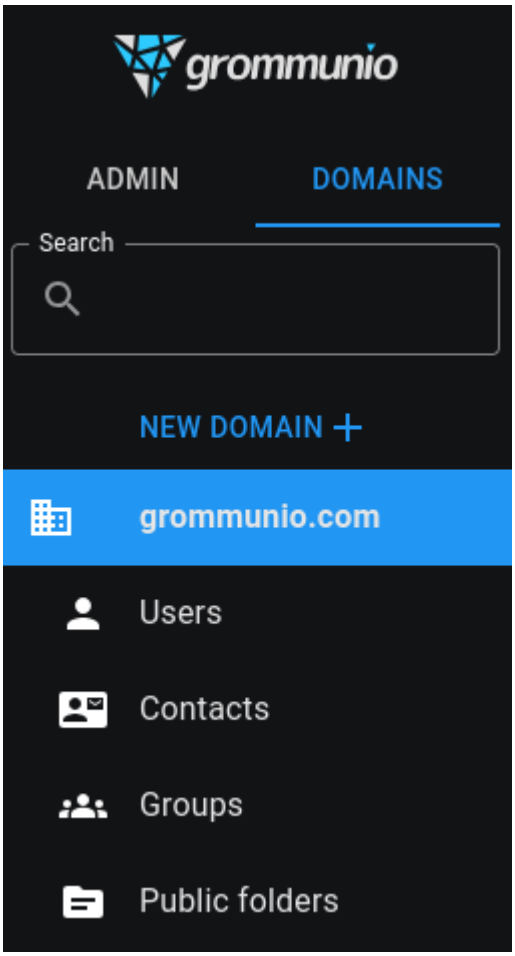
After successfully installing the grommunio Appliance, you can access the UI through your browser on port 8080 (8443 with https soon).

Since you most likely set a password for admin UI while installing the Appliance, you can immediately use these credentials to login.



To navigate through the UI, simply use the drawer on the left side of the page.


grommunio
ADMIN
DOMAINS
Overview
Dashboard
Spam History
Management
Organizations
Domains
Users
Contacts
Roles
Defaults
Configuration
LDAP Directory
Configuration DB
Servers
Monitoring
Mail queue
Task queue
Mobile devices
Live status

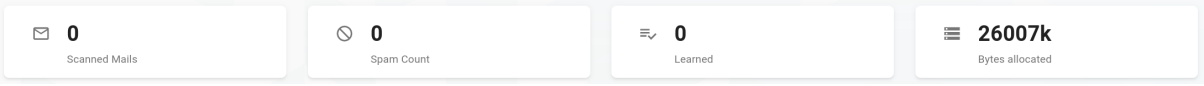

grommunio
ADMIN
DOMAINS
Search
NEW DOMAIN +
grommunio.com
Users
Contacts
Groups
Public folders

6.1.1 Dashboard

After a successful login, you can see the dashboard with live data of the machine grommunio runs on.

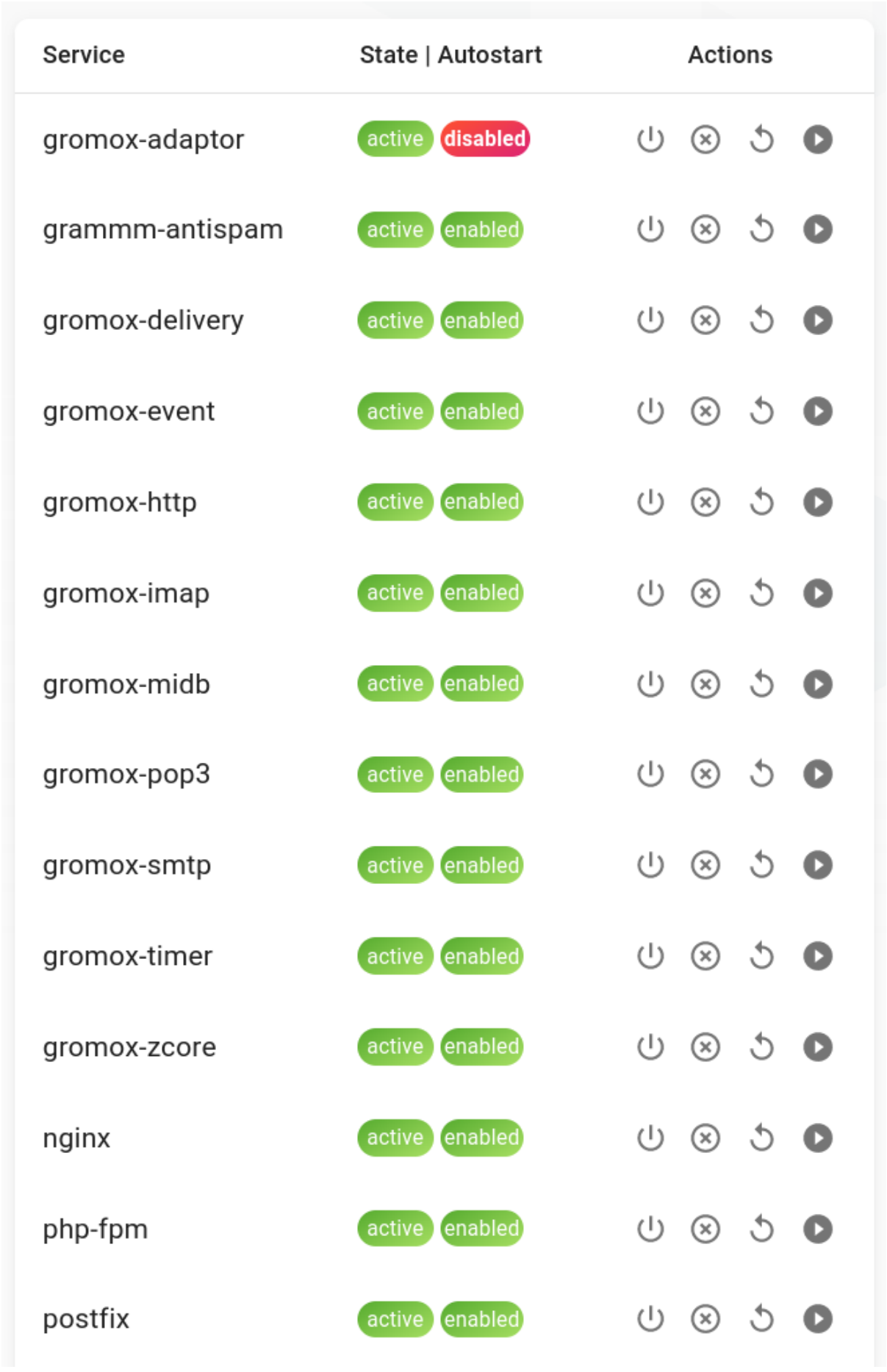
6.1.1.1 Antispam

Since grommunio has its own antispam service, according data can be displayed in the Dashboard.



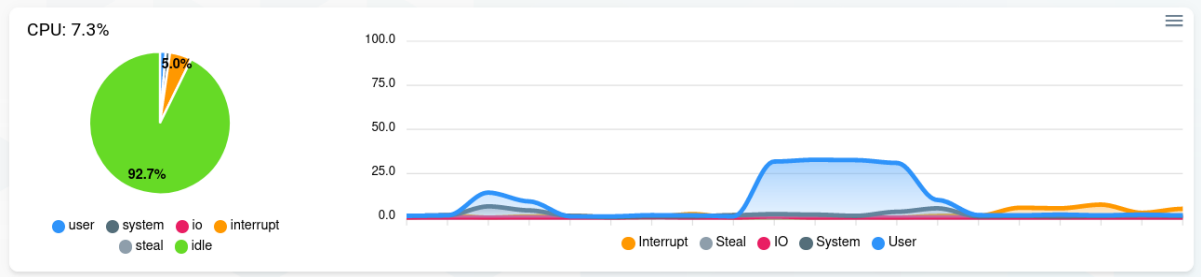
6.1.1.2 Services

Antispam isn't the only grommunio service, in fact there are lots more. The current state of these services can be seen on the left side of the dashboard.


Service
State | Autostart
Actions
gromox-adaptor
active
disabled
grammm-antispam
active
enabled
gromox-delivery
active
enabled
gromox-event
active
enabled
gromox-http
active
enabled
gromox-imap
active
enabled
gromox-midb
active
enabled
gromox-pop3
active
enabled
gromox-smtp
active
enabled
gromox-timer
active
enabled
gromox-zcore
active
enabled
nginx
active
enabled
php-fpm
active
enabled
postfix
active
enabled

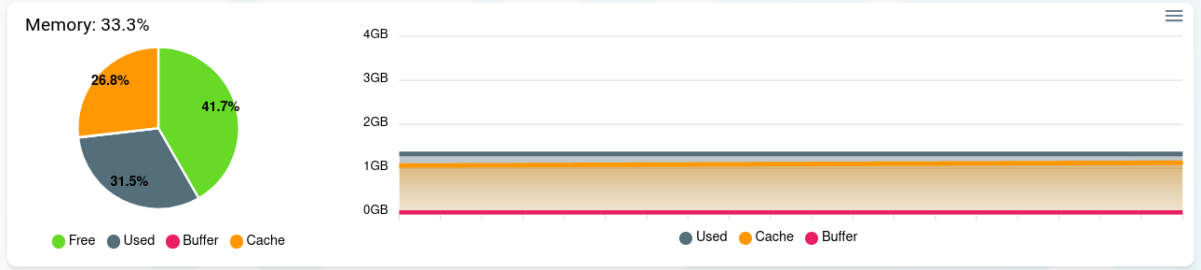
You can stop, restart or start these services from here by clicking the action buttons of a service in the list.

6.1.1.3 CPU



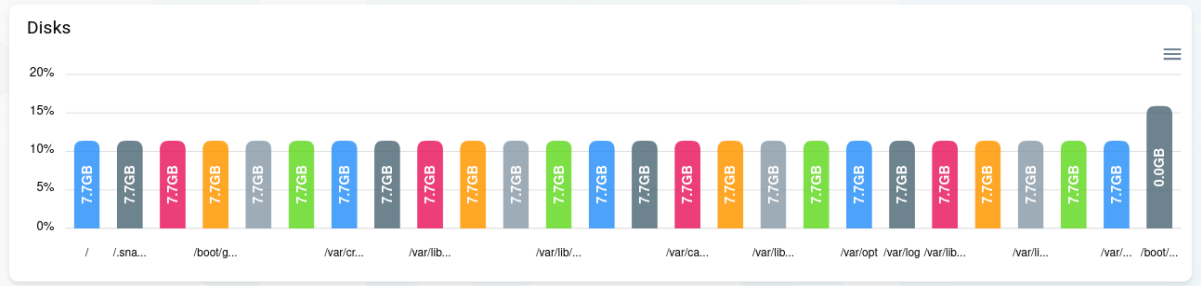
A live and history display of the CPU usage.

6.1.1.4 Memory



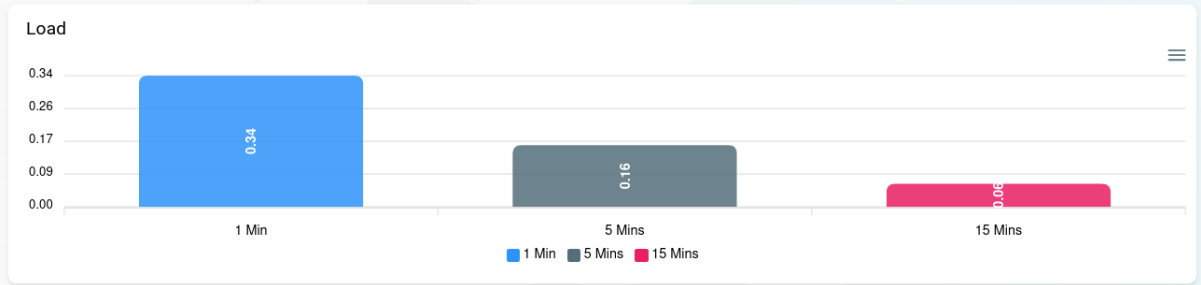
A live and history display of the memory usage.

6.1.1.5 Disks and swap



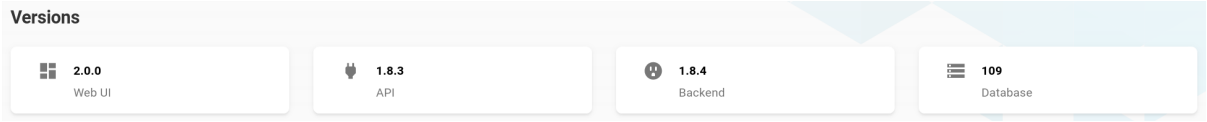
A live display of the disks and swap.

6.1.1.6 Load



A display of the system load over the last 1, 5 and 15 minutes.

6.1.1.7 Versions



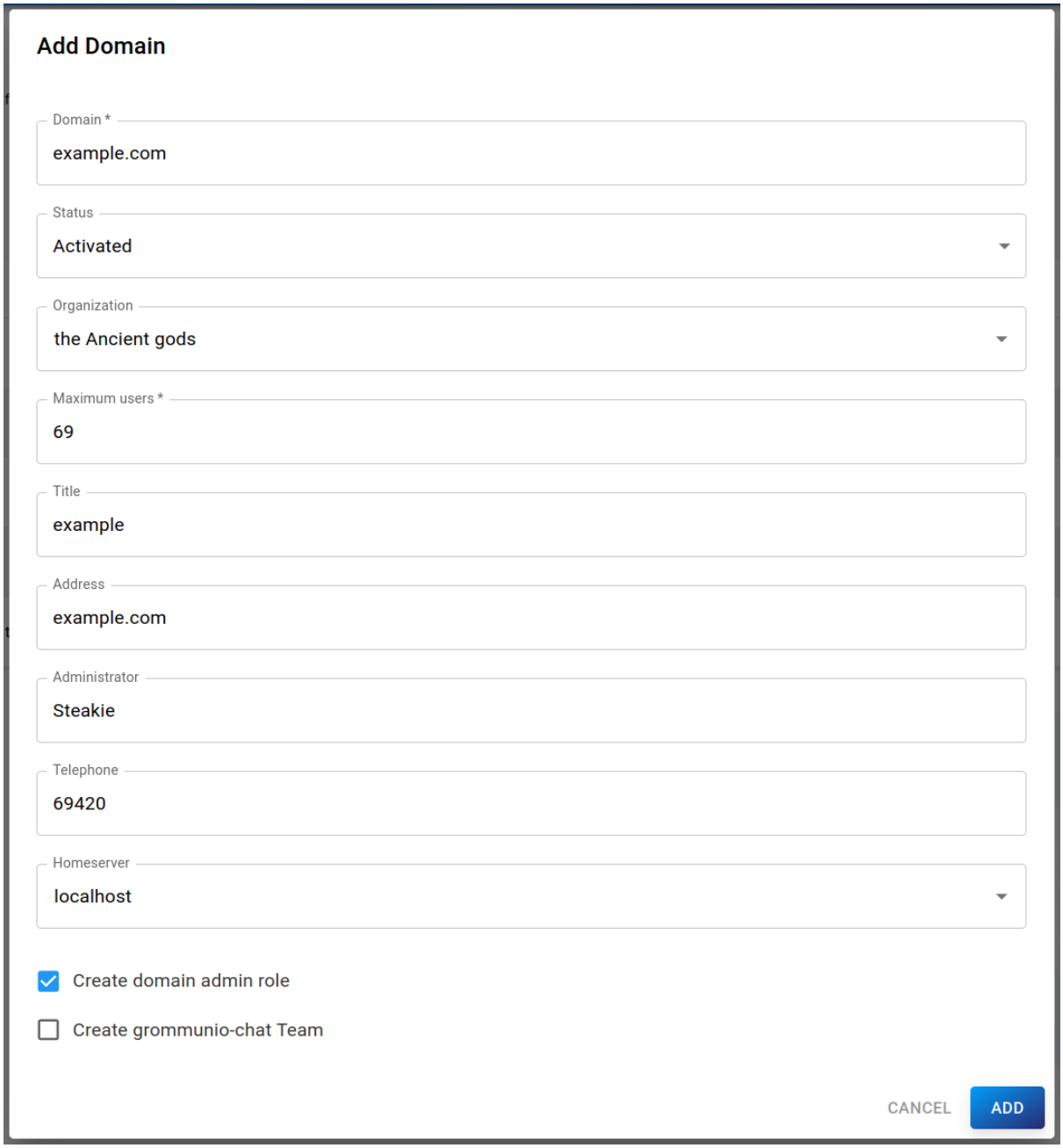
A display of installed component versions.

6.1.2 Domains

Click on *Domains* in the drawer, which will redirect you to the list view of existing domains. If you just set up grommunio, the table will be empty. If you want to show currently deactivated domains check the checkbox *show deactivated*.

6.1.2.1 Adding a domain

To add a new domain, click the blue *NEW DOMAIN* button to open the form dialog:



The following properties can be set:

- Domain (required): The name of the domain (cannot be changed afterwards)
- Status: Whether the domain should be currently activated or deactivated
- Organization: Organization of the domain
- Maximum users (required): The maximum amount of users (e-mails) of this domain
- Title: Title of the domain
- Address: Address of the domain
- Administrator: Administrator of the domain
- Telephone: Hotline for problems
- Homeserver: The server on which the domain's data is stored
- Create domain admin role: Creates a role for users, who will be admins for this domain

- Create grommunio-chat team: Creates a new grommunio-chat team for this domain. If you want users of this domain to be able to log into grommunio-chat, this has to be checked.

Click *Add* to confirm or *Cancel* to cancel.

6.1.2.2 Editing a domain

To edit an existing domain, click on a domain in the list to open the detailed view of a domain.

Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

To change the current password of the domain, click *Change password* next to the domain name. You will be prompted to set and repeat your new password.

6.1.2.3 Deleting a domain

To delete a domain, click on the trash icon of a domain in the domain list view.

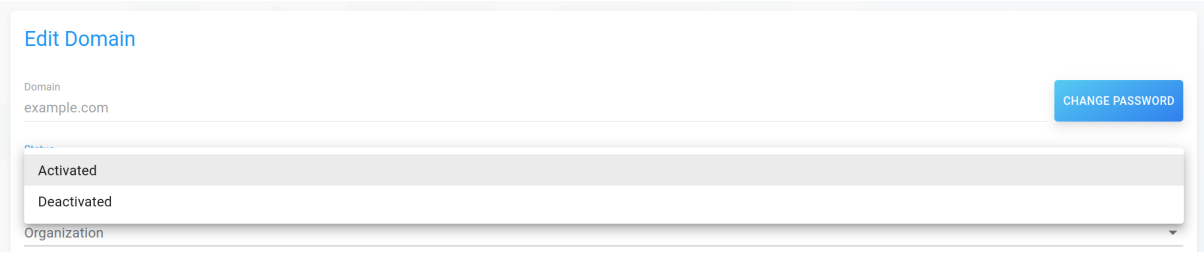
The following flags can be set:

- Delete permanently: Checking this, will completely remove the domain out of the database, not just deactivate it
- Delete files: Only available if permanently deleting, will delete all files of this domain

Click *Confirm* to confirm or *Cancel* to cancel

6.1.2.4 Reactivating domains

If you didn't delete a domain permanently, it will automatically be set to deactivated. To reactivate a domain, click on a domain in the list to get to the detailed view. Now change the status from deactivated to activated.



6.1.3 Users

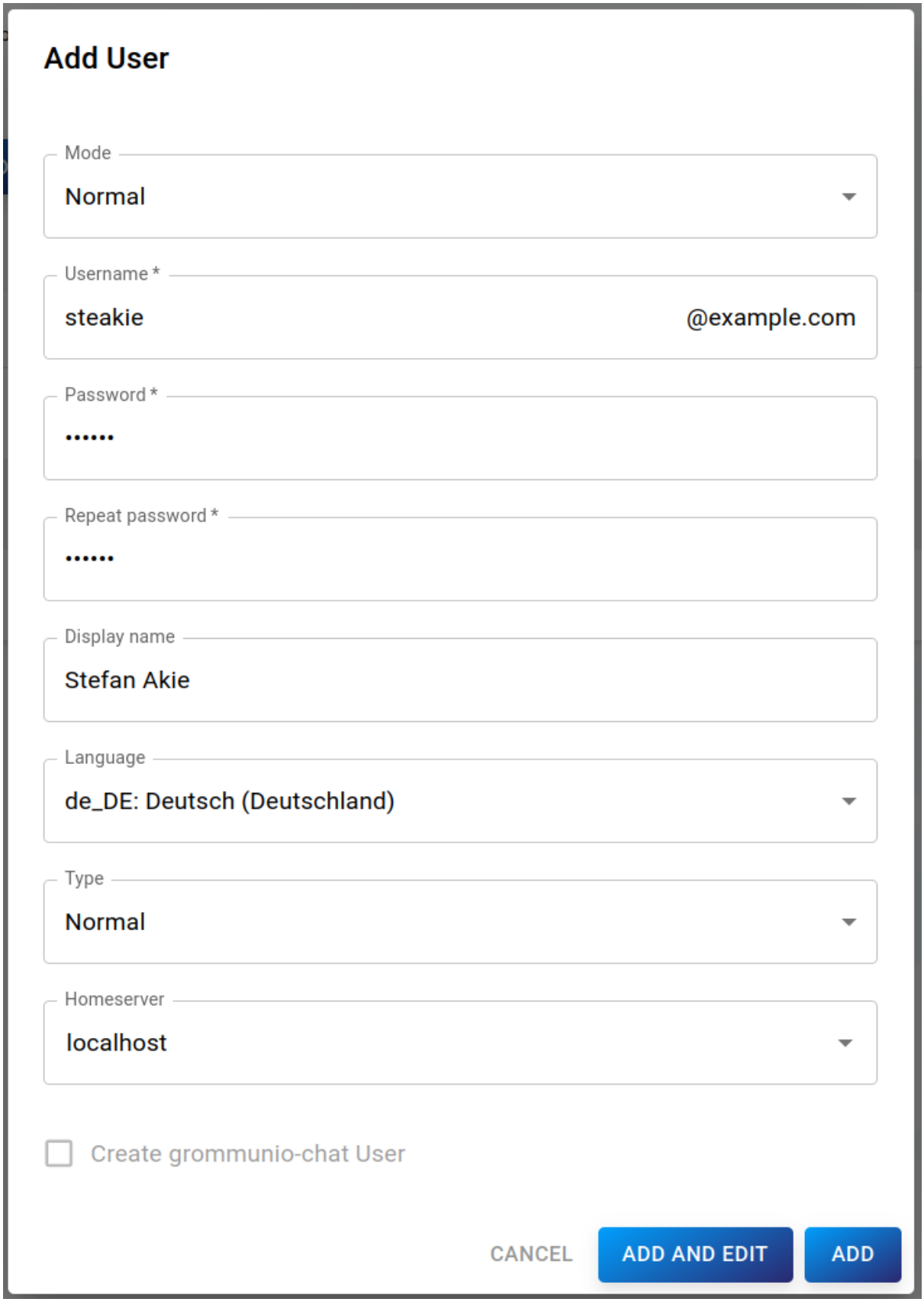
If at least one domain exists in the database, users can be added to a domain. To show existing users of a domain, navigate to the domain view in the drawer (*Domains* tab).

Click on a domain to expand available sub-pages and click on *Users*, which will redirect you to the list of users of this domain. If you just installed grommunio or added the domain, the list will be empty.

Alternatively, to see all users across all domains, click on *Global users* in the drawer.

6.1.3.1 Adding a user

To add a new user, click the blue *NEW USER* button to open the form dialog:



The following properties can be set:

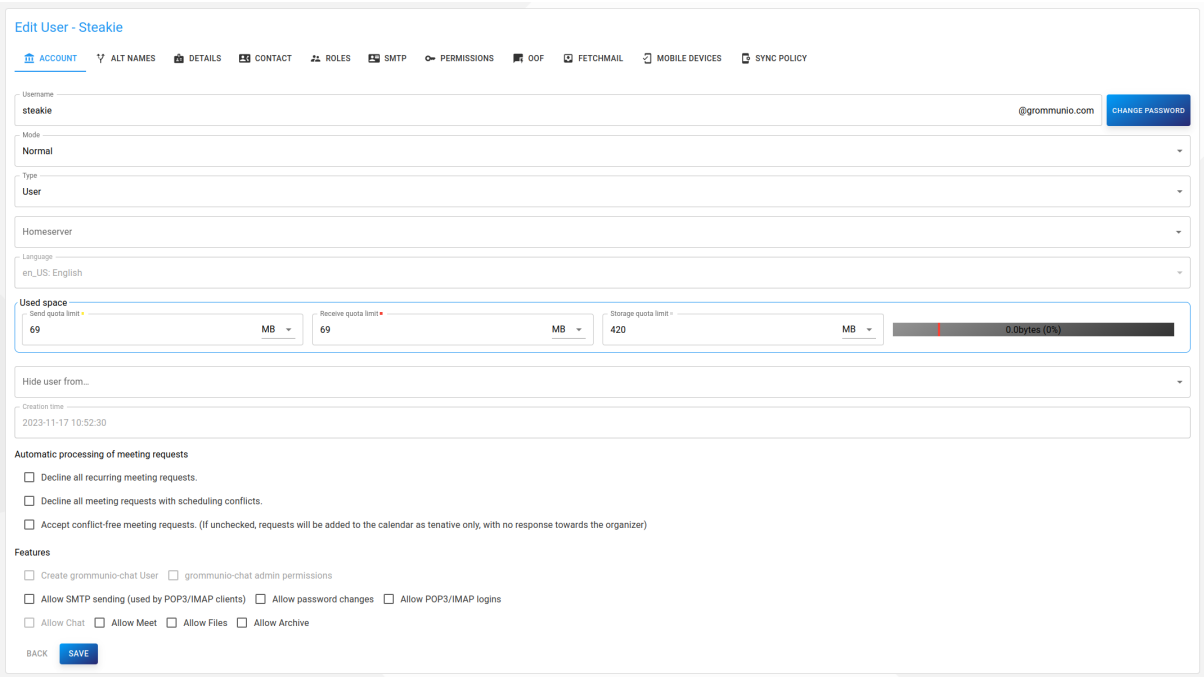
- Mode: Normal or shared user
- Username (required): Username of the user

- Password (required): Password of the user
- Display name: Name to be displayed for this user
- Storage quota limit: Storage limit of the user
- Type: Type of user
- Homeserver: The server on which the user's data is stored

Click *Add* to confirm or *Cancel* to cancel. If you need to further specify user properties, click *Add and Edit* to open the detailed view of this user.

6.1.3.2 Editing a user

To edit an existing user, click on a user in the list to open the detailed view of a user.



There are 10 main categories of user properties:

- Account: RPC/HTTP (Outlook Anywhere), MAPI/HTTP, IMAP, POP3 etc. configuration
- Alt names: Alternative usernames to log into mail-clients with (does not have to be an e-mail address)
- Details: MAPI props
- Contact: Additional MAPI props
- Roles: Roles of the user
- SMTP: Additional e-mails for this user (aliases) and forwarding rules
- Permission: Select users which have special permissions for this user's mailbox
- OOF: Out of office settings
- Fetchmail: Configuration to fetch mails from other servers via fetchmail
- Mobile devices: List of user's mobile devices (via MDM)
- Sync policy: MDM sync policy (specifically for this user)

6.1.3.2.1 Account

The following properties can be edited:

- Username
- Mode: Mailbox mode, select between a normal user, a suspended user and a shared mailbox
- Type: Type of user
- Homeserver: Server on which the user's data is stored
- Language: Store language of the user (does not effect the language of the UI)
- **Used space**
  - Send quota limit: Maximum size of the mailbox before sending messages is blocked
  - Receive quota limit: Maximum size of the mailbox before message reception is blocked
  - Storage quota limit: Maximum size of the mailbox before storing (any kind of) objects is blocked
- Hide user from: Hide user from specific user lists (e.g. the global address list)
- Automatic processing of meeting requests: Trivial
- Create grommunio-chat user: Creates a grommunio-chat account for this user. If this checkbox is disabled, there is no grommunio-chat team for this domain.
- grommunio-chat admin permission: Gives administrative permissions for grommunio-chat to this user's grommunio-chat account.
- grommunio-chat permissions: Grants grommunio-chat admin permissions
- Allow SMTP sending: Allows the user to send e-mails via SMTP
- Allow password changes: Allows the user to change his/her password
- Allow POP3/IMAP logins: Allows logins via POP3 or IMAP
- Hide from GAL: Hides the user from the Global Address List
- Allow Chat/Meet/Files/Archive: Allows access to respective feature

Note that, because a message needs to exist internally before it can be sent, the storage quota limit is also relevant for sending. Conversely, for reception, the storage quota limit must allow storing messages. (Thus, the storage quota should always be more than receive quota, and more than send quota.)

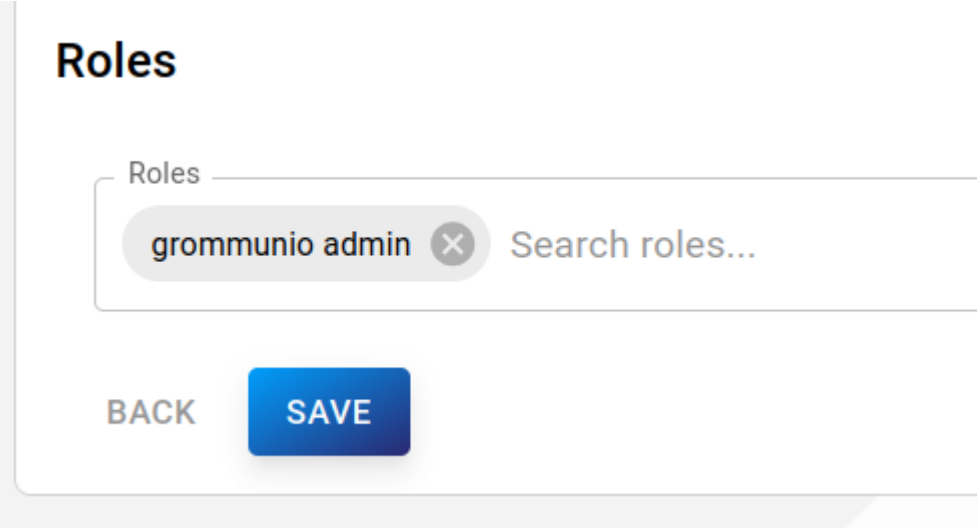
To change the current password of the user, click *Change password* next to the username. You will be prompted to set and repeat the new password.

6.1.3.2.2 User & Contact

Common MAPI props. These are self-explanatory.

6.1.3.2.3 Roles

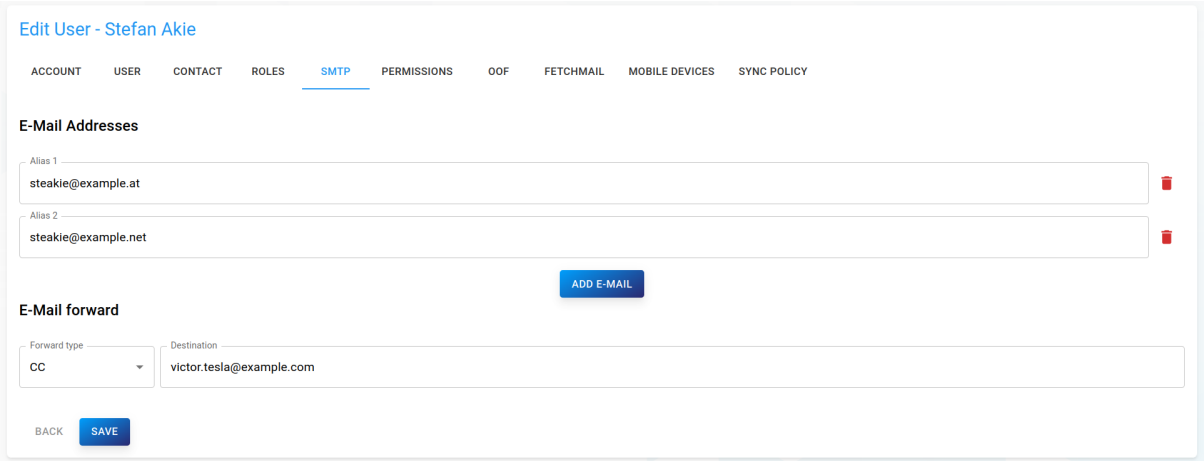
Roles of the user, which can be edited with the autocompleting textfield



6.1.3.2.4 SMTP

User aliases:. Edit the textfield to edit an alias, click *ADD E-MAIL* to add or click the delete icon to delete an alias.

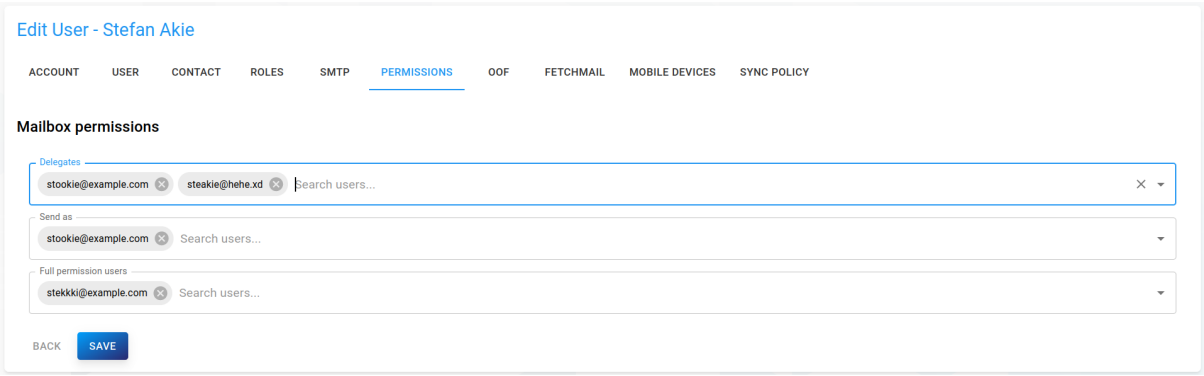
E-Mail forward: Select forward type (CC or Redirect) and the destination e-mail.



6.1.3.2.5 Permissions

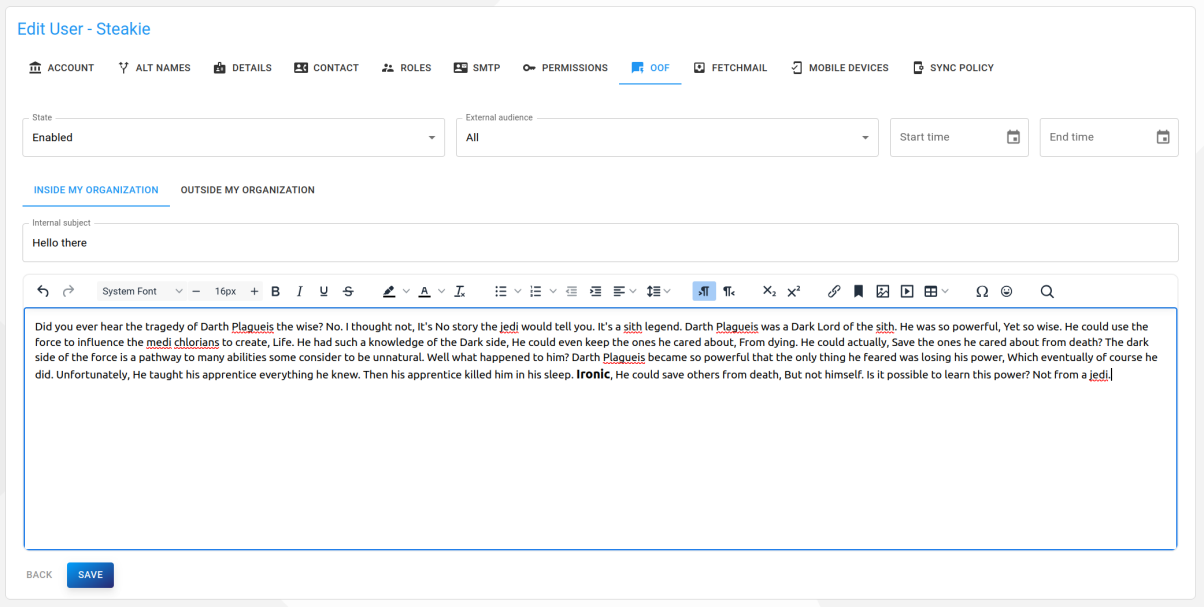
Granting other users access to this mailbox. Available permissions:

- Delegates: Permission to send 'on behalf of' this user
- Send as: Permission to set this user as the sender of an email
- Full permissions: Grants full mailbox permission



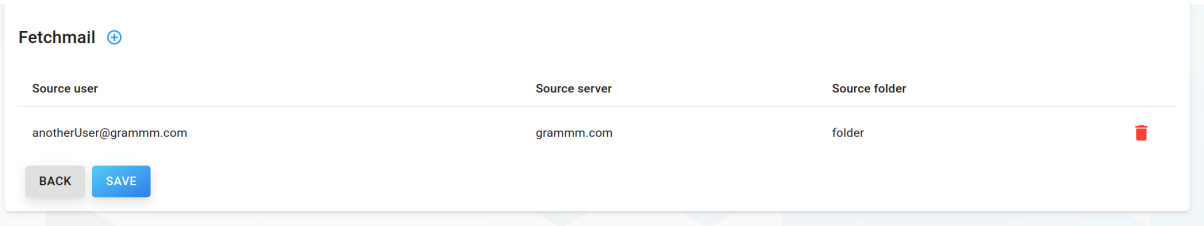
6.1.3.2.6 OOF

Out of office settings (auto-reply messages).

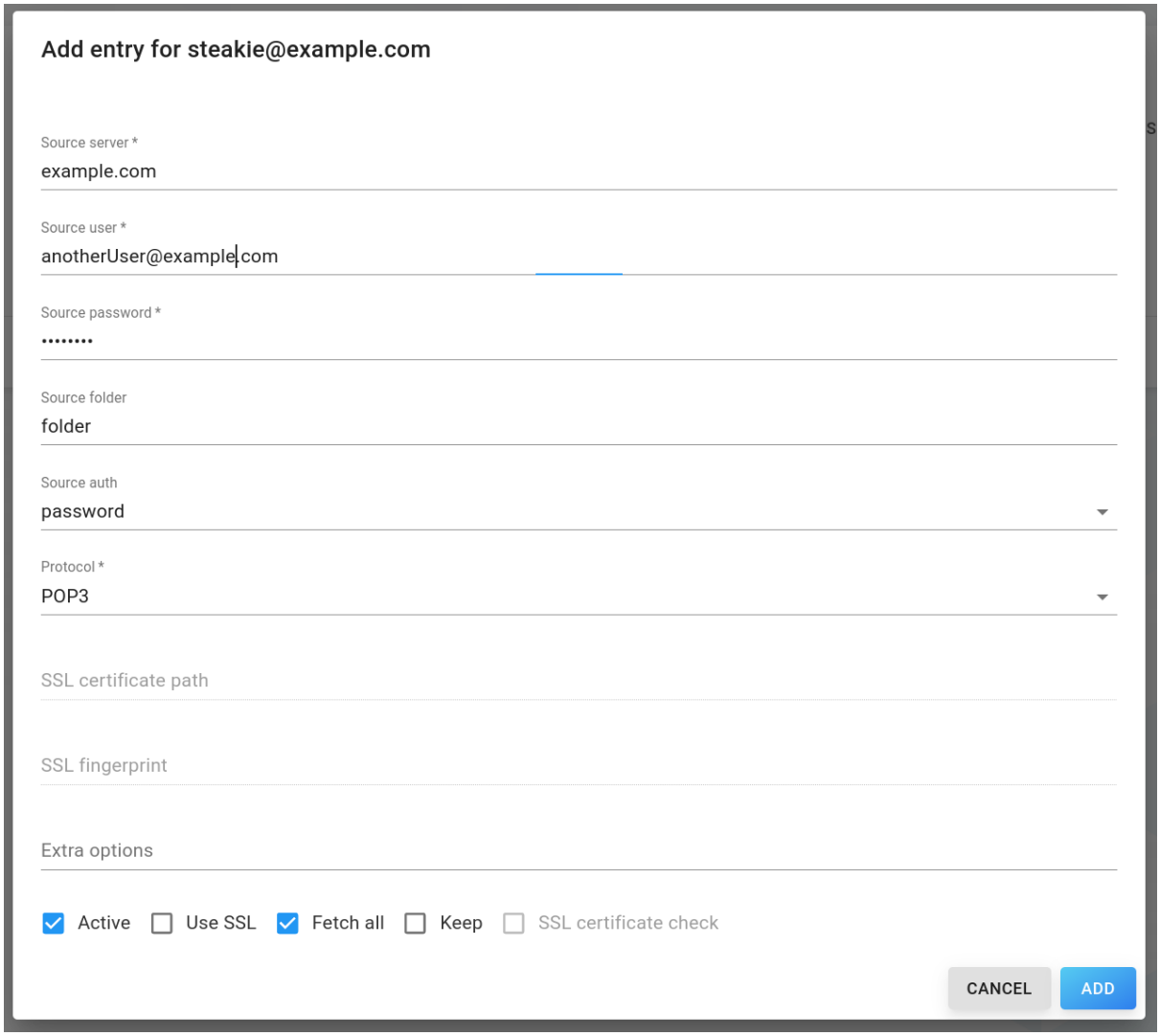


6.1.3.2.7 Fetchmail

It is possible to fetch e-mails from other mailserver via fetchmail. To configure this feature, you can add several e-mail servers and/or users to fetch mails from.



To add new fetchmail entry, click the circled plus icon, which will open the following input form:



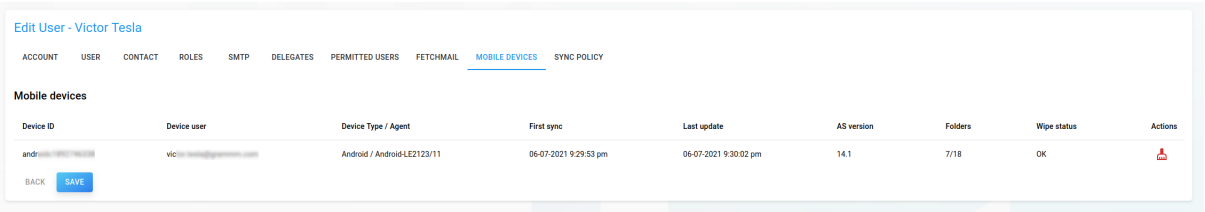
- Source server (required): E-Mail server to fetch from
- Source user (required): E-Mail address to fetch from
- Source password (required): Password to the source users account (Hint: Single or double quotes are not supported)
- Source folder (required): Source folder to sync from
- Source auth: Type of authentication to use
- Protocol (required): Protocol to use
- SSL certificate path (if *Use SSL* is checked): Path to local certificate directory or empty to use local default
- SSL fingerprint (if *Use SSL* is checked): Fingerprint of the server certificate
- Extra options: (if *Use SSL* is checked): Additional fetchmail options
- Active: Whether fetchmail is currently activated
- Use SSL: Whether to use SSL
- Fetch all: Whether to fetch seen mails
- Keep: Keep original e-mails
- SSL certificate check: Check ssl certificate

To edit these properties, click on a row in the table. To delete an entry, click the trash icon of a table row.

Important: Any changes will only be saved after clicking the click *Save* on the bottom of the page.

6.1.3.2.8 Mobile Devices

Synchronized mobile devices of this user

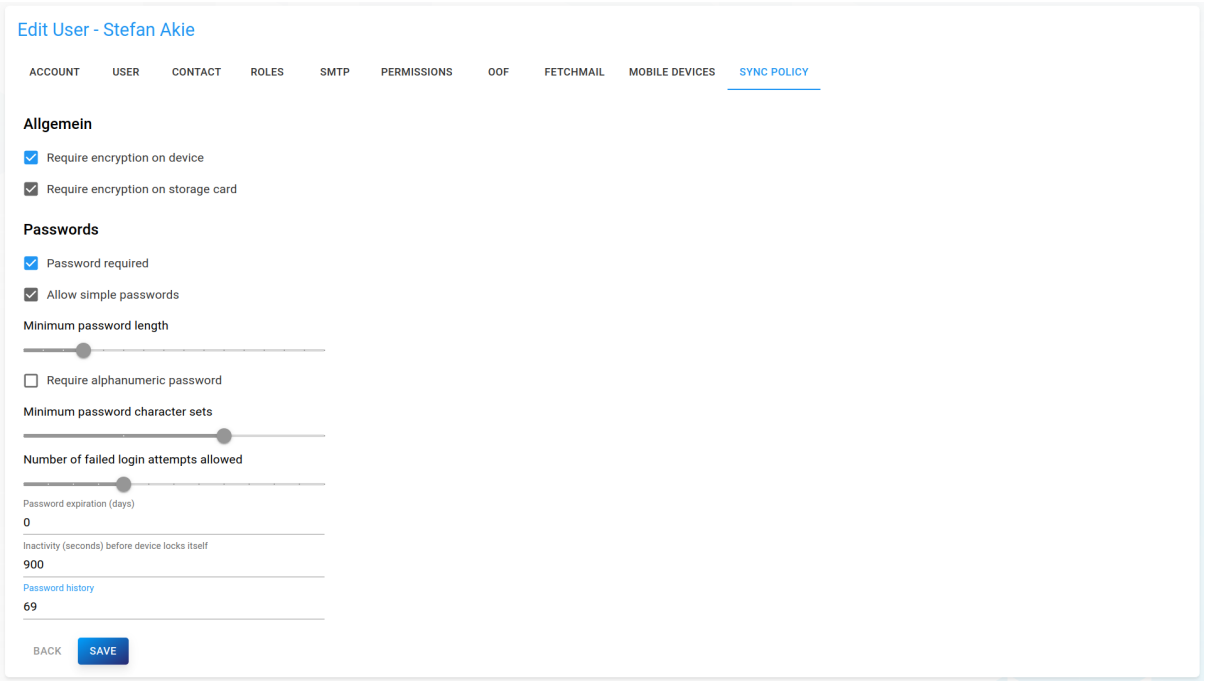


Actions:

- Remote wipe: Engages a remote wipe for a device via MDM (Mobile Device Management)
- Cancel remote wipe: Cancel above action

6.1.3.2.9 Sync policy

Specific MDM rules for this user. Unedited rules (greyed out) are adopted from the domain's policy.



6.1.3.3 Deleting a user

To delete a user, click on the trash icon of a user in the user view.

The following flags can be set:

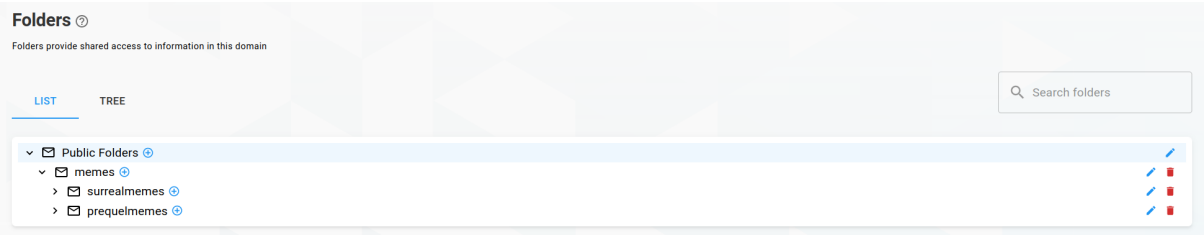
- Delete files: Will delete all files of this user

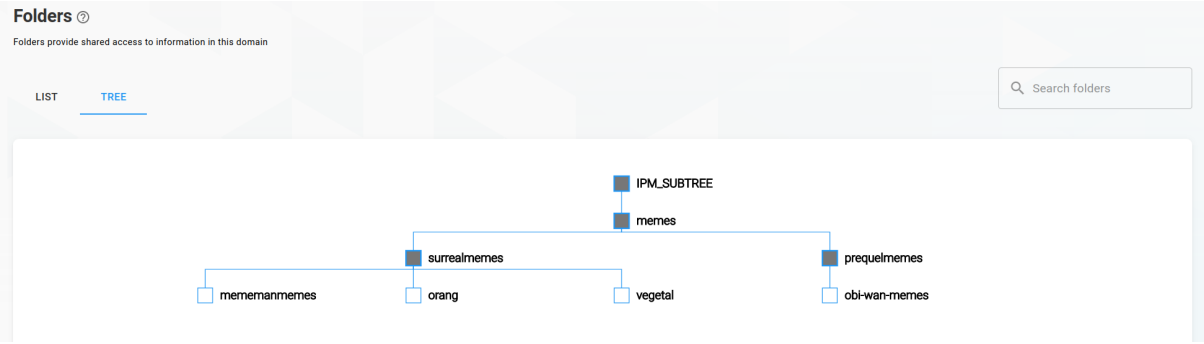
Click *Confirm* to confirm or *Cancel* to cancel.

6.1.4 Public folders

If at least one domain exists in the database, public folders can be added to a domain. To show existing public folders of a domain, navigate to the domain view in the drawer.

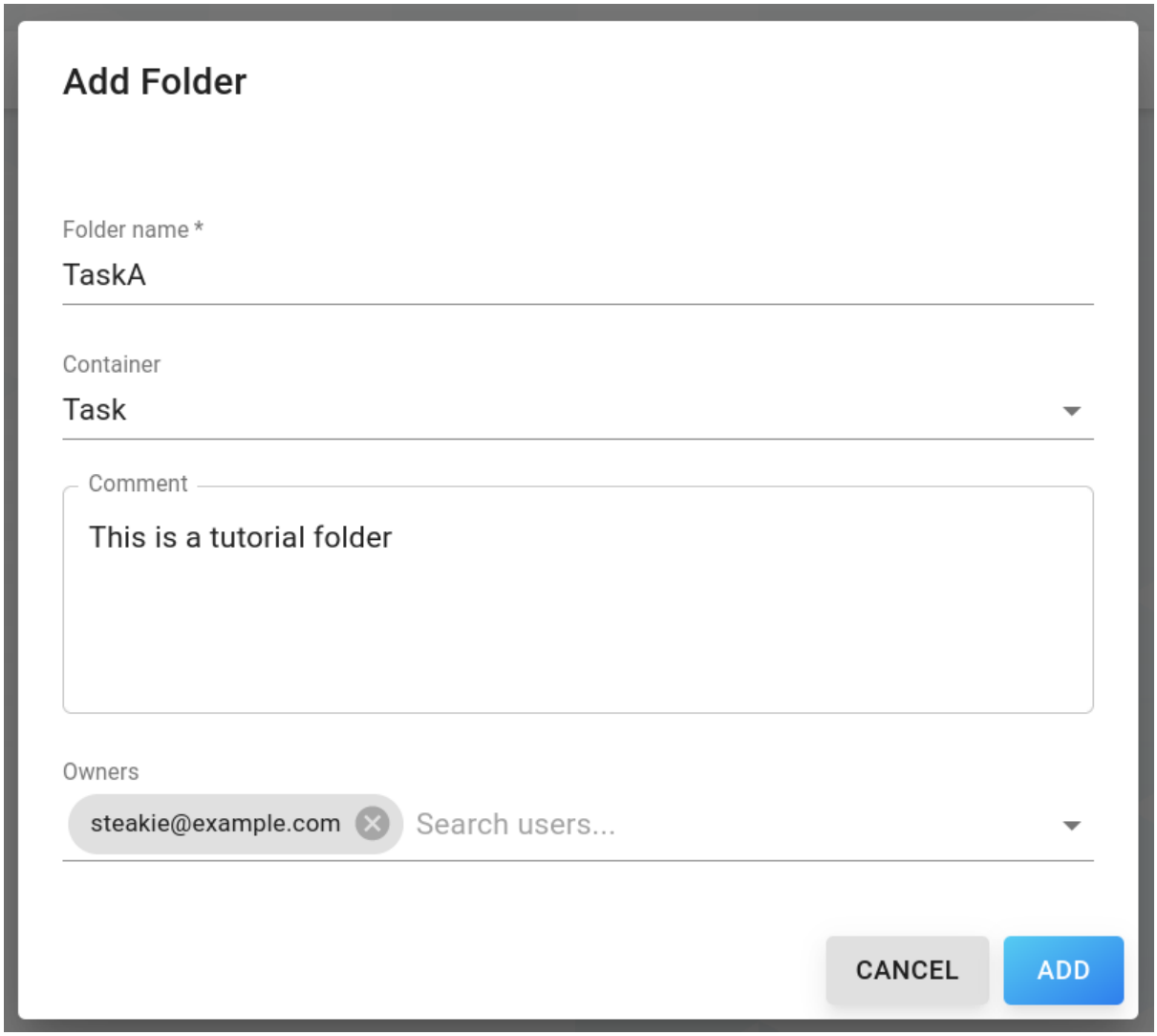
Click on a domain to expand available sub-pages and click on *Public folders*, which will redirect you to the list of folders of this domain. There are two views: A hierarchical view, like a common folder structure and a tree-like graph view.





6.1.4.1 Adding a folder

To add a folder, click the *Plus Circle* icon of the folder's parent folder. *Public Folders* is the root folder, all other folders are put into. Thus the first folder is always within *Public Folders (IPM_SUBTREE)*.



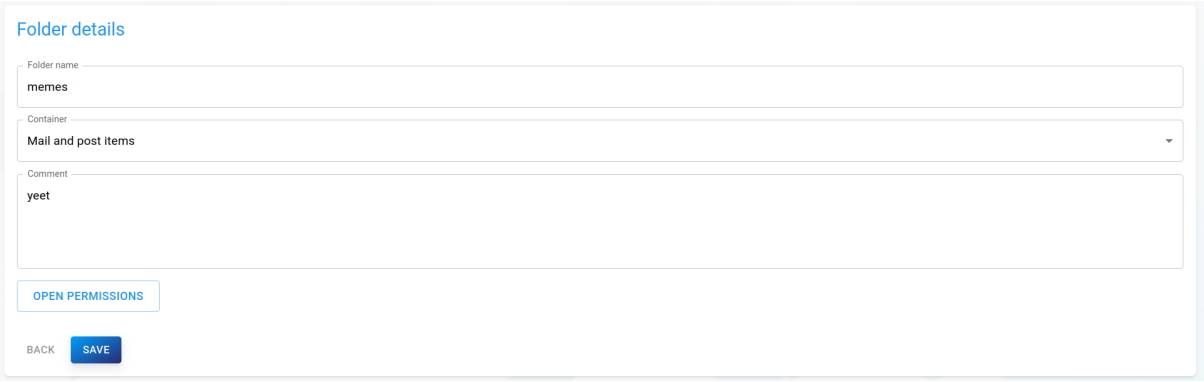
The following properties can be set:

- Folder name (required): Name of folder
- Container: Type of folder container
- Comment: Comment
- Owners: Owners of this folder (Multi-select of users in the database)

Click *Add* to confirm or *Cancel* to cancel.

6.1.4.2 Editing a folder

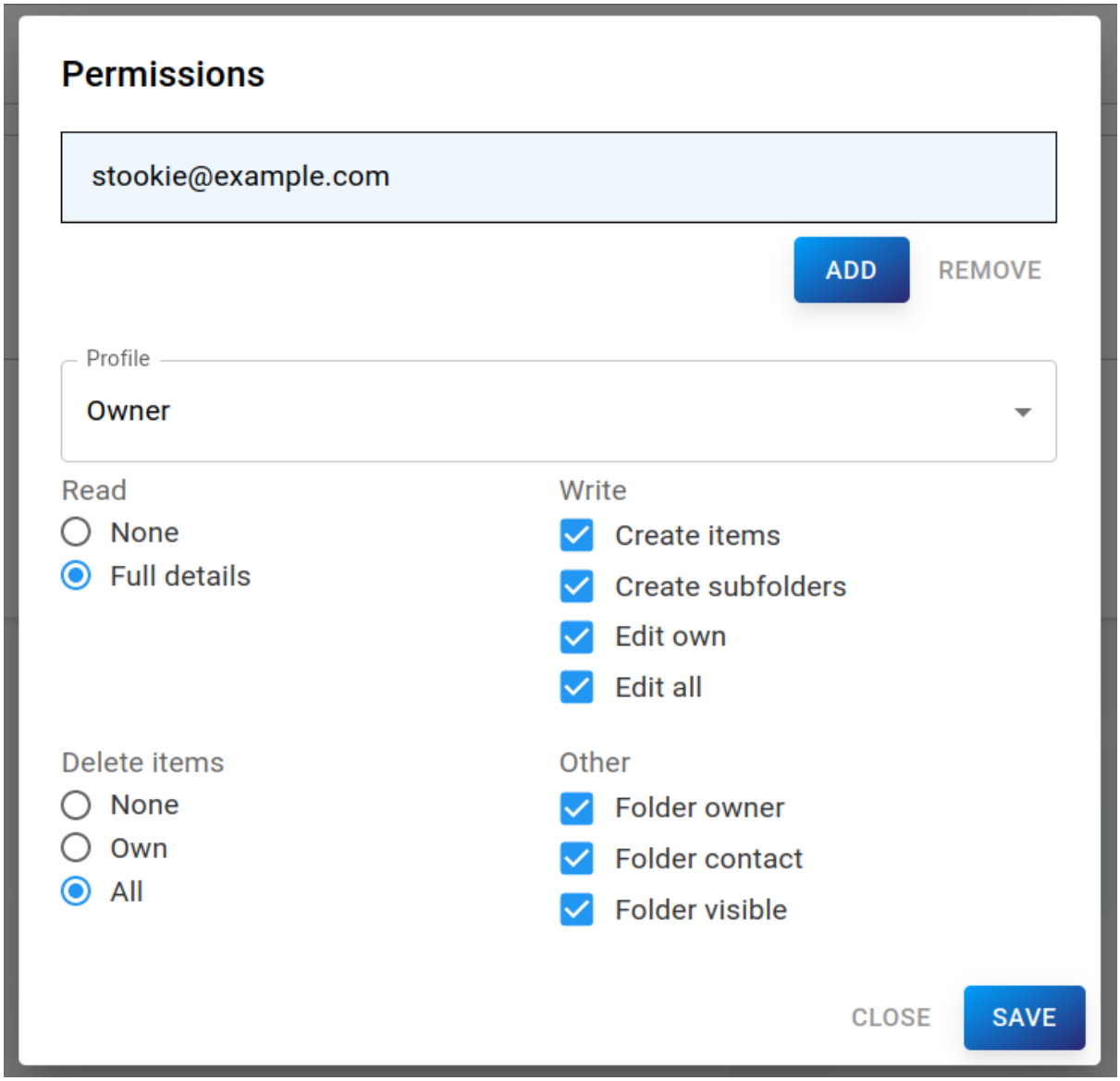
To edit an existing folder, click on the right *Edit* icon inside the hierarchy view to open the folder details.



Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

6.1.4.2.1 Folder permissions

To edit folder permission click on *Open permissions* to open the permissions dialog.



This form is a direct replica of grommunio-web's and outlook's folder permission settings. Select users to grant permissions at the top and set their permissions at the bottom.

6.1.4.3 Deleting a folder

To delete a folder, click on the trash icon of a folder in the folder view. Click *Confirm* to confirm or *Cancel* to cancel.

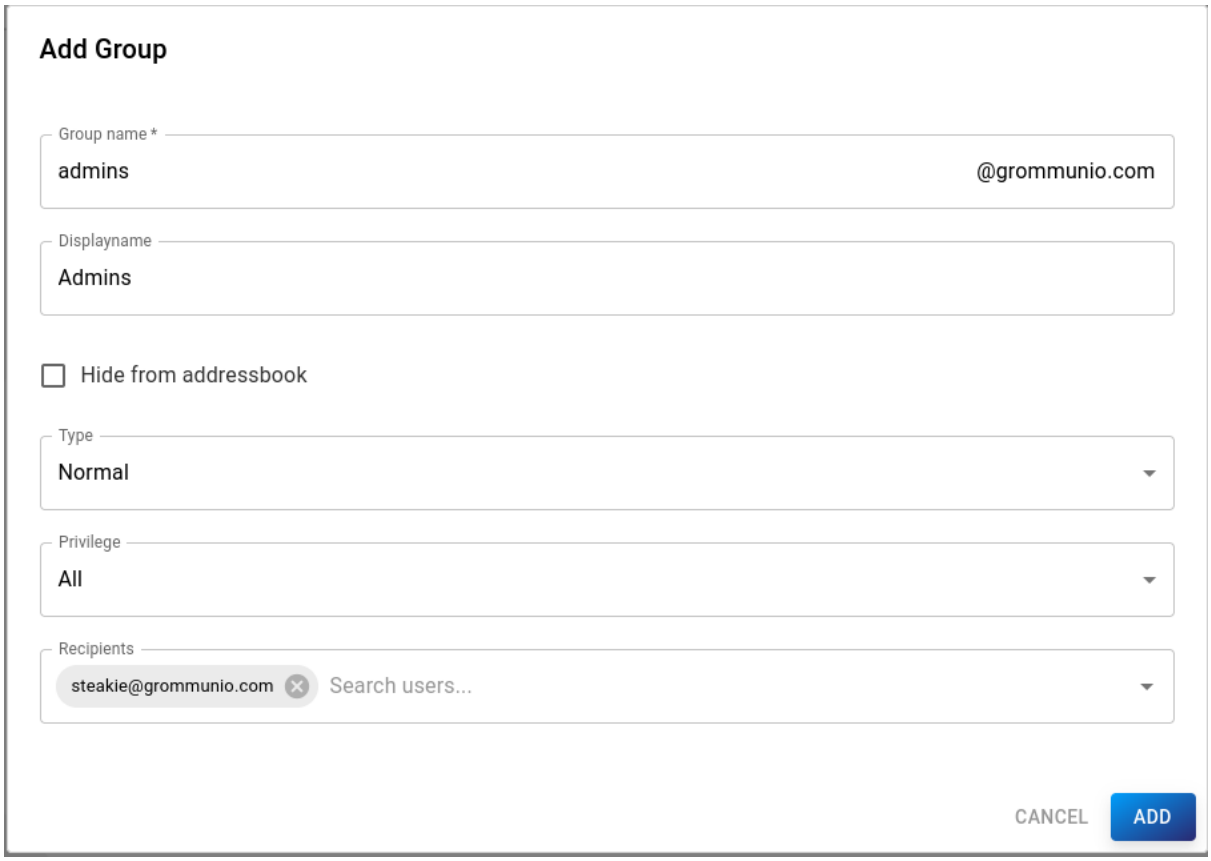
6.1.5 Groups

If at least one domain exists in the database, groups can be added to a domain. To show existing groups of a domain, navigate to the domain view in the drawer.

Click on a domain to expand available sub-pages and click on *Groups*, which will redirect you to the list of groups of this domain. If you have just installed grommunio or added the domain, the list will be empty.

6.1.5.1 Adding a group

To add a new group, click the blue *NEW GROUP* button to open the form dialog:



The following properties can be set:

- Group name (required): E-Mail address of the group
- Displayname: Displayed name of the group
- Hide from addressbook: If selected, the mailing list won't be visible in the Global Address Book
- **Type:**
  - Normal: Select users as recipients
  - Domain: All users of the domain will be recipients
- **Privilege: Users who are allowed to send E-Mails to the group**
  - All: Everyone
  - Internal: All users of the group
  - Domain: All users in the domain
  - Specific: Specific users (*Senders*)
- Recipients: Users of the domain, who are part of the group (not available if type=Domain)
- Senders: Users, who are allowed to send e-mails to the group (only available if privilege=Specific)

Click *Add* to confirm or *Cancel* to cancel.

6.1.5.2 Deleting a group

To delete a group, click on the trash icon of a group in the list view. Click *Confirm* to confirm or *Cancel* to cancel.

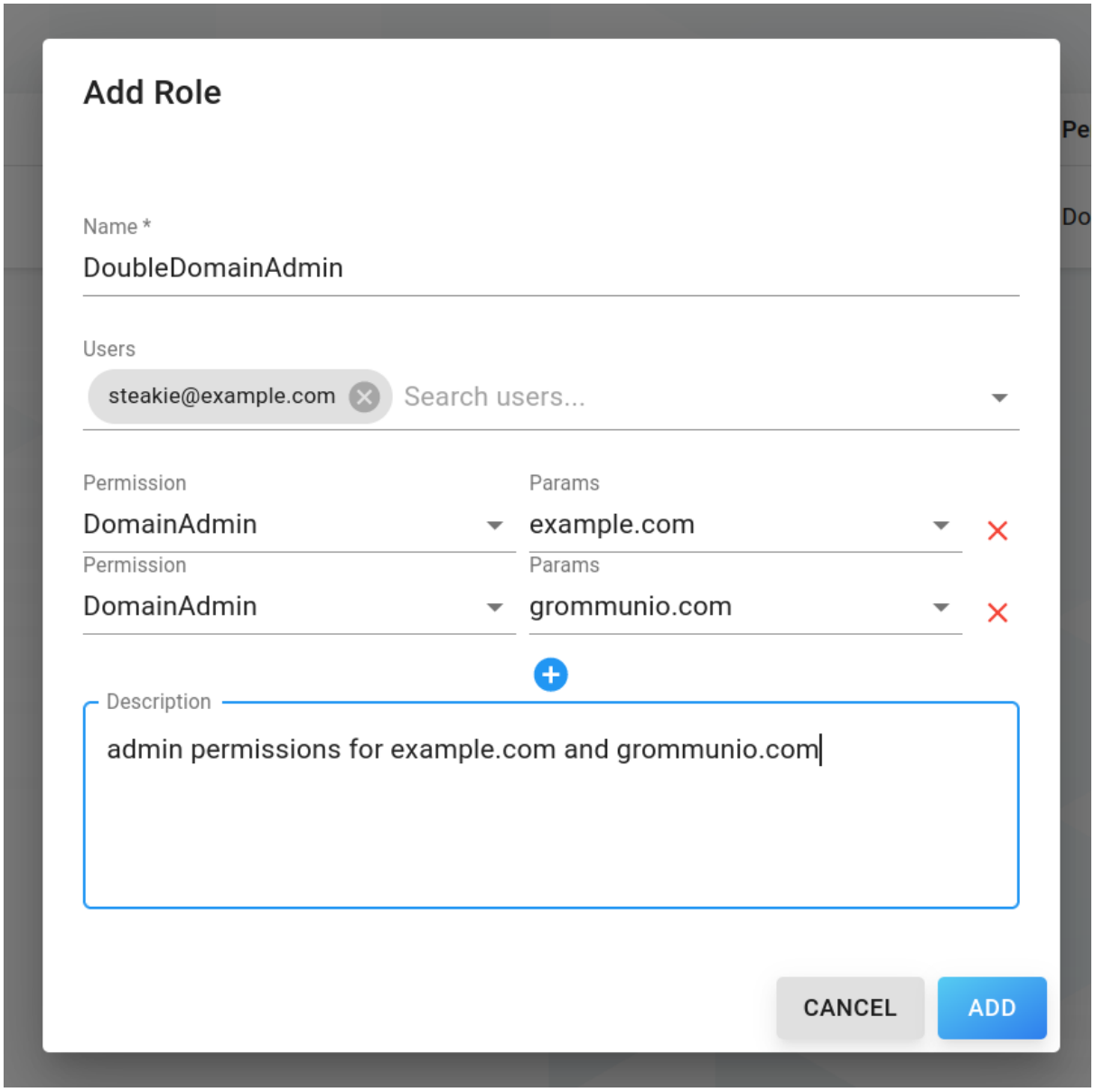
6.1.6 Roles

Click on *Roles* in the drawer, which will redirect you to the list view of existing roles. If you have just set up grommunio, the table will be empty.

By default, every time a domain is added, a new role with rights for the new domain will be added. Additionally, you can create your own roles to specify access rights for multiple domains.

6.1.6.1 Adding a role

To add a new role, click the blue *NEW ROLE* button to open the form dialog:



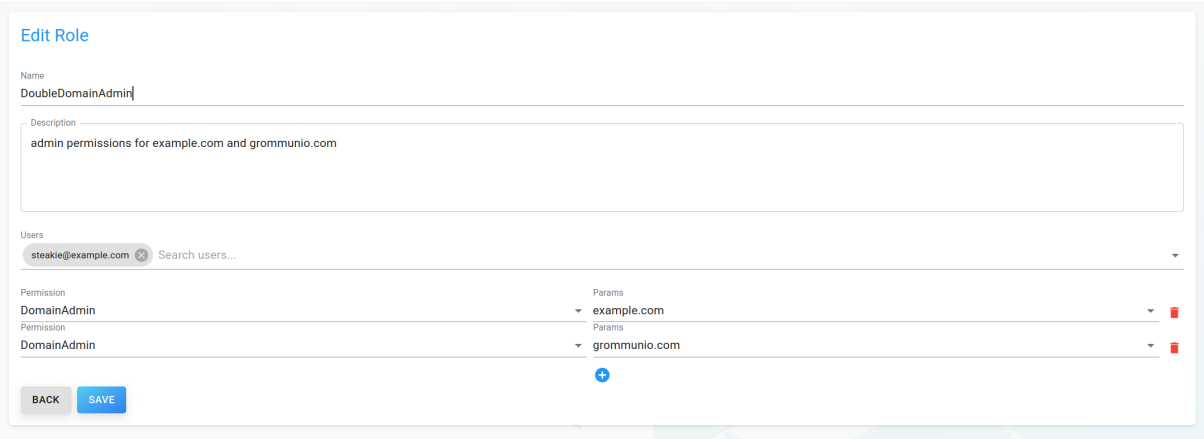
The following properties can be set:

- Name (required): Name of the role
- Users: Users to which this role will be assigned to
- **Permissions:**
  - SystemAdmin: Permits any operation
  - SystemAdminRO: Grants read-only permissions to system settings
  - DomainAdmin: Permits operations on for specific domain
  - DomainAdminRO: Grants read-only permissions to specific domain
  - DomainPurge: If present, grants permission to purge any writable domain
  - OrgAdmin: Grants DomainAdmin permission to any domain with matching orgID
  - Params: Domain/Organisation to get access to with this role
- Description: Role description

Click *Add* to confirm or *Cancel* to cancel.

6.1.6.2 Editing a role

To edit an existing role, click on a role in the list to open the detailed view of a role.



Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

6.1.6.3 Deleting a role

To delete a role, click on the trash icon of a role in the list view. Click *Confirm* to confirm or *Cancel* to cancel.

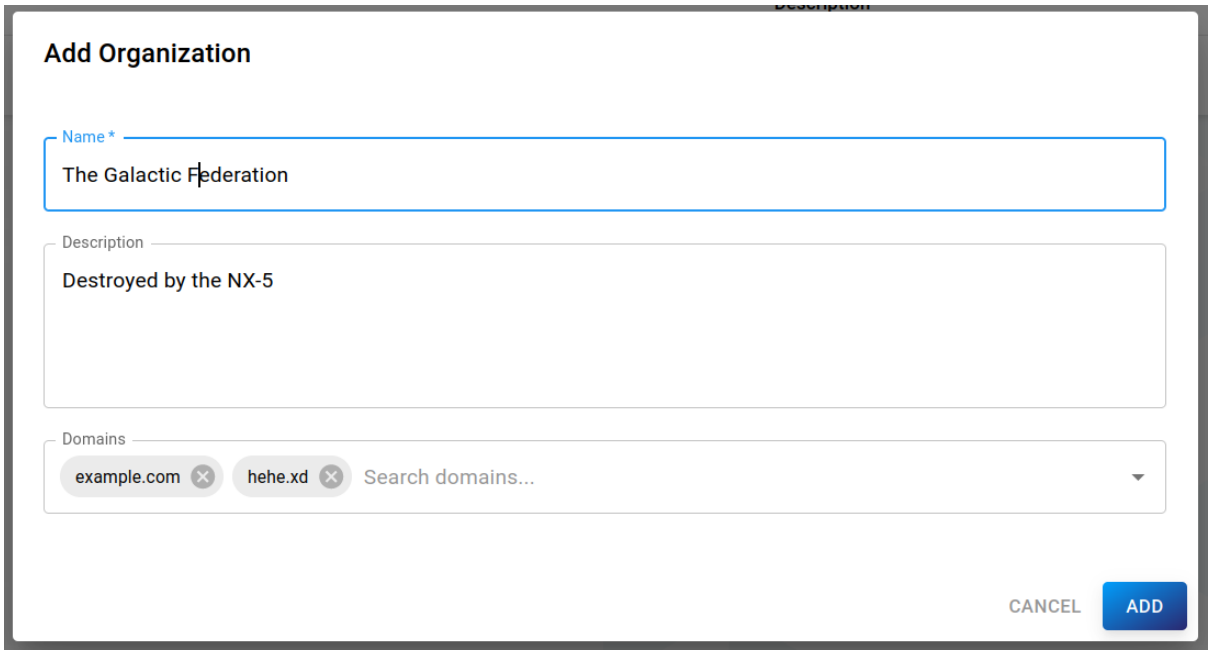
6.1.7 Organizations

Click on *Organizations* in the drawer, which will redirect you to the list view of existing organizations. If you have just set up grommunio, the table will be empty.

Organizations are used to group domains, and give access to multiple domains in the system by using the *OrgAdmin* role. Every domain can be associated with at most one organization.

6.1.7.1 Adding an organization

To add a new organization, click the blue *NEW ORGANIZATION* button to open the form dialog:



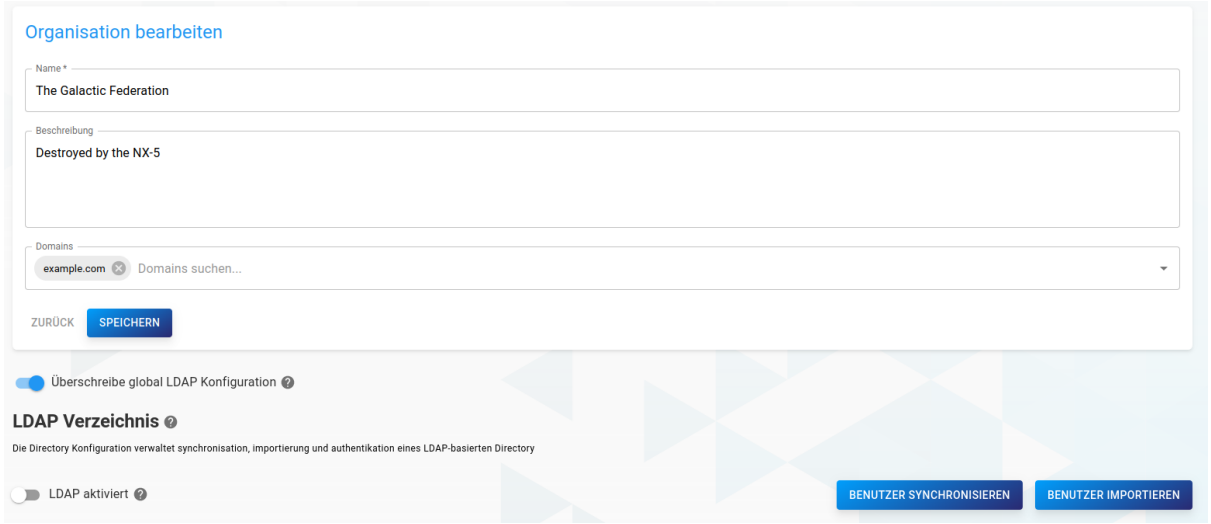
The following properties can be set:

- Name (required): Name of the organization
- Description: Detailed description of the organization
- Domains: Domains which are part of this organization

Click *Add* to confirm or *Cancel* to cancel.

6.1.7.2 Editing an organization

To edit an existing organization, click on an organization in the list to open the detailed view of an organization.



In this view, it is also possible to override the global LDAP configuration for domains in this organisation. To get more information about creating an LDAP config, see the *LDAP* section of this documentation.

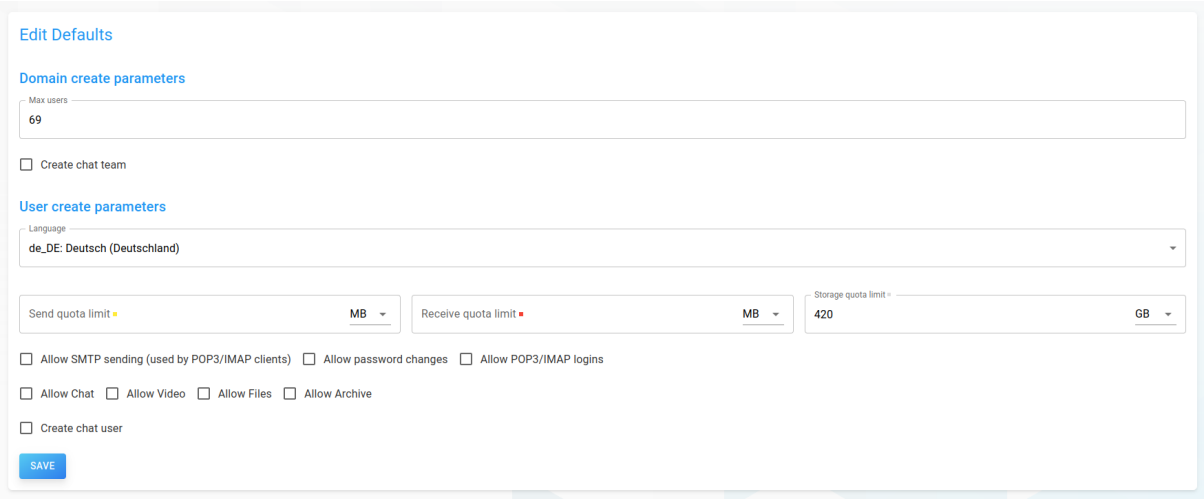
6.1.7.3 Deleting an oranization

To delete an oranization, click on the trash icon of a role in the list view. Click *Confirm* to confirm or *Cancel* to cancel.

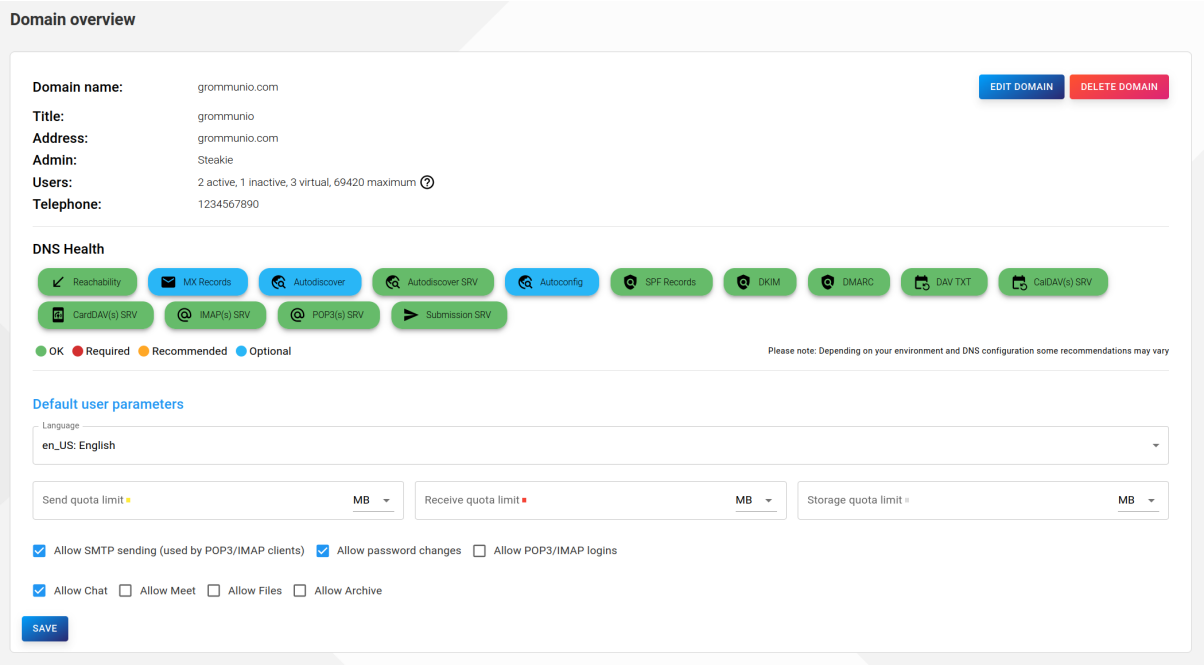
6.1.8 Defaults

To simplify the creation of domains and especially users, it is possible to set default create parameters. If set, the input masks for adding a domain or user will automatically be filled with these values.

Users with SystemAdmin permissions, can set global defaults by clicking on *Defaults* in the drawer.



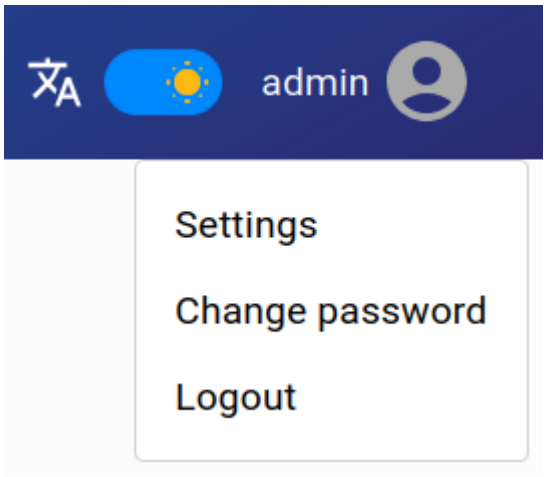
These values can be overwritten for each domain in the domain overviews:



6.1.9 Settings

Settings are split into server and user settings.

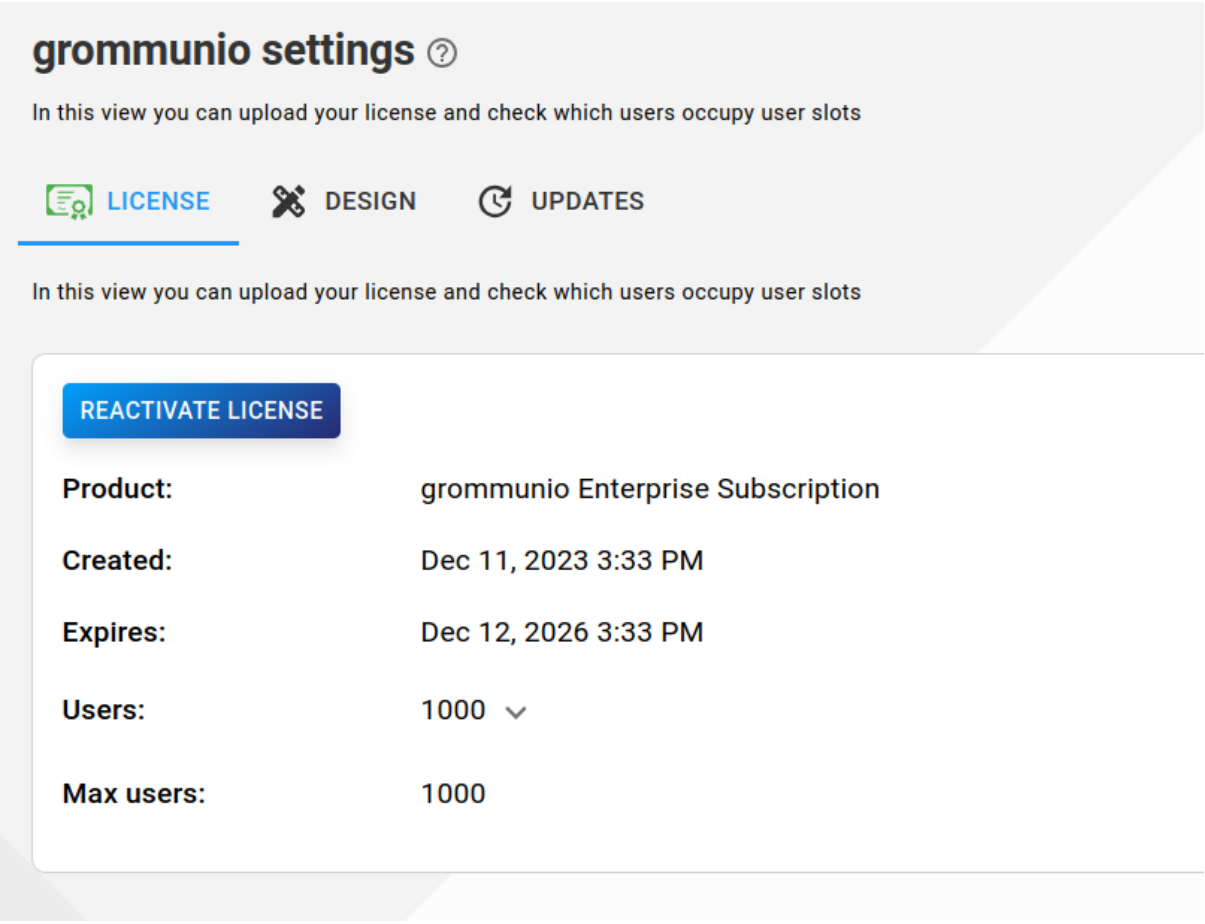
To change language, darkmode or theme, use the respective buttons in the topbar or the user icon



Server settings can be changed by clicking the settings icon in the topbar:

6.1.10 License

To use the full potential of grommunio you can upload your license by clicking *Upload* and selecting your purchased license. If you do not have a grommunio license yet, but want to upgrade, you can click on *Buy now*.



The following license properties are display:

- Product: Type of grommunio subscription (Community, Business, etc...)
- Created: Date on which the license was created
- Expires: Last day on which the license needs to be renewed
- Users: Current amount of users on this license
- Max users: Maximum amount of users that can be created with the current license

If you click on the expansion icon next to the users count, you can see what users are occupying user slots of the license.

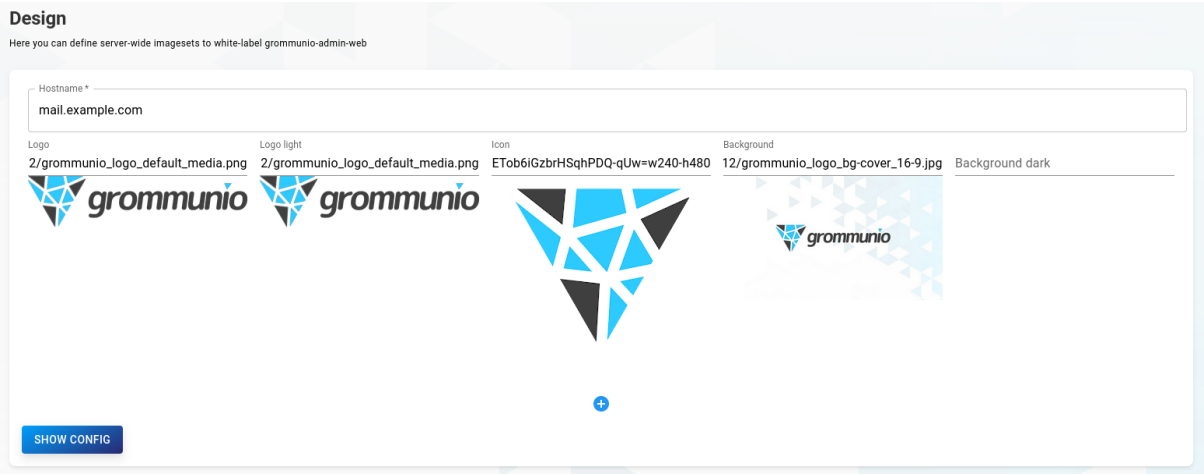
6.1.11 Design

Admins have the ability to whitelabel the grommunio-admin-web for *all* grommunio-admin-web users. There is a basic input mask, which helps you set custom app icons and background images.

It is possible to create separate sets of images for different hostnames. Click the plus icon to create a new set of images for a hostname. Following images can be set:

- logo: The logo in the login form
- logoLight: The logo in the expanded drawer
- icon: The icon in the collapsed drawer
- background: The background image in light mode
- backgroundDark: The background image in dark mode

Each of these keys must be an URL to an image file.



As you can see, it is not necessary to overwrite every image, but the hostnames need to be accurate.

Click on the *Show config* button to display the customImages config object, which needs to be copied into /etc/grommunio-admin-common/config.json on the server.

6.1.12 Application links

Links to external applications need to be configured in /etc/grommunio-admin-common/config.json on the server.

Following attributes are available:

- rspamdWebAddress:String: Url of rspamd server (default: '')
- mailWebAddress:String: Url of mail webapp (e.g. grommunio-web) (default: '')
- chatWebAddress:String: Url of grommunio-chat (default: '')
- videoWebAddress:String: Url of grommunio-meet (default: '')
- fileWebAddress:String: Url of grommunio-files (default: '')
- archiveWebAddress:String: Url of grommunio-archive (default: '')

6.1.13 Additional server-side configuration

Following additional attributes can be configured at /etc/grommunio-admin-common/config.json on the server.

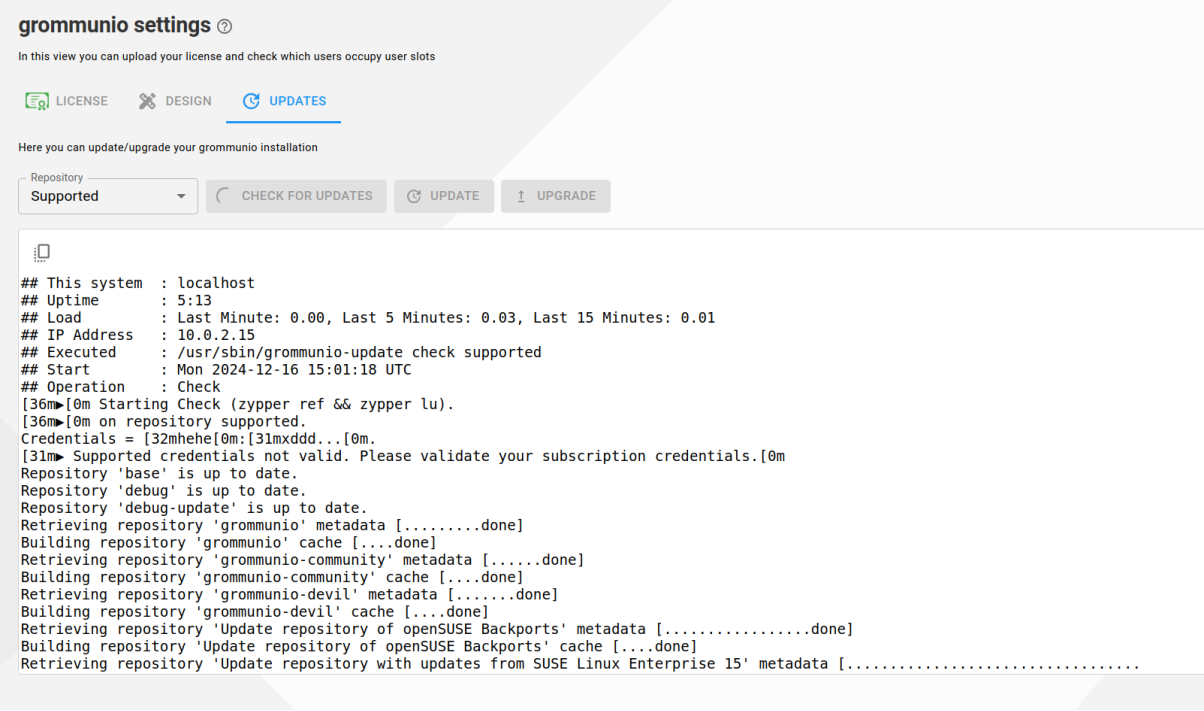
- devMode:boolean: For development, enables redux logger (default: false)
- tokenRefreshInterval:int: Sets token refresh interval in seconds. (default: 86400 (24h))
- defaultDarkMode:boolean: If true, the app will be set to dark mode, if not explicitly set by the user/browser (default: false)
- defaultTheme:string: Name of the default theme to use. Available themes: grommunio, green, purple, magenta, teal, orange, brown, bluegrey (default: "grommunio")

- loadAntispamData:boolean: Whether or not to load antispam data in the dashboard (default: true)
- searchAttributes:Array<String>: Array of strings, possible LDAP Search attributes (default: All attributes)

6.1.14 Updates

It is possible to update and upgrade server packages within grommunio-admin-web.

- Choose repository: Community (publicly available) or Supported (license required)
- Check for updates, update or upgrade packages by clicking the respective buttons



6.1.15 LDAP

It it possible to synchronise users from external user directories using LDAP. To configure LDAP, click on *LDAP* in the drawer, which will redirect you to the LDAP form to define a global LDAP configuration. This config can be overwritten for each individual organisation. To do so, navigate to *Organisations* and open the detailed view of an organisation. Flip the *Override global LDAP config* switch and set a config according to the following specification.

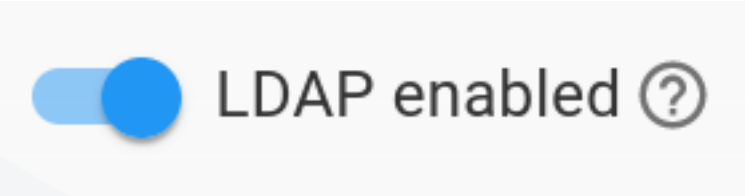
Important: Please note that configuration changes are not automatically applied to the services already running. Make sure to restart the services to be able to pick up the LDAP authentication first.

After applying a new LDAP configuration, the services are intentionally not automatically restarted as this would result into possibly inconvenient downtime if existing internal users are already used by the authentication manager (authmgr). Services can either be restarted through admin UI in the dashboard section or via systemd directly:

```
systemctl restart gromox-{http,zcore,pop3,delivery,delivery-queue,midb,imap}
```

6.1.15.1 Availability

LDAP not available means the LDAP config isn't set up correctly or the server can't be reached. If you want to disable LDAP manually, flip the *LDAP enabled* switch.



6.1.15.2 Configuration

Through this form, you create a *ldap.yaml* file, which configures an LDAP connection.

Properties are split into the following categories:

- LDAP Server
- Attribute Configuration
- Custom Mapping

To save a configuration, click *Save* at the bottom or click *Delete Config* to delete the current configuration.

6.1.15.3 LDAP Server

The following properties are available:

- LDAP Server (server): Address of the LDAP server to connect to
- LDAP Bind User (bindUser): DN of the user to perform initial bind with
- StartTLS: Whether to utilize the StartTLS mechanism to secure the connection
- LDAP Base DN (baseDn): Base DN to use for user search

6.1.15.4 Authentication manager

Primary authentication mechanism

- Always MySQL (default): MySQL authentication
- Always LDAP: LDAP authentication
- Automatic: The choice between LDAP/MySQL occurs dynamically, depending on whether the user was imported from LDAP originally.

6.1.15.5 Attribute Configuration

The following properties are available:

- LDAP Templates (templates): Template to prefill any fields below. Available are:
  - OpenLDAP
  - ActiveDirectory
- LDAP Filter (filters): LDAP search filter to apply to user lookup
- Unique Identifier Attribute (objectID): Name of an attribute that uniquely identifies an LDAP object

- LDAP Username Attribute (username): Name of the attribute that corresponds to the username (e-mail address)
- LDAP Default Quota (defaultQuota): Storage quota of imported users if no mapping exists
- LDAP Display Name Attribute (displayName): Name of the attribute that contains the name

6.1.15.6 LDAP Search Attributes

Controls which attributes the "Search in LDAP" functionality will look at when searching using an arbitrary search string.

6.1.15.7 Custom Mapping

LDAP attribute -> PropTag mapping to use for LDAP import. Any mappings specified take precedence over active templates.

You can create a list of *(Name, Value)* pairs

- Name: Name of the PropTag the attribute maps to
- Value: Value of the PropTag the attribute maps to

6.1.15.8 User import and synchronisation

To import/sync users from all domains, you have to have SystemAdmin permissions. If you do, click on *IMPORT USERS* or *SYNC USERS*. This will import/sync all users of all domains.

If you don't have these permissions, you can import/sync users for your domain. To do that, navigate to the user list(s) of your domain(s).

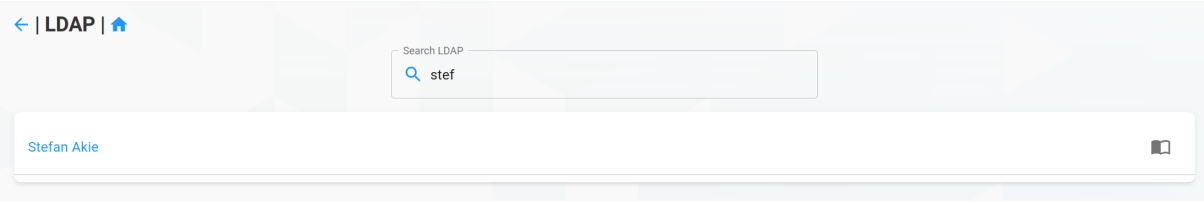
Importing users will synchronise all already imported users and also import new ones. Synchronising will only do the first.

6.1.15.9 Domain user import and synchronisation

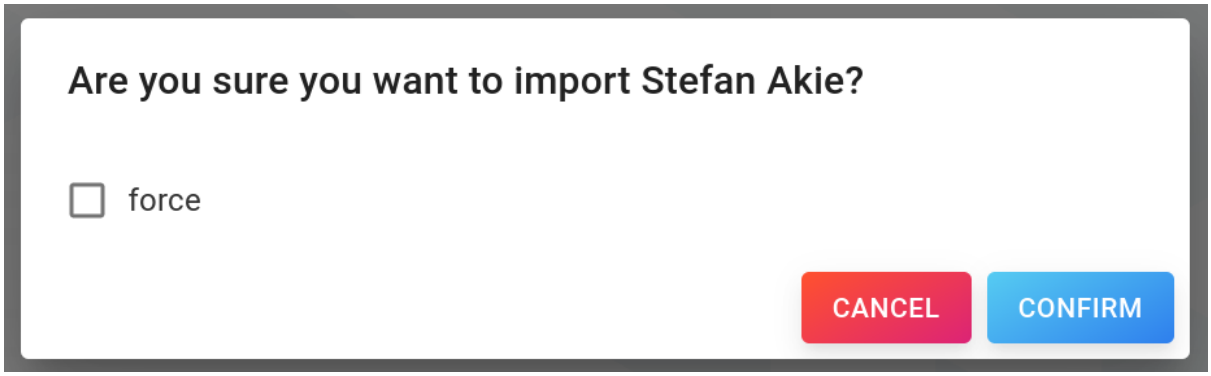
In the users list, you can either import/sync all users of this domain by clicking *Import/Sync ldap users*. If you want to import specific users, you can do the following:

6.1.15.10 User import

Click on *Search in ldap* to open a list view of ldap users. Simply enter a username at the searchbar and click the import icon of a user to import.



There is the option to force the import. If checked, an existing user with this usename in the grommunio database will be overwritten.



You can sync these specific users by clicking on them in the list view and clicking the *Sync* button in the detailed view (only for LDAP users).

6.1.15.11 Detaching a user

If you want to modify an ldap user, you need to detach it from ldap. You can achieve this by clicking *Detach* in the detailed user view. This essentially removes the synchronisation until forcefully overwriting the user via another import.

6.1.15.12 Removing orphaned users

If a user was removed from the ldap directory, the imported user will be orphaned. To show and/or delete currently orphaned users, click on *Check ldap users*.

6.1.16 DB Configuration

It is possible to create config files in the database to manage services. Every config file manages exactly one file and includes lines of *(key, value)* pairs.

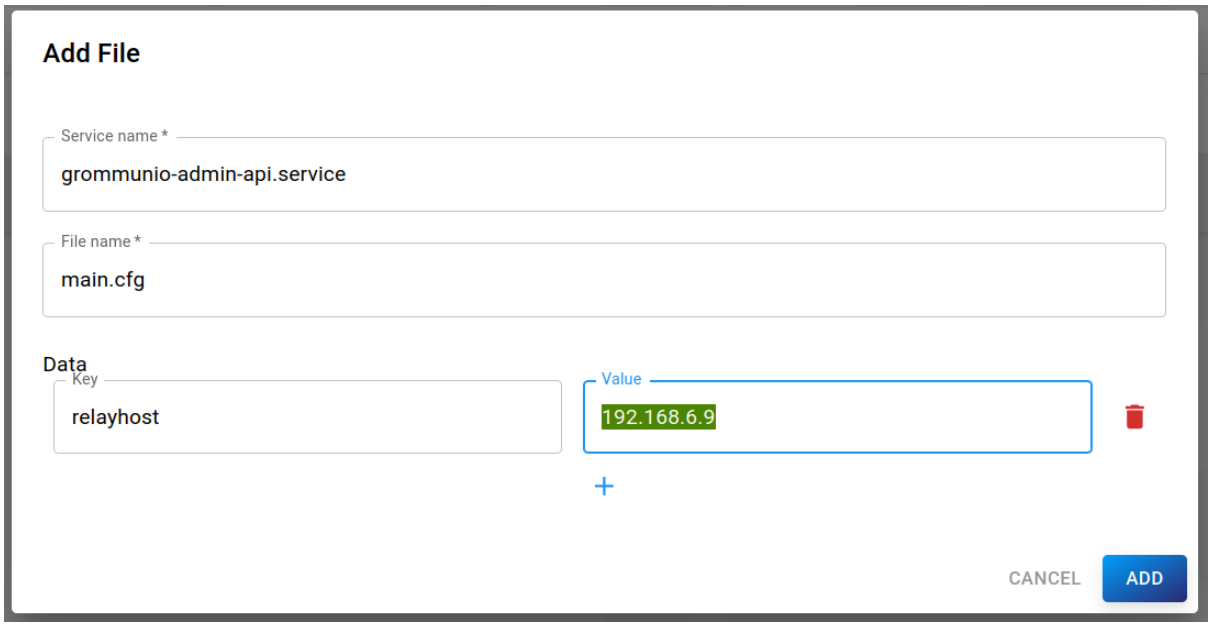
This creates a hierarchical structure:

- **ServiceA**
  - **FileA**
    - foo=bar
  - **FileB**
    - test=example
    - test2=example2
- **ServiceB**
  - **FileC**

* key=value

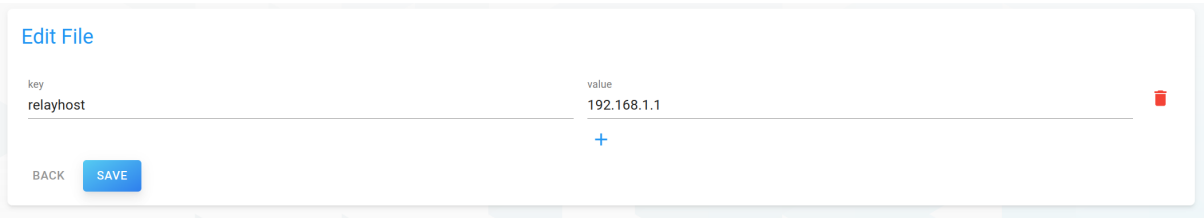
6.1.16.1 Adding a file

A useful example would be to configure a relayhost in postfix:



6.1.16.2 Editing a file

To edit a file, click on the service the file belongs to. This will open a detailed view of the service with a list of its files. Click on a file to open its detailed view and edit the *(key, value)* pairs to your needs.



Click *Save* to confirm or *Cancel* to discard your changes.

6.1.16.3 Deleting a file

To delete a file, click on the service the file belongs to. This will open a detailed view of the service with a list of its files. Click on the trash icon of a file to delete it and confirm.

6.1.16.4 Configuring grommunio-dbconf

grommunio-dbconf is an internal service, that will execute actions/commands when configs change. These actions can be specified for every service separately.

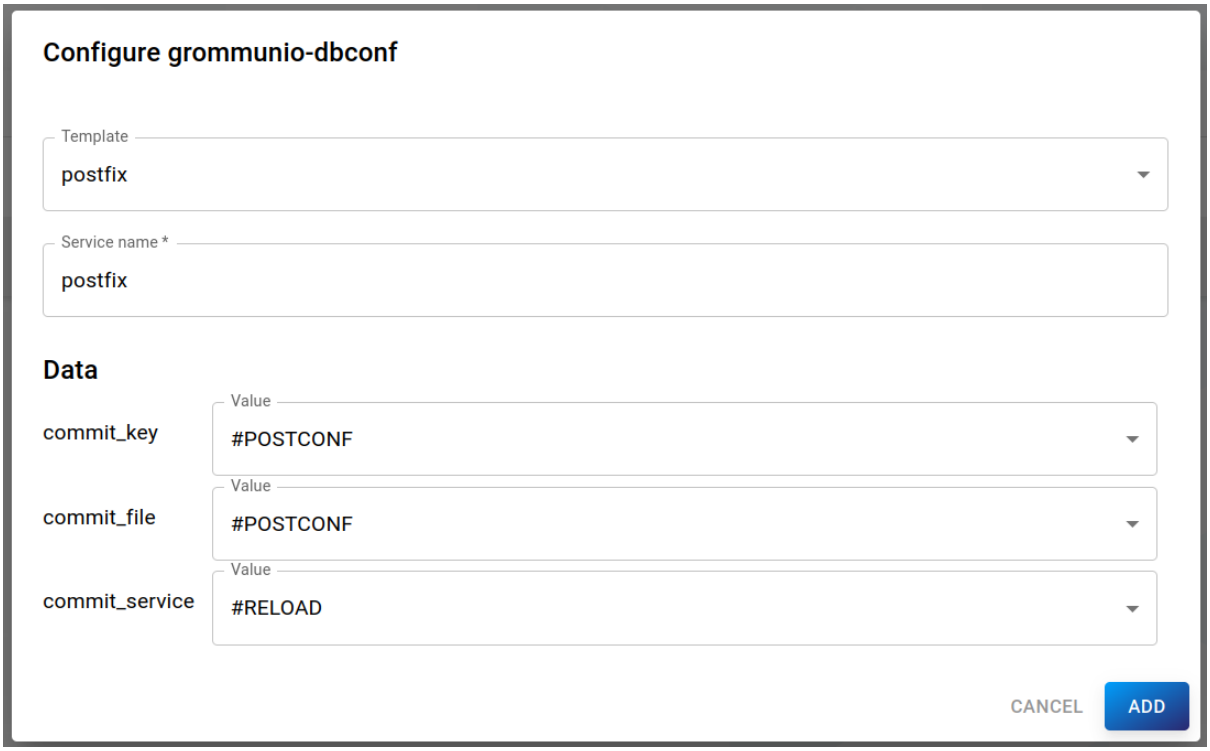
6.1.16.5 Adding a grommunio-dbconf file

Actions to be executed when a config of a service <*servicename*> changes, need to be set in the file *grommunio-dbconf/<servicename>*.

There are pre-made commands to set for either key, file or service changes. Those can be found on the *Commands* tab.

If a command doesn't exist, the next lower level command will be executed (service -> file -> key).

For example, you could configure *postfix* changes like this:

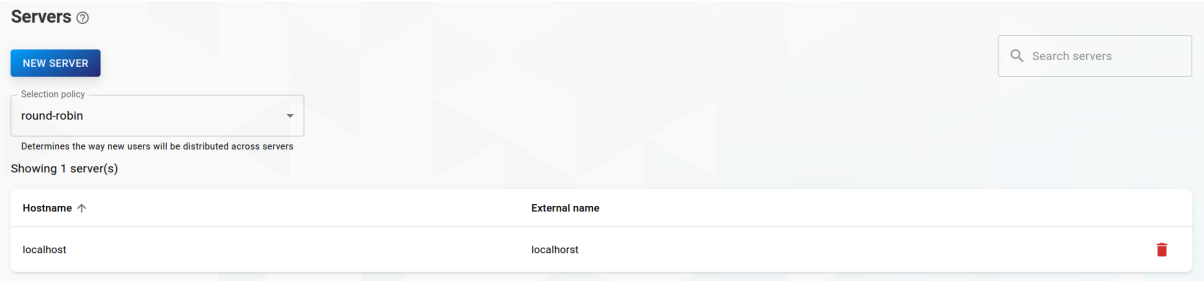


This will, among else, restart the service if the service config changes.

6.1.17 Servers

If grommunio is running on a distributed system, the list of servers can be added in this view. It is possible to specify the selection policy for user distribution. You can select from:

- round-robin: Always use the server on which a user has *not* been added for the longest time (in a circle-like manner).
- balanced: Put new user on server with least workload
- first: Always use the first server
- last: Always use the last server
- random: Pick a random server



6.1.17.1 Adding a server

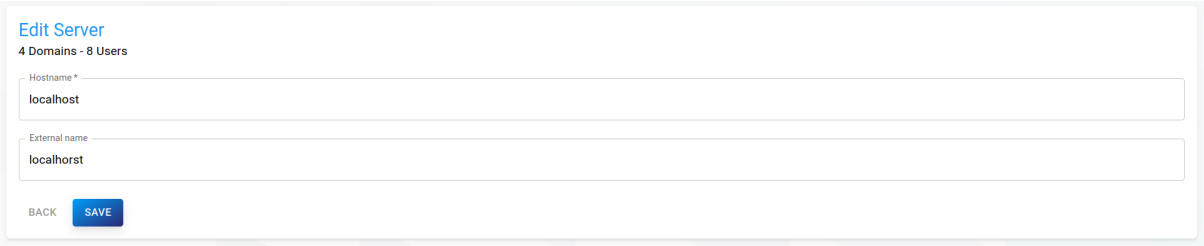
To add a new server, click the blue *NEW SERVER* button to open the form dialog:

The following properties can be set:

- Hostname (required): Internal server hostname
- Extname (required): Hostname for external access (DNS-Name)

6.1.17.2 Editing a server

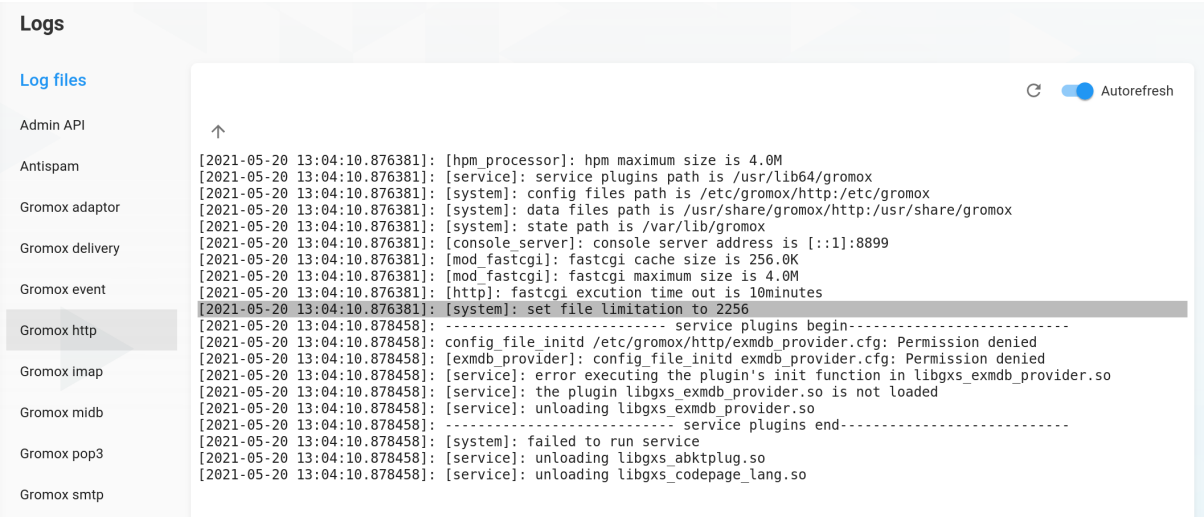
To edit an existing server, click on a server in the list to open the detailed view.



Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

6.1.18 Logs

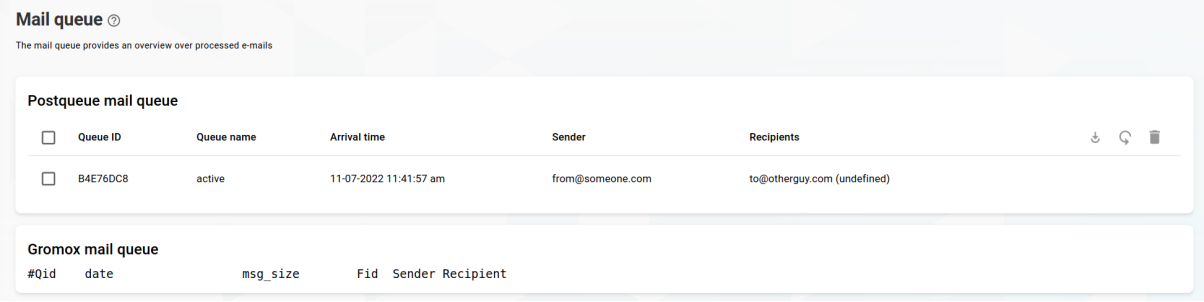
Click on *Logs* in the drawer, which will redirect you to the list of available logs. Usually, you will see a list of grommunio/gromox services, which *journalctl* logs you can view here.



Click on the uparrow to show previous logs. Click on the the refresh button to fetch new logs or toggle the autorefresh switch to automatically refresh logs of the selected service every 5 seconds. Click on a log line to fetch every log *after* the timestamp of the clicked line.

6.1.19 Mail queue

Click on *Mail queue* in the drawer, which will redirect you to the view of the current postfix and gromox mail queue.



These lists will update automatically every 10 seconds.

6.1.19.1 Actions

Select table rows by clicking the checkboxes. Mail queue actions will be used on the selected entries.

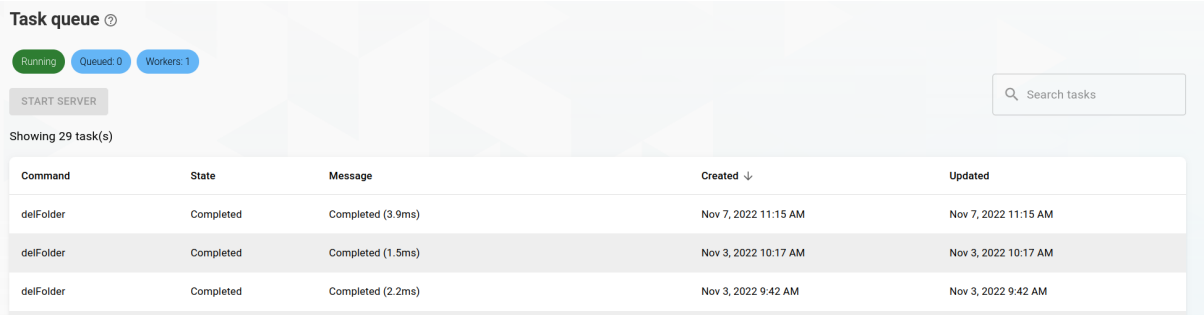
The actions are:

- Flush: Try to continue mail processing
- Requeue: Remove mail from queue and add queue the same mail as new entry
- Delete: Permanently remove mail from queue

6.1.20 Tasks

Click on *Mail queue* in the drawer, which will redirect you to the Tasks view.

Tasks are created for operations which could potentially take a long time. Currently, this includes LDAP sync and folder deletion. If one of these operations take too long, a background task is created, which can be viewed in this table.

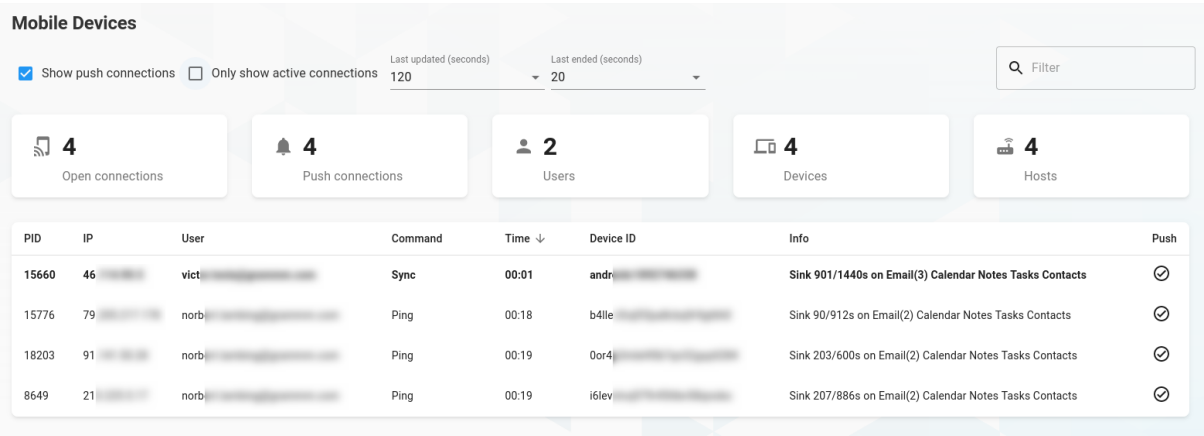


In case the internal task processor is not running, it can be started manually by clicking the *Start server* button.

Further task details can be seen in the task details view, by clicking on a task in the table.

6.1.21 Mobile devices

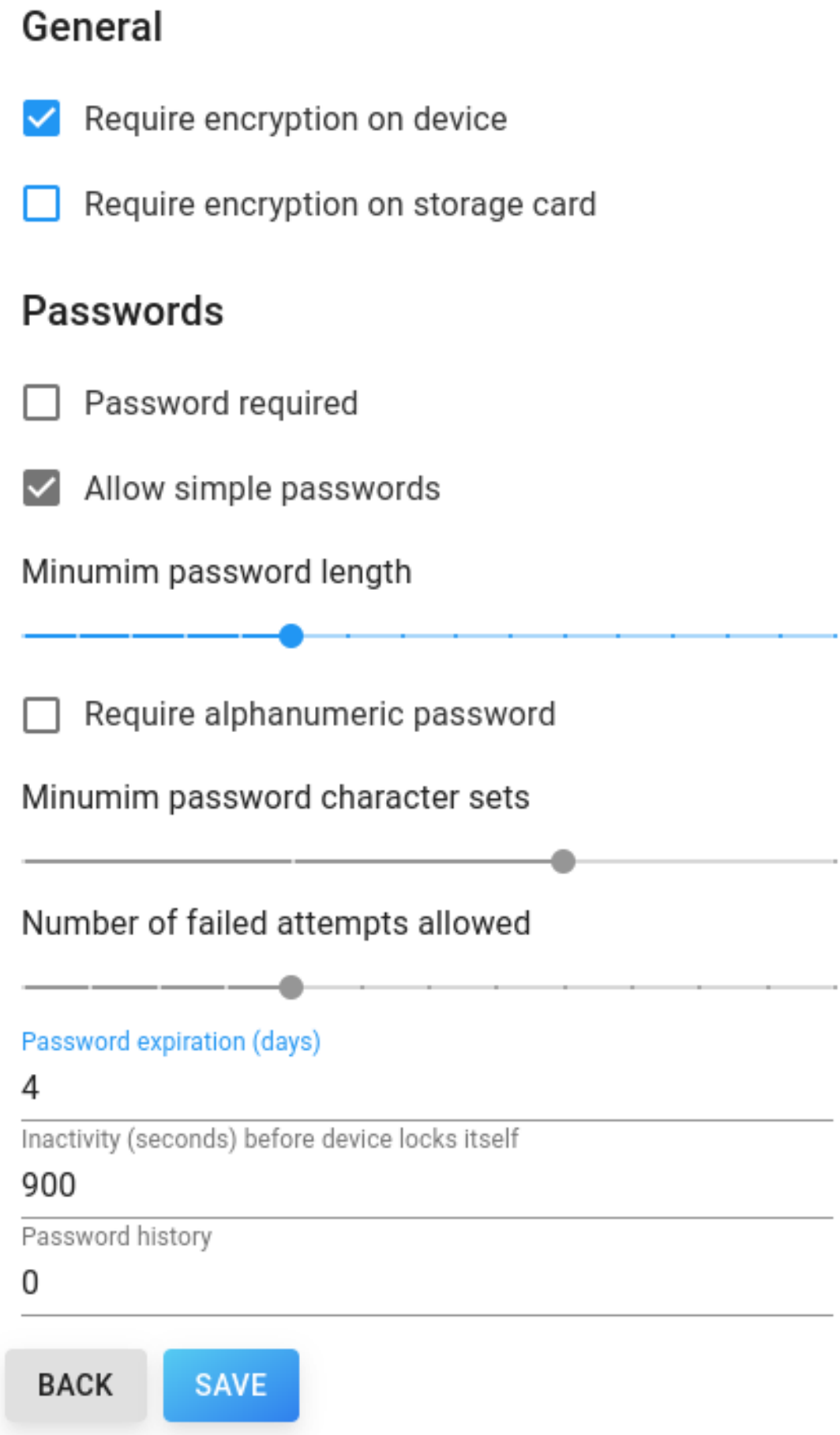
Click on *Mobile devices* in the drawer, which will redirect you to the list of synchronised mobile devices. This view is a recreation of the grommunio-sync-top CUI.



The view will update the devices every 2 seconds. On the top, you can specify filters for the table, like text-based search or activity of devices.

6.1.22 Sync policies

The synchronisation behavior of devices is specified by the sync policies, which are a set of rules. When a user logs into an account, these policies will be applied to the device and updated as soon as the policy is changed. It is not possible to change the policies globally, but per domain (all users of a domain) or per user. To change the policy for all users of a domain, navigate to the list of domains and click on the domain for which you want to change the policy. Under the *Sync policy* tab, you can see the current rules.

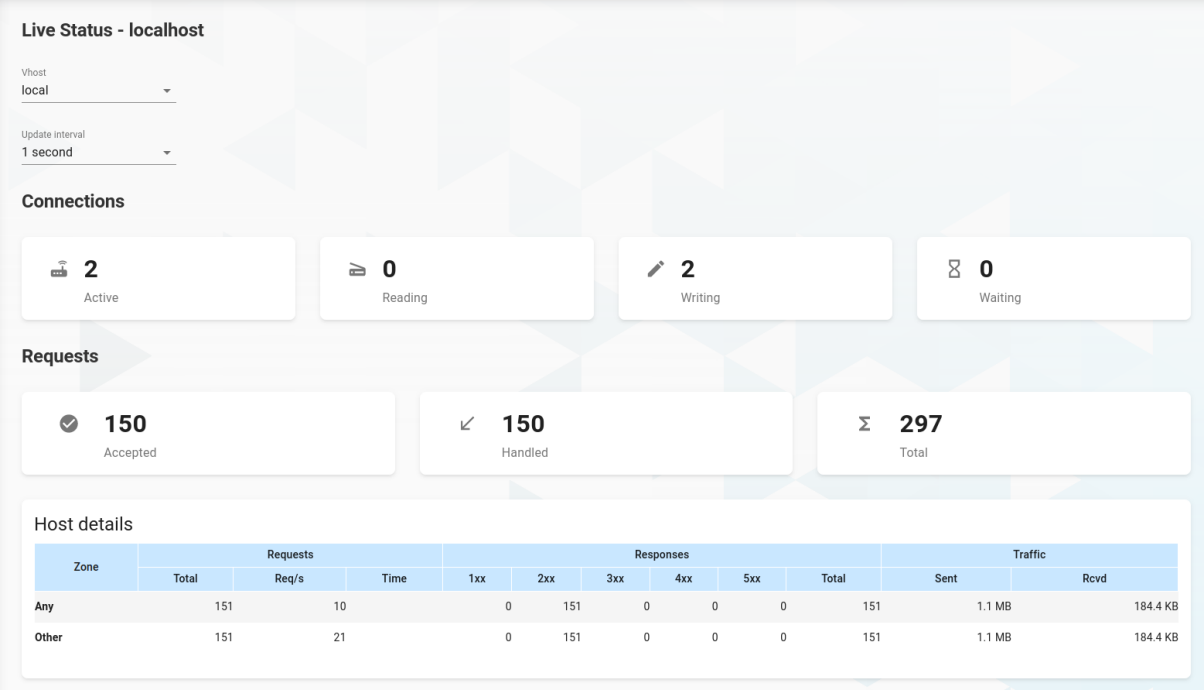

General
Require encryption on device
Require encryption on storage card
Passwords
Password required
Allow simple passwords
Minumim password length
Require alphanumeric password
Minumim password character sets
Number of failed attempts allowed
Password expiration (days)
4
Inactivity (seconds) before device locks itself
900
Password history
0
BACK
SAVE

Blue checkboxes, sliders or textfields indicate deviations from the default policy, grey ones match it.

To specify specific rules for a user, navigate to list of users and click on the user for whom you want to change the policy. Just like domain-specific policies, current rules are displayed under the *Sync policy* tab. Again, blue checkboxes, sliders or textfields indicate deviations from the *domain* policy of this user, grey ones match it.

6.1.23 Live Status

Click on *Live Status* in the drawer, which will redirect you to the live, realtime view of the grommunio web services. Any HTTP request shows up in live status, including MAPI/HTTP, EAS, EWS and other requests made. All connections other than grommunio Groupware, e.g. Chat and Files are also viewable and can be tracked by the entrypoint URL in the list.



At the top you can select one of the available vhosts and the update interval.

6.2 grommunio admin CLI (ACLI)

6.2.1 grommunio-admin

grommunio-admin is the command line interface tool of the grommunio Admin API. grommunio-admin is a low level administrative tool for grommunio configuration and provides a large number of subcommands to administrate grommunio accordingly.

grommunio-admin also provides bash completion functionality and an interactive shell, with the following subcommands available:

config	Show or check configuration. See **'grommunio-admin-config'_**.
connect	Connect to remote CLI. See **'grommunio-admin-connect'_**.
dbconf	Database-stored configuration management. See *grommunio-admin-dbconf*.
domain	Domain management. See **'grommunio-admin-domain'_**.
fetch-mail	Fetchmail management for retrieval of remote mails. See **'grommunio-admin-fetchmail'_**.
fs	Filesystem operations. See **'grommunio-admin-fs'_**.
ldap	LDAP/Active Directory configuration, diagnostics and synchronization. See **'grommunio-admin-ldap'_**.
mconf	Managed configurations manipulation. See **'grommunio-admin-mconf'_**.
mlist	Mailing/distribution list management. See **'grommunio-admin-mlist'_**.
passwd	Internal user password management. See **'grommunio-admin-passwd'_**.
run	Run the REST API. See **'grommunio-admin-run'_**.
shell	Start interactive shell. See **'grommunio-admin-shell'_**.
taginfo	Print information about MAPI property tags. See **'grommunio-admin-taginfo'_**.
user	User management. See **'grommunio-admin-user'_**.
version	Show version information. See **'grommunio-admin-version'_**.

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021 grommunio GmbH

6.2.1.1 Name

grommunio-admin config — grommunio-admin config introspection

6.2.1.2 Synopsis

grommunio-admin config *check*
grommunio-admin config (*dump*|*get*) [*KEY*]
grommunio-admin config *trace [-s]* (*files*|*values*) [*KEY*]

6.2.1.3 Commands

6.2.1.3.1 check

Check the structural validity of the configuration.
Does currently not validate the semantic integrity, i.e. existence of referenced files, LDAP or database connectivity etc., although this functionality may be added in the future.

6.2.1.3.2 dump, get

Print the complete configuration.
As the grommunio-admin configuration can (and probably will) be distributed over multiple files, the get command provides an easy way to see the effective configuration.
The output can be reduced to a single *KEY*, if specified. Sub-levels can be specified in dotted notation (e.g. sync.defaultPolicy)
The *dump* command is an alias for *get* and remains for backward compatibility.

6.2.1.3.3 trace

Trace source of effective configuration.
Results can be presented either by file (files), showing which parts of a file are actually used, or by value (values), showing which file each value originates from.
Installation of the Python termcolor package is advised for a more readable output. See section *Tracing* for more information.

6.2.1.4 Options

KEY
: Only view specified key.

-s, --show-history
: Display more value history (see section *Tracing* for more information)

6.2.1.5 Tracing

6.2.1.5.1 By-File

Print annotated contents of each file.
Each line is marked and color coded to show its status. The following annotations are used:

- +, green: The value is part of the final configuration
- *x*, red: The value is overwritten by a later file
- *, yellow: The object or list is extended by a later file
- ~, grey: The value is overwritten with the same value

Additionally, lines overwriting or extending previous entries are printed in boldface.
When specifying *--show-history*, each value that is overwritten or extended is annotated with the files doing so (each being color coded with the effect it has on the value).

6.2.1.5.2 By-Value

Print annotated effective configuration.
Each line is annotated with the file it originates from. In case of objects and lists, all contributing files are listed.
When specifying *--show-history*, overwritten files containing that value are listed as well. The effective source file is underlined.
For better visualization, color coding is done on a per-file basis: Each file is assigned an individual style which is used for its contributions. Objects and lists originating from multiple files are always shown in boldface white.

6.2.1.6 Name

grommunio-admin connect — Connect to remote CLI

6.2.1.7 Synopsis

grommunio-admin connect [*-c COMMAND*] [*--no-verify*] [*--redirect-fs* [*--auto-save (local|remote|discard|print)*]] [*-v*] [*HOST* [*USER* [*PASSWORD*]]]

6.2.1.8 Description

Connect to a remote server to invoke CLI commands on.
Requires a running admin API with active remote CLI and a user with SystemAdminPermission.

Note that the remote CLI currently uses a REST interface which does not provide a standard input, rendering commands that rely on user interaction useless.

6.2.1.9 Options

HOST
: Host to connect to, in the format *protocol*://*hostname*:*port*, where protocol is either http or https. If omitted, the protocol is auto-detected, with https taking precedence over http. If no port is specified, the default ports 8080 (http) and 8443 (https) are used. *hostname* can either be a resolvable host name, an IPv4 address or an IPv6 address in brackets. The default hostname is *localhost*.

PASSWORD
: Password to use for authentication. Default is to prompt.

USER
: User to use for authentication. Default is *admin*.

--auto-save ACTION
: Choose automatic action for received files when filesystem redirection is enabled. Possible actions are:

  discard - discard any received file
  local - save at local path
  print - print file contents to stdout and discard
  remote - save at path reported from remote server

-c, --command
: Execute command on remote server and exit instead of starting an interactive shell.

--no-verify
: Continue with https even if the TLS certificate presented by the server is invalid. Required if the server uses a self-signed certificate that is not installed on the system. Use with caution.

-p, --password
: Prompt for password even when connecting to localhost.

--redirect-fs
: Redirect CLI initiated file operations to local filesystem. See section *Filesystem Emulation* for details.

-v, --verbose
: Print more detailed information about the connection process.

6.2.1.10 Filesystem Emulation

When the *--redirect-fs* option is given, CLI initiated file operations are performed in an emulated filesystem and written files are sent back to the client.

Note that this does only apply to files which are opened by CLI operations, while module-level operations (e.g. loading of configurations) are unaffected.

Files received from the remote server can then be viewed or saved locally.

6.2.2 grommunio-admin-dbconf

6.2.2.1 Name

grommunio-admin dbconf — Database-stored configuration management.

6.2.2.2 Synopsis

grommunio-admin dbconf (*commit* | *delete*) *SERVICE* [*FILE* [*KEY*]]
grommunio-admin dbconf *get SERVICE FILE* [*KEY*]
grommunio-admin dbconf *list* [*SERVICE* [*FILE* [*KEY*]]]
grommunio-admin dbconf *set* [*-b*] [*-i*] [--] *SERVICE FILE KEY VALUE*

6.2.2.3 Description

grommunio dbconf provides the ability to store and manage configurations at a single location while making it available across distributed systems. The configurations are stored in the central MySQL database and can be accessed via *grommunio-dbconf* and *grommunio-admin-dbconf*.
While both tools essentially provide the same functionality, *grommunio-dbconf* provides far better performance and is intended to be used for quickly accessing the configuration.

Configurations consist of key/value pairs organized in files, grouped by service. Each service can have an arbitrary number of configuration files, which in turn can contain an arbitrary number of unique keys.

6.2.2.4 Commands

commit
: Trigger commit hook for service, file or key

delete
: Delete service, file or key

get
: Get file or single key

list
: List available services, files or keys

set
: Set a configuration key

6.2.2.5 Options

SERVICE
: Name of the service to configure

FILE
: Name of the configuration file

KEY
: Name of the configuration key

VALUE
: Value to store in the key

--
: Indicate that all options have been specified and only names follow

-b, --batch
: Do not auto-commit

-i, --init
: Only set if configuration key does not exist yet

6.2.2.6 grommunio-admin

The grommunio-admin API and CLI are also dbconf consumers. This allows system adiministrators to change certain configurations without filesystem access and the need to restart the API.

The following files and keys are meaningful when placed under the *grommunio-admin* service:

6.2.2.6.1 multi-server

policy
: Server selection policy for newly created users and domains in multi-server environments. Possible values are *balanced*, *first*, *last*, *random* and *round-robin*. Default is *round-robin*.

6.2.2.7 Commit Hooks

When modifying values, potential consumers can be notified of this change via commit hooks, for example by restarting the service using the configuration. For security reasons only a few white-listed commands are available (see section *AVAILABLE COMMIT COMMANDS*).

Commit hooks can be defined on key, file or service level. *set* operations always trigger commits at key level, while the *commit* command can directly trigger key or service level hooks depending on whether a file or key is specified.

If no hook is defined for a specific trigger level, it automatically falls through to the next lower level, in the order *key* > *file* > *service*.

Commit hooks for a service can be defined by setting commit_key, commit_file and commit_service keys under *grommunio-dbconf/<service>* to a valid command (see below).

6.2.2.8 Available Commit Commands

The following commands are available:

6.2.2.8.1 Key

postconf -e $ENTRY

6.2.2.8.2 File

postconf -e $FILE_S && systemctl reload postfix

6.2.2.8.3 Service

systemctl reload $SERVICE
systemctl restart $SERVICE

6.2.2.9 Macros

As the whitelisted commands might be hard to memorize and may be changed in the future, macros are provided that expand to whitelisted commands.

The following macros are defined:

6.2.2.9.1 Key

#POSTCONF -> postconf -e $ENTRY

6.2.2.9.2 File

#POSTCONF -> sudo postconf -e $FILE_S && systemctl reload postfix

6.2.2.9.3 Service

#RELOAD -> systemctl reload $SERVICE
#RESTART -> systemctl restart $SERVICE

6.2.2.10 Command Variable Expansion

Commands can contain $-prefixed variables that are expanded before execution. The literal *$$* can be used to generate a single $.

The following variables are valid:

ENTRY
: Expands to $KEY=$VALUE (key level only)

FILE
: Complete content of the modified file as newline separated key=value entries (file level only)

FILE_S
: Complete content of the modified file as space separated key=value entries (file level only)

FILENAME
: Name of the modified file (key and file level)

KEY
: The modified key (key level only)

SERVICE
: Name of the modified service

VALUE
: New value of the modified key (key level only)

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.11 Name

grommunio-admin domain — Domain management

6.2.2.12 Synopsis

grommunio-admin domain create [*--create-role*] [*--homeserver HOMESERVER*] [*--no-defaults*] [*--skip-adaptor-reload*] [*<FIELDS>*] *-u MAXUSER DOMAINNAME*
grommunio-admin domain delete *DOMAINSPEC*
grommunio-admin domain list [*-f FIELD=<value>*] [*-s FIELD*] [*DOMAINSPEC*]
grommunio-admin domain modify [*<FIELDS>*] *DOMAINSPEC*
grommunio-admin domain purge [*--files*] [*-y*] *DOMAINSPEC*
grommunio-admin domain query [*-f ATTRIBUTE=<value>*] [*--format FORMAT*] [*--separator SEPARATOR*] [*-s FIELD*] [*ATTRIBUTE ...*]
grommunio-admin domain recover *DOMAINSPEC*
grommunio-admin domain show [*-f FIELD=<value>*] [*-s FIELD*] *DOMAINSPEC*

6.2.2.13 Description

Subcommand to show and manipulate domains.

6.2.2.14 Commands

create
: Create a new domain

delete
: Soft-delete a domain

list
: List domains

modify
: Modify domain

purge
: Permanently delete domain

query
: Query domain attributes

recover
: Recover a soft-deleted domain

show
: Show detailed information about a domain

6.2.2.15 Options

ATTRIBUTE
: Attributes to query. Available attributes are *ID*, *activeUsers*, *address*, *adminName*, *chat*, *chatID*, *displayname*, *domainStatus*, *domainname*, *endDay*, *homedir*, *homeserverID*, *inactiveUsers*, *maxUser*, *orgID*, *tel* and *title*

  If no attributes are specified, *ID*, *domainname* and *domainStatus* are shown.

DOMAINNAME
: Complete name of the domain

DOMAINSPEC
: Domain name prefix or domain ID

--create-role
: Automatically create a domain administrator role for the new domain

--files
: Also delete files from disk

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

--format FORMAT
: Output format. Can be one of *csv*, *json-flat*, *json-kv*, *json-object*, *json-structured* and *pretty*. Default is *pretty*.

" --homeserver HOMESERVER"
: ID of the homeserver to place the domain on

--no-defaults
: Do not apply configured default values

--separator SEPARATOR
: String to use for column separation (*csv* and *pretty* only). Must have length 1 if format is *csv*. Default is "," for *csv* and " " for pretty.

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

-y, --yes
: Assume yes instead of prompting

6.2.2.16 Fields

--address ADDRESS
: Content of address field

--adminName ADMINNAME
: Name of the domain administrator or primary contact

--endDay ENDDAY
: Date of domain expiration in YYYY-MM-DD format

--orgID ID
: ID of the organization to assign the domain to

--tel TEL
: Telephone number of domain administrator or primary contact

-u MAXUSER, --maxUser MAXUSER
: Maximum number of users in the domain

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2022 grommunio GmbH

6.2.2.17 Name

grommunio-admin exmdb — User or domain store management

6.2.2.18 Synopsis

grommunio-admin exmdb *TARGET folder create* [*--comment COMMENT*] [*-t TYPE*] NAME [*PARENTID*]
grommunio-admin exmdb *TARGET folder delete* [*-a*] [--clear] *FOLDERSPEC*
grommunio-admin exmdb *TARGET folder find* [*-x*] *NAME* [*ID*]
grommunio-admin exmdb *TARGET folder grant* [*-f*] [*-r*] *ID USERNAME PERMISSION* [*PERMISSION* ...]
grommunio-admin exmdb *TARGET folder list* [*-r*] [*--format FORMAT*] [*ID*]
grommunio-admin exmdb *TARGET folder revoke* [*-r*] *ID USERNAME* [*PERMISSION* ...]
grommunio-admin exmdb *TARGET store delete PROPSPEC* [*PROPSPEC* ...]
grommunio-admin exmdb *TARGET store get* [*--format FORMAT*] [*--separator SEPARATOR*] [*PROPSPEC* ...]
grommunio-admin exmdb *TARGET store set* [*PROPSPEC=VALUE* ...]

6.2.2.19 Description

Subcommand to access and modify a domain's or user's store via exmdb protocol.

6.2.2.20 Commands

6.2.2.20.1 Folder subcommand

create
: Create a new folder

delete
: Delete folder by ID or name.

find
: Find folders with given name

grant
: Grant permissions on this folder to a user

list
: List subfolders of a folder. If no folder ID is specified, list subfolders of root folder.

revoke
: Revoke permissions on this folder from a user. If not permission is specified, revoke all permissions.

6.2.2.20.2 Store subcommand

delete
: Delete properties

get
: Get store properties

set
: Set store properties

6.2.2.21 Options

ID
: ID of the folder

FOLDERSPEC
: ID or name of the folder

NAME
: Name of the folder

PARENTID
: ID of the parent folder

PERMISSION
: Name or numeric value of the permission

PROPSPEC
: Name or numeric value of the property

TARGET
: Name of the domain or e-mail address of the user

USERNAME
: E-Mail address of a user

-a, --all
: Do not stop if target is ambiguous but apply to all.

--clear
: Delete folder contents. Required for non-empty folders.

--comment COMMENT
: Folder comment

-f, --force
: Grant permissions to non-existing user

--format FORMAT
: Output format. Can be one of *csv*, *json-flat*, *json-kv*, *json-object*, *json-structured* and *pretty*. Default is *pretty*.

-r, --recursive
: Apply recursively to subfolders

--separator SEPARATOR
: String to use for column separation (*csv* and *pretty* only). Must have length 1 if format is *csv*. Default is "," for *csv* and " " for pretty.

-t TYPE, --type TYPE
: *CONTAINERCLASS* property, defaults to *"IPF.Note"*

-x, --exact
: Only match exact folder names instead of case-insensitive substrings

6.2.2.22 Notes

- Folder IDs and permissions can be given in decimal, hexadecimal (*0x*-prefix), octal (*0*-prefix) or binary (*0b*-prefix).
- Currently, the permission value echoed by the *grant* and *revoke* commands is the one sent to the server and might differ from the value actually assigned.
- The *create*, *find* and *list* commands operate on the *IPMSUBTREE* folders (*0x9* for users, *0x2* for domains) by default, which can be overridden by the *ID* parameter.

6.2.2.23 Name

grommunio-admin fetchmail — Manage fetchmail settings and generate rc file

6.2.2.24 Synopsis

grommunio-admin fetchmail create [*<FIELDS>*] *--srcPassword PASSWORD --srcServer SERVER --srcUser USER USERSPEC* [*MAILBOX*]
grommunio-admin fetchmail delete [*-y*] *MBSPEC*
grommunio-admin fetchmail list [*-f FILTER*] [*-s SORT*] [*MBSPEC*]
grommunio-admin fetchmail modify [*<FIELDS>*] *MBSPEC*
grommunio-admin fetchmail print [*-q*] *MBSPEC*
grommunio-admin fetchmail show [*--password*] *MBSPEC*
grommunio-admin fetchmail write-rc [*--force*] [*-o FILE*] [*-p*] [*-t MINUTES*] [*-v*]

6.2.2.25 Description

Subcommand to show and manipulate fetchmail entries and generate fetchmailrc file.

6.2.2.26 Commands

create
: Create a new fetchmail entry

delete
: Delete fetchmail entry

list
: List fetchmail entries

modify
: Modify fetchmail entry

print
: Print fetchmail configuration line generated by the entry

show
: Show detailed information about fetchmail entry

write-rc
: Write fetchmail configuration file (fetchmailrc)

6.2.2.27 Options

MAILBOX
: E-Mail address of the local mailbox to deliver the mails to. Defaults to e-mail address of the specified user

MBSPEC
: Mailbox prefix or ID of the fetchmail entry

USERSPEC
: Username prefix or ID of the user to attach the entry to

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

--force
: Write rc file even if no entries were changed since the last write

-o, --out-file
: Path to write configuration to. Default is */etc/fetchmailrc*

--password
: Print the source password

-p, --print
: Additionally print rc file to stdout

-q, --quiet
: Do not print additional info

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

-t, --time
: Time in minutes since the last write. Default is to autodetect by file mtime

-v, --verbose
: Be more verbose

-y, --yes
: Delete multiple entries without prompting

6.2.2.28 Fields

--active STATE
: Whether the entry is active. STATE can be one of *0*, *1*, *yes* or *no*. Default is 1

--extraOptions EXTRAOPTIONS
: Space separated list of options to write into the fetchmailrc

--fetchall STATE
: Whether to fetch mails marked as seen on the source server. STATE can be one of *0*, *1*, *yes* or *no*. Default is 0

--keep STATE
: Whether to keep fetched mails on the source server. STATE can be one of *0*, *1*, *yes* or *no*. Default is 1

--protocol PROTOCOL
: Protocol to use for fetching. Can be one of *POP3*, *IMAP*, *POP2*, *ETRN* or *AUTO*. Default is IMAP

--srcAuth AUTH
: Authentication method to use. Can be one of *password*, *kerberos_v5*, *kerberos*, *kerberos_v4*, *gssapi*, *cram-md5*, *otp*, *ntlm*, *msn*, *ssh*, *any*. Default is password

--srcFolder FOLDER
: Source folder to fetch from

--srcPassword PASSWORD
: Password of the source user. Single (') and double (") quotes are automatically removed.

--srcServer SERVER
: Source server to fetch from

--srcUser USER
: Source user to fetch mails from

--sslCertCheck STATE
: Whether to force SSL certificate check. STATE can be one of *0*, *1*, *yes* or *no*. Default is 0

--sslCertPath SSLCERTPATH
: Path to a directory containing trusted certificates or empty to use system default

--useSSL STATE
: Enable SSL

> SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.29 Name

grommunio-admin fs — Filesystem operations

6.2.2.30 Synopsis

grommunio-admin fs clean [*-d*] [*-s*] [*PARTITION*]
grommunio-admin fs du [*PARTITION*]

6.2.2.31 Description

Show space used by user and domain home directories or remove unused files.

Unused files may remain when users or domains are deleted without removing their files.

6.2.2.32 Commands

clean
: Remove directories and files that are not used by any domain or user.

du
: Show data usage statistics

6.2.2.33 Options

PARTITION
: Apply only to selected partition. Can be either *domain* or *user*

-d**,** --dryrun
: Do not delete anything, just print what would be deleted

-s**,** --nostat
: Do not collect disk usage statistics of deleted files

> SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.34 Name

grommunio-admin ldap — LDAP tools

6.2.2.35 Synopsis

grommunio-admin ldap check [*-o ORGSPEC*] [*-r* [*-m*] [*-y*]]
grommunio-admin ldap configure [*-d*] [*-o ORGSPEC*]
grommunio-admin ldap downsync [*-c*] [*-f*] [*-l*] [*-o ORGSPEC*] [*-p PAGE_SIZE*] [*USER* [*USER* ...]]
grommunio-admin ldap dump [*-o ORGSPEC*] *USER*
grommunio-admin ldap info [*-o ORGSPEC*]
grommunio-admin ldap reload [*-o ORGSPEC*]
grommunio-admin ldap search [*-a*] [*--format FORMAT*] [*-n MAX_RESULTS*] [*-o ORGSPEC*] [*-p PAGE_SIZE*] [*USER*]

6.2.2.36 Description

The grommunio admin ldap module provides functions for configuring and testing the LDAP connection and downloading or updating users.

6.2.2.37 Commands

check
: Check if the LDAP objects imported users are linked to can still be found, optionally removing orphaned users

configure
: Interactively configure or modify LDAP connection

downsync
: Synchronize or import users from LDAP

dump
: Print LDAP object

info
: Show connection status

reload
: Reload the LDAP configuration and reconnect

search
: Search for users

6.2.2.38 Options

USER
: LDAP object ID or search string

-a**,** --all
: Show all results, not only importable objects

-c**,** --complete
: Import or update all users from the LDAP tree

-f**,** --force
: Force update users that are linked to a different or no LDAP object

--format FORMAT
: Output format. Can be one of *csv*, *json-flat*, *json-kv*, *json-object*, *json-structured* and *pretty*. Default is *pretty*.

-l, --lang
: Set language for imported users. Default is to not set any language.

-m, --remove-maildirs
: Also remove user files from disk

-n MAX_RESULTS, --max-results MAX_RESULTS
: Maximum number of results or 0 to disable limit (default 0). Note that the actual number of results may exceed the limit due to paging and filtering.

-o ORGSPEC, --organization ORGSPEC
: Use organization specific LDAP connection. Supports organization ID or name.

-p PAGE_SIZE, --page-size PAGE_SIZE
: Set batch size for paged search. Can be decreased when running into timeout errors with slow LDAP servers. Default is 1000.

-r, --remove
: Remove imported users of which the linked LDAP object could not be found

-t TYPES, --types TYPES
: Comma separated list of object types to search for. Supported are *user*, *contact* and *group*.

-y, --yes
: Do not prompt for confirmation, assume yes

6.2.2.39 Name

grommunio-admin mconf — Managed configuration manipulation

6.2.2.40 Synopsis

grommunio-admin mconf dump [*-c*] *CONFIG*
grommunio-admin mconf modify *CONFIG unset KEY*
grommunio-admin mconf modify *CONFIG ACTION* [*-i* | *-b*] *KEY VALUE*
grommunio-admin mconf print *CONFIG*
grommunio-admin mconf reload *CONFIG*
grommunio-admin mconf save *CONFIG*

6.2.2.41 Description

grommunio managed configuration (mconf) offers the possibility to manipulate configuration files used by gromox.

6.2.2.42 Commands

dump
: Print configuration file that would be generated from internal state

modify
: Modify internal configuration state

print
: Print internal configuration state

reload
: Reload configuration from disk

save
: Save configuration file to disk

6.2.2.43 Options

ACTION
: Modification action:

  add - Add entry to list
  remove - Remove entry from list
  set - Add key
  unset - Remove key

CONFIG
: Configuration file, either *authmgr* or *ldap*

KEY
: Configuration key

VALUE
: Configuration value for numeric or boolean values use *-b* and *-i* respectively

-b, --bool
: Convert value to boolean, valid values are *y*, *n*, *yes*, *no*, *true*, *false*, *1*, *0*

-c, --censor
: Hide confidential information

-i, --int
: Convert value to integer, octal (*0o*) and hexadecimal (*0x*) prefixes are supported

  SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021 grommunio GmbH

6.2.2.44 Name

grommunio-admin mlist — Mailing/distribution list management

6.2.2.45 Synopsis

grommunio-admin mlist add *MLISTSPEC* (*sender*|*recipient*) *ENTRY* **grommunio-admin mlist create** [*-p PRIVILEGE*] [*-r RECIPIENT*] [*-s SENDER*] [*-t TYPE*] NAME
grommunio-admin mlist delete [*-y*] *MLISTSPEC*
grommunio-admin mlist list [*-f FIELD=<value>*] [*-s FIELD*] [*MLISTSPEC*]
grommunio-admin mlist modify [*-p PRIVILEGE*] [*-r RECIPIENT*] *MLISTSPEC* **grommunio-admin mlist remove** *MLISTSPEC* (*sender*|*recipient*) *ENTRY* **grommunio-admin mlist show**

6.2.2.46 Description

Create, modify or delete mailing lists.

6.2.2.47 Commands

add
: Add sender or recipient to list

create
: Create a new mailing list

delete
: Delete mailing list

list
: List mailing lists

modify
: Modify mailing list

remove
: Remove sender or recipient from list

show
: Show detailed information about mailing list

6.2.2.48 Options

-p PRIVILEGE, --privilege PRIVILEGE
: Set who is allowed to send mails to the list, one of *all*, *domain*, *internal*, *outgoing* or *specific*

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

-t TYPE, --type TYPE
: List type (recipient selection), one of *normal* or *domain*

6.2.2.49 Name

grommunio-admin org — Organization management

6.2.2.50 Synopsis

grommunio-admin org create [*--description DESCRIPTION*] [*--domain DOMAIN ...*] *ORGNAME*
grommunio-admin org delete *ORGSPEC*
grommunio-admin org modify [*<FIELDS>*] *ORGSPEC*
grommunio-admin org query [*-f ATTRIBUTE=<value>*] [*--format FORMAT*] [*--separator SEPARATOR*] [*-s FIELD*] [*ATTRIBUTE ...*]
grommunio-admin org show [*-f FIELD=<value>*] [*-s FIELD*] *ORGSPEC*

6.2.2.51 Description

Subcommand to show and manipulate organizations.

6.2.2.52 Commands

create
: Create a new organization

delete
: Delete an organization

modify
: Modify organization

query
: Query organization attributes

show
: Show detailed information about an organization

6.2.2.53 Options

ATTRIBUTE
: Attributes to query. Available attributes are *ID*, *name*, *description* and *domainCount*

  If no attributes are specified, *ID*, *name* and *domainCount* are shown.

ORGNAME
: Complete name of the organization

ORGSPEC
: Organization name prefix or organization ID

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

--format FORMAT
: Output format. Can be one of *csv*, *json-flat*, *json-kv*, *json-object*, *json-structured* and *pretty*. Default is *pretty*.

--separator SEPARATOR
: String to use for column separation (*csv* and *pretty* only). Must have length 1 if format is *csv*. Default is "," for *csv* and " " for pretty.

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

-y, --yes
: Assume yes instead of prompting

6.2.2.54 Fields

--description DESCRIPTION
: Description of the organization

--domain DOMAINSPEC
: Name prefix or ID of the domain to adopt. Can be given multiple times

--name ORGNAME
: Name of the organization

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021 grommunio GmbH

6.2.2.55 Name

grommunio-admin passwd — Set user password

6.2.2.56 Synopsis

grommunio-admin passwd [*-a*] [*-l LENGTH*] [*-p PASSWORD*] [*USER*]

6.2.2.57 Description

Set user password.
If no user is specified, the password is set for the *admin* user, which is created automatically if necessary.
If neither *-a* nor *-p* is provided, the user is prompted for a password.

6.2.2.58 Options

USER
: User to set password for (default *admin*)

-a, --auto
: Automatically generate a password

-l LENGTH, --length LENGTH
: Length of the automatically generated password (default 16)

-p PASSWORD, --password PASSWORD
: Password to set (do not prompt)

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.59 Name

grommunio-admin run — Start a stand-alone HTTP server

6.2.2.60 Synopsis

grommunio-admin run [*-d*] [*-i IP*] [*--no-config-check*] [*-p PORT*]

6.2.2.61 Description

Run REST API in a stand-alone HTTP server.

——DO NOT USE IN PRODUCTION!——
This command is intended for development and testing. A production instance should use an external WSGI server like *uwsgi*.

6.2.2.62 Options

-d**,** --debug
: Enable debug mode

-i IP**,** --ip IP
: Host address to bind to (default ::)

--no-config-check
: Skip configuration check

-p PORT**,** --port PORT
: Host port to bind to (default 5001)

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2022 grommunio GmbH

6.2.2.63 Name

grommunio-admin server — Multi-server management

6.2.2.64 Synopsis

grommunio-admin server create *-H HOSTNAME -e EXTNAME*]
grommunio-admin server delete *SERVERSPEC*
grommunio-admin server list [*-f FIELD=<value>*] [*-s FIELD*] [*SERVERSPEC*]
grommunio-admin server modify [*<FIELDS>*] *SERVERSPEC*
grommunio-admin server show [*-f FIELD=<value>*] [*-s FIELD*] *SERVERSPEC*

6.2.2.65 Description

Subcommand to show and manipulate server entries.

If at least one server is specified, newly created users and domains will be associated with one of the servers. The destination server may be specified explicitly, or is chosen automatically according to *options.serverPolicy*.

6.2.2.66 Commands

create
: Register a new server

delete
: Soft-delete a server

list
: List domains

modify
: Modify server

show
: Show detailed information about a server

6.2.2.67 Options

SERVERSPEC
: Server hostname or ID

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

6.2.2.68 Fields

-H HOSTNAME, --hostname HOSTNAME
: Internal hostname of the server

`-e EXTNAME, --extname EXTNAME
: External hostname (e.g. FQDN) of the server.

6.2.2.69 Name

grommunio-admin service — grommunio-admin external service interface control

6.2.2.70 Synopsis

grommunio-admin service [*-r*] *load SERVICE* [*ARGS* ...]
grommunio-admin service [*-v*] *status* [*SERVICE* [*SERVICE* [...]]]

6.2.2.71 Description

grommunio-admin connects to several external services to either provide means of configuration via API (e.g. grommunio chat) or to retrieve additional information (e.g. LDAP).
grommunio-admin service can be used to introspect the connection status of these services.
Note that the CLI runs separately from the API backend. If introspection of the running server instance is required, use the *connect* command to access the server instance.
As of version 1.9, each service acts as a blueprint for parameterized instances. Currently only the LDAP service supports parameters, allowing for organization-specific ldap connections.
Each instance has a state, reflecting the connection status. The following states are used:

UNLOADED
: The service has not been loaded yet. It will be loaded automatically when needed.

LOADED
: The service has been initialized successfully.

UNAVAILABLE
: An error occurred that indicates that the service is not available, but might become available in the future. No reload is necessary to reconnect.

SUSPENDED
: Àn error occurred that indicates that the service is not available, but might become available in the future. The service will be reloaded automatically on next usage.

ERROR
: The service is not available either because initialization failed or because to man errors occurred. It will remain unavailable until reloaded manually.

DISABLED
: The service has been manually disabled (either by configuration or command).

6.2.2.72 Commands

6.2.2.72.1 load

Load or reload services.
Only services in UNLOADED or SUSPENDED state will be affected unless the *--reload* option is given.

6.2.2.72.2 status

Show status of all services.

6.2.2.73 Options

SERVICE
: Name of the service.

-r, --reload
: Force reload of service.

-v, --verbose
: Show more information.

6.2.2.74 Services

The following services are currently connected via the service interface:

chat
: grommunio chat. Connected via REST interface.

exmdb
: gromox exmdb provider (gromox-http). Connected via custom TCP protocol.

ldap
: External LDAP service. Connected via LDAP(s).

redis
: Redis instance (used by grommunio sync). Connected via redis driver (TCP).

systemd
: Systemd shell execution.

6.2.2.75 Name

grommunio-admin shell — Start interactive shell

6.2.2.76 Synopsis

grommunio-admin shell [*-d*] [*-n*] [*-x*]

6.2.2.77 Description

The interactive shell mode allows execution of multiple (new line separated) commands in a single session. Command syntax is identical to the CLI arguments, with addition of the *exit* command which ends the interactive shell.

If possible, typed history will be saved in *~/.grommunio-admin.history*.

6.2.2.78 Options

-d, --debug
: Enable more verbose debug output

-n, --no-history
: Disable loading/saving of the typed history

-x, --exit
: Exit immediately if a command results in a non-zero exit code

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.79 Name

grommunio-admin taginfo — Show information about proptags

6.2.2.80 Synopsis

grommunio-admin taginfo TAG [TAG ...]

6.2.2.81 Description

Display information about a property tag, as defined in the Microsoft Exchange Server Protocols Master Property List.
Note that property names used by grommunio may differ from the names defined by Microsoft.

6.2.2.82 Options

TAG
: Decimal or hexadecimal (with 0x prefix) Tag ID or grommunio tag name glob

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021-2022 grommunio GmbH

6.2.2.83 Name

grommunio-admin user — User management

6.2.2.84 Synopsis

grommunio-admin user create [*--no-defaults*] [*<FIELDS>*] *USERNAME*
grommunio-admin user delegate *USERSPEC* (*clear* | *list*)
grommunio-admin user delegate *USERSPEC* (*add* | *remove*) *USERNAME* ...
grommunio-admin user delete [*-c*] [*-k*] [*-y*] *USERSPEC*
grommunio-admin user devices *USERSPEC* (*list* | *resync* | *remove* | *show*) [*DEVICE* ...]
grommunio-admin user devices *USERSPEC* wipe [*--mode MODE*] *DEVICE*
grommunio-admin user list [*-f ATTRIBUTE=<value>*] [*-s FIELD*] [*USERSPEC*]
grommunio-admin user login [*--nopass*] [*--password PASSWORD*] [*--token*] *USERNAME*
grommunio-admin user modify [*<FIELDS>*] [*--delete-chat-user*] [*--no-ldap*] [*--remove-alias ALIAS*] [*--remove-altname ALTNAME*] [*--remove-property PROPSPEC*] [*--remove-storeprop PROPSPEC*] *USERSPEC*

grommunio-admin user query [*-f ATTRIBUTE=<value>*] [*--format FORMAT*] [*--separator SEPARATOR*] [*-s FIELD*] [*ATTRIBUTE ...*]
grommunio-admin user sendas *USERSPEC* (*clear* | *list*)
grommunio-admin user sendas *USERSPEC* (*add* | *remove*) *USERNAME ...*
grommunio-admin user show [*-f ATTRIBUTE=<value>*] [*-s FIELD*] *USERSPEC*

6.2.2.85 Description

Subcommand for user management.

6.2.2.86 Commands

create
: Create a new user

delegate
: Manage delegate permission

delete
: Delete user

devices
: User mobile device management

list
: List users **Deprecated.** Use query instead.

login
: Test user login

modify
: Modify a user

query
: Query user attributes

sendas
: Manage send-as permission

show
: Show detailed information about a user

6.2.2.87 Options

ATTRIBUTE
: Attributes to query. Available attributes are *ID*, *aliases*, *changePassword*, *chat*, *chatAdmin*, *domainID*, *forward*, *homeserverID*, *lang*, *ldapID*, *maildir*, *pop3_imap*, *privArchive*, *privChat*, *privFiles*, *privVideo*, *privWeb*, *privDav*, *privEas*, *publicAddress*, *smtp*, *status* and *username*.

  If no attributes are specified, *ID*, *username* and *status* are shown.

DEVICE
: Limit command to given device ID(s)

USERNAME
: E-Mail address of the user

USERSPEC
: User name prefix or user ID

-c, --keep-chat
: Deactivate but do not permanently delete chat user

--delete-chat-user
: Permanently delete chat user

-f FIELD=<value>, --filter FIELD=<value>
: Filter expression in the form of 'field=value'. Can be specified multiple times to refine filter

--format FORMAT
: Output format. Can be one of *csv*, *json-flat*, *json-kv*, *json-object*, *json-structured* and *pretty*. Default is *pretty*.

-k, --keep-files
: Do not delete user files from disk

--mode MODE
: Specify wipe status to set. Possible values are *account* and *normal*, or *cancel* to stop a pending wipe.

--no-defaults
: Do not apply configured default values

--no-ldap
: Detach user from LDAP object

--nopass
: Skip password check

--password
: User password. If omitted, password is retrieved from prompt.

--remove-alias ALIAS
: Remove ALIAS from user (can be given multiple times)

--remove-altname ALTNAME
: Remove ALTNAME from user (can be given multiple times)

--remove-property PROPSPEC
: Remove property from user (can be given multiple times)

--remove-storeprop PROPSPEC
: Remove property from user's store (can be given multiple times)

--separator SEPARATOR
: String to use for column separation (*csv* and *pretty* only). Must have length 1 if format is *csv*. Default is "," for *csv* and " " for pretty.

-s FIELD, --sort FIELD
: Sort by field. Can be given multiple times

--token
: Generate access and CSRF token on successful login

-y, --yes
: Assume yes instead of prompting

6.2.2.88 Fields

--changePassword <bool>
: Whether the user can change the password

--chat <bool>
: Whether to create a chat user

--chatAdmin <bool>
: Whether the user has chat admin privileges

--homeserver ID
: ID of the home server or 0 for local user

--lang LANG
: User store language

--ldapID LDAPID
: Identifier of the LDAP object linked to the user

--pop3-imap <bool>
: Whether the user has the POP3/IMAP privilege

--privArchive <bool>
: Whether the user has the archiving privilege

--privChat <bool>
: Whether the user has the chat privilege

--privFiles <bool>
: Whether the user has the files privilege

--privVideo <bool>
: Whether the user has the video privilege

--privWeb <bool>
: Whether the user has the web privilege

--privDav <bool>
: Whether the user has the DAV privilege

--privEas <bool>
: Whether the user has the EAS privilege

--public-address <bool>
: Whether the user has the public address privilege

--smtp <bool>
: Whether the user has the SMTP privilege

--status STATUS
: User address status. Either numeric value or one of *normal*, *suspended*, *deleted* or *shared*.

--alias ALIAS
: Add alias

--altname ALTNAME
: Add ALTNAME to user alternative login name list (can be given multiple times)

--property propspec=value
: Set property defined by propspec to value

--storeprop propspec=value
: Set store property defined by propspec to value

--username
: Rename user

SPDX-License-Identifier: CC-BY-SA-4.0 or-later SPDX-FileCopyrightText: 2021 grommunio GmbH

6.2.2.89 Name

grommunio-admin version — Show backend and/or API version

6.2.2.90 Synopsis

grommunio-admin version [*-a*] [*-b*] [*-c*]

6.2.2.91 Description

Show the current version of the API (specification) or the backend (code).
The combined mode (default) appends the difference between between backend and API version at the end of the API version.

If multiple options are given, each requested version is printed on a separate line. The order is always API – backend – combined.

6.2.2.92 Options

-a, --api
: Print API version

-b, --backend
: Print backend version

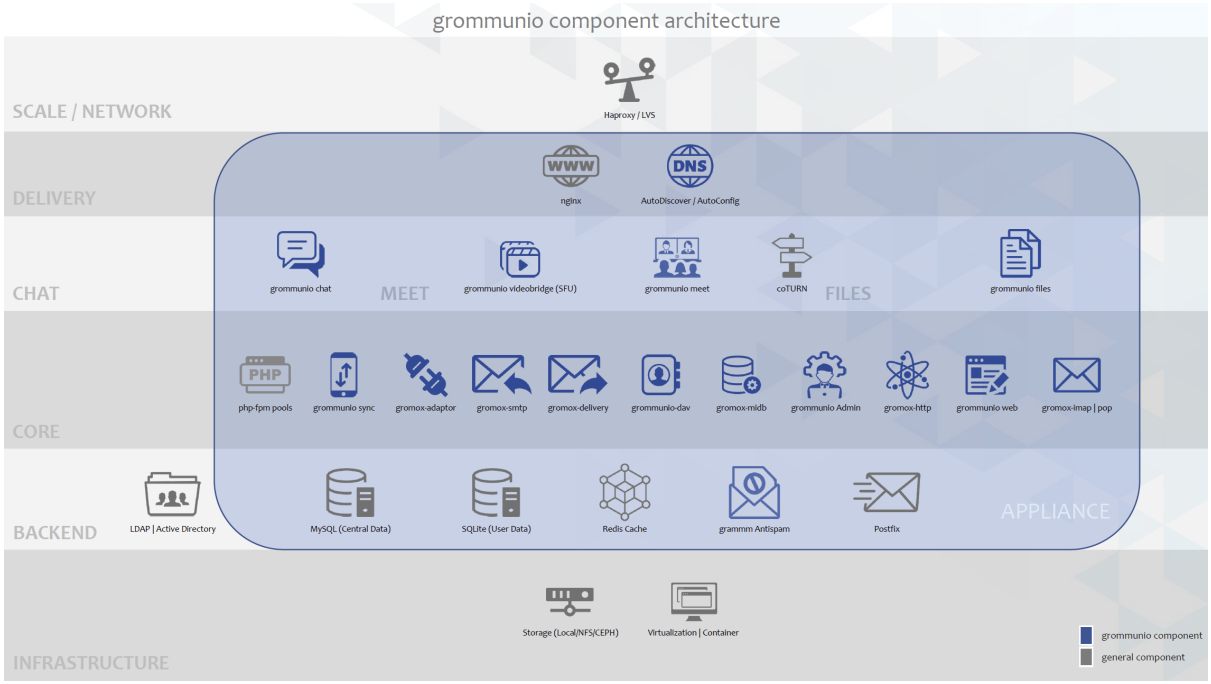
-c, --combined
: Print combined version

CHAPTER 7

Architecture

7.1 Component architecture

While grommunio is distributed as appliances, grommunio is also available as packaged software. The packaging is oriented on the modular component structure of the software modules available by grommunio. The modular component layout of grommunio allows a component-based deployment for all sorts of deployments and operations - ranging from single deployments with high-density component layout to scale-out, distributed component layout.



The grommunio component layout supports a wide range of deployment types:

- Containers (Docker, LXC)
- Virtual machines (KVM, VMware, Hyper-V)

The appliances shipped by grommunio contain all components required for operation by the use of

packages which are available for updates via repositories. The appliances ship these packages as part of the appliance distribution to be able to operate the installation without external repository activation necessary (whilst deploying with active internet connection for updates at deployment is strongly encouraged).

To understand the component architecture and the interconnectivity of these components, the following chapters show the single components and how they interoperate with other components in the entire component stack.

7.2 Multi-Server architecture

Scaling grommunio across multiple instances ("Multi-Server") requires a fundamental understanding of scale-out solutions. For most users, the Single Point of Entry (SPOE) approach is preferable, as it eliminates the need to remember which specific server their profile resides on. However, in large-scale environments with tens of thousands of users, even the most advanced server systems require distribution across multiple nodes to handle the load efficiently.

The role of Autodiscover, Autoconfig and Load Balancers

Autodiscover, in combination with load balancers, provides a unified access point, ensuring that users can seamlessly connect to their mailboxes without concern for backend distribution. To make this work, the system must know where each user's data is stored—whether they are using an IMAP client or the web interface.

When a request arrives at a load balancer, it may be directed to Node A, even though the user's data resides on Node B. Depending on the protocol in use, the component handling the request will either:

Proxy the request to the appropriate backend node, or access the data directly through RPC calls (exmdb traffic).

7.2.1 Setting up Multi-Server

To ensure smooth operation across multiple nodes, the following configurations must be in place:

- Defining Servers in grommunio Admin:
  - The internal hostname should reflect the actual hostname of the server.
  - The external hostname should match the name communicated in client requests (e.g., for AutoDiscover).
- User Assignment:
  - Users must be associated with specific servers. If multiple servers are configured, the system's selection policy will determine automatic placement.
- Network Service Configuration:
  - Core services must be configured to listen for network requests based on the architecture.

Important: Functional redirects require that internal hostnames remain accessible even in a proxied environment. Load balancers must respond to both internal and external hostnames. Additionally, inter-component traffic must resolve correctly via internal hostnames. A proper TLS certificate setup is critical for secure traffic exchange between components. A multi-SAN or wildcard certificate is recommended.

Important: Depending on your environment, additional tuning may be required. Parameters like notify_stub_threads_num or rpc_proxy_connection_num should be adjusted based on your specific

scale and workload. Refer to the man pages for details on these settings.

7.2.2 Shared storage

To configure gromox for multi-server operation, follow these steps for a three-node setup (with IPv4 traffic enabled):

Configuration Files

```
# /etc/gromox/exmdb_provider.cfg
exmdb_hosts_allow=:: ::ffff:10.10.10.20 ::ffff:10.10.10.30 ::ffff:10.10.10.40 ::1
listen_ip=::

# /etc/gromox/exmdb_list.txt
/var/lib/gromox/user/mail1.gro.at/ private ::ffff:10.10.10.20 5000
/var/lib/gromox/user/mail2.gro.at/ private ::ffff:10.10.10.30 5000
/var/lib/gromox/user/mail3.gro.at/ private ::ffff:10.10.10.40 5000
/var/lib/gromox/domain/ public ::ffff:10.10.10.20 5000

# /etc/gromox/midb.cfg
midb_hosts_allow=:: ::ffff:10.10.10.20 ::ffff:10.10.10.30 ::ffff:10.10.10.40 ::1
midb_listen_ip=::

# /etc/gromox/midb_list.txt
/var/lib/gromox/user/mail1.gro.at ::ffff:10.10.10.20 5555
/var/lib/gromox/user/mail2.gro.at ::ffff:10.10.10.30 5555
/var/lib/gromox/user/mail3.gro.at ::ffff:10.10.10.40 5555
/var/lib/gromox/domain/ ::ffff:10.10.10.20 5555

# /etc/gromox/event.cfg
event_hosts_allow=:: ::ffff:10.10.10.20 ::ffff:10.10.10.30 ::ffff:10.10.10.40 ::1
event_listen_ip=::

# /etc/gromox/timer.cfg
timer_hosts_allow=:: ::ffff:10.10.10.20 ::ffff:10.10.10.30 ::ffff:10.10.10.40 ::1
timer_listen_ip=::
```

Important: Please note that the *.txt files include the ACLs and are named as such (not *.cfg)

Important: Please make sure that your firewall configuration enables the additional ports used: exmdb(tcp/5000), midb (5555/tcp), timer (6666/tcp) and event (33333/tcp).

Central Admin API control

Since using multiple servers requires some logic also for the mailbox store creation process, using the topology of Share-nothing clusters still would require the gromox-mbop mailbox creation process to be able to access the nodes storage.

Generally, in multi-server environments, it is recommended to use the hostname prefixing technique as outlined in the above section. This way, the mailbox is ”pinned” as per its directory to the mailbox directory it has defined. This only means that this mailbox is associated with the hostname (primarily), however the real ”processing” hostname is defined by the user<>mailbox relation.

To create the mailboxes with the hostname-included pathname, this setting is required for grommunio-admin API to create the mailbox appropriately:

```
options:
  serverExplicitMount: true
```

Ideally, this configuration block would be included in the grommunio Admin API config tree, for example as a new yaml under /etc/grommunio-admin-api/conf.d/multiserver.yaml.

Important: Just because your nodes have shared access to all the nodes configured does not mean that the applications serving the mailboxes are strictly tied to the hostname. The effective processing is determined by the relationship of the user<>server association.

Storage Structure

This setup distributes user directories across multiple nodes while maintaining a unified logical structure. Traditionally, shared-storage clusters use a clustered filesystem, ensuring efficient replication by storing different user directories as separate inode entries. This reduces filesystem load and improves performance.

Note: Before the release of 2025.01.1 it was required for all nodes to have access to /var/lib/gromox/user in this example, because some components (especially IMAP and POP3) were accessing the object files via direct IO.

Note: Depending on the type of setup for the Admin API, it might still be viable to have shared storage access available. If the Admin API is a headless node (for example), you might want to have access to the storage so that the API is able to create the stores for you.

7.2.3 Share-nothing clusters

With grommunio 2025.01.1, components no longer require direct I/O access to mailbox storage. Instead, requests are handled as follows:

- If a request does not belong to the local node, the system will:
  - Retrieve data via RPC from the node where the mailbox resides, or
  - Use proxy mode to forward the request and return the response.

Key Benefits of Share-Nothing Clusters

- No shared storage required: Each node operates independently, eliminating the need for a common filesystem.
- Improved high availability: Nodes can be distributed across different locations without centralized storage dependencies.
- Compatibility with modern HA solutions:

This architecture natively supports cloud-native environments such as Kubernetes, other containerized environments, and various replication techniques for failover scenarios.

From a configuration standpoint, share-nothing clusters remain identical to shared-storage setups, except that nodes do not need access to each other's mailboxes. Production deployments may benefit from additional replication techniques for high availability.

7.2.4 Failover

A cluster can also have strictly high availability requirements (e.g. five nines >99.999%). This level of availability does require some cluster suite software to be able to failover in these second-level failover switches. For implementing such scenarios covered by the enterprise subscription, the workflow is roughly:

- Detect application/container/VM fail
- Activate standy application/container/VM
- Switch over application-level requests to new standby system

Failing over the entire cluster-stack of grommunio requires in its core a simple mysql-query to be executed as well as a reload signal (SIGHUP) to main services (gromox-*). As per example:

```
UPDATE users SET homeserver=8 WHERE homeserver=4;
```

and a subsequent following reload signal to reload any application caches.

All of this can be well controlled by well-established cluster services, such as Pacemaker or similar products.

Switching "back" to normal operation can be done by for example re-balancing all users to all available nodes in your installation:

```
-- prepare environment
SET @rownum = 0;
SELECT COUNT(*) INTO @server_count FROM servers;

-- reassign homeservers evenly
UPDATE users
JOIN (
    SELECT
        maildir,
        (@rownum := @rownum + 1) AS rownum
    FROM users
    WHERE homeserver != 0
    ORDER BY maildir
) AS u_ordered
ON users.maildir = u_ordered.maildir
SET users.homeserver = (u_ordered.rownum - 1) % @server_count + 1
WHERE users.homeserver != 0;
```

and again, followed by a subsequent reload of the core services (gromox-*).

7.3 Protocol / Component Flow

grommunio is a comprehensive communication and collaboration solution that covers and delivers protocols with a vast variety of computer standards for communication. The main protocols delivered by grommunio are:

Wire-level protocols:

- SMTP
- IMAP
- POP3

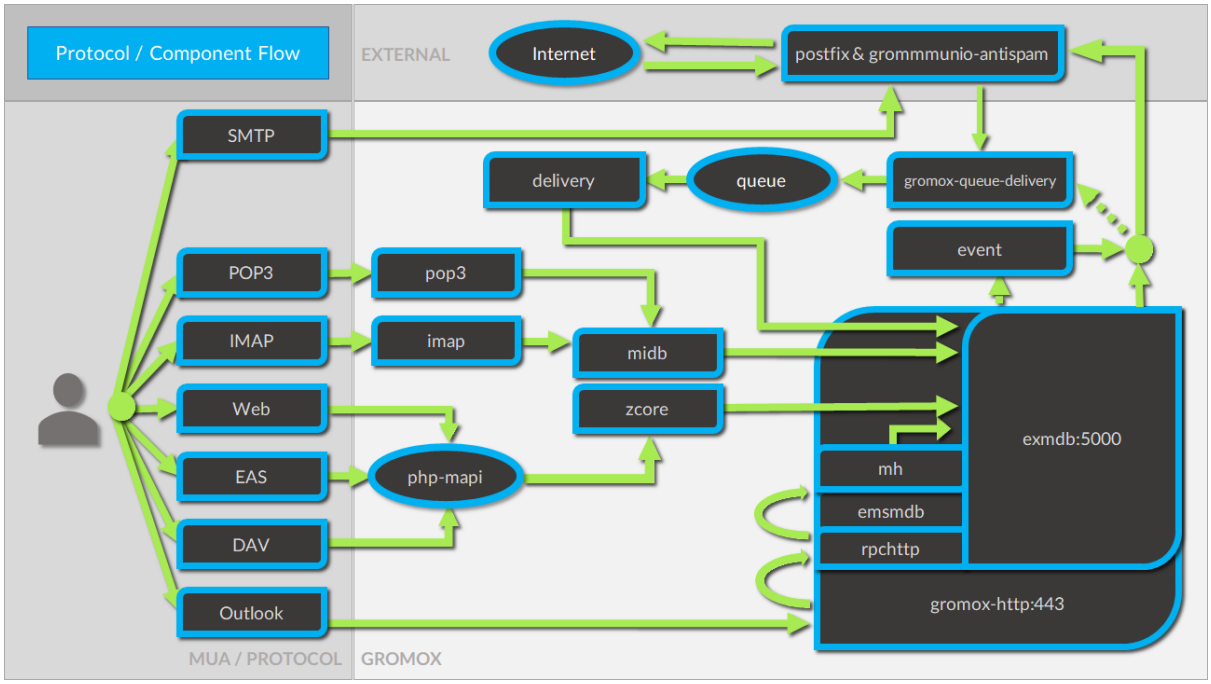
Application-level protocols (HTTP-based):

- RPC/HTTP (OutlookAnywhere)

- MAPI/HTTP
- EWS (Exchange Web Services)
- EAS (Exchange ActiveSync)
- CalDAV
- CardDAV

With these numerous protocols available, grommunio needs to have an effictient component flow. Since protocols may be accessed in parallel for the same dataset, grommunio takes care of parallelization and protocol tracking. To ensure operation, security and functionality, grommunio uses a set of different components as well as a plugin-based structure for larger components. This way, components may be extended for future feature expansion and allows nearly-realtime patches and updates. More complex setups gain from the component/plugin architecture as the scalability of the components allow various flavors of containerization and orchestration.

The following illustration shows the combined protocol and component flow for grommunio Groupware based components:



7.3.1 Proxy capabilities

By default, grommunio's HTTP-based services are exposed through nginx. This recommended mode of operation adds an additional layer of protection for gromox's components, as nginx validates incoming HTTP requests before they are processed by gromox. The internal nginx proxy configuration is not designed (nor required) to horizontally scale requests; instead, grommunio supports load balancers placed in front of it. These load balancers effectively serve as reverse proxies with built-in load balancing logic. In such cases, it is advisable to use a separate proxy in front of any services provided by grommunio.

When gromox needs to process requests for a different node it is running on, the internal exmdb logic code comes into play and forwards the traffic to the appropriate node.

Grommunio supports various load balancers capable of handling tens of thousands of connections per node. Since each installation may have unique configuration requirements, the following sections aim to provide a foundation and inspire custom setups. Please note that there are various extra options not directly covered which are provided by other load balancers as well, such as NGINX Plus, KEMP and/or others.

Important: Please use these configuration sections as mere inspiration for a template of your own requirements. These examples do not claim to be complete in any way, as for example the forwarding of POP3 and IMAP are not available and your individual installation requirements might vary. The below shows an example with a distributed setup for users of a ~75k user environment. Also, these configuration files do not take specialized OpenID Connect or 2FA installations into account.

7.3.1.1 HAPROXY

```
global
 chroot /var/lib/haproxy
 daemon
 log /dev/log local0

 group haproxy
 user haproxy

 maxconn 80000
 stats timeout 30s
 ulimit-n 165000

 ca-base /etc/ssl/certs
 crt-base /etc/ssl/private
 ssl-default-bind-ciphers AES128-GCM-SHA256:AES128-SHA:AES128-SHA256:AES256-GCM-
↪SHA384:AES256-SHA:AES256-SHA256:DES-CBC3-SHA:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-
↪AES128-SHA256:DHE-RSA-AES256-GCM-SHA384:DHE-RSA-AES256-SHA256:DHE-RSA-CHACHA20-
↪POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES128-SHA:ECDHE-ECDSA-
↪AES128-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-ECDSA-AES256-SHA:ECDHE-ECDSA-
↪AES256-SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-
↪RSA-AES128-SHA:ECDHE-RSA-AES128-SHA256:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-RSA-
↪AES256-SHA:ECDHE-RSA-AES256-SHA384:ECDHE-RSA-CHACHA20-POLY1305:TLS_AES_128_
↪GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
 ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets
 tune.ssl.default-dh-param 2048

defaults
 log global
 mode http
 option httplog
 option dontlognull

 retries 3
 timeout connect 5s
 timeout queue 30s
 timeout client 300s
 timeout server 300s

frontend fe_http

 bind :80
 http-response set-header Strict-Transport-Security max-age=31536000
 http-response set-header X-Content-Type-Options nosniff
 http-response set-header X-Forwarded-Proto https
 http-response set-header X-Frame-Options SAMEORIGIN

 acl whitelist-ip src -f /etc/haproxy/ha_whitelist_main.txt
 http-request silent-drop if HTTP_1.0
 acl blacklist-ip src -f /etc/haproxy/ha_blacklist_main.txt
 http-request deny if blacklist-ip
```

(continues on next page)

(continued from previous page)

```
 mode http
 maxconn 80000

 bind *:443 ssl crt /etc/haproxy/proxy.pem alpn h2,http/1.1
 no option httpclose
 option forwardfor
 redirect scheme https code 301 if !{ ssl_fc }

 # bind quic4@:443 ssl crt /etc/haproxy/proxy.pem alpn h3
 # http-after-response add-header alt-svc 'h3=":443"; ma=60'

 acl fe_haproxy hdr(host) -i mail.grommunio.at
 acl admin dst_port 8443
 acl auth path_beg /auth
 acl autodiscover path_beg -i /autodiscover
 acl chat path_beg /chat
 acl colibri path_beg /colibri-ws
 acl dav path_beg /dav
 acl default path_beg /
 acl eas path_beg /Microsoft-Server-ActiveSync
 acl ews path_beg /EWS
 acl files path_beg /files
 acl hdr_connection_upgrade hdr(Connection) -i upgrade
 acl hdr_upgrade_websocket hdr(Upgrade) -i websocket
 acl mapi path_beg /mapi
 acl meet path_beg /meet
 acl oab path_beg /OAB
 acl office path_beg /office
 acl rpc path_beg /rpc/rpcproxy.dll
 acl web path_beg /web

 use_backend be_adminnodes if admin fe_haproxy
 use_backend be_authnodes if auth fe_haproxy
 use_backend be_chatnodes if chat fe_haproxy
 use_backend be_filesnodes if files fe_haproxy
 use_backend be_gromoxnodes if autodiscover
 use_backend be_gromoxnodes if ews fe_haproxy
 use_backend be_gromoxnodes if mapi fe_haproxy
 use_backend be_gromoxnodes if rpc fe_haproxy
 use_backend be_meetnodes if colibri fe_haproxy
 use_backend be_meetnodes if hdr_connection_upgrade hdr_upgrade_websocket meet fe_haproxy
 use_backend be_meetnodes if meet fe_haproxy
 use_backend be_officenodes if office fe_haproxy
 use_backend be_webnodes if dav fe_haproxy
 use_backend be_webnodes if default fe_haproxy
 use_backend be_webnodes if eas fe_haproxy
 use_backend be_webnodes if web fe_haproxy

frontend fe_imaps
 mode tcp
 option tcplog
 bind :993 name imaps
 acl blocklist-imap src -f /etc/haproxy/ha_blacklist_imap.txt
 tcp-request connection reject if blocklist-imap
 default_backend be_imaps

frontend fe_pop3s
 mode tcp
 option tcplog
 bind :995 name pop3s
```

(continues on next page)

(continued from previous page)

```
 acl blocklist-pop3s src -f /etc/haproxy/ha_blacklist_pop3.txt
 tcp-request connection reject if blocklist-pop3s
 default_backend be_pop3s

frontend fe_smtp
 mode tcp
 option tcplog
 bind :25 name smtp
 acl blocklist-smtp src -f /etc/haproxy/ha_blacklist_smtp.txt
 tcp-request connection reject if blocklist-smtp
 default_backend be_smtp

frontend fe_submission
 mode tcp
 option tcplog
 bind :587 name submission
 acl blocklist-submission src -f /etc/haproxy/ha_blacklist_submission.txt
 tcp-request connection reject if blocklist-submission
 default_backend be_submission

frontend fe_admin
 mode http
 option httplog
 option forwardfor
 bind *:8443 ssl crt /etc/haproxy/proxy.pem alpn h2,http/1.1
 acl whitelist-admin src -f /etc/haproxy/ha_whitelist_admin.txt
 http-request deny if !whitelist-admin
 default_backend be_adminnodes

backend be_gromoxnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option redispatch
 server gromox01 mail01.grommunio.at:443 check ssl verify none
 server gromox02 mail02.grommunio.at:443 check ssl verify none
 server gromox03 mail03.grommunio.at:443 check ssl verify none
 server gromox04 mail04.grommunio.at:443 check ssl verify none
 server gromox05 mail05.grommunio.at:443 check ssl verify none

backend be_chatnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server chat01 chat01.grommunio.at:443 check ssl verify none
 server chat02 chat02.grommunio.at:443 check ssl verify none

backend be_webnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server web01 web01.grommunio.at:443 check ssl verify none
 server web02 web02.grommunio.at:443 check ssl verify none
```

(continues on next page)

(continued from previous page)

```
backend be_meetnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance url_param room
 hash-type consistent
 option forwardfor
 option http-server-close
 option redispatch
 server meet01 meet01.grommunio.at:443 check ssl verify none
 server meet02 meet02.grommunio.at:443 check ssl verify none

backend be_filesnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server files01 files01.grommunio.at:443 check ssl verify none
 server files02 files02.grommunio.at:443 check ssl verify none

backend be_officenodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server office01 office01.grommunio.at:443 check ssl verify none

backend be_authnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server auth01 auth01.grommunio.at:443 check ssl verify none

backend be_adminnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
 option http-server-close
 option redispatch
 server admin01 admin01.grommunio.at:8443 check ssl verify none

backend be_imaps
 stick-table type ip size 10240k expire 60m
 mode tcp
 balance source
 stick on src
 server imap01 classic01.grommunio.at:993 check
 server imap02 classic02.grommunio.at:993 check

backend be_pop3s
 stick-table type ip size 10240k expire 60m
 mode tcp
 balance source
 stick on src
```

(continues on next page)

(continued from previous page)

```
 server pop01 classic01.grommunio.at:995 check
 server pop02 classic02.grommunio.at:995 check

backend be_smtp
 mode tcp
 balance source
 server smtp01 classic01.grommunio.at:25 send-proxy
 server smtp02 classic02.grommunio.at:25 send-proxy

backend be_submission
 mode tcp
 balance source
 server submission01 classic01.grommunio.at:587 send-proxy
 server submission02 classic02.grommunio.at:587 send-proxy
```

7.3.1.2 NGINX

Please note that this configuration does not cover other relevant settings from nginx in a large scale-out installation, please consult nginx manual of certain sclability related configuration directives, for example (but not limited to) *worker_processes*.

The optimal value depends on many factors including the the number of available CPU cores, the load pattern and more. When in doubt, setting the number of available CPU cores is a good starting point.

```
upstream be_smtp {
 server classic01.example.com:25;
 server classic02.example.com:25;
}

upstream be_submission {
 server classic01.example.com:587;
 server classic02.example.com:587;
}

upstream be_imaps {
 server classic01.example.com:993;
 server classic02.example.com:993;
}

upstream be_pop3s {
 server classic01.example.com:995;
 server classic02.example.com:995;
}

upstream be_gromoxnodes {
 server mail01.grommunio.at:443;
 server mail02.grommunio.at:443;
 server mail03.grommunio.at:443;
 server mail04.grommunio.at:443;
 server mail05.grommunio.at:443;
}

upstream be_adminnodes {
 server admin01.grommunio.at:8443;
}

upstream be_archivenodes {
 server archive01.grommunio.at:443;
}
```

(continues on next page)

(continued from previous page)

```
upstream be_chatnodes {
 server chat01.grommunio.at:443;
 server chat02.grommunio.at:443;
}

upstream be_webnodes {
 server web01.grommunio.at:443;
 server web02.grommunio.at:443;
}

upstream be_filesnodes {
 server files01.grommunio.at:443;
 server files02.grommunio.at:443;
}

upstream be_officenodes {
 server office01.grommunio.at:443;
}

upstream be_meetnodes {
 server meet01.grommunio.at:443;
 server meet02.grommunio.at:443;
}

upstream be_authnodes {
 server auth01.grommunio.at:443;
}

stream {
 server {
  listen 25;
  proxy_pass be_smtp;
 }
 server {
  listen 587;
  proxy_pass be_submission;
 }
 server {
  listen 993;
  proxy_pass be_imaps;
 }
 server {
  listen 995;
  proxy_pass be_pop3s;
 }
}

server {
 listen 80;
 listen [::]:80;

 server_name _;

 error_log /var/log/nginx/error.log;
 access_log /var/log/nginx/access.log;

 return 301 https://$server_name$request_uri;
}

server {
```

(continues on next page)

(continued from previous page)

```
listen 443 ssl http2;
listen [::]:443 ssl http2;
# listen 443 quic reuseport;
# listen [::]:443 quic reuseport;

server_name _;

ssl_certificate /etc/nginx/proxy.pem;
ssl_certificate_key /etc/nginx/proxy.key;
include ssl_params;

error_log /var/log/nginx/error.log;
access_log /var/log/nginx/access.log;

charset utf-8;

proxy_buffers 4 256k;
proxy_buffer_size 128k;
proxy_busy_buffers_size 256k;
proxy_http_version 1.1;
proxy_pass_header Authorization;
proxy_pass_header Date;
proxy_pass_header Server;
proxy_pass_request_headers on;
proxy_read_timeout 3h;
proxy_read_timeout 60s;

more_set_input_headers 'Authorization: $http_authorization';
more_set_headers -s 401 'WWW-Authenticate: Basic realm="mail.grommunio.at"';
proxy_set_header Accept-Encoding "";
proxy_set_header Connection "Keep-Alive";
proxy_set_header Host $host;
proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
proxy_set_header X-Forwarded-Proto $scheme;
proxy_set_header X-Real-IP $remote_addr;

client_max_body_size 0;

location ~* /admin { proxy_pass https://be_adminnodes; }
location ~* /auth { proxy_pass https://be_authnodes; }
location ~* /antispam { proxy_pass https://be_adminnodes/antispam; }
location ~* /archive { proxy_pass https://be_gromoxnodes/archive; }
location ~* /autodiscover { proxy_pass https://be_gromoxnodes/Autodiscover; }
location ~* /colibri-ws { proxy_pass https://be_meetnodes/meet; }
location ~* /chat { proxy_pass https://be_chatnodes/chat; }
location ~* /EWS { proxy_pass https://be_gromoxnodes/EWS; }
location ~* /files { proxy_pass https://be_filesnodes/files; }
location ~* /mapi { proxy_pass https://be_gromoxnodes/mapi; }
location ~* /meet { proxy_pass https://be_meetnodes/meet; }
location ~* /office { proxy_pass https://be_officenodes/office; }
location ~* /Microsoft-Server-ActiveSync { proxy_pass https://be_webnodes/Microsoft-Server-ActiveSync; }
location ~* /oab { proxy_pass https://be_gromoxnodes/OAB; }
location ~* /Rpc { proxy_pass https://be_gromoxnodes/Rpc; }
location ~* /web { proxy_pass https://be_webnodes/web; }

location / { proxy_pass https://be_gromoxnodes/; }
}
```

7.3.2 SMTP

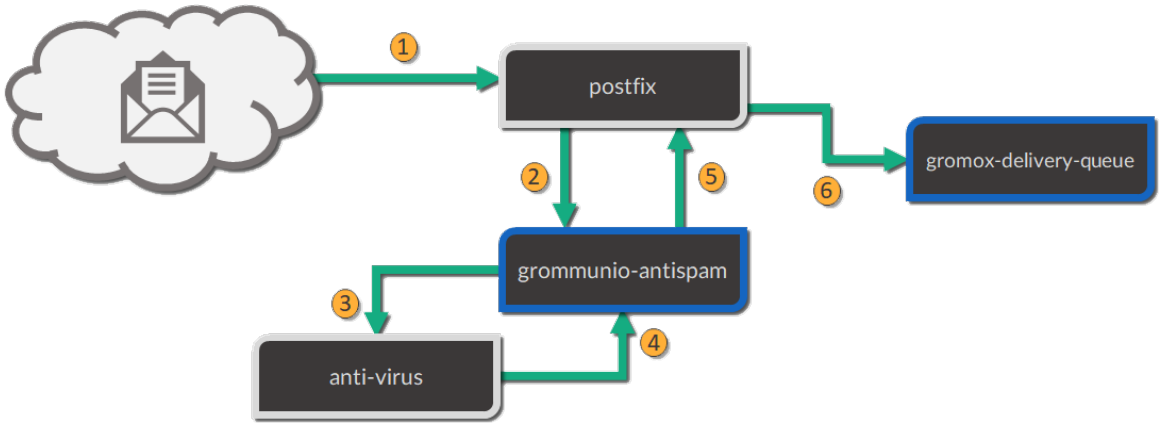
SMTP is the main protocol used for mail transport. For illustration purposes, there is a distinction made of the internal mail flow as well as external mail flow.

The entire transport is configured to be gapless in terms of email processing. This way, grommunio protects also from internal outbreaks (for example spam or virus distribution).

The configuration outlined here defines the default configuration set. In many cases, even more sophisticated setups might be envisioned, as with extended integration of security appliances. The following workflows provide the process definition which provides a view to where a preferred hook might be implemented.

7.3.2.1 Incoming

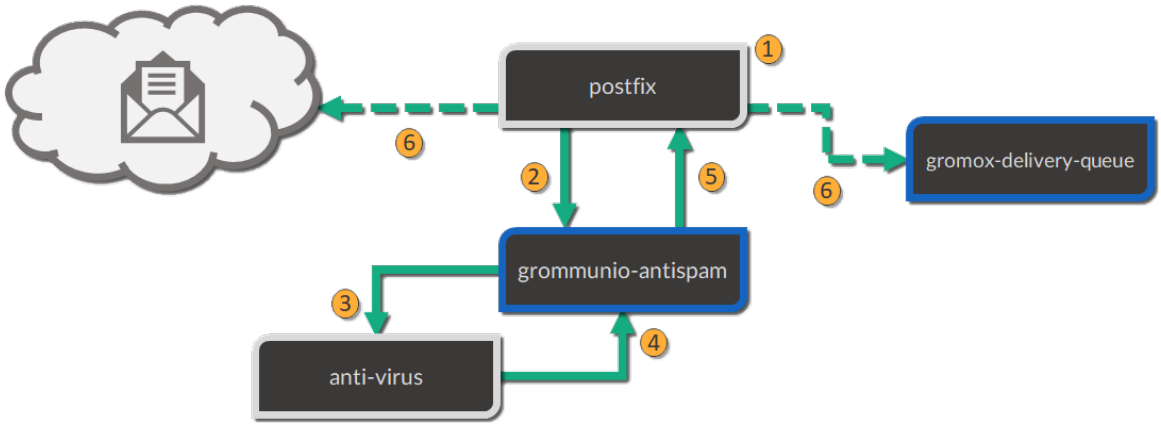
SMTP incoming



Mails are processed as follows (applies to incoming and outgoing):

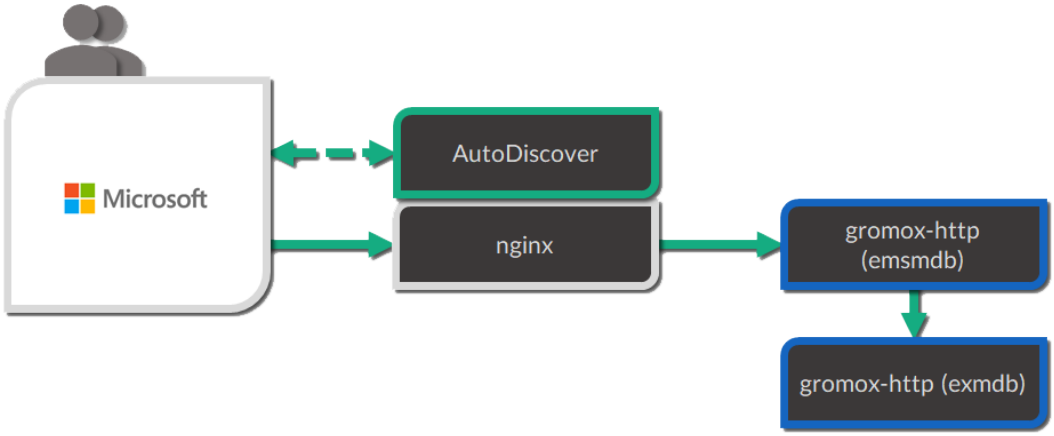
1. The included Postfix MTA receives messages and passes them to grommunio-antispam via the *Milter* mail filter protocol.
2. grommunio-antispam checks the message for spam. If configured, grommunio-antispam (optionally) passes the message to an anti-virus processing service.
3. The response from the anti-virus check is read back by antispam.
4. The response from antispam is read back by Postfix.
5. Postfix evaluates the contents of the Envelope-From and Envelope-To address pair to make the decision if this is i incoming or outgoing mail.
6. Incoming mail is relayed to the gromox-delivery process, which converts the mail to a MAPI object and places it in the user's mailbox.
7. Outgoing mail is delivered to a configured relayhost or to the next MX destination that is responsible for the target address.

SMTP outgoing/internal



7.3.3 RPC/HTTP, MAPI/HTTP & EWS workflow

RPC/HTTP, MAPI/HTTP & EWS connection flow



The main protocols used by grommunio for MAPI-based connectivity - as used for example with Microsoft Outlook - are:

- RPC/HTTP (OutlookAnywhere)
- MAPI/HTTP
- EWS (Exchange Web Services)

All of these protocols are HTTP-based which is why these are routed through the shipped nginx web server, primarily for security, scalability and monitoring reasons.

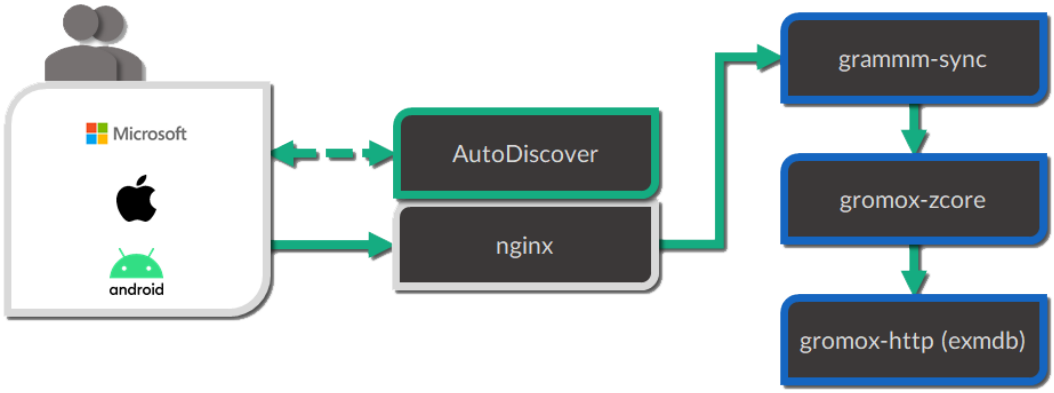
MAPI-based connections are processed as follows:

1. In the first stage, the endpoint utilizes AutoDiscover (https://docs.microsoft.com/en-us/exchange/architecture/client-access/autodiscover) technology (with Authentication) to discover which service endpoint URL is responsible for it.

2. If the AutoDiscover endpoint ends up at the same service (If not, it will be redirected to the other endpoint URL), nginx routes the connection directly to the gromox-http service which handles the connection.
3. For access to the users' mailbox, gromox-http's emsmdb plugin connects to the exmdb plugin for mailbox data delivery.

7.3.4 Exchange ActiveSync (EAS)

EAS connection flow



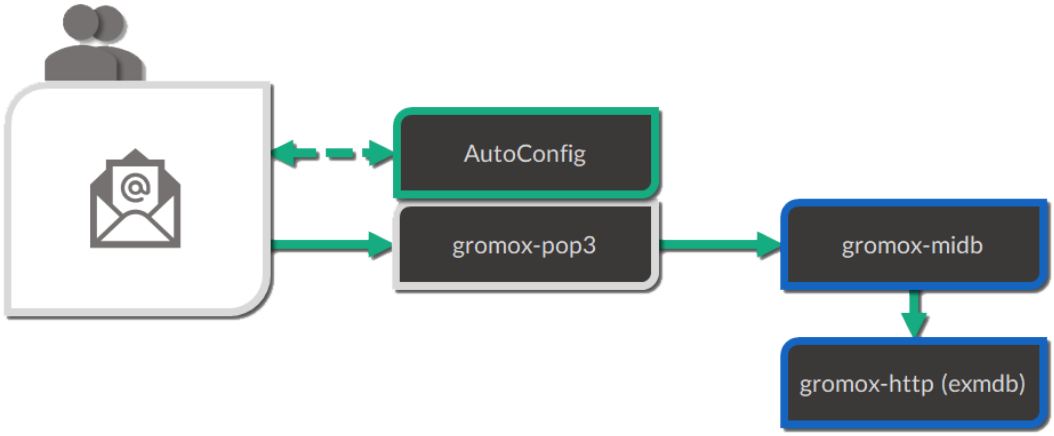
The main protocol used for mobile devices and tablets is Exchange ActiveSync (EAS). EAS is a synchronization state-based protocol which uses state data to determine its current synchronization status. EAS is often synonymously refered to as "Push Mail", since it is permanently connected to its service and listening for updates. As such, EAS is recommended as protocol for mobile devices especially over unreliable networks, such as cellular networks. While it is possible to connect certain clients, including Microsoft Mail and Microsoft Outlook, it is strongly discouraged to do so. Compared to its more performing alternatives, such as MAPI/HTTP, the EAS protocol is slower for bulk data transfer or large to very large (10 GB+) mailboxes. At last, the EAS protocol only delivers a subset of features available to other protocols.

EAS-based connections are processed as follows:

1. In the first stage, the endpoint utilizes AutoDiscover (https://docs.microsoft.com/en-us/exchange/architecture/client-access/autodiscover) technology (with Authentication) to discover which service endpoint URL is responsible for it.
2. If the AutoDiscover endpoint ends up at the same service (If not, it will be redirected to the other endpoint URL), nginx routes the connection to grommunio-sync which natively provides the /Microsoft-Server-ActiveSync endpoint to its device.
3. For access to the users' mailbox, grommunio-sync connects to gromox-zcore which delivers PHP-MAPI interfaces to access
4. gromox-http via exmdb plugin for mailbox data delivery.

7.3.5 POP3

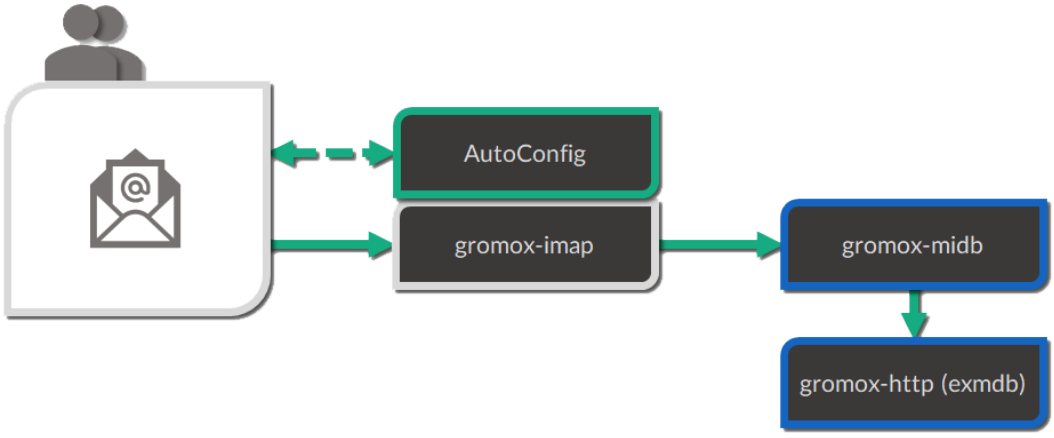
POP3 connection flow



POP3 workflow

7.3.6 IMAP

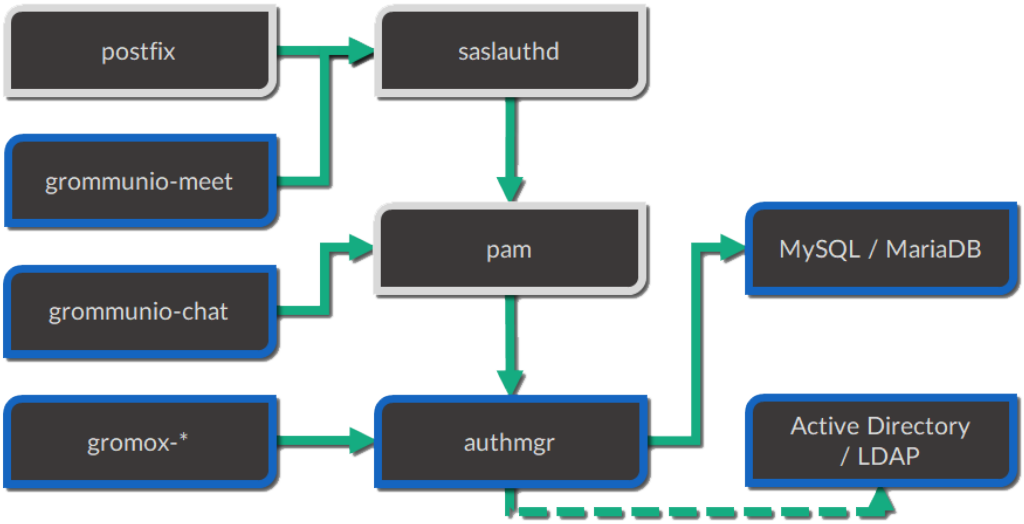
IMAP connection flow



IMAP workflow

7.3.7 Authentication

authentication flow



Authentication workflow

CHAPTER 8

Operations

8.1 Configuration

8.1.1 Admin API TLS configuration

Since the process of the Admin API is relevant for the initial provisioning stage, it is per default made available via port 8080 and unencrypted. As soon as the setup process has finished, it is advised to switch to a TLS-based configuration.

The shipped grommunio configuration files are prepared for setting up TLS configuration with the existing configuration. To activate the TLS configuration of grommunio-admin, execute the following steps:

```
ln -s /etc/grommunio-common/nginx/ssl_certficate.conf /etc/grommunio-admin-common/nginx-ssl.conf
```

This assumes the configuration of the TLS certificates has been installed successfully by the provisioning of grommunio Setup.

As a final step, uncomment the prepared configuration directive in the last line of the configuration file /etc/nginx/conf.d/grommunio-admin.conf as follows:

```
vhost_traffic_status_zone shared:vhost_traffic_status:8m;

# If you want to disable HTTP, take note that your configuration might
# need adaptation in the admin api configuration in
#   config.yaml -> options: -> vhosts: -> local:
include /usr/share/grommunio-admin-common/nginx.conf;

# Uncomment the following line to enable TLS for the admin interface.
# Make sure to create /etc/grommunio-admin-common/nginx-ssl.conf
# containing the certificate configuration
include /usr/share/grommunio-admin-common/nginx-ssl.conf;
```

After a subsequent successful configuration check of the webserver configuration, nginx may be restarted, and the Admin API is available on port 8443, e.g. https://mail.example.com:8443 :

```
# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
```

(continues on next page)

(continued from previous page)

```
nginx: configuration file /etc/nginx/nginx.conf test is successful

# systemctl restart nginx
```

Note that by restarting the webserver, existing connections are terminated.

8.1.2 Certificate management

For the operation of grommunio, the use of TLS-based security is mandatory. With TLS certificates in place, any communication with grommunio's services are protected by state-of-the-art encryption, which is mandatory for many clients and protocols.

If following the grommunio Setup path, also see also https://docs.grommunio.com/admin/administration.html#tls-configuration. Throughout the installation process, the administrator has multiple choices for TLS-based installation. For seamless operation, it is recommended to have a basic understanding of PKI concepts and the X.509 standard for certificates. Generally, grommunio uses PEM-encoded certificates.

If certificates need to be replaced, the certificates used by grommunio can be found by default in the following locations:

- /etc/grommunio-common/ssl/server-bundle.pem (certificate bundle including certificate authority)
- /etc/grommunio-common/ssl/server.key (private key)

By changing the certificates, all services using these certificates need to be restarted for the certificate to be used.

If Let's Encrypt has been chosen for installation, the service grommunio-certbot-renew.timer automatically runs weekly to perform any new certificate request. The status of the timer service can be checked with:

```
systemctl status grommunio-certbot-renew.timer
```

8.2 Updating grommunio

8.2.1 Package Updates

During every installation of grommunio Appliance, it attempts to connect to the community repository of grommunio. This way, community updates are directly available to community users and can update the Appliance accordingly. Furthermore, grommunio provides the operating system repositories which provide state-of-the-art packages with latest updates available to the Linux operating system based on openSUSE Leap, a binary compatible distribution of SUSE Linux Enterprise Server.

Note: Community repositories are delivered on a best-effort basis and are not supported. While grommunio welcomes community members to use grommunio, the software distribution available with the subscription repositories include production-relevant benefits. Subscription repositories (available only with a valid subscription) include quality-tested packages, hotfixes and extra features not available in community repositories.

For package management, the grommunio Appliances use zypper. Zypper is the package manager primarily used by SUSE-based distributions and is therefore default for the grommunio Appliances. Zypper has many similarities to other well-known package managers, such as dnf or apt.

The default repository file, /etc/zypp/repos.d/grommunio.repo is shipped with the following contents:

```
[grommunio]
enabled=1
autorefresh=1
baseurl=https://download.grommunio.com/community/packages/openSUSE_Leap_15.5/?ssl_verify=no
type=rpm-md
```

The default configuration does not verify SSL/TLS certificates intentionally. This enables support for:

- configuration-less automated proxy environments with SSL/TLS interception
- repository mirroring with selected partners and customers (hosting, large installations)

The integrity of all packages is secured by signatures on all packages distributed by grommunio with the grommunio GPG key, of which the public key is available at https://download.grommunio.com/community/packages/RPM-GPG-KEY-grommunio.

Your subscription credentials are provided to you via your grommunio partner and enables the availability of production-grade grommunio packages. These packages are quality-tested and only available to subscription customers.

To update your grommunio appliance with the most recent available updates, execute the following steps:

```
# zypper ref
Repository 'base' is up to date.
Repository 'debug' is up to date.
Repository 'debug-update' is up to date.
Repository 'grommunio' is up to date.
Repository 'update' is up to date.
All repositories have been refreshed.

# zypper up
Loading repository data...
Reading installed packages...

The following package is going to be upgraded:
 grommunio-admin-web

 1 package to upgrade.
 Overall download size: 1.8 MiB. Already cached: 0 B. After the operation, additional 696.0 B will be used.
 Continue? [y/n/v/...? shows all options] (y):
 Retrieving package grommunio-admin-web-1.0.1.8.6c8842f-lp153.1.1.noarch      (1/1), 1.8 MiB ( 15.0 MiB␣
↪unpacked)
 Retrieving: grommunio-admin-web-1.0.1.8.6c8842f-lp153.1.1.noarch.rpm .....[done]
 Checking for file conflicts: .....[done]
 (1/1) Installing: grommunio-admin-web-1.0.1.8.6c8842f-lp153.1.1.noarch .....[done]
```

After the installation/update of some packages, services are not always restarted automatically due to the nature of the potential implications of such a restart during a package installation. For packages that have been updated however, a manual restart of the service is recommended. The command zypper ps -s lists such services that should be restarted at a convenient time to have the new update in place. An example of such an operation is:

```
# zypper ps -s

zypper ps -s
The following running processes use deleted files:

PID  | PPID | UID | User | Command   | Service
-----+------+-----+------+-----------+----------
1553 | 1    | 0   | root | saslauthd | saslauthd

You may wish to restart these processes.
```

(continues on next page)

(continued from previous page)

```
See 'man zypper' for information about the meaning of values in the above table.

No core libraries or services have been updated since the last system boot.
Reboot is probably not necessary.

# systemctl restart saslauthd
```

8.3 Backup & Disaster Recovery

grommunio fully supports snapshot-based backups of all modern filesystems and/or appliances. The snapshot mechanisms of the following filesystems, backup solutions or storage systems are tested and supported:

- Acronis Backup
- Arcserve Unified Data Protection (UDP)
- Amanda Backup
- Amazon EBS snapshots
- Azure VM snapshots
- Bacula Backup
- Bareos Backup
- btrfs-based snapshots
- CephFS/RBD snapshots
- Commvault Hyperscale
- Dell EMC
- Docker-based snapshots (docker checkpoint)
- Google cloud persistent disk snapshots
- HP StoreVirtual
- Hitachi Vantara
- Huawei OceanStor
- Hyper-V snapshots
- KVM-based snapshots
- Kubernetes volume snapshots
- LVM-based snapshots
- LXC-based snapshots (lxc snapshot)
- NetApp
- NovaStor DataCenter
- Nutanix
- Pure Storage
- VMware snapshots
- Veeam Backup
- Veritas

- Xen-based snapshots
- ZFS-based snapshots

With the snapshot mechanism provided by the storage provider, snapshots can be easily used to backup and restore entire mailboxes in a matter of seconds. For restoring mailboxes to another mailbox's identity, it is recommended to ensure the mailbox is not in active use (such as mobile devices, profile synchronization). After the restore operation has completed, it is advised to restart the services gromox-http and gromox-midb to invalidate any existing runtime caches:

```
# systemctl restart gromox-http
# systemctl restart gromox-midb
```

To backup your grommunio installation, the following backup artifacts are relevant (per default):

1. grommunio Groupware (gromox):
   - /var/lib/gromox/user: directory hierarchy for private mailboxes
   - /var/lib/gromox/domain: directory hierarchy for public mailboxes (public folders)
   - /var/lib/gromox/user/account@domain: individual mailbox container
   - MySQL database: grommunio
2. grommunio Files:
   - /var/lib/grommunio-files
   - MySQL database: grofiles
3. grommunio Chat:
   - /var/lib/grommunio-chat
   - MySQL database: grochat
4. grommunio Archive:
   - /var/lib/grommunio-archive
   - MySQL database: groarchive
5. grommunio Appliance:
   - File backup of /etc/grommunio*
   - File backup of /etc/nginx (if any non-standard configuration changes have been made)
   - File backup of /etc/php7/fpm/php-fpm.d (if any non-standard configuration changes have been made)
   - File backup of /etc/letsencrypt (if Let's Encrypt certificates are used)
   - File backup of /etc/postfix (if any non-standard configuration changes have been made)

Note: By using grommunio-dbconf, many file-based backups are not required. This is because dbconf stores configuration directives within the main grommunio database.

8.3.1 Database backup

Backup the grommunio databases grommunio, grofiles, groarchive and grochat using standard procedures. Most backup solutions provide MySQL database backup agents for easy integration. For detailed backup options of your MySQL databases, refer to: https://dev.mysql.com/doc/refman/8.0/en/backup-types.html. If in doubt, the built-in utility mysqldump (https://dev.mysql.com/doc/refman/8.0/en/mysqldump.html can create single SQL backup files of databases. A manual MySQL backup dump can be issued with:

```
mysqldump --single-transaction --routines --triggers --events --add-drop-database > grommunio-mysql-backup.
↪sql
```

8.3.2 File-based backup

Since grommunio works entirely on the basis of transactions, any file-based backup is consistent at sync time, as long as it utilizes a "deltasync" based operation. It is also possible to sync files from the original operating location to a remote/mounted location for disk-to-disk backup scenarios, if so desired. With rsync, the grommunio Appliance offers a simple tool to synchronize data for this backup method. A manual file backup based on deltasync functionality by rsync can be issued with:

```
rsync -HPavS <from-directory> <to-directory>
```

8.4 Mail requeueing

To requeue messages in Gromox that have failed delivery for any reason, follow the steps outlined below. This process is particularly useful when dealing with messages that have not been delivered due to various errors, and can be found in the /var/lib/gromox/queue/save directory.

Locate Failed Messages: Navigate to the directory where failed messages are stored by using the command cd /var/lib/gromox/queue/save. This directory contains messages that have not been successfully processed.

Listing Messages: Use the command ls -ltr to list the messages in the directory. This will display all the files sorted by modification time, making it easier to identify and select the messages you wish to requeue.

Requeue Messages: To requeue the messages, they must be moved to the /var/lib/gromox/queue/mess directory. This can be done using the mv command. For example, to move a specific message, use:

```
mv <filename> /var/lib/gromox/queue/mess/
```

Replace <filename> with the name of the message file you intend to requeue. If you wish to requeue multiple messages, you can use wildcards or specify multiple filenames separated by spaces.

Reprocessing: Once the messages are moved to the /var/lib/gromox/queue/mess directory, Gromox will automatically attempt to reprocess and deliver them. This step does not require any additional action from the user.

Verification: After requeuing, it's a good practice to monitor the system logs to ensure that the messages are being processed successfully. Use journalctl -u gromox-delivery -u gromox-delivery-queue to check for any log entries related to the requeued messages.

This process is essential for managing delivery issues within Gromox and ensuring that all messages reach their intended recipients. If requeuing does not resolve the delivery issues, further investigation into the cause of the failure might be necessary, including checking system logs for errors and reaching out to Grommunio Support for assistance.

A simple loop to re-queue all failed messages can be achieved by the following snippet:

```
cd /var/lib/gromox/queue/
j=0
for i in save/*; do
        mv "$i" "mess/0$j"
        j=$(($j+1))
        echo "requeued $i"
done
```

8.5 Size limits

Grommunio operates using native MAPI storage, which imposes certain (soft) restrictions. Although these are primarily theoretical, client limitations when accessing grommunio are dictated more by the MAPI standard's guidance rather than by MAPI's inherent limitations.

Most restrictions stem from the use of Microsoft Outlook as the primary client. Outlook imposes its own set of limits, which we advise adhering to in order to maintain compatibility and full functionality with Microsoft's suite of products.

An object, in this context, could be an email, a calendar appointment, a task entry, a contact, or a note.

We recommend observing the following limitations:

- Maximum size per object (Message size limit): 150 MB
- Maximum number of objects per folder (Message count limit): 1,000,000
- Maximum parts in multipart messages (MIME): 250 parts
- Maximum storage size: 100 GB (with a default of 50 GB)

Registry path for adjusting max OST size in Microsoft Outlook:

- HKEY_CURRENT_USER\Software\Microsoft\Office\<version>\Outlook\PST MaxLargeFileSize (DWORD(32-bit), Decimal: 102400)

Please note:

Note: These are not strict limitations set by grommunio; rather, they are recommended for smooth integration with Microsoft Outlook. These guidelines also align with those used in Microsoft Exchange On-premises and Exchange Online environments. Should you opt not to use Microsoft Outlook as your client, grommunio can support much larger limits. However, exceeding these recommended limits may affect performance, depending on your configuration.

CHAPTER 9

Troubleshooting

9.1 Support package

Subscription customers can generate a support package by executing the command `grommunio-support` and send the created support package to grommunio's support for analysis to support@grommunio.com.

The archive generated is made available under the web root of grommunio admin archive, which is why it is stongly recommended to remove the generated support archive as soon as it has been transmitted to grommunio support. The support archive can be removed by accessing the console and executing the command `rm -f /usr/share/grommunio-admin-web/grommunio-support.txz`.

The information collected by grommunio-support contains:

- Crash relevant information (Coredumps)
- Disk layout (incl. LVM layout and SMART and Software RAID)
- Configuration dump (incl. /etc and more specific information e.g. from webserver)
- High-availability information
- Memory-related information
- Network configuration
- Process-relevant information
- Sysconfig information

Important: The support package might contain sensitive information. If this is a concern to you, it is recommended to prune specific private data from the generated archive before sending it to grommunio support. Support data is used only for diagnostic purposes and is considered confidential information.

Note: The support is solely available on the appliances provided by grommunio. On SUSE-based distributions it can also be made available by repository installation via the "grommunio-setup" package, which has a dependency on the package "supportutils".

9.2 Installation logs

The setup wizard of the grommunio Appliance saves its log to /var/log/grommunio-setup.log. If, for example, the wizard fails the certificate generation, the reasons should be visible in that file.

9.3 System logs

The grommunio Appliance inherits system logging settings from systemd. Refer to the systemd-journald(8) manpage for details. To display logs, use the journalctl(8) command from a root login shell prompt:

```
journalctl -u gromox-http -n 1000
journalctl -f
```

Useful options that can independently be combined are:

- -f for follow mode
- -n to show that many of the most recent lines
- -u to limit the display to one particular service unit

Some logs are emitted to files rather than journald. These include:

URI Prefix	Process	Files
/dav	nginx	/var/log/nginx/grommunio-web-access.log, /var/log/nginx/grommunio-web-error.log
/dav	php-fpm	/var/log/grommunio-dav/grommunio-dav-php.log
/Microsoft-Server-ActiveS	nginx	/var/log/nginx/grommunio-web-access.log, /var/log/nginx/grommunio-web-error.log
/Microsoft-Server-ActiveS	php-fpm	/var/log/grommunio-sync/grommunio-sync-fpm.log
/web	nginx	/var/log/nginx/grommunio-web-access.log, /var/log/nginx/grommunio-web-error.log
/web	php-fpm	/var/log/gromox

9.4 Coredumps

The grommunio Appliance ships with systemd-coredump installed by default and is thus configured to emit dumps to /var/lib/systemd/coredump. If a crash occurred and left a dump behind in this directory, make available the dump file to the support team, and specify the version details of packages (e.g. the command rpm -qi gromox grommunio-index libexmdbpp0 will give Version: and Distribution: field). Note that because it is a complete memory dump, the files can contain sensitive information like usernames, passwords, mail texts, etc.

For systems not based on the appliance, consider the following points:

When systemd-coredump is installed, that package normally sets the systemd-coredump.socket to active, and places a fragment file in /usr/lib/sysctl.d/:

```
kernel.core_pattern = |/usr/lib/systemd/systemd-coredump %P %u %g %s %t %c %e
```

The presence of this fragment file will make this setting effective at the next boot. The presence of _another_ coredump middleware, including, but not limited to, Ubuntu apport or Fedora abrt, may cause multiple sysctl fragment files to compete and only one win. It is best not to have more than one such middleware.

Furthermore, systemd versions before 251 have a rather low dump limit of just 2 GB. To raise this, see /etc/systemd/coredump.conf.

It is possible to do without middleware and instead exercise the direct-to-file dump functionality from the Linux kernel, e.g. by setting the particular sysctl variable to:

```
kernel.core_pattern = /var/tmp/core.%E.%p
```

This emits files without compression, which may be beneficial during development but less so much for transferring dumps.

CHAPTER 10

Release Notes

10.1 grommunio 2025.01.2

- Release type: Minor
- Release date: 18th of April 2025
- General availability: Yes

Highlights

- Polished grommunio Web with updated editors (TinyMCE 7.8.0) and viewers (PDF.js 5.1.91), plus improved handling of shared distribution lists.
- Per-user service controls are now fully enforceable – administrators can enable/disable Web, ActiveSync, and CalDAV/CardDAV access per user via Admin API/CLI, and these restrictions are honored across all components.
- Enhanced mobile and CalDAV synchronization reliability, including better compatibility with iOS all-day calendar events and support for alternate login names.
- Licensing improvement: Only active (non-disabled) user accounts count toward license limits now, aligning license usage with actual active users.
- Numerous stability and performance fixes across the stack (mail processing, logging, memory management, etc.) further improve reliability.
- grommunio Setup for DEB: Shipping through the package grommunio-setup, first semi-automatic installations can be made on DEB-based distributions (Debian, Ubuntu)
- EWS: Further improvements in our EWS improve interoperability, especially with eM Client.

Enhancements

- User Service Management: Introduced support for per-user service enablement toggles. The Admin API/CLI now allows toggling user access to Web, EAS (ActiveSync), and DAV services, and the groupware components respect these settings (enforcing service restrictions for disabled users).
- Licensing: Improved licensing logic by counting only active users against license limits. Disabled or archived users no longer consume a license slot, providing more accurate license utilization for organizations.

- Web Interface Updates: Upgraded grommunio Web's third-party components for a better user experience. The rich text editor was updated to TinyMCE 7.8.0, the PDF viewer to pdf.js 5.1.91, and the HTML sanitizer to DomPurify 3.2.5, bringing performance, security, and functionality improvements. Additionally, the calendar's monthly view now once again displays the recurring-event icon, and the Web UI can show details of public and shared distribution lists (making it easier to view members of shared contacts lists).
- Plugin and Compatibility Improvements: The optional Kendox plugin is now disabled by default to streamline the Web interface and avoid issues with unused integrations. Also, grommunio Web and related services have officially dropped support for PHP 7.x, requiring PHP 8+ — this update aligns the platform with modern PHP versions for better performance and security.
- Mailing List and Address Book: Gromox now supports nested groups in permission checks. This enhancement means distribution lists can contain other lists and still resolve correctly, improving flexibility in complex group permissions. Furthermore, internal address-book handling was improved for internationalized entries – additional UTF-16/32 codepage variants are recognized, enhancing support for contacts or attachments with non-Latin characters and internationalized domain names.
- CalDAV/CardDAV (grommunio DAV): Refined the DAV service for better performance and interoperability. Logging verbosity has been reduced by removing overly extensive debug output (resulting in cleaner logs and lower overhead), and the default fastcgi_read_timeout for the DAV web service was extended (to 360 seconds) to accommodate lengthy calendar or address book operations without timing out. The DAV service also now passes through error responses to clients correctly (ensuring CalDAV/CardDAV clients receive proper error codes), and its dependency stack was updated for stability.
- General Performance & Stability: Numerous low-level enhancements were made in the core services (Gromox). Memory management was improved in several modules (e.g., automatic buffer reallocation and proper out-of-memory signaling in zcore and exmdb components) to increase scalability under high load. These changes, along with other under-the-hood optimizations, reduce the likelihood of service crashes and improve overall system efficiency.

Bug Fixes

- Shared Mailbox Distribution Lists: Fixed issues with shared and public distribution lists in grommunio Web. Users can now successfully send emails to a shared distribution list, and the UI properly expands and displays members of shared/public distribution lists. (Previously, attempts to use or view members of these lists could fail.)
- Alternate Login Name Fixes: Resolved multiple problems related to alternate user login names (aliases). Users who log in with an alternate email/username can now change their password from the user portal (this was not possible before). In addition, synchronization issues in grommunio Sync when using alternate logins have been addressed, so mobile devices and EAS clients will sync correctly even if the user is logged in via an alias.
- Calendar All-Day Events: Corrected an ActiveSync calendaring bug that affected Apple iOS clients. All-day events created on one day would sometimes appear spanning two days on iOS devices – this has been fixed to ensure all-day events consistently show on the intended single day across all clients.
- IMAP Protocol Compliance: Fixed a minor formatting error in IMAP responses – the BODYSTRUCTURE response now includes a needed space that was previously omitted. This compliance fix improves compatibility with IMAP email clients and ensures no parsing issues due to the missing whitespace.
- Email Content Conversion: Fixed an issue in the email conversion library that could cause HTML-formatted emails to be converted incorrectly. Email content (HTML to plain text or other formats) now converts as expected, preserving formatting and ensuring the message is readable in all clients.
- Stability Fixes: Addressed rare crashes in the mail processing backend. In particular, issues in the rule processor and mail delivery modules caused by memory allocator mismatches have been resolved. These fixes eliminate certain intermittent crashes (for example, when processing

server-side mail rules or delivering messages under high load), resulting in a more robust and reliable server.

- PST Export: Resolved a problem that prevented Outlook PST exports in certain scenarios. Gromox no longer includes an unintended PR_MESSAGE_SIZE property in export streams, which means exporting mailboxes to PST format will now complete successfully (the extra data that caused PST exports to fail has been removed).

General Notes

This version is the last version to include builds for openSUSE 15.5. Any future updates demand strict PHP 8.1+ compatibility. Please update installations still running on openSUSE 15.5 accordingly (for example by use of `grommunio-update upgrade`)

The above lists cover the most significant changes in grommunio 2025.01.2. Dozens of smaller fixes and improvements are included in this release to refine overall functionality and security.

10.2 grommunio 2025.01.1

- Release type: Major
- Release date: 29th of January 2025
- General availability: Yes

Highlights

Appliances now based on openSUSE 15.6

The latest grommunio appliance releases ship with openSUSE 15.6 as their foundation, benefiting from up-to-date security patches, improved stability, and modern hardware support.

Performance Boost & Lower Resource Requirements

Thanks to extensive enhancements in parallelization (especially for single-store, highly parallelized scenarios), the overall performance of the grommunio stack has improved while resource requirements (RAM, CPU, disk) have decreased.

Keycloak 26.1 Integration

grommunio now ships with Keycloak 26.1, including:

- Refined SSO & identity management with expanded security controls.
- Improved user federation for large-scale deployments, simplifying integration with heterogeneous directory services.
- Advanced admin console features for streamlined configuration and audit trails.

TinyMCE Upgrade from 4.9.11 to 7.6.1

The grommunio Web's email editor now leverages TinyMCE 7.6.1, providing:

- Modernized UI/UX, especially on mobile and touch devices.
- Enhanced performance and security, ensuring a smoother editing experience (like the content hover-bar).

PHP 8.2 and 8.3 Support

grommunio's core and associated services are now fully compatible with PHP 8.2 and 8.3. Key benefits include:

- Better performance and memory optimization.
- Enhanced type and error-handling features for developers.
- Extended grommunio Stack Upgrades and compliance.

Enhanced Internet Mail Compliance

grommunio continues to refine support for Internet mail standards. This ensures more robust and accurate parsing and generation of emails across a variety of clients and mail servers.

New Features & Enhancements Share-Nothing Clusters

Expanded from the previous release, clusters can be scaled out without relying on shared storage. This provides maximum flexibility in multi-node deployments and reduces potential bottlenecks or single points of failure.

Parallelized Single Mailbox Access

A key promise fulfilled: significant performance gains when multiple users or processes access large mailboxes simultaneously. The new parallelization logic helps distribute loads more efficiently, avoiding lock contention scenarios.

Overhauled Indexing & Search

Building on recent indexing improvements, search across emails, contacts, and other items is now quicker and more accurate while requiring less storage overhead.

Massively Improved S/MIME

Updates include refined clear-signed message handling, upgraded certificate validation, and improved out-of-the-box interoperability with various device classes.

Per-User Feature Enablement

Administrators can continue to leverage granular toggles to enable or disable Web, Sync (ActiveSync), and DAV services on a per-user basis, helping organizations fine-tune resource access.

Timezone & Migration Compatibility

Ongoing refinements ensure consistent scheduling across multiple protocols (CalDAV, EWS, MAPI) and more accurate data migration from legacy systems (Exchange, Communigate Pro, Kerio, Kopano, Zarafa).

grommunio-Web Signature Templating

A new feature allowing administrators and end users to define, customize, and manage standardized email signatures across the organization. This includes variables (e.g., name, title, department) for dynamic insertion, ensuring a consistent brand identity while reducing manual signature maintenance.

EWS processing enhancements

With a growing number of EWS clients using grommunio, certain specific flavors of EWS client implementations show the need for adoption in our EWS server-side processing code for enhanced compatibility. For example, 2025.01.1 includes improved timezone management for example with Apple clients and further enhancements for enhanced compatibility with emClient and Evolution.

Development Process Updates

- Monthly Point Releases: Starting with 2025.01.1, grommunio will deliver monthly point releases (e.g., 2025.01.2 in February).
- Annual Major Upgrade: There will be at least one major release each year, with larger feature overhauls and architectural improvements.

Certification Initiatives

With growing adoption by public sector and defense organizations, grommunio is actively pursuing certifications such as FedRAMP/NIST, FISMA, and BSI. This underscores the commitment to higher security standards and regulatory compliance.

Roadmap for 2025.01.2

- RFC 2184/2231: Enhanced handling of extended parameters in MIME headers.
- Trashed Mailboxes & Migration: Improvements for advanced mailbox handling across multiple migrations, including x400 addressing and undocumented MAPI attributes.

- grommunio Support v2: Expanding support for the setup stage for RHEL9, Debian 12, and Ubuntu 24.04.
- grommunio-files: Updated version with group folder management and modern authentication.

Forthcoming in the next releases (previews available to selected partners)

- Modern Authentication (OAuth2) for Outlook, IMAP, and POP3.
- Full HTML-based MR (Meeting Request) Processing in the Web UI.
- AI-Powered Features for enhanced user productivity.
- Extended rules & autoprocessing support

Supported Distributions

As of 2025.01.1, grommunio actively supports installation and operation on the following Linux distributions:

- RHEL9 / EPEL9
- openSUSE 15.5+ / SLES 15.5+
- Debian 12
- Ubuntu 24.04

Acknowledgements

We would like to extend our sincere gratitude to our community, customers, and partners for their continued support, feedback, and contributions. A special thanks goes out to our active contributors: crpb, dahan, brad0, kasperk81, robert-scheck, orandev01, rnagy, walter, liske, steve, milotype, clique2015 any many others. Your insights drive our roadmap and make grommunio more robust, secure, and performant with each release.

10.3 grommunio 2023.11.3

- Release type: Minor
- Release date: 16th of February 2024
- General availability: Yes

Highlights

- EWS is now fully supported to run with Microsoft Outlook for Mac as well as Apple native organization apps (Mail, Calendar, ...)
- S/MIME received updates for validation across various device classes
- IDN (internationalized) domains are now supported in GAL (Global Address Lists)
- CalDAV now supports iCalender free/busy information
- grommunio Web received polishing fixes since the last major design upgrade
- Support for Passkey authentication with grommunio Auth
- Documentation has received numerous updates, including a major documentation overhaul

New Features

- EWS has left the beta stage and is now enabled by default (See notes)
- The new rule processor (twostep_ruleproc) now supports Outlook-style public folders
- grommunio now provides 389DS schema via a selector in grommunio Admin
- Outgoing messages submission via postdrop is now supported

- grommunio Next is now available as technology preview in the repositories (requires Graph API)

Enhancements

- S/MIME related fixes to Web now enable multiple attachment download
- Unintended double-quotes in mails are now dropped around RFC 2047-style encodings
- Resolved a rare case where PR_TRANSPORT_MESSAGE_HEADERS had an extra byte
- Resolved a case where four extra bytes where added in front of the first transport header
- Semicolons in "Reply-To" headers are now handled correctly
- Proper handling for log messages enabling better fail2ban processing
- ICS requests can now be dumped for developer inspection
- Extensive dependency updates for Debian/Ubuntu based installations
- Various improvements to migration toolset
- Various mail processing enhancements (e.g. dot-stuffing)

Notes on EWS

As mentioned above, with EWS leaving the beta stage, the parameter $\mathrm{ews\_beta{=}1}$ in $/\mathrm{etc/gromox/ews.cfg}$ is now obsolete. EWS is now enabled per default and the parameter is not required anymore.

Acknowledgements

We extend our heartfelt thanks to our customers, partners, and the community for their valuable input and feedback. Thanks to feedback from customers and the community, we have been able to track down EWS-related issues properly and have included the feedback in our evaluation process, leading to a better product for all.

We would especially like to thank the community for the overwhelming feedback, especially at FOSDEM https://fosdem.org/2024/ <https://fosdem.org/2024/>_.

Last remarks

The development, QA, and release teams, apologize that our public communication has been occasionally delayed. We've been very busy not only delivering a better product to you with a plethora of fixes and new features but also integrating new resources into the entire organization and infrastructure. It's amazing how many installations have hit production in the holiday season which required additional prioritization. Rest assured, there's big news coming up from grommunio, and you'll notice it.

10.4 grommunio 2023.11.2

- Release type: Minor
- Release date: 28th of December 2023
- General availability: Yes

Highlights

- The appliance now ships with XFS as the default main filesystem
- IMAP performance has improved overall by a factor of 2 or more (SELECT/LIST/FETCH seqid renumbering removal)
- IMAP compatibility has significantly improved by handling EXPUNGE and STATUS commands properly
- Windows Mail now also works as an EAS client
- Enable Room and Equipment stores for AutoDiscover with Delegation (Shared Store)

- Enhanced search folder notifications (more improvements to come)

New Features

- IMAP now receives deletion events from other clients (OL/Web/EAS/EWS)
- gromox-mbop now supports time specifications to limit the deletion of messages of a certain age
- All daemons have received various config directives for file descriptor limits, with 512K instead of 2256 in systemd environments
- Support for XFS snapshots

Enhancements

- Enable gromox-mbop path specifications, such as *SENT/2024*
- RTF compressed MAPI properties now generate a complete header
- Free busy information is now more resilient to non-existing data (no information available)
- The basic authentication header is now fully RFC 7617 compliant
- The name service provider (NSP) now fully supports the Windows UTF-8 locale (Beta feature by Microsoft)
- Improved calendar item coverage for EWS
- Enhanced EWS CreateItem for Apple Mac Mail
- Repair Property ID/Tag swapping with TNEF objects
- Enhancements to ICS now reduce the number of sync issues due to broken items (imported e.g. from defective Kopano datasets)
- Better processing for calendar appointments (RDATE, Weekorder), displaying correct all-day events from broken sources as per OXCICAL spec recommendations
- Heap-use-after-free fix for free/busy requests in EWS
- Multi-LDAP has received robustness fixes for special cases (such as 389DS)
- Various fixes to free busy handling (related to scheduling)

Acknowledgements

Since the number of contributors keeps growing with each release, we now refrain from compiling a hand-curated list and instead ask anyone interested to head over to our git repositories and see the evolving community for yourself. Rest assured, grommunio thanks all its stakeholders: customers, partners, and the community alike.

10.5 grommunio 2023.11.1

- Release type: Major
- Release date: 18th of November 2023
- General availability: Yes

Highlights

We are excited to announce the release of grommunio 2023.11.1. This update marks a significant milestone in our journey as a leading open-source groupware platform. With a suite of new features and enhancements, this release underscores our commitment to providing an enterprise-grade communication solution that is both comprehensive and secure.

What's New

- Enhanced EWS Functionality with support for Microsoft Outlook for Mac, Apple Mail, and Microsoft Outlook for Mobile
- Advanced Single Sign-On (SSO) with Active Directory environments (SPNEGO support)
- Redesigned User Interfaces, adhering to WCAG 2.1 guidelines for improved accessibility
- Performance improvements with grommunio Web and 25% faster end-to-end processing
- Alternative Logon Names Support, offering greater flexibility in identity management for complex enterprise needs
- Online Update and Upgrade Capabilities integrated with grommunio Admin
- Recipient Plus-Addressing and enhanced Mailbox DB Operations with grommunio-mbop
- Modern Authentication in grommunio Web with OpenID Connect including support for 2FA (Two-Factor Authentication)

Enhancements

- Various Fixes: Including support for non-receiving shared mailboxes and enhancements in imap, exmdb, and alias_resolve modules.
- Comprehensive IMAP (Large Literals and RFC 7888) and Productivity Enhancements
- Support for vCard 4.0 and improvements in 'oxvcard'
- Refined Folder and Message Delivery including improved 'create_folder' and 'movecopy_folder' RPCs

Notes on EWS

- To activate EWS Beta features, add $\mathrm{ews\_beta=1}$ to $\mathrm{/etc/gromox/ews.cfg}$
- Activation of $\mathrm{ews\_pretty\_response}$ is not supported by Mac Mail and is recommended not to be enabled as such
- The best supported EWS Client is currently Microsoft Outlook for Mac
- The upcoming EWS operations FindFolder and FindItem are expected to be released within the upcoming 2 weeks after release which enhances Apple's macOS apps most.

Disclaimer: Public Beta Release of EWS Functionality

- Intensive Development and Testing: The EWS functionality has undergone extensive development to achieve a modern and solid software architecture. This rigorous process ensures a high standard of quality, security, and functionality. However, as with any complex software endeavor, there may be unforeseen nuances in diverse real-world environments.
- Current Limitations: We acknowledge that two features – the FindItem operation and the Impersonation feature – are not yet included in this beta release. These features are currently undergoing thorough quality assurance testing. We anticipate their inclusion still within the 2023 release timeline, further enhancing the EWS functionality.
- Commitment to quality and security: Our team, in collaboration with our technology partners, has repeatedly validated the EWS functionality to ensure its security, data protection, and stability. We adhere to the highest standards to safeguard your experience.
- Feedback and continuous improvement: While we have invested considerable effort in testing, we acknowledge that the diverse and dynamic nature of IT environments can present unique scenarios. Therefore, we welcome and appreciate any feedback or reports of issues from our users. Your insights are invaluable in helping us refine and improve the EWS functionality.
- Support for subscription holders: With the release of this EWS functionality, it becomes a fully supported protocol within grommunio. Subscription holders are entitled to our full support for any queries or assistance related to EWS. For customers and hosters: Please approach your support representative if you need any planing for EWS rollout. As with every new big feature, it is recommended to plan the availability with care and our staff is committed to support you well.

Acknowledgements

We extend our heartfelt thanks to our customers, partners, and the community for their invaluable input and feedback, especially to:

- clique2015, robert-scheck, General-Aussie, steve, prandev01, crpb, rnagy, walter any many others

10.6 grommunio 2022.12.1

- Release type: Major
- Release date: 24th of December 2022
- General availability: Yes

Highlights

- grommunio Appliance now on openSUSE 15.4 with many improvements, such as PHP 8.0
- General Availability of Multi-LDAP, worlds-first multi-backend groupware engine
- General Availability of Admin API for PowerShell (AAPIPS), a PowerShell interface for grommunio Admin
- General Availability of grommunio Desktop, a multi-platform client for grommunio Web
- General Availability of grommunio Meet for Outlook, a plugin for Microsoft Outlook and grommunio Meet
- General Availability of grommunio Auth, SSO availability with grommunio (based on Keycloak)
- General Availability of native Dockerfiles and Kubernetes recipes for Gromox
- High performance data compression with zStandard (zstd)
- Public Folder synchronization for mobile devices
- High-performance rewrite of Autodiscover and Autoconfig
- High-performance rewrite of EWS (Exchange Web Services)
- DNS-Name based OEM whitelabeling for custom branding

Enhancements

- Availability of EAS 16.1 FIND command
- Full user resolution for Kopano migrations (--mbox-name/--user-map)
- Centralization of MAPI header files
- grommunio CUI is now fully translated in 22 languages
- Enhanced navivation controls of grommunio CUI
- Support for hidden contacts
- Automatic mapping of AD/Exchange Store Types (msExchRecipientDisplayType)
- Centralized MAPI header files for PHP consumers
- Default integration of grommunio-dbconf
- Implementation of hierarchy and permission model (ACLs) for public folders in Admin
- Mail-Queue mangement in grommunio Admin
- Large documentation updates, launch of Knowledge Base in Documentation Portal

The above list is not conclusive. As usual, numerous bug fixes and features have been included. The release notes just highlight major changes; Feel free to check out the detailed logs at GitHub (https://github.com/grommunio).

The official documentation covers the necessary steps for the update procedure.

Contributions & Thanks

Thanks to customers, partners and the entire community - the community for their ongoing contributions, especially to:

- MrPikPik, tiredofit, maddin200, artem, steve, thermi, milo, Bheam, crpb, rnagy, walter any many others

Special thanks to Microsoft Corporation for the productive cooperation on standards and protocols and to T-Systems International for the collaborative work on scale-out installations with highest enterprise demands.

10.7 grommunio 2022.05.2

- Release type: Minor
- Release date: 31st of August 2022
- General availability: Yes

Highlights

- Support for PHP 8.0 and 8.1
- "SendAs" support (additionally to "Send on behalf of")
- Improved admin interface design and handling, including topic search
- Multi-Language Support with 22 languages
- Multiple dependency extensions for Platforms EL 8, Debian 11 and Ubuntu 22.04
- Hierarchy for Public Folders in grommunio Admin (API, CLI and Web)
- Public Folder ACL support admin grommunio Admin (API, CLI and Web)

New Features

- Support for multi-iCal and multi-vCard formats
- Unification of MAPI libraries throughout web components
- Configurable midb command buffer size for large IMAP migrations (80GB+ per mailbox)
- Migration: Ignore Kopano Archiver stub elements

Enhancements

- Support for pooled LDAP connections via TLS (restartable Policy)
- Enhanced Timezone handling based on most recent IANA Timezone policies
- kdb2mt: support recovering broken attachments lacking PR_ATTACH_METHOD
- kdb2mt: remove PK-1005 warning since now implemented
- delmsg: support mailbox lookup using just the mailbox directory name
- http: added the "msrpc_debug" config directive
- nsp: added the "nsp_trace" config directive
- mh_nsp: make the addition of delegates functional
- kdb2mt: support recovering broken attachments lacking PR_ATTACH_METHOD

- imap: emit gratuitous CAPABILITY lines upon connect and login
- imap, pop3: support recognizing LF as a line terminator as well (other than CRLF)
- Added a config directive tls_min_proto so one can set a minimum TLS standard when your distro doesn't have crypto-policies (https://gitlab.com/redhat-crypto/fedora-crypto-policies)
- autodiscover.ini: new directives advertise_mh and advertise_rpch for finer grained control over individual protocol advertisements; replaces mapihttp.
- exmdb_provider: lifted the folder limit from 10k to 28 billion
- oxcmail: cease excessive base64 encoding.
- Improvements to Outlook online/interactive search for improved responsiveness in Online Mode.
- Messages are now preferably encoded as quoted-printable during conversion to Internet Mail format. This might help with spam classification.
- delivery-queue: the maximum mail size is now strictly enforced rather than rounded up to the next 2 megabytes
- gromox-dscli: the -h option is no longer strictly needed, it will be derived from the -e argument if absent

The above list is not conclusive. As usual, numerous bug fixes and features have been included. The release notes just highlight major changes; Feel free to check out the detailed logs at GitHub (https://github.com/grommunio).

The official documentation covers the necessary steps for the update procedure.

Did you know?

grommunio strives for precise documentation underlying the standards and protocols grommunio builds upon, since these are the foundation for stable communication and functionality. We at grommunio also regularly fix incorrect portions of Microsofts' own documentation - example: https://github.com/MicrosoftDocs/office-developer-client-docs/pull/613/commits/09c4ada5114d8e2d9f65ce29a25f40a6fc6c2278

In this spirit, we have published the grommunio documentation online (https://github.com/grommunio/grommunio-documentation), available for contributions from any source to make the documentation of grommunio as good as possible.

Contributions

Thanks to customers, partners and the entire community - the community for their ongoing contributions, especially to:

- Robert, who has provided various contributions to support BSD.
- Walter, for his various contributions in the migration tools area.
- Christopher, for his role-model involvement in grommunio community as maintainer.
- Michael, for reports on admin api resiliency in distributed environments.
- Stefan, Bob and Andreas for large scale container setup feedback.
- Rob and Hannah, for guidances path on F5 nginx plus/unit.
- Microsoft, for review, feedback and acceptance of errors in Microsofts' documentation.
- ILS, for intense collaborative contributions to deliver grommunio in over 22 languages.
- Artem, Milo, Hugel and many more for various language contributions.

10.8 grommunio 2022.05.1

- Release type: Major
- Release date: 16th of May 2022
- General availability: Yes
- grommunio: Support for Ubuntu 22.04
- grommunio: Support for NetIQ eDirectory
- grommunio: Support for 389 Directory Server
- grommunio: Support for Multi-Forest Active Directory installations
- grommunio: Support for IBM z15 (T02) mainframe
- grommunio: API extensions to support store-level operations, e.g. setting store permissions and store properties
- grommunio: Automatic restore of connections for long-lived and/or error-prone connections (libexmdbpp)
- grommunio: Availability in OTC (Open Telekom Cloud) via T-Systems
- grommunio: Availability of grommunio Antispam web interface via grommunio Admin API
- grommunio: Enhancements to BSD and library compatibility (e.g. LibreSSL)
- grommunio: Integration of grommunio Office and grommunio Archive now also for appliance users (grommunio-setup)
- grommunio: Multi-Server management with integrated placement policy engine, integrated in Admin API
- grommunio: Several documentation upgrades, including Debian and Ubuntu
- grommunio: Several security-related enhancements and optimizations
- grommunio: Simplification of deployment architecture ultra-scalable container deployments (docker, kubernetes)
- grommunio: Switch to AF_LOCAL sockets eliminating TCP overhead for socket connections
- grommunio: User template defaults for user creation (via CLI and UI) for mass deployment
- grommunio Groupware: Configuration parameters enabling enhanced analysis for professionals, e.g. imap_cmd_debug
- grommunio Groupware: Enhancements to service plugins and additional capabilities such as store cleanup (deleted items)
- grommunio Groupware: Extension of analytic tools, such as gromox-dscli for autodiscover connectivity analysis
- grommunio Groupware: Introduction of public folder read-state management flags
- grommunio Groupware: New migration tools for EML (rfc5322), iCalendar (ics) and vCard (vcf) import
- grommunio Groupware: Search enhancements, resulting in ~15-fold performance improvement with online search operations
- grommunio Groupware: Several enhancements to IMAP & POP daemons for more performance and stability
- grommunio Groupware: Several enhancements to existing migration tools (imapsync, kdb2mt, ...), filtering and partially even repairing broken data and migrating permissions where possible from the source

- grommunio Groupware: Several optimizations to cached mode handling, also making use of alternative return of states
- grommunio Groupware: Upgrade to FTS5 search index
- grommunio Groupware: Upgrade-capability of user stores for further extensibility in feature set
- grommunio Web: Allow setting recursive permissions by copying changes to lower hierarchy objects
- grommunio Web: Enhancements to multiple contactfolder scenarios with logical filters (contacts with e-mail addresses)
- grommunio Web: Integration of S/MIME management with support for multiple S/MIME keys and key management
- grommunio Web: Integration of grommunio Archive
- grommunio Web: Integration of grommunio Files with multiple account management
- grommunio Web: Integration of grommunio Office with realtime collaboration editing on Office Documents
- grommunio Web: Integration of online maps, based on OSM (OpenStreetMap), for contacts and global contacts
- grommunio Web: Performance optimizations, delivering with intermediary caches and large object size reduction, resulting in 4+-fold delivery speed to user
- grommunio Web: Several editor enhancements, e.g. extensive copy & paste compatibility with office documents
- grommunio Web: Several style and compatibility enhancements, e.g. enhanced printing format and favorite folder handling
- grommunio Web: Support for multi-hierarchy-level search without performance penalties
- grommunio Web: Support for prefix-based search operations, e.g. ”gro” -> ”grommunio”
- grommunio Web: Translation updates, now including all modules of grommunio Web
- grommunio Sync: Enhanced MIME (rfc822, rfc2822) and S/MIME support
- grommunio Sync: Performance improvements with redis-based state management > 100 kops (thousand operations per second) per instance possible
- grommunio Sync: Public folder sharing capabilities
- grommunio Chat: Support for enhanced operations (delete)
- grommunio Meet: Automatic disabling of media sharing when video sender limit reached
- grommunio Meet: Dynamic rate limiting, automatic video stream prioritization
- grommunio Meet: Integration of polls and polls management
- grommunio Meet: Various bridge-related enhancements, especially with stream bridges
- grommunio Meet: Various enhancements to breakout room management (notifications)
- grommunio Archive: Automatic key generation, sphinx enhancements
- grommunio Archive: Simplified installation via grommunio-setup
- grommunio Office: Automatic font management/generation via system-installed fonts (ds-fontgen)
- grommunio Office: Simplified installation via grommunio-setup

Only Available for customers/partner with privileged access (beta approval):

- grommunio: Preliminary Support for Red Hat Enterprise 9 (Stream, beta)
- grommunio: Preliminary Support for SUSE Liberty Linux

- grommunio Meet: Microsoft Outlook plug-in for meeting management
- grommunio Meet: Office/Meet integration
- grommunio Meet: Whiteboard integration
- grommunio Chat: Integration of Matrix (Homeserver+Element)

As usual, numerous bug fixes and features have been included. The release notes just highlight the major changes - Feel free to check out the detailed logs at GitHub

The official documentation covers the necessary steps for the update procedure.

We would like to thank the community for their ongoing contributions, but especially to:

- Jens Schleusener, who has provided tools for spell checking via FOSSIES codespell
- Robert Nagy, who has provided various contributions to support OpenBSD
- Walter Hofstädtler, who has provided various contributions for automating imports from MS Exchange and Kopano.

10.9 grommunio 2021.08.3

- Release type: Minor
- Release date: 8th of February 2022
- General availability: Yes
- grommunio: Support for Univention Corporate Server 5
- grommunio: Support for Red Hat Directory Server
- grommunio: Support for FreeIPA, incl. duplicate primary attributes
- grommunio: Support for Kong gateway
- grommunio: Support for APISIX gateway
- grommunio: Support for Kemp load balancer
- grommunio: Support for IBM Power10
- grommunio: Enhancements to haproxy scaling with support for 100k+ concurrent ingres connections
- grommunio: New index service for pre-indexing of web contents
- grommunio: Availability of submission service
- grommunio: Highest SSL/TLS standards according to QualysLabs A+ certification
- grommunio: Enhanced security/privacy by use of HSTS, CSP and HTTP Permissions-Policy
- grommunio: Advanced compression of HTTP(S)-enabled streams (Brotli)
- grommunio: Introduction of privilegeBits (Chat, Video, Files, Archive)
- grommunio: Mainstream availability of grommunio-archive (also to community)
- grommunio: Task management for asynchronous handling of tasks with longer duration (TasQ)
- grommunio: Thread-safe LDAP adaptor service (API)
- grommunio Groupware: Full support for S/MIME and GPG via (Outlook) MAPI/HTTP, MAPI/RPC and other clients (IMAP/POP/SMTP)
- grommunio Groupware: Auto-attach of shared mailboxes via AutoDiscover/Web with full owner permissions
- grommunio Groupware: Language-independent folder migration mapping

- grommunio Groupware: Migration script for Exchange (online/on-premise) to grommunio
- grommunio Groupware: Hidden folder control with migrations
- grommunio Groupware: Enhanced support for multi-value variable-length property types
- grommunio Groupware: Support for language-based stores at creation time (mkprivate / mkpublic)
- grommunio Web: Automatic addition of stores with full owner permissions (additional mail-boxes)
- grommunio Web: Set Out of Office information for other users (with full permissions)
- grommunio Web: Enhancements to session & store management (Performance, Languages, ...)
- grommunio Web: Support for Microsoft Exchange compatible ACLs and profiles (editor, author, ...)
- grommunio Web: Enhance search result limit to 1000 results
- grommunio Web: Editor upgrade to TinyMCE 4.9.11 with preparation to Tiny 5+
- grommunio Web: Language updates (English, German, Russian, Hungarian, Danish, ...)
- grommunio Web: Enhancements to user experience (style, compatbility, performance)
- grommunio Web: Fix missing font definition for new mails and inline comments
- grommunio Web: Fix Task requests with Outlook interoperability
- grommunio Web: Fingerprinting fixes (Firefox ESR)
- grommunio Web: Support for shallow MDM devices
- grommunio Web: W3C CSS 3 + SVG certification
- grommunio Web: Update dompurify (XSS protection)
- grommunio Web: Web application static resource delivery (payload reduction & performance) enhancements
- grommunio Sync: Reduction of memory footprint per EAS device by 24%
- grommunio Sync: Fixes/Enhancements based on static code analysis
- grommunio Chat: Update to 6.2.1

Only Available to customers/partner access (beta approval):

- grommunio Chat: Integration of Matrix (Homeserver+Element)
- grommunio: Support for IBM z15 (T02) mainframe
- grommunio: Preliminary Support for Ubuntu 22.04 (finished at Ubuntu's release date)
- grommunio: Preliminary Support for SUSE Liberty Linux

The official Documentation covers the necessary steps for the update procedure.

10.10 grommunio 2021.08.2

- Release type: Minor
- Release date: 24th of November 2021
- General availability: Yes

Major changes:

- grommunio: Production availability of Debian 11 via repository
- grommunio: Availability of grommunio mobile apps via the App Store and Playstore
- grommunio: Support for stretched cluster installations
- grommunio: Preliminary support for OpenID Connect via Keycloak
- grommunio Web: Major upgrade including over 230 fixes, updated WYSIWYG editor, design and performance improvements
- grommunio Groupware: Enhanced Out-of-Office autoresponder implementation
- grommunio Groupware: Enhanced support for OP_MOVE rules processing
- grommunio Groupware: Enhanced vCard processing
- grommunio Groupware: Full multilingual mailbox support for 91 languages
- grommunio Groupware: Full support for mailbox owner mode
- grommunio Groupware: Full support for shared mailboxes
- grommunio Groupware: Import into public stores
- grommunio Groupware: Support for public folder access via EAS (Exchange ActiveSync)
- grommunio Groupware: Synchronization resiliency for offline mode with broken objects (named properties)
- grommunio Admin: Enhanced Active Directory Alias Support (Exchange compatible)
- grommunio Admin: Inline help for better understanding and easier administration
- grommunio Admin: Integration of remote wipe for Administrators via Admin UI/CLI
- grommunio Admin: License manager integration within Admin UI
- grommunio Admin: Reorganization of Admin UI for better usability
- grommunio Chat: Major upgrade to 6.1.1 with many fixes, style adoptions and seamless upgrade procedure
- grommunio Setup: Support for special characters under special circumstances with grommunio Meet and grommunio Files

The official Documentation covers the necessary steps for the update procedure.

10.10.1 Post-update tasks

When using the grommunio appliance, some packages (depending on your configuration) might require your configuration to be adapted:

The list of known files that can require adoption are due to configuration file extensions:

1. /etc/grommunio-antispam/local.d/redis.conf.rpm*
2. /etc/grommunio-web/config.php.rpm*
3. /etc/grommunio-chat/config.json.rpm*

4. /etc/prosody/prosody.cfg.lua.rpm*

If the configuration file has been replaced by a package update, the minimal approach is to copy the original configuration file back in place. It is recommended to make a backup beforehand and restart the respective service either via Admin UI/CLI or system console/ssh:

.. code-block: bash

cp /etc/prosody/prosody.cfg.lua /etc/prosody/prosody.cfg.lua.rpmnew cp /etc/prosody/prosody.cfg.lua.rpmsave /etc/prosody/prosody.cfg.lua systemctl restart prosody

10.11 grommunio 2021.08.1

- Release type: Major
- Release date: 17th of August 2021
- General availability: Yes

Major changes:

- Extension of distribution support and available repositories (SUSE Linux Enterprise Server 15, Red Hat Enterprise Linux 8 incl. derivatives)
- Extension of available processor architectures: ARM64, PowerPC (ppc64le) and IBM zSeries (s390x)
- New installation images: OVA (VMware), Docker, Raspberry Pi (4+)
- Live Status Overview and Mobile Device Status
- Support for Mobile Policies (MDM)
- Extensive enhancements to migration tools for migrating Exchange (PST), Kopano (DB/Attachments) and generic mail systems (IMAP/CalDAV/CardDAV)
- Support for Active Directory Forest installations
- Support for deputy configuration
- Extensions of the Free/Busy functionality
- Support for special control characters
- Configuration based integration of grommunio Files, Meet, Chat into grommunio Web
- Inclusion of grommunio Files, Meet, Chat and Archive in the installation images

---

Important: Due to https://grommunio.com/en/news-en/aus-grommunio-wird-grommuniogrommunio-becomes-grommunio , grammm was renamed to grommunio. We are aware that this creates some challenges for the migration of existing platforms. All subscription holders are eligible for free professional services for the migration process. For the migration process, the estimated time required to for the completion of migration is 5000 users per hour.

Due to the nature of the rebranding from grammm to grommunio, a simple, automated upgrade mechanism was not created. Subscription holders with update services enabled automatically have access to the services available by the distribution upgrade process. The configuration switchover (configuration, data) has not changed much, and therefore the migration process is possible with the respective configuration dumps.

10.11.1 grommunio Admin API

Repository: https://github.com/grommunio/admin-api

Code statistics:

- +15323 lines added
- -5131 lines removed

Commits:

- 2021-08: 16
- 2021-07: 33
- 2021-06: 22
- 2021-05: 15
- 2021-04: 20
- 2021-03: 14

10.11.1.1 New (Improvements)

- Add (in)active user count to domain
- Add CLI documentation
- Add CLI fs operations
- Add CLI config tracing
- Add CLI mconf reload
- Add IDN support and input validation
- Add LDAP server pooling
- Add access to user store properties
- Add authmgr configuration management
- Add database connection check and CLI safeguard
- Add device delete (resync) endpoint
- Add domain effective sync policy endpoint
- Add endpoints for user delegates
- Add fetchmail management
- Add format validation endpoint
- Add journald log viewer
- Add log message for failed logins
- Add mailq endpoint
- Add man pages
- Add nginx vhost status proxies
- Add permanent domain deletion to API
- Add possibility to filter sync top data
- Add public folder detail endpoint
- Add read-only permissions

- Add separate permissions and ownerships for mconf
- Add support for JSON serialized device states
- Add support for numeric permission strings
- Add systemctl enable/disable commands
- Add user device sync information endpoint
- Allow force updating LDAP config
- Automatically adapt to new schema version
- Change public folder IDs to string
- Change user sync data to normal array
- Enforce user delegate format
- Implement database-stored configurations
- Implement dbconf commit hooks
- Implement domain management via CLI
- Implement grommunio-chat interface
- Implement import of aliases from LDAP
- Implement organizations
- Implement public folder editing
- Implement remote CLI
- Improve API documentation
- Improve CLI logging output
- Improve LDAP configuration check
- Improve LDAP configuration via CLI
- Improve LDAP import ”no users” message
- Improve LDAP usability
- Improve automatic service reload
- Improve handling of unreadable config files
- Invalidate redis cache on sync policy update
- Move domain creation to orm
- Move user creation to orm
- Move user store access to separate endpoint
- Optimize domain and user setup
- Provide sync policies
- Relax startup database connection test
- Reload additional services on domain creation
- Reload gromox-adaptor service on domain creation
- Reload gromox-http service on user creation
- Reload services on LDAP config change
- Reload systemd after en- or disabling units
- Reorganize system admin capabilities

- Sort dbconf services and files alphabetically
- Support loading of JSON OpenAPI spec
- Support unlimited storage quotas
- Switch to shell-exec systemd control

10.11.1.2 Bugfixes

- Fix LDAP check crashing on invalid externalID
- Fix LDAP check not working with AD
- Fix PATCH roles not working properly
- Fix Python version lock in Makefile
- Fix autocomplete
- Fix bad response on domain creation failure
- Fix broken login with PyJWT 2
- Fix clean target grommunio-dbconf
- Fix crashes when MySQL is unavailable on startup
- Fix dbconf service endpoint not working
- Fix declarative base query using wrong session
- Fix handling of broken LDAP IDs
- Fix missing user delegates request body
- Fix numerical file permissions not working
- Fix traceback when aborting password reset
- Fix unaligned reads/writes exmdbpp
- Fix user password attribute
- Fix wrong HTTP status on dashboard service signal
- Fix wrong redis key used for policy invalidation
- Fix wrong service signal response code
- Ignore incomplete LDAP objects

10.11.1.3 Removed

- Remove database URL quoting
- Remove fetchmail entries from profile endpoint
- Remove Flask-SQLAlchemy dependency
- Remove groups
- Remove old systemd code
- Remove permissions and roles on domain purge
- Remove PyJWT version constraint
- Remove unused dbus import

10.11.2 grommunio Admin Web

Repository: https://github.com/grommunio/admin-web

Code statistics:

- +43319 lines added
- -18542 lines removed

Commits:

- 2021-08: 10
- 2021-07: 52
- 2021-06: 28
- 2021-05: 46
- 2021-04: 53
- 2021-03: 47

10.11.2.1 New (Improvements)

- Add Circular progress to login button while logging in
- Add LDAP config parameter 'aliases'
- Add LDAP filter defaults
- Add auth manager config
- Add autocompletes for domain.org and mlist.class
- Add checkbox to set when putting LDAP config
- Add confirm dialog for stop/restart service buttons
- Add count of tablerows above tables
- Add createRole query param to POST /system/domains
- Add dashboard for domain admin
- Add displayname to headline of user details
- Add email to fetchmail dialog headline
- Add form autofill attributes to LDAP config
- Add human readable MSE to slider
- Add icon to get back to users view when in LDAP view
- Add indication of LDAP user sync at LDAP config view
- Add missing autocompletes
- Add more LDAP tooltips
- Add name and id attribute to login form
- Add new LDAP import buttons
- Add new orgAdmin and DomainPurge role
- Add new table view wrapper
- Add org to domain
- Add placeholder to LDAP server TF

- Add possibility to set 0 MB as quota limits
- Add scroll: auto to drawer
- Add send and receive quota to AddUser dialog
- Add service detail page
- Add sync statistics
- Add sync tab to user page
- Add tooltip with service description to service list
- After successfully adding an item, set loading to false
- After successfully importing/syncing users, refetch users
- Always divide quotas by 1024 before calculating size unit
- Automatically uppercase ssl fingerprints of fetchmail entries
- Button colors expanded with signal colors and adapted according to their function.
- Change AD to ActiveDirectory template
- Change default values of fetchmail dialog
- Change endpoint for quota values
- Change helpertexts of custom mapping
- Change logs hover color to work on light and dark mode
- Change role multiselect to autocomplete
- Check email and domain format with backend endpoint
- Completely remove swap chart if it's 0
- Convert folder match to local filtering
- Convert maxattrsize to MB
- Fetch domain lvl2 in user details to get chat-attribute
- Fill form when selecting LDAP template
- Fully reset store when logging out
- Get command name from code
- Implemented new responsive grid layout for the dashboard
- Implement CRUD for orgs
- Implement DBConf Filecreation
- Implement anti spam statistics into dashboard with a responsive layout
- Implement auto refresh of logs
- Implement autocomplete for AddRoles
- Implement autocomplete for Folders
- Implement class-members /-filters XOR
- Implement db file deletion
- Implement domain editing and deletion for OrgAdmins
- Implement dynamic table row fonts according to device status
- Implement fancy sorting algorithm for domain admin dashboard
- Implement fetchmail crud

- Implement file editing
- Implement folder editing
- Implement full domain deletion
- Implement grommunio chat team/user management
- Implement live server status page
- Implement local services filter
- Implement log viewer
- Implement mailq
- Implement minified sync policy prototype
- Implement new Chart designs
- Implement proper login form autocompletion
- Implement read-only capabilities/permissions
- Implement send/receive quota limit
- Implement service autostart
- Implement service deletion
- Implement service renaming
- Implement sync policy for users
- Implement sync policy prototype
- Implement sync table
- Implement sync table filters
- Implement used space bar
- Implement user delegates
- Implement vhost status endpoints
- Improve design of mailQ
- Improve design of quota graph
- Improve fetchmail
- Improve log viewer
- Improve sync table header
- Improve wording of owner removal
- Improved strings for LDAP configuration
- Increase size of services chart to prevent wrapping of deactivating chip
- LDAP: update textual requirements for server field
- Make all multiline textfields outlined
- Make deactivated domains re-activatable
- Make quotas optional for adding users
- Mark deleted domains as deleted in drawer
- More details in per-domain view
- Move used space percentage to center of bar
- New service chart design

- Rectify default values for LDAP fields
- Redesign quota chart
- Reduce count of mlists when deleting
- Relabel buttons for CNF clause
- Relabel quota error
- Rename RemoveOwner class
- Rename classes to groups on the outside (only displayed text)
- Reorganize ldap config
- Reorganize permission handling
- Resolve eqeqeq warning
- Resolve fetchmail warning
- Separate user and storeprops fetch in 2 different try/catch blocks
- Show domain displayname if it's different than the domainname
- Significantly improve data management
- Significantly improve design of sync policy mask
- Slightly improve padding and margin
- Split spam and performance into 2 chapters by headlines
- Translations
- Trim message about LDAP fields being optional
- Update LDAP tooltip strings
- Update counter after softdeleting domain
- Update mconf and ldap url
- When updating domainStatus, also update drawer domains
- Wrap detail view components in new wrapper
- View: fix also update timestamp

10.11.2.2 Bugfixes

- Fix broken classes fetch
- Fix broken dashboard layout
- Fix broken default vhost
- Fix broken domain patch
- Fix broken fetchOrgs and edit maillist
- Fix broken folder details
- Fix broken folder sorting
- Fix broken format check
- Fix broken grochat checkbox
- Fix broken ldap template select
- Fix broken parent groups
- Fix broken role editing

- Fix broken service disableing
- Fix broken table filters
- Fix broken toggleswitch
- Fix broken used space labels
- Fix broken user edit
- Fix chart issues
- Fix crashing empty-ldap view
- Fix crashing mlist details
- Fix crashing views
- Fix disk labels
- Fix doubling visual feedback of ldap responses
- Fix non-resizing charts
- Fix non-updating authBackendSelection
- Fix potential live status crashes
- Fix quota absence not displayed properly
- Fix tooltip warnings for link button
- Fix uncaught config.json error
- Fix valid domain names rejection
- Fix warnings
- Fix wrong default searchAttribute
- Fix wrong implementation of ldap enable-available-switch
- Properly show ldap ok-status

10.11.2.3 Removed

- Remove availability text if LDAP is disabled
- Remove chat user option in post dialog
- Remove empty limit parameter from entire app
- Remove error color from cancel button in AddDialogs
- Remove groups
- Remove password and make maxUser mandatory
- Remove redundant home icons in views
- Remove sorting from user list, besides username
- Remove srcFolder from required textfields and disable save-button if a required tf isn't filled

10.11.3 grommunio CUI

Repository: https://github.com/grommunio/grommunio-cui

Code statistics:

- +2565 lines added
- -2879 lines removed

Commits:

- 2021-08: 10
- 2021-07: 48
- 2021-06: 1
- 2021-05: 50
- 2021-04: 0
- 2021-03: 37

10.11.3.1 New (Improvements)

- Add cancel button to admin pw change dialog
- Add cancel button to reboot and shutdown question box
- Add checked information to homescreen
- Add footerbar for better keyboard shortcut readability
- Add help note to ”Change password” dialog
- Add last login time to bottom half of homescreen
- Add launcher script
- Add load average to footerbar and introduce quiet mode
- Add menu entry to reset AAPI password
- Add padded Edit class GEdit
- Add shutdown to menu
- Add some kbd layouts
- Add space to ”Average load”
- Add status messagebox after admin pw reset
- Add status messagebox after tymesyncd configuration
- Add timesyncd config to main menu
- Add timezone configuration via yast2
- Change Buttons to RadioButtons
- Change column size of menu field descriptions
- Change hidden keyboard switcher to menu guided
- Change netmask to cidr
- Change stupid cat command to pythons internal open
- Change wrap mode of all editable fields to ellipsis
- Check content of netifaces before getting default gw

- Correct indenting after event refactoring
- Create a general input box for changing admin-web password
- Create header for log viewer
- Create message after dns settings apply
- Delete redundant copy of README
- Disable mouse support as mentioned in #9
- Ditch ordered_set from requirement
- Ditch urwid>=2.1 requirement
- Do not check for timesyncd configuration
- Do not show gateway on lo
- Drop menu element number
- Enable /etc/hosts writing
- Enhance GText class with some additional methods
- Enhance dialog sizes of IP address and DNS config
- Escape the quote at the system call for changing admin-web password
- Finish log viewer
- Give menu items more contrast
- Handle footerbar correctly if screen width changes
- Introduce a general Text class padding the correct chars
- Keyboard layout switcher
- Make function check_if_password_is_set available for all
- Make getty upbranding compatible
- Make homescreen more readable
- Make it upbranding compatible
- Make rest upbranding compatible
- Make some checks more exact
- Move timsyncd configuration behind timezone configuration
- New program names in help texts
- Optimize further wording
- Optimize logging support
- Optimize wording
- Read *grommunio-admin config dump* and extract the log units
- Reboot when asked for reboot, don't poweroff
- Recolor footerbar
- Rectify indent of docstrings
- Reduce from unnecessary 3 digits to 2 digits in average load view
- Reduce length of keyb/color line
- Replace custom netconfig implementation by yast2
- Replace incorrect credentials message

- Replace windowed shell by fullscreen one
- Restore termios setting when CUI exits
- Revert "Remove systemd from requirements because it is already in systemd-python."
- Reword main menu texts
- Set up environment variables for terminal shell
- Show IPv6 addresses in overview
- Split large handle_event function
- Stop abusing str() to test for classes/enums
- Suppress messages of shell commands
- Switch to RGB444 format
- Tone down brightness of the "dark" scheme
- Tone down reverse color in light mode
- Trim excessive sentence punctuation/structuring
- Update header to be more suitable to the new footerbar
- Update systemd module requirement
- Use "reboot" command without path
- Use autologin if no initial password is set
- Use long names in binaries again and rename gro* to grommunio-*
- Use systemd-journal instead of viewing log files directly

10.11.3.2 Bugfixes

- Fix admin api pw reset and use better wording
- Fix bug on keyboard change while in main menu
- Fix correct display of distro and version
- Fix crash on starting if no grommunio-admin was present
- Fix hanging in menu while colormode or kbd switching
- Fix missing captions on some formatting calllls of GEdit
- Fix not closing password change dialog on hitting close with enter
- Fix out of bounds on the right side of log viewer
- Fix returning back from unsupported shell
- Fix shell injection bug on resetting admin pw
- Fix some config file issues on writing
- Fix suboptimal contrast in "light" mode
- Fix tab handling lock after message- or input box call
- Fix that only one time logging is needed
- Fix wrong 'NOTHING' message if only enter being pressed
- Fix wrong admin interface url
- Fix wrong color switching in menues
- Fix wrong current window setting on input boxes

- Fix wrong explaining text on first menu start
- Fix wrong logging formating
- getty: do set up stderr as well

10.11.3.3 Removed

- Remove ”activated by what” and check privileges.
- Remove arbitrary startup wait phase
- Remove extraneous HL coloring
- Remove inconsistent status bar coloring
- Remove systemd from requirements because it is already in systemd-python.
- Remove the ’heute’ clockstring.
- Remove unnecessary border around mainwindow
- Remove wrong hint to yast.

10.11.4 grommunio Core (gromox)

Repository: https://github.com/grommunio/gromox

Code statistics:

- +65616 lines added
- -95032 lines removed

Commits:

- 2021-08: 78
- 2021-07: 207
- 2021-06: 197
- 2021-05: 159
- 2021-04: 308
- 2021-03: 256

10.11.4.1 New (Improvements)

- adaptor: reduce main() unwinding boilerplate
- adaptor: use stdlib containers for data_source
- alias_translator: add PLUGIN_RELOAD functionality
- alias_translator: expand mailaddr buffers to UADDR_SIZE
- all: add <cerrno> include for errno
- all: avoid integer underflow in qsort comparators
- all: check return values of ext_buffer_push_*
- all: delete extra blank lines from header files
- all: disambiguate multiply assigned error/warning codes
- all: drop C (void) argument filler

- all: drop _stop() function return values
- all: ease setting breakpoints on thread entry functions
- all: enlarge buffers for IPv6 addresses
- all: favor simpler x[j] over *(x+j)
- all: log all pthread_create failures
- all: make use of EXT_PULL::g_*bin* member functions
- all: make use of EXT_PULL::g_bool member functions
- all: make use of EXT_PULL::g_bytes member functions
- all: make use of EXT_PULL::g_guid* member functions
- all: make use of EXT_PULL::g_proptag_a member functions
- all: make use of EXT_PULL::g_restriction member functions
- all: make use of EXT_PULL::g_str* member functions
- all: make use of EXT_PULL::g_tpropval_a member functions
- all: make use of EXT_PULL::g_uint* member functions
- all: make use of EXT_PULL::* member functions
- all: make use of EXT_PUSH::{advance,p_proptag_a} member functions
- all: make use of EXT_PUSH::{check_ovf,p_tpropval_a,p_tarray_set} member functions
- all: make use of EXT_PUSH::{init,p_guid,p_bool} member functions
- all: make use of EXT_PUSH::* member functions
- all: make use of EXT_PUSH::{p_bin,p_bin_s,p_bin_a,p_restriction} member functions
- all: make use of EXT_PUSH::p_int* member functions
- all: make use of EXT_PUSH::{p_msgctnt,p_eid_a,p_abk_eid} member functions
- all: make use of EXT_PUSH::{p_store_eid,p_folder_eid,p_msg_eid} member functions
- all: make use of EXT_PUSH::{p_str,p_wstr,p_bytes} member functions
- all: make use of EXT_PUSH::{p_tagged_pv,p_oneoff_eid,p_proprow} member functions
- all: make use of EXT_PUSH::p_uint* member functions
- all: make use of EXT_PUSH::{release,p_xid,p_bin_ex} member functions
- all: print connecting module together with gx_inet_connect error messages
- all: reduce verbosity of pext->alloc()
- all: replace awkward multiply-by-minus-1
- all: replace memset by shorter initialization
- all: replace memset with hardcoded sizes
- all: replace sprintf by snprintf
- all: reset deserializer struct counts on allocation failure
- all: resolve instances of -Wunintialized
- all: speedier shutdown of sleepy threads
- all: switch plugins to return true for unhandled plugin calls
- all: switch ports to uint16 / resolve instances of -Wformat
- all: switch *_stop variables to atomic<bool>

- all: switch to EXT_PULL::init
- all: use anonymous namespaces for TU-local struct declarations
- authmgr: delete unused mode argument
- authmgr: implement ”allow_all” auth mode
- authmgr: make login check isochronal
- authmgr: move up too-late return value check of mysql_meta
- authmgr: support config reloading
- authmgr: switch default mode to ”externid”
- bodyconv: add rtfcptortf to option summary
- bodyconv: better error message when rtfcptortf fails
- build: add another symbol to zendfake
- build: add cryptest.cpp
- build: add ldd check for mapi.so
- build: add libgromox_common to pffimport link
- build: add libgromox_mapi to pffimport link
- build: add missing <mutex> include
- build: add plugin support functions
- build: change qconf to use -Oo
- build: deal with php-config which has no --ini-dir
- build: delete sa_format_area.sh
- build: installation order of LTLIBRARIES is significant
- build: libpthread is needed for logthru
- build: make struct BINARY_ARRAY trivial again
- build: make struct PROPTAG_ARRAY trivial again
- build: move ext_buffer.cpp into libgromox_common.la
- build: move pffimport manpage to section 8gx
- build: pass -fsanitize to linker as well when using --with-asan/ubsan
- build: quench compiler warnings on autolocking libcrypto implementations
- build: quench gcc-7 compiler warnings for -Wunused*
- build: reorder php-config calls and show immediate results
- build: resolve instance of -Wformat-overflow
- build: resolve attempts at narrowing conversion under -funsigned-char
- build: scan for more variants of php-config
- build: support OpenLDAP 2.5
- build: use AC_PATH_PROGS to make deptrace recognize the PHP dependency
- build: zendfake needs a non-noinst LTLIB
- daemons: add ctor/dtor for main process contexts
- daemons: add missing reporting of gx_inet_connect failures
- daemons: delete use of ip6_container, ip6_filter

- daemons: set up SIGINT handler like SIGTERM
- daemons: upgrade to POSIX signal functions
- daemons: use inheritance to base off SCHEDULE_CONTEXT
- dbop: add ”fetchmail” table
- dbop: add fetchmail table for dbop -C
- dbop: add missing classes.filters for new db setups
- dbop: add table ”configs”
- dbop: add users.chat_id and domains.chat_id
- dbop: add users.sync_policy and domains.sync_policy
- dbop: error when schema version unobtainable
- dbop: make user_properties table fit for multivalue props
- delivery: abolish pthread_cancel
- delivery: abolish unnecessary (a+i)-> syntax
- delivery: add missing mutex unlock
- delivery: add missing pthread_join calls
- delivery: delete unneeded pthread_setcanceltype call
- doc: add Autodiscovery manpage
- doc: add document for the RWZ stream/file format
- doc: add general notes for logon_object_get_properties
- doc: add manpage for gromox-abktconv
- doc: add manpage for gromox-abktpull
- doc: add manpages for gromox-kpd2mt
- doc: add Name sections to all pages
- doc: add notes about character set woes
- doc: authmgr has relaxed requirement on ldap_adaptor
- doc: Autodiscover corrections to mod_fastcgi
- doc: bulletize FILES sections
- doc: delete obsolete digest.8gx manpage
- doc: detail on addressEntryDisplayTableMSDOS
- doc: do not escape (
- doc: expand on the relationship between DCERPC, EMSMDB and OXCROPS
- doc: mark up tcp_mss_size default value
- doc: mention caching behavior for PR_EC_WEBACCESS_JSON
- doc: mention exchange_emsmdb.cfg:rop_debug
- doc: mention openldap as build requirement
- doc: move exrpc_debug explanation to exmdb_provider.4gx
- doc: note about variability of ${libdir}
- doc: rearrange aux utilities in gromox.7
- doc: replace roff SS command by TP

- doc: show right option combinations for gromox-pffimport
- doc: turn oxoabkt.txt to rST
- doc: update documentation pertaining to MAPIHTTP and norms
- doc: update event.8gx
- doc: upgrade changelog.txt to changelog.rst
- doc: use default indent for RS command
- doc: use the right rST syntax for literal code blocks
- doc: use the right syntax for literal blocks
- email_lib: qp_decode_ex's return value needs proper type
- emsmdb: deindent logon_object_get_named_{propids,propnames}
- event: add another termination checkpoint
- event: add missing pthread_join for accept/scan threads
- event: kick threads with a signal upon termination request
- event_proxy: reduce excess gx_inet_connect messages
- event: reduce main() unwinding boilerplate
- event: replace pthread_cancel by pthread_join
- event: resolve buffer overrun in ev_deqwork
- event: switch g_dequeue_lists to a stdlib container
- event: switch g_enqueue_lists to a stdlib container
- event: switch g_host_list to a stdlib container
- event: switch HOST_NODE::phash to a stdlib container
- event: switch listnode allocations to new/delete
- event: switch to std::mutex
- exch: add length parameter to common_util_addressbook_entryid_to_username
- exch: add length parameter to common_util_check_delegate
- exch: add length parameter to common_util_essdn_to_username
- exch: add length parameter to common_util_parse_addressbook_entryid
- exch: add length parameter to *_to_essdn functions
- exchange_emsmdb: add directive exrpc_debug
- exchange_emsmdb: add length parameter to common_util_entryid_to_username
- exchange_emsmdb: add length parameter to common_util_essdn_to_username
- exchange_emsmdb: add variable for enabling trivial ROP status dumps
- exchange_emsmdb: allow setting rop_debug from config file
- exchange_emsmdb: change ATTACHMENT_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change FASTDOWNCTX_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change FASTUPCTX_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change FTSTREAM_PARSER freestanding functions to member funcs
- exchange_emsmdb: change ICSDOWNCTX_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change ICSUPCTX_OBJECT freestanding functions to member funcs

- exchange_emsmdb: change MESSAGE_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change STREAM_OBJECT freestanding functions to member funcs
- exchange_emsmdb: change SUBSCRIPTION_OBJECT freestanding functions to member funcs
- exchange_emsmdb: collect magic array size into a mnemonic
- exchange_emsmdb: compact common subexpressions
- exchange_emsmdb: compact common_util hook definitions
- exchange_emsmdb: compact exmdb_client declaration boilerplate
- exchange_emsmdb: compact exmdb_client hook definitions
- exchange_emsmdb: compact if-1L-1L blocks to use ?:
- exchange_emsmdb: compact if-1L-1L into ?:
- exchange_emsmdb: compact repeated expression (T*)expr
- exchange_emsmdb: const qualifiers for logon_object_check_readonly_property
- exchange_emsmdb: deindent ftstream_parser_read_element
- exchange_emsmdb: deindent oxcfold_deletemessages
- exchange_emsmdb: deindent rop_syncimportdeletes
- exchange_emsmdb: delete unused function folder_object_get_tag_access
- exchange_emsmdb: delete unused function table_object_get_table_id
- exchange_emsmdb: emit MID during rop_sendmessage as hex
- exchange_emsmdb: kick threads with a signal upon termination request
- exchange_emsmdb: make folder_object_* member functions
- exchange_emsmdb: make logon_object_check_private a member function
- exchange_emsmdb: make logon_object_get_account a member function
- exchange_emsmdb: make logon_object_get_dir a member function
- exchange_emsmdb: make logon_object_guid a member function
- exchange_emsmdb: make logon_object_* member functions
- exchange_emsmdb: quench repeated ((T*)expr)
- exchange_emsmdb: reduce indent in ftstream_producer_write_groupinfo
- exchange_emsmdb: reduce indent in rop_querynamedproperties
- exchange_emsmdb: repair botched access check in rop_syncconfigure
- exchange_emsmdb: replace folder_object_get_calculated_property silly casts
- exchange_emsmdb: restore MOH functions
- exchange_emsmdb: rework return codes for emsmdb_interface_connect_ex
- exchange_emsmdb: source inline folder_object_get_id
- exchange_emsmdb: source inline folder_object_get_type
- exchange_emsmdb: source inline logon_object_get_account_id
- exchange_emsmdb: source inline logon_object_get_logon_mode
- exchange_emsmdb: source inline logon_object_get_mailbox_guid
- exchange_emsmdb: source inline table_object_get_rop_id
- exchange_emsmdb: store ownership bit

- exchange_emsmdb: substitute lookalike variable names
- exchange_emsmdb: switch to std::mutex
- exchange_emsmdb: trim goto from emsmdb_interface_connect_ex
- exchange_emsmdb: trim single-use variables in ftstream_producer
- exchange_emsmdb: turn freestanding FTSTREAM_PRODUCER functions into member ones
- exchange_emsmdb: turn freestanding ICS_STATE functions into member ones
- exchange_emsmdb: use ”auto” specifier with common_util_get_propvals
- exchange_emsmdb: use ”auto” specifier with emsmdb_interface_get_emsmdb_info
- exchange_emsmdb: use ”auto” specifier with rop_processor_get_logon_object
- exchange_emsmdb: use mnemonic names for RPC opnums
- exchange_emsmdb: wrap FASTDOWNCTX_OBJECT in unique_ptr
- exchange_emsmdb: wrap FASTUPCTX_OBJECT in unique_ptr
- exchange_emsmdb: wrap FTSTREAM_PARSER in unique_ptr
- exchange_emsmdb: wrap FTSTREAM_PRODUCER in unique_ptr
- exchange_emsmdb: wrap ICS_STATE in unique_ptr
- exchange_emsmdb: wrap LOGON_OBJECT in unique_ptr
- exchange_emsmdb: wrap STREAM_OBJECT in unique_ptr
- exchange_emsmdb: wrap SUBSCRIPTION_OBJECT in unique_ptr
- exchange_nsp: add length parameter to ab_tree_get_display_name
- exchange_nsp: add PLUGIN_RELOAD functionality
- exchange_nsp: adjust ab_tree code to zcore ab_tree again
- exchange_nsp: clear some type overlaps
- exchange_nsp: comapct if-1L-1L blocks to use ?:
- exchange_nsp: combine LPROPTAG_ARRAY / MID_ARRAY
- exchange_nsp: combine STRING_ARRAY / STRINGS_ARRAY
- exchange_nsp: compact repeated expression (T*)expr
- exchange_nsp: deindent ab_tree_get_node_type, ab_tree_get_server_dn
- exchange_nsp: dissolve 11 type aliases
- exchange_nsp: dissolve 4 type aliases
- exchange_nsp: drop implicit conversion of AB_BASE_REF
- exchange_nsp: replace custom AB_BASE_REF by unique_ptr-with-deleter
- exchange_nsp: resolve some copy-paste flagged code
- exchange_nsp: switch g_base_hash to a stdlib container
- exchange_nsp: switch to documented MAPI type names
- exchange_nsp: switch to std::mutex
- exchange_nsp: use implicit conversion from nullptr to AB_BASE_REF
- exchange_nsp: use mnemonic names for RPC opnums
- exchange_rfr: add length parameter to rfr_get_newdsa
- exchange_rfr: use mnemonic names for RPC opnums

- exch: centralize pidlid constants
- exch: change overlapping variable names g_cache_interval
- exch: compact conditional expressions around sqlite3_step
- exch: compact repeated logic involving rop_make_util_*_guid
- exch: compact return expressions
- exch: compact tag list modifications
- exch: construct SQL queries with snprintf rather than sprintf
- exch: CSE-combine permission checks
- exch: cure overlapping variable names (improve debugging)
- exch: deduplicate exmdb_ext.cpp
- exch: deduplicate struct DB_NOTIFY_DATAGRAM
- exch: deduplicate struct EXMDB_REQUEST
- exch: deduplicate struct EXMDB_RESPONSE
- exch: delete empty functions
- exch: delete xstmt::finalize calls before return
- exch: delete xstmt::finalize calls near end of scope
- exch: expand char arrays to hold usernames (emailaddrs)
- exch: implement send quota
- exch: make IDL-generated **exmdb_client_** functions part of a namespace
- exch, mda, mra: add SIGHUP handler
- exch: MH support
- exchnage_nsp: make calls to ab_tree_put_base automatic
- exch: read delegates.txt with a consistent list format
- exch: reduce excess gx_inet_connect messages
- exch: reduce verbosity of ndr_stack_alloc
- exch: rename source directory str_filter to match plugin name
- exch: resolve instances of -Wmissing-braces
- exch: resolve cov-scan reports
- exch: roll nullptr check into xstmt::finalize
- exch: switch to std::mutex
- exch: switch to std::shared_mutex
- exch: trim nullptr post-assignment for xstmt
- exch: use ”auto” specifier with get_rpc_info
- exch: wrap ATTACHMENT_OBJECT in unique_ptr
- exch: wrap FOLDER_OBJECT in unique_ptr
- exch: wrap ICSDOWNCTX_OBJECT in unique_ptr
- exch: wrap ICSUPCTX_OBJECT in unique_ptr
- exch: wrap MESSAGE_OBJECT in unique_ptr
- exch: wrap TABLE_OBJECT in unique_ptr

- exmdb_client: drop extra payload_cb==0 check
- exmdb_local: silence a cov-scan warning
- exmdb_provider: add destructor for IDSET_CACHE
- exmdb_provider: add length parameter to common_util_entryid_to_username
- exmdb_provider: add missing pointer advancements in message_rectify_message
- exmdb_provider: add missing return statements after db_engine_put_db
- exmdb_provider: add unwinding for plugin startup
- exmdb_provider: add/utilize xstmt::finalize
- exmdb_provider: add variable for enabling trivial RPC status dumps
- exmdb_provider: allow reduction of cache_interval down to 1s
- exmdb_provider: bump default limits for stub threads and router connections
- exmdb_provider: change g_connection_list to a stdlib container
- exmdb_provider: change g_router_list to a stdlib container
- exmdb_provider: compact common subexpressions
- exmdb_provider: compact common_util hook definitions
- exmdb_provider: compact exmdb_client hook registrations
- exmdb_provider: compact if-1L-1L into ?:
- exmdb_provider: compact long common subexpressions
- exmdb_provider: compact repeated error checking
- exmdb_provider: compact repeated expression (T*)expr
- exmdb_provider: cure nullptr dereferences in ext_rule OP_FORWARD processing
- exmdb_provider: cure "SELECT count(idx)" error messages
- exmdb_provider: decide for sqlite3_finalize based upon pointer to be freed
- exmdb_provider: deindent table_load_content_table
- exmdb_provider: deindent table_load_hierarchy
- exmdb_provider: dissolve goto statements in db_engine_notify_content_table_add_row
- exmdb_provider: dissolve goto statements in exmdb_server_get_content_sync
- exmdb_provider: dissolve goto statements in table_load_content_table
- exmdb_provider: emit log message when sqlite DBs cannot be opened
- exmdb_provider: emit warning when folder_type is indeterminate
- exmdb_provider: enable ctor/dtor on OPTIMIZE_STMTS
- exmdb_provider: factor out folder name test into separate function
- exmdb_provider: reduce indent in exmdb_parser.cpp:thread_work_func
- exmdb_provider: reduce indent in folder_empty_folder
- exmdb_provider: reduce variable scope in folder_empty_folder
- exmdb_provider: reload exrpc_debug variable on SIGHUP
- exmdb_provider: reorder error case handling in exmdb_server_create_folder_by_properties
- exmdb_provider: reorder if-else blocks in table_load_content_table to facilitate deindent
- exmdb_provider: reorder if-else blocks in table_load_hierarchy to facilitate deindent

- exmdb_provider: replace pthread_cancel by join procedure
- exmdb_provider: retire W-1299 warning
- exmdb_provider: scoped cleanup for DB_ITEM objects
- exmdb_provider: set PR_READ based upon PR_MESSAGE_FLAG
- exmdb_provider: show exrpc requests with succinct result code
- exmdb_provider: silence unchecked return values in exmdb_server_set_message_instance_conflict
- exmdb_provider: simplify parts of folder_empty_folder
- exmdb_provider: split common_util_get_properties into more sensible subfunctions
- exmdb_provider: stop using strncpy
- exmdb_provider: switch g_hash_list to a stdlib container
- exmdb_provider: switch largely to std::mutex
- exmdb_provider: use ”auto” keyword around gx_sql_prep
- exmdb_provider: use ”auto” specifier with instance_get_instance
- exmdb_provider: warn when store directory inaccessible
- exmdb_provider: wrap DB_ITEM in a unique_ptr
- exmdb_provider: wrap sqlite3_close in an exit scope
- freebusy: centralize pidlid constants
- freebusy: compact if-1L-1L blocks to use ?:
- http: add idempotent return stmts to facilitate deindent
- http: add plugin support functions
- http: better status codes when FastCGI is not available
- http: centralize call to http_end
- http: compact read/SSL_read calls in http_parser_process
- http: deindent htparse_*
- http: deindent pdu_processor_destroy
- http: drop implicit conversion of VCONN_REF
- http: emit status 503 for ”out of resources” cases
- http: factor out building of 408-typed response
- http: factor out building of 4xx-typed response
- http: factor out building of 5xx-typed response
- http: factor out END_PROCESSING code block from http_parser_process
- http: make calls to http_parser_put_vconnection automatic
- http: make the different 503 response codes more discernible
- http: move rfc1123_dstring to lib and add a size argument
- http: narrow the scope of http_parser_process local variables
- http: quench ”unloading <nothing>” messages
- http: reduce messages’ log level from 8 to 6
- http: reorder if-else branches to facilitate deindent
- http: section htparse_* into lambdas for function splitting

- http: section http_parser_process into lambdas for function splitting
- http: split function http_parse_process
- http: split functions htparse_rdhead, htparse_rdbody, htparse_wrrep, htparse_wait
- http: switch g_vconnection_list to a stdlib container
- http: switch HPM plugin list to a stdlib container
- http: switch largely to std::mutex
- http: switch PDU plugin list to a stdlib container
- http: switch service plugin list to a stdlib container
- http: trim use of strncpy / adjust buffer sizes
- http: use ”auto” keyword around http_parser_get_vconnection
- imap: break up imap_parser_process into more sensible subfunctions
- imap: cleanup unused variables
- imap: clear ineffective unsigned comparison
- imap: compact repeated expression (T*)expr
- imap: compact repeated midb error reporting
- imap: compact standardized response line emission
- imap: cure an uninitialized variable issue in ps_stat_appending
- imap: deindent imap_cmd_parser.cpp
- imap: deindent imap_cmd_parser_password2
- imap: deindent imap_parser_process subfunctions
- imap: delete IMAP_CODE enum and reduce numeric range
- imap: delete netconsole routine for imap_code
- imap: delete parsing of imap_code.txt
- imap: do not advertise RFC2971 commands when so disabled
- imap: invert imap_parser_process’s if conditions to facilitate deindent
- imap: pass full buffer size to sprintf
- imap: quote folder names in LIST, LSUB, XLIST, STATUS results
- imap: reduce scope of variables imap_parser_process
- imap: reduce scope of variables in imap_parser_process 2
- imap: resolve CHECKED_RETURN cov-scan warning
- imap: resolve memory leak in resource_load_imap_lang_list
- imap: standardized reporting of midb responses
- imap: trim some gotos from imap_parser_process
- imap: unbreak parsing of {} literals
- imap: use ”auto” specifier with resource_get_imap_code
- imap: use stdlib container for g_lang_list
- kdb2mt: heed SRCPASS environment variable
- kpd2mt: abandon enable_shared_from_this
- kpd2mt: add YError exception printer

- kpd2mt: support reading attachments
- ldap_adaptor: add missing std::forward<>()
- ldap_adaptor: add option to disable auth connection persistence
- ldap_adaptor: compact config log messages
- ldap_adaptor: establish all server connections on first demand only
- ldap_adaptor: guard against bad_alloc during reload
- ldap_adaptor: ignore search referrals emitted by MSAD
- ldap_adaptor: support config reloading
- ldap_adaptor: unconditionally initialize plugin
- ldap_adaptor: use proper parameters for ldap_sasl_bind simple binding
- lib: add allocator support for EXT_PUSH
- lib: add config_file_get_uint
- lib: add const qualifiers to stream functions
- lib: add const variants for the double_list API
- lib: add ctor/dtor for RTF_READER
- lib: add ctor/dtor to RTF_WRITER
- lib: add default functions for exmdb_rpc hooks
- lib: add dtor to EXT_PUSH
- lib: add exmdb_rpc_free hook
- lib: add generational support to resource_pool
- lib: add hex2bin function
- lib: add initializers for binhex.cpp:READ_STAT
- lib: add ip_filter_add to list of exempted warnings about svc funcs
- lib: add length parameter to GET_USERNAME
- lib: add **MAPI_E_** constants as comments to standard ec* codes
- lib: add member initialization to EXT_PULL/EXT_PUSH
- lib: add missing newline in slurp_file
- lib: add more codes to exmdb_rpc_strerror
- lib: add new fields for orgs user table
- lib: add OOP-style interface/member functions to EXT_PULL class
- lib: add OOP-style interface / member functions to EXT_PUSH class
- lib: add plugin call type RELOAD
- lib: add PST properties to mapidefs.h
- lib: add rights flag combinations
- lib: add SCHEDULE_CONTEXT::context_id to easier backreference program contexts
- lib: add textual descriptions for all known EC/RPC errors
- lib: add wrapper for sqlite3_stmt
- lib: adjust mime_get_mimes_digest, mime_get_structure_digest argument and return types
- lib: adjust parse_mime_field argument and return types

- lib: adjust qp_decode return type
- lib: allow redirecting HX_strlcpy to snprintf
- lib: automatic finalization of xstmt
- lib: automatic memory mgt for FOLDER_CONTENT
- lib: avoid double UTF-8 encoding by html_to_plain
- lib: avoid joining a non-existing thread
- lib: cease treating '#' in config values as comment
- lib: change FOLDER_CONTENT freestanding functions to member funcs
- lib: change mail_get_length return type to ssize_t
- lib: change overquota report code to MAPI_E_STORE_FULL
- lib: class maintenance on resource_pool
- lib: collect magic array size into a mnemonic
- lib: combine common expressions into function strange_roundup
- lib: combine copy-and-pasted code into exmdb_rpc_strerror
- lib: combine duplicated unique_tie implementation
- lib: combine underflow/overflow protection logic near add/subtract
- lib: comment out all unused proptags
- lib: compact repeated expression (T*)expr
- lib: consolidate exmdb socket read/write functions
- lib: convert incomplete and syntactically broken RTF anyway
- lib: deduplicate and use ACTTYPE names from documentation
- lib: deduplicate decls for ADVISE_INFO, NOTIF_SINK
- lib: deduplicate decls for FLATUID, FLATUID_ARRAY
- lib: deduplicate decls for MESSAGE_STATE, STATE_ARRAY
- lib: deduplicate decls for NEWMAIL_ZNOTIFICATION, OBJECT_ZNOTIFICATION
- lib: deduplicate decls for PERMISSION_ROW, PERMISSION_SET
- lib: deduplicate decls for PROPERTY_NAME, PROPNAME_ARRAY
- lib: deduplicate decls for PROPID_ARRAY
- lib: deduplicate decls for PROPTAG_ARRAY
- lib: deduplicate decls for RESTRICTION*
- lib: deduplicate decls for RULE_DATA
- lib: deduplicate decls for RULE_LIST
- lib: deduplicate decls for {SHORT,LONG,LONGLONG,STRING}_ARRAY
- lib: deduplicate decls for SORT_ORDER, SORTORDER_SET
- lib: deduplicate decls for struct BINARY, BINARY_ARRAY
- lib: deduplicate decls for struct GUID, GUID_ARRAY
- lib: deduplicate decls for TAGGED_PROPVAL, TPROPVAL_ARRAY, TARRAY_SET
- lib: deduplicate decls for ZNOTIFICATION, ZNOTIFICATION_ARRAY
- lib: deduplicate display type constants

- lib: deduplicate **exmdb_client_** declarations
- lib: deduplicate exmdb_rpc.cpp
- lib: deduplicate PidLid constants
- lib: deduplicate **PLUGIN_** definitions
- lib: deduplicate **resource_get_** defines
- lib: deindent exmdb_ext.cpp
- lib: delete empty function ext_buffer_pull_free
- lib: delete empty function single_list_free
- lib: delete redundant buffer packing functions
- lib: delete unused array.cpp
- lib: delete unused implementation of strcasestr
- lib: delete unused PT_STRING8 variants of MAPI property definitions
- lib: dissolve goto statements in exmdb_ext.cpp
- lib: do away with contexts_pool function pointer casting
- lib: drop 3rd argument from gx_sql_prep
- lib: drop pthread_cancel from console_server_notify_main_stop
- lib: ensure mime_get_length callers check for <0
- lib: expand char arrays to hold usernames (emailaddrs)
- lib: expand field sizes of EMAIL_ADDR
- lib: expand mapidefs comment about MS-OAUT
- lib: handle BinHex repetition char 0x90 at start of buffer
- lib: have unique_tie::operator~ clear all private members
- lib: make arglist part of the EXMIDL/ZCIDL macro
- lib: make ext_buffer_push run in amortized linear
- lib: make LONG_ARRAY et al trivial again
- lib: mark EXT_PULL::init as requiring an allocator
- lib: more detailed error return values for rtf_convert_group_node
- lib: pick a better initial size for dynamic EXT_PUSH buffers
- lib: put Olson tz code into a namespace
- lib: rectify syntax error for beXX_to_cpu
- lib: reduce indent of html_init_library
- lib: reduce requirements for ext_buffer.hpp inclusion
- lib: rename **MAPI_** to **ZMG_** constants
- lib: replace hard-to-read byteswapping macros
- lib: replace PROP_TAG_ADDRESBOOK* with standardized **PR_** names
- lib: rewrite config_file_save for size
- lib: sort proptag lists
- lib: split mysql parts off database.h
- lib: stay silent on absence of optional service functions

- lib: support for reading type-2 ABKT templates
- lib: switch bounce_producer's g_resource_list to a stdlib container
- lib: switch bounce_producer to C++ stdlib mutexes
- lib: switch service.context_num to uint
- lib: switch to ABK display template control type/flag names from the docs
- lib: switch to std::mutex
- lib: trim 3rd arg to contexts_pool_init
- lib: trim gotos from rtf_convert_group_node
- lib: turn MIME_FIELD length values into unsigneds
- lib: use common-place PR_OOF_*/PR_EC_* tag names
- lib: use full 8-char salt for md5crypt
- lib: use size_t for LIST_FILE members
- lib: use standardized folder deletion flag names
- lib: use standardized fright* flag names
- lib: use standardized **MAPI_** object type names
- lib: use standardized **MSGFLAG_** message flag names
- lib: use standardized PR_ACCESS* tag names
- lib: use standardized PR_ATTACH_DATA_BIN/OBJ tag name
- lib: use standardized PR_ATTACH_* tag names
- lib: use standardized PR_BODY tag name
- lib: use standardized PR_CHANGE_KEY tag name
- lib: use standardized PR_CREATION_TIME tag name
- lib: use standardized PR_DELETED_* tag names
- lib: use standardized PR_DISPLAY_NAME tag name
- lib: use standardized PR_DISPLAY_* tag names
- lib: use standardized PR_EMAIL_ADDRESS tag name
- lib: use standardized PR_ENTRYID tag name
- lib: use standardized PR_INTERNET_CPID, PR_LOCALE_ID
- lib: use standardized PR_IPM_* tag names
- lib: use standardized PR_LAST_MODIFICATION_TIME tag name
- lib: use standardized PR_MESSAGE_FLAGS tag name
- lib: use standardized PR_MESSAGE_SIZE tag name
- lib: use standardized PR_MESSAGE_* tag names
- lib: use standardized PR_OBJECT_TYPE tag name
- lib: use standardized PR_PARENT_* tag names
- lib: use standardized PR_PREDECESSOR_CHANGE_LIST tag name
- lib: use standardized PR_READ tag name
- lib: use standardized PR_RECORD_KEY tag name
- lib: use standardized PR_SMTP_ADDRESS tag name

- lib: use standardized PR_SOURCE_KEY tag name
- lib: use standardized PR_STORE_* tag names
- lib: use standardized PR_*SUBJECT* tag names
- lib: use standardized PR_* tag names
- lib: use stdlib containers for html.cpp
- lib: use std::min for memcpy
- lib: use STREAM_SEEK, BOOKMARK names from documentation
- logthru: add logfile support and reloading
- mapi_lib: add length parameter to common_util_entryid_to_username
- mapi_lib: add length parameter to oxcical_get_smtp_address
- mapi_lib: add length parameter to oxcmail_export_address
- mapi_lib: add length parameter to oxcmail_export_addresses
- mapi_lib: add length parameter to oxcmail_get_smtp_address
- mapi_lib: centralize element growth parameters
- mapilib: combine oxcical pidlid constants
- mapilib: combine oxcmail pidlid constants
- mapi_lib: compact busy status int/string mapping
- mapi_lib: compact calendar scale int/string mapping
- mapilib: compact oxcical if-1L-1L to ?:
- mapi_lib: compact replicated busystatus emission code
- mapi_lib: complete tpropval_array conversion to stdbool
- mapi_lib: delete unnecessary memcpy during EXT_PULL::g_wstr
- mapi_lib: guard against integer underflow in pull_svreid
- mapi_lib: repair RECIPIENT_ROW::pdisplay_type pointing to stack
- mapi_lib: replace address property magic values by standardized mnemonics
- mapi_lib: replace busy status magic values by standardized mnemonics
- mapi_lib: resolve instances of -Wabsolute-value
- mapi_lib: rework code to soothe clang analyzer warning
- mapi_lib: silence clang warning about uninitialized value in RTF parser
- mapi_lib: support for the olWorkingElsewhere busy status
- mapi_lib: support MH encodings
- mapilib: switch oxcical from INT_HASH to unordered_map<int>
- mapilib: switch oxcmail from INT_HASH to unordered_map<int>
- mapi_lib: use standardized calendar scale enum names
- mda, mra: add const/unsigned qualifiers
- mda, mra: compact system_service hook definitions
- mda, mra: expand char arrays to hold usernames (emailaddrs)
- mda, mra: handle multipurpose dispatch return codes
- mda, mra: turn dispatch value into a multi-purpose field

- mda, mra: use stdlib container for g_def_code_table
- mda: switch to std::mutex
- mda: switch to std::shared_mutex
- midb: add additional locking needed for g_server_list
- midb_agent: compact get_connection code
- midb_agent: deindent fetch_detail, fetch_detail_uid
- midb_agent: deindent get_connection
- midb_agent: reduce excess gx_inet_connect messages
- midb_agent: speed up termination during midb connection trying
- midb_agent: use ”auto” specifier with get_connection()
- midb: break up if stmt for static analysis
- midb: change silly FDDT return code on absent folder
- midb: check return value of tpropval_array_set_propval
- midb: compact repeated expression (T*)expr
- midb: default REMOTE_CONN_floating(&&)
- midb: drop implicit conversion of IDB_REF
- midb: emit log message when sqlite DBs cannot be opened
- midb: make calls to mail_engine_put_idb automatic
- midb: mark IDB_REF(IDB_ITEM *) as explicit
- midb: reduce indent in midcl_thrwork
- midb: reduce main() unwinding boilerplate
- midb: replace custom IDB_REF by unique_ptr-with-deleter
- midb: replace magic return values by mnemonics
- midb: replace pthread_cancel by join procedure
- midb: restore str_hash_iter_get_value semantics
- midb: switch g_hash_list to a stdlib container
- midb: switch largely to std::mutex
- midb: use ”auto” keyword around gx_sql_prep
- midb: use ”auto” keyword around mail_engine_get_idb, mail_engine_get_folder_id
- midb: utilitze xstmt::finalize
- midb: wrap sqlite3_close in an exit scope
- midb: zero-initialize AGENT_THREAD, REMOTE_CONN struct members
- misc: replace more strncpy sites by HX_strlcpy
- mlist_expand: expand mailaddr buffers to UADDR_SIZE
- mod_cache: add missing include <atomic>
- mod_cache: implement fallback to built-in defaults
- mod_cache: move cache.txt reading to separate function
- mod_cache: switch to std::mutex
- mod_cache: use stdlib containers for g_directory_list

- mod_fastcgi: avoid using /../ in path
- mod_proxy: move proxy.txt reading into separate function
- mod_proxy: pick better variable names
- mod_proxy: switch g_proxy_list to a stdlib container
- mra: switch to std::mutex
- mra: switch to std::shared_mutex
- mt2exm: set PR_LAST_MODIFICATION_TIME if not present
- mt2exm: start exmdb connection after base maps have been read
- mysql_adaptor: add manpage reference to logmsg about schema_upgrade skip/abort
- mysql_adaptor: add schema_upgrades=hostid
- mysql_adaptor: change default schema_upgrades action to "skip"
- mysql_adaptor: collect magic array size into a mnemonic
- mysql_adaptor: compact config log messages
- mysql_adaptor: complain if there is an overlap between user and alias table
- mysql_adaptor: deindent svc_mysql_adaptor
- mysql_adaptor: deindent verify_password
- mysql_adaptor: delete duplicate get_username <> get_username_from_id
- mysql_adaptor: disable firsttime password feature by default
- mysql_adaptor: establish server connections on demand only
- mysql_adaptor: heed user_properties.order_id from now on
- mysql_adaptor: move z_null to single user .cpp file
- mysql_adaptor: new config loader with std::string and direct parameter init
- mysql_adaptor: pass length parameter to firsttime_password
- mysql_adaptor: reorder functions
- mysql_adaptor: silence successful reconnect messages
- mysql_adaptor: support config reloading
- mysql_adaptor: use SHA512 crypt for firsttime_pw functionality
- oxdisco: add built-in defaults
- oxdisco: handle empty input XML document
- pam_gromox: set global config file object
- pff2mt: dump MNID_ID names with hex ID
- pff2mt: resolve instances of -Wmismatched-new-delete
- pff2mt: restore folder progress message
- pffimport: abandon libpff item type for parent descriptor
- pffimport: add command for splicing PFF objects into existing store hierarchy
- pffimport: add const qualifiers to some functions
- pffimport: add more dry-run mode checks
- pffimport: add -p option for property detail view
- pffimport: add support for transferring PT_CLSID propvals

- pffimport: attachment support
- pffimport: avoid running into PF-1034/PF-1038 assertions
- pffimport: consistently report errors to stderr
- pffimport: consistent return value checks
- pffimport: cure occurrence of PF-1036 exception
- pffimport: ditch extraneous argument to az_item_get_propv
- pffimport: do not abort when treevisualizing u-0 type nodes
- pffimport: drop extra set of braces from -p output
- pffimport: dump NID_MESSAGE_STORE during -t walk
- pffimport: dump NID_NAME_TO_ID_MAP during -t walk
- pffimport: dump raw mvprop data for analysis
- pffimport: emit all messages to stderr
- pffimport: emit terse progress report in absence of -t
- pffimport: facilitate debugging 0-byte multivalue properties
- pffimport: factorize initial destination mailbox discovery
- pffimport: factor out folder map dumping
- pffimport: factor out part of the namedprop resolution
- pffimport: handle Unicode properties with bogus data
- pffimport: hook up attachments to their message objects
- pffimport: implement named property translation
- pffimport: infrastructure for folder mapping
- pffimport: let az_item_get_string_by_propid take a proptag
- pffimport: limit ASCII string dumps like Unicode dumps
- pffimport: lookup named properties ahead of time
- pffimport: move generic functions to another file
- pffimport: move to pipeline-based importer architecture (pff2mt, mt2exm)
- pffimport: new way to track each item level's parent
- pffimport: partial multivalue property support
- pffimport: recognize --help option
- pffimport: reduce az_item_get_record_entry_by_type arguments
- pffimport: refine check for broken mvprop blocks
- pffimport: reorder blocks in do_item2 for function split
- pffimport: replace manual msg dumper by MESSAGE_CONTENT dumper
- pffimport: replace recordent dumper by TAGGED_PROPVAL dumper
- pffimport: report and skip over broken attachments
- pffimport: report NID_MESSAGE_STORE presence as normal condition
- pffimport: resolve instance of -Wmain
- pffimport: resolve static analyzer warnings
- pffimport: separate function for folder map population

- pffimport: skip server-side propname resolution in dry mode
- pffimport: skip transfer message in dry run
- pffimport: spacing adjustments in tree output
- pffimport: split do_item2 per pff item type
- pffimport: split do_print_extra off do_item2
- pffimport: start analysis at the absolute PFF root
- pffimport: stop showing empty summary displayname/subject in tree mode
- pffimport: stop showing too many commas in -t/-p output
- pffimport: switch mostly to exception-based error reporting
- pffimport: treat contacts, notes, tasks like email messages
- php-lib-db: add log functions and replace die
- php_mapi: address a potential future use-after-free
- php_mapi: better error descriptions for exceptions
- php_mapi: compact if-1L-1L blocks to use ?:
- php_mapi: compact repeated error checking
- php_mapi: deduplicate ext_pack_pull_*
- php_mapi: deduplicate ONEOFF_ENTRYID
- php_mapi: deduplicate PULL_CTX/PUSH_CTX
- php_mapi: deduplicate types.h declarations
- php_mapi: unbreak STREAM_OBJECT seeking
- plugins: compact config file reading
- pop3: add notes for POP3_CONTEXT::array
- pop3: compact standardized response line emission
- pop3: delete netconsole routine for pop3_code
- pop3: delete parsing of pop3_code.txt
- pop3: delete POP3_CODE enum and reduce numeric range
- pop3: delete unused units_allocator.cpp
- pop3: make ip6_filter optional
- pop3: use a stdlib container for MSG_UNIT arrays
- Rebranding followup
- rebuild: employ documented option parsing
- rebuild: trim dead stores
- rebuild: use ”auto” keyword around gx_sql_prep
- Revert ”ldap_adaptor: add option to disable auth connection persistence”
- smtp: add config directive ”command_protocol”
- smtp: bump logmsg severity for rejected deliveries
- smtp: collect smtp_parser_init parameters in a struct
- smtp: compact standardized response line emission
- smtp: delete netconsole routine for smtp_code

- smtp: delete parsing of smtp_code.txt
- smtp: delete SMTP_CODE enum and reduce numeric range
- smtp: join overlapping struct definitions and move to stdlib containers
- smtp: reduce indent in smtp_cmd_handler_check_onlycmd
- smtp: rename to delivery-queue
- str_filter: indent reduction in audit_filter.cpp
- str_filter: replace internal condition for audit-disabled case
- str_filter: switch g_audit_hash to a stdlib container type
- str_table(domain_list): add PLUGIN_RELOAD functionality
- system: add ProtectSystem=yes to systemd units
- system: delete target units
- tests: add more zendfake symbols
- timer: add missing pthread_join for accept thread
- timer: add pthread_kill for speedier shutdown
- timer_agent: reduce excess gx_inet_connect messages
- timer: avoid crash on shutdown
- timer: lambda-ify block of code for outfactoring
- timer: move to std::mutex
- timer: replace pthread_cancel by pthread_join
- timer: split code block into separate function
- timer: switch connection list to std::list
- timer: switch timer list to std::list
- timer: use exit scopes and compact repeated teardown code
- tools: add documented -? option
- tools: add gromox-pffimport script with replacement notice
- tools: construct SQL queries with snprintf rather than sprintf
- tools: delete digest utility
- tools: new utility "gromox-kpd2mt"
- tools: PFF importer
- tools: print conn info when database connection has failed
- tools: reduce code nesting level
- tools: rename kpd2mt to kdb2mt
- tools: utilize xstmt::finalize
- tools: wrap sqlite3_close in an exit scope
- tools: wrap sqlite3_shutdown in an exit scope
- zcore: add directive zrpc_debug
- zcore: add directive zrpc_debug
- zcore: add length parameter to ab_tree_get_display_name
- zcore: add missing free() call when object_tree_create fails

- zcore: add variable for enabling trivial RPC status dumps
- zcore: change ATTACHMENT_OBJECT freestanding functions to member funcs
- zcore: change CONTAINER_OBJECT freestanding functions to member funcs
- zcore: change ICSDOWNCTX_OBJECT freestanding functions to member funcs
- zcore: change ICSUPCTX_OBJECT freestanding functions to member funcs
- zcore: change MESSAGE_OBJECT freestanding functions to member funcs
- zcore: change TABLE_OBJECT freestanding functions to member funcs
- zcore: change USER_OBJECT freestanding functions to member funcs
- zcore: collapse zarafa_server.cpp nested ifs into one
- zcore: compact common subexpressions
- zcore: compact if-1-1 blocks to use ?:
- zcore: compact if-1L-1L near return into ?:
- zcore: compact repeated expression (T*)expr
- zcore: compact repeated logic involving rop_make_util_*_guid
- zcore: compact repeated static_cast exprs
- zcore: CSE-combine multiflag checks
- zcore: defer a few unique_ptr::reset calls on specific paths
- zcore: deindent ab_tree_get_node_type, ab_tree_get_server_dn
- zcore: deindent folder_object.cpp, store_object.cpp
- zcore: deindent object_tree_free_root
- zcore: deindent store_object_get_named_{propids,propnames}
- zcore: deindent zarafa_server_deletemessages
- zcore: deindent zarafa_server_logon
- zcore: deindent zarafa_server_notification_proc
- zcore: deindent zarafa_server_openabentry
- zcore: deindent zarafa_server_submitmessage
- zcore: do not switch to Chinese when store language unresolvable
- zcore: drop implicit conversion of AB_BASE_REF
- zcore: drop implicit conversion of USER_INFO_REF
- zcore: factor PROP_TAG_ECUSERLANGUAGE handling out to split function
- zcore: lambdaify sections of hierconttbl_query_rows
- zcore: lambdaify sections of table_object_get_folder_permission_rights
- zcore: log attempts to send mail to no recipients
- zcore: log failed attempts to use delegate FROM
- zcore: make calls to ab_tree_put_base automatic
- zcore: make calls to zarafa_server_put_user_info automatic
- zcore: make g_notify_table a stdlib container
- zcore: make g_session_table a stdlib container
- zcore: make g_user_table a stdlib container

- zcore: make object_tree_* member functions
- zcore: make OBJECT_TREE::phash a stdlib container
- zcore, php_mapi: deduplicate RPC_REQUEST
- zcore, php_mapi: deduplicate RPC_RESPONSE
- zcore: reduce main() unwinding boilerplate
- zcore: reload zrpc_debug variable on SIGHUP
- zcore: repair inaccurate BOOL value passed to container_object_get_container_table_num
- zcore: replace custom AB_BASE_REF by unique_ptr-with-deleter
- zcore: replace custom USER_INFO_REF by unique_ptr-with-deleter
- zcore: replace pthread_cancel by join procedure
- zcore: resolve instances of -Wformat*
- zcore: resolve deadcode warning for FOLDER_OBJECT::updaterules
- zcore: skip call to table_object_set_table_id for unhandled table types
- zcore: source code indent reduction
- zcore: source-inline folder_object_get_id function calls
- zcore: source-inline folder_object_get_store function calls
- zcore: source-inline folder_object_get_type function calls
- zcore: source-inline store_object_check_private function calls
- zcore: source-inline store_object_get_account_id function calls
- zcore: source-inline store_object_get_mailbox_guid function calls
- zcore: split functions off hierconttbl_query_rows
- zcore: split functions off table_object_get_folder_permission_rights
- zcore: stop using strncpy
- zcore: store ownership bit
- zcore: switch ab_tree from INT_HASH to unordered_map
- zcore: trim braces on if blocks with trivial condition /FALSE == .*b_/
- zcore: trim braces on if blocks with trivial condition /TRUE == .*b_/
- zcore: trim braces on single-expr blocks
- zcore: trim redundant unique_ptr::reset calls
- zcore: turn freestanding FOLDER_OBJECT functions into member ones
- zcore: turn freestanding STORE_OBJECT functions into member ones
- zcore: turn store_object_check_owner_mode into a member function
- zcore: turn store_object_get_account into a member function
- zcore: turn store_object_get_dir into a member function
- zcore: turn store_object_guid into a member function
- zcore: unbreak deletion of origin message during copy-delete moves
- zcore: use ”auto” specifier with zarafa_server_get_info
- zcore: use ”auto” specifier with zarafa_server_query_session/USER_INFO
- zcore: use stdlib types for USER_INFO members

- zcore: variable scope reduction in table_object_get_folder_permission_rights
- zcore: wrap CONTAINER_OBJECT in unique_ptr
- zcore: wrap OBJECT_TREE in unique_ptr
- zcore: wrap STORE_OBJECT in unique_ptr
- zcore: wrap USER_OBJECT in unique_ptr

10.11.4.2 Bugfixes

- all: fix instances of -Wmaybe-uninitialized
- all: fix instances of unchecked return values
- all: fix instances of TOCTOU
- all: fix instances of -Wodr
- all: fix instances of -Wformat-truncation
- all: fix instances of -Wsign-compare
- all: fix instances of -Wshadow
- authmgr: fix type mismatch on dlname ldap_auth_login2
- daemons: fix type mismatch on log_info
- daemons: fix unbalanced reference counts on service plugins
- daemons: switch thread numbers to unsigned
- doc: fix wrong file reference in mod_fastcgi.4gx
- email_lib: fix evaluation of undefined variable
- exch: fix instances of -Wunused-*
- exch: fix instances of -Wunused-variable
- exch: fix a number of dead stores
- exch: fix incomplete module teardown on init failure
- exch: fix potential null deref on plugin unload
- exchange_emsmdb: fix an instance of type punning
- exchange_emsmdb: fix comparison against unsigneds
- exchange_emsmdb: fix compiler warning for casting to whacky type
- exchange_emsmdb: fix copy paste error
- exchange_emsmdb: fix crash during getpropertiesall
- exchange_emsmdb: fix crash upon retrieval of some calculated properties
- exchange_emsmdb: fix dereference null return value
- exchange_emsmdb: fix failed substitution logon_object_get_account -> plogon->get_dir
- exchange_emsmdb: fix ftstream_parser_create running into EISDIR error
- exchange_emsmdb: fix incorrect sleep amount
- exchange_emsmdb: fix integer arithmetic and truncation issues in rop_readstream, rop_seekstream
- exchange_emsmdb: fix integer multiplication overflow during quota check
- exchange_emsmdb: fix logical vs. bitwise operator

- exchange_emsmdb: fix read from uninitialized variable
- exchange_emsmdb: fix resource leaks
- exchange_emsmdb: fix ropGetPropertiesList name
- exchange_emsmdb: fix signed arithmetic issues in rop_seekrow
- exchange_emsmdb: fix wrong size argument
- exchange_nsp: fix function signature mismatches
- exchange_nsp: fix nullptr deref in nsp_interface_resolve_names
- exchange_rfr: fix out-of-bounds access
- exmdb_client: fix unspecified state after std::move
- exmdb_provider: fix instance of -Wmissing-declarations
- exmdb_provider: fix instances of FORWARD_NULL
- exmdb_provider: fix a set of unterminated strings
- exmdb_provider: fix an incomplete permission check
- exmdb_provider: fix an out-of-bounds write in common_util_get_proptags
- exmdb_provider: fix an unterminated string buffer in common_util_username_to_essdn
- exmdb_provider: fix broken recursive deletion of folders
- exmdb_provider: fix crash on shutdown near pthread_kill
- exmdb_provider: fix double call to db_engine_put_db
- exmdb_provider: fix hang when aborting midway through db_engine_run
- exmdb_provider: fix illegal mutex double unlock
- exmdb_provider: fix missing calls to db_engine_put_db
- exmdb_provider: fix null dereference in exmdb_parser_stop
- exmdb_provider: fix out-of-bounds write
- exmdb_provider: fix resource leak in exmdb_server_set_message_instance_conflict
- exmdb_provider: fix too early db_engine_put_db calls
- exmdb_provider: fix unchecked return value
- exmdb_provider: fix unchecked return value in exmdb_server_load_message_instance
- exmdb_provider: fix unchecked return values in exmdb_server_flush_instance
- exmdb_provider: fix unused value in exmdb_server_query_table
- exmdb_provider: fix unused value in exmdb_server_store_table_state
- exmdb_provider: fix unused values in table_load_content_table
- exmdb_provider: fix use of wrong quota property
- exmdb_provider: fix wrong serialization of REQ_SET_MESSAGE_READ_STATE
- http: fix a number of dead stores
- http: fix crash when user_default_lang is unset
- http: fix dereference null return value
- http: fix destination buffer too small
- http: fix explicit null dereference
- http: fix ignored return values from ndr_pull_data_*

- http: fix out-of-bounds read
- http: fix out-of-bounds write
- imap: dissolve uses of snprintf to fixed buffer in imap_parser_process
- imap: fix absence of starttls capability keyword
- imap: fix double free during shutdown
- imap: fix garbage listing of folders
- imap: fix off-by-one in literal processing
- imap: fix wrong strptime format for internaldate parsing
- ldap_adaptor: fix incorrect comparison
- ldap_adaptor: fix null deref when LDAP server is away
- lib/mapi: fix possible unsigned underflow
- lib: fix a number of dead stores
- lib: fix comparison against unsigneds (related to mime_get_length)
- lib: fix crash when zcore uses a zero-length name during zcore_callid::COPYFOLDER
- lib: fix inconsistent capacity allocations in ext_buffer
- lib: fix intended return value of gx_snprintf1
- lib: fix multiplication overflow in Olson tz code
- lib: fix out-of-bounds write in parse_mail_addr, parse_mime_addr
- lib: fix parenthesis bugged expression in threads_pool
- lib: fix use-after-destruction near ext_buffer_push_release
- lib: spello fix for pidTag* in comments
- mapi_lib/rtf: fix passing an undefined value between functions
- mapi_lib: fix PidLidIntendedStatus always being olTentative
- mapi_lib: fix an allocation too short
- mapi_lib: fix an out-of-bounds write in oxvcard_import
- mapi_lib: fix memory leak in rtf_load_element_tree
- mapi_lib: fix memory leak in rule_actions_dup
- mapi_lib: fix returns with garbage values
- mda: fix a number of dead stores
- mda: fix spello ”envelop”
- midb: fix concurrent use of sqlite data structure
- midb: fix leftover debugging breakpoint infinite loop
- midb: fix out-of-bounds read
- midb: fix unchecked return value
- midb: fix wrong serialization of REQ_LOAD_PERMISSION_TABLE
- misc: fix instances of NULL_RETURNS
- misc: fix two overlapping copy operations
- misc: fix unbounded strcpy calls
- misc: fix uninitialized pointers/scalars

- mod_cache: fix spello ”defualt”
- mod_proxy: fix out-of-bounds access parsing proxy.txt
- mra: fix occasional compile error
- mt2exm: add small prefix to log messages
- mt2exm: fix inverted meaning of exm_create_folder::o_excl parameter
- mysql_adaptor: fix unchecked return value
- oxcical: fix possible null deref in oxcical_parse_tzdefinition
- oxdisco: fix incorrect XML tag name ”DelpoymentId”
- pff2mt: support oddly-encoded subject prefix length marker
- pffimport: fix cov-scan reports
- pffimport: fix i586 build error
- php_mapi: fix a number of dead stores
- php_mapi: fix signed arithmetic issues in stream_object_seek
- tools: fix crash when /etc/gromox is unreadable
- zcore: fix a number of dead stores
- zcore: fix logical vs. bitwise operator
- zcore: fix mismatch of RESP_CONFIGSYNC, RESP_SYNCMESSAGECHANGE structs
- zcore: fix null deref in delegate rule scenario
- zcore: fix resource leak
- zcore: fix signed arithmetic issues in zarafa_server_seekrow
- zcore: fix unsigned compared against 0
- zcore: fix use after free in zarafa_server_openabentry
- zcore: fix wrong deserialization of DB_NOTIFY_DATAGRAM/FOLDER_MODIFIED
- zcore: fix zarafa_server_openembedded adding wrong message to objtree

10.11.4.3 Removed

- adaptor: remove unused functions
- all: remove config_file_set_value calls with no effect
- all: remove outdated, inaccurate and trivial function descriptions
- all: remove some unused includes
- all: remove unused pthread.h includes
- all: remove unused variables
- exch: remove log_plugin service plugin
- exch: remove mod_proxy
- exchange_emsmdb: remove logically dead code
- exmdb_provider: delete remove() call with garbage parameter
- http: remove unused functions
- ldap_adaptor: remove unnecessary base discovery
- lib: abolish itoa function

- lib: remove **ext_pull_** freestanding function variants
- lib: remove **ext_push_** freestanding function variants
- lib: remove ineffective unsigned comparison
- lib: remove pointer indirection for PROPERTY_NAME::plid
- lib: remove unused definitions from plugin.hpp
- mda, mra: remove unnecessary decorative comment lines
- midb: remove mail_engine_sync_mailbox's goto spaghetti
- midb: remove unused functions
- midb: remove unused midb protocol commands
- mod_fastcgi: remove unnecessary braces for 1-line blocks
- mysql_adaptor: remove config_file_set_value calls
- mysql_adaptor: remove unused function z_strlen
- php_mapi: remove unused zcore RPC structs
- smtp: remove unused smtp_param::threads_num member
- system: remove obsolete PartOf= directives of systemd units
- zcore: remove constant 2nd argument to table_query_rows
- zcore: remove dead code from storetbl_query_rows
- zcore: remove spurious break in table_object_query_rows
- zcore: remove unused functions

10.11.5 grommunio Sync

Repository: https://github.com/grommunio/grommunio-sync

Code statistics:

- +23138 lines added
- -25155 lines removed

Commits:

- 2021-08: 5
- 2021-07: 23
- 2021-06: 6
- 2021-05: 1
- 2021-04: 0
- 2021-03: 0

10.11.5.1 New (Improvements)

- add missing ADMIN_API_POLICY_ENDPOINT to config.php
- added ProvisioningManager
- added TTL to InterprocessData setData
- added TTL to setKey
- check if contentdata is set before accessing it
- deviceManager is available only when authenticated, adjusting code to match
- enable provisioning by default
- let sync have its own user
- log: assign su permissions for logrotate
- log: update paths
- refactoring provisioning process
- retrieving policies from admin api
- save state data as json
- save states in redis and the user store
- set missing properties for signed emails
- use microtime for start

10.11.5.2 Bugfixes

- don't serialize json ASDevice in redis
- fix Utils::PrintAsString() to recognize null correctly
- fix fallback to default policies if API endpoint is not available
- rename redis key to statefoldercache

10.11.5.3 Removed

- remove default policies and policyname
- remove grommunio-sync-admin.php

10.11.6 grommunio Setup

Repository: <internal-only>

Code statistics:

- +2180 lines added
- -1278 lines removed

Commits:

- 2021-08: 18
- 2021-07: 19
- 2021-06: 2
- 2021-05: 8

- 2021-04: 7
- 2021-03: 91

10.11.6.1 New (Improvements)

- log: redirect ssl self-generation to log file
- move fullca function to separate script
- move logfile to /var/log/ for persitence
- mysql_adaptor: set schema_upgrade in the right file
- new SQL setup
- new TLS setup dialog
- new hostname dialog
- new repo dialog
- new repo setup
- new setup finish screen
- new welcome screen
- plugin: add onlyoffice as default enabled plugin
- query admin for relayhost and set it in postfix
- rebranding: update URLs / mail
- reject some path injections for FQDN & hostname
- replace cron entry by a persistent systemd timer
- repos: enable autorefresh
- req: add redis new grommunio default service for operation
- res: rename certbot service and timer
- restore sh compatibility
- reword the Lets_Encrypt prompt
- services: don't enable prosody if not checked as to be installed in the first place
- set +x bit on certbot-renew-hook
- setup: be more specific than "Admin UI"
- ssl: adjust to new nginx config structure
- ssl: switch to certbot standalone mode
- strip filler wording from dialog texts
- style/log: re-add indications at which stage the configuration stage runs
- style: avoid mixing double and single quotes in a config file
- style: better dialog in case of failure
- style: better readability through spacing
- style: change idents to one standard
- style: make code-style consistent
- style: put init vars on top, static anyways
- style: re-add unused progress indicators

- style: readability/style
- support ”localhost” as a default domain for dirty setups
- support PHP8
- support: add support package
- support: silence killing of bgid
- tls: add the link to the current terms of service from Let’s Encrypt
- tls: inform admin about failed certbot command
- tls: move recommended domains to optional
- trim filler wording
- typo: replace _ with space
- typo: stls->starttls
- upgrade to 15.3
- use IPv6 transport and privileged port for LDA
- use mysql to provide virtual_mailbox_domains
- use systemctl, not service
- verify installed amount of memory and warn user
- visibility: don’t show all the logs to terminal, pipe to logfile instead.
- workflow: nginx failing start
- write php config to new location

10.11.6.2 Bugfixes

- nginx: correct replacement of vars
- postfix: FQDN fix
- shellcheck: fix SC2004
- shellcheck: fix SC2006
- shellcheck: fix SC2016
- shellcheck: fix SC2027 && SC2086
- shellcheck: fix SC2046
- shellcheck: fix SC2086
- shellcheck: fix SC2102
- shellcheck: fix SC2129
- shellcheck: fix SC2148
- shellcheck: fix SC2166
- shellcheck: fix SC2223
- shellcheck: fix SC2254
- ssl: fix providing owncert unresolvable loop
- style: readability / style fixes

10.11.6.3 Removed

- Remove Zb, Zu escape codes
- Remove inconsistent step counter

10.11.7 grommunio Web

Repository: https://github.com/grommunio/grommunio-web

Code statistics:

- +15712 lines added
- -4891 lines removed

Commits:

- 2021-08: 18
- 2021-07: 8
- 2021-06: 2
- 2021-05: 3
- 2021-04: 3
- 2021-03: 47

10.11.7.1 New (Improvements)

- Add CSS to style popout window
- Add default domain configuration
- Added Development section to Readme
- Disable password plugin server side
- Explicitly show English as British English
- Highlight 'open shared folders' button
- Implement another way to make textareas white without changing notes colors
- Improve darktheme
- Let contact detail dialog show business address by default
- Let web have its own user
- MAPI: add error code to action rejection message dialogs
- MAPI: emit textual error strings
- Plugin: MDM plugin
- Plugin: Meet plugin
- Plugin: Passwd - plugin improvements (handling)
- Plugin: Passwd - reorder conditions for enhancement
- Plugin: Passwd - restore ability to use zcore setpasswd
- Rebrand grammm to grommunio
- Remove redundant error log
- Remove unused themes

- Rename current themes and rename intern light and dark theme
- Reword "Unknown MAPI Error: 0x000003eb"
- Send request to admin API to change the password
- Set page title to something useful
- Sort the language list in the settings dialog
- Style: gradient header in light theme
- Translate ecUnknownUser to a sensible error message
- Try fixing broken popout CSS
- Update border color
- Use DOMPurify as XSS sanitizer
- Use anchored gitignores

10.11.7.2 Bugfixes

- Fix: ERROR - variable customItems is undeclared
- Fix: broken 'Additional information' textarea
- Fix: color in dropdown box
- Fix: copy & paste from certain browsers end in copy of steuerzeichen.
- Fix: css on firefox
- Fix: dark theme bugs and added css variables to make the code more maintainable
- Fix: invisible settings icon
- Fix: make manifest.xml build and source aware
- Fix: presentation of the topbar with linear-gradient & changed svg color.
- Fix: weird background-color of addressbook

10.11.7.3 Removed

- Core: remove obsolete CmdAgent
- Disable nwjs usage

CHAPTER 11

Roadmap

Current roadmap of grommunio (as of 29th of January, 2025):

- The **Current Stable** release of grommunio has been released on the 29th of January 2025, with a support lifecycle of 3 years and optional extended lifecycle extensions available with grommunio's subscription program.
- The **Next point release** is planned for release in the last week of February 2025 with the following features:
  - RFC 2184/2231: Enhanced handling of extended parameters in MIME headers.
  - Trashed Mailboxes & Migration: Improvements for advanced mailbox handling across multiple migrations, including x400 addressing and undocumented MAPI attributes.
  - grommunio Setup v2: Expanding support for the setup stage for RHEL9, Debian 12, and Ubuntu 24.04.
  - grommunio-files: Updated version with group folder management and modern authentication.
- Following 2025.01.2 the following features are in the upcoming release funnel (in QA and/or deployed already in pilot-installations):
  - Modern Authentication (OAuth2) for Outlook, IMAP, and POP3.
  - Full HTML-based MR (Meeting Request) Processing in the Web UI.
  - AI-Powered Features for enhanced user productivity.
  - Extended rules & autoprocessing support.

11.1 Release strategy

grommunio is committed to delivering quality software products in a customer-friendly and predictable way.

The release model of grommunio is divided into 2 different chains:

- Major releases (e.g. 2025.01.1, 2023.11.1, 2022.12.1)
- Minor releases (e.g. 2025.01.2, 2023.11.3, 2022.12.2)

Major releases contain are determined to larger feature sets as well as including potential architectural changes whereas minor releases are focused on bugfixes, security updates and smaller features.

grommunio provides a major release annually with patch-level releases in monthly cycles.

11.2 Supported Distributions

As of 2025.01.1, grommunio actively supports installation and operation on the following Linux distributions:

- RHEL9 / EPEL9
- openSUSE 15.5+ / SLES 15.5+
- Debian 12
- Ubuntu 24.04

With 2025.01.2, grommunio additionally will start to provide automatic deployment tools (grommunio-setup) for these distributions, too.

11.3 Disclaimer

While grommunio does everything to provide the highest level of transparency, the roadmap document is subject to change based on factors such as customer demand, technological adaption or community response. While this document is crafted with the highest level of care and accuracy, there might be changes, which grommunio communicates through its available communication channels, such as social networks, newsletters and on grommunio's other news sections.

CHAPTER 12

Legal Notice

- Amazon Web Services, AWS, AWS CloudFront, CloudFront, Elastic Compute Cloud, Amazon EC2, EC2, Elastic Load Balancer, ELB, and related graphics, logos, page headers, button icons, scripts, and service names are trademarks, registered trademarks, or trade dress of Amazon Web Services in the U.S. and/or other countries.
- Apache and the Apache feather logo are registered trademarks of The Apache Software Foundation in the United States and other countries.
- Azure, Azure Container Service, ACS, Azure Resource Manager, Windows, Windows Server, Active Directory, Active Directory Federation Services, Outlook and Microsoft are registered trademarks or trademarks of the Microsoft Corporation.
- Debian is a registered trademark of SOFTWARE IN THE PUBLIC INTEREST, INC.
- Docker is a registered trademark of Docker, Inc.
- FreeBSD and the FreeBSD Logo are registered trademarks of The FreeBSD Foundation.
- Google is a registered trademark, and Google Cloud Platform, Google Cloud Platform service, GCP infrastructure platform, Google Cloud enterprise services, and TensorFlow open-source software library are trademarks of Google LLC.
- Grafana, Tempo, Pyroscope are trademarks of Raintank, Inc. dba Grafana Labs.
- grommunio contains software provided by NGINX and its contributors. nginx is a trademark of Nginx Software, Inc.
- HAProxy is a registered trademark of HAProxy Technologies LLC in the United States and France.
- IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.
- JavaScript, MySQL, and Oracle are trademarks or registered trademarks of the Oracle Corporation and/or its affiliates.
- Jitsi is a trademark of 8x8, Inc.
- Kubernetes is a graduated project of the Cloud Native Computing Foundation, or CNCF. Cloud Native Computing Foundation, CNCF, Kubernetes, K8s, Prometheus, and The Linux Foundation are registered trademarks of the The Linux Foundation. CNI and Spinnaker are trademarks of The Linux Foundation. Additional trademarks and registered trademarks can be found at https://www.linuxfoundation.org/trademark-list/.

- Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.
- Mac, MacBook, and macOS are registered trademarks of Apple, Inc in the United States and other countries.
- Matrix.org, Synapse are trademarks of Matrix.org Foundation C.I.C.
- Mattermost is a trademark of Mattermost, Inc.
- Nagios, the Nagios logo, and Nagios graphics are the service marks, trademarks, or registered trademarks of Nagios Enterprises, LLC.
- NetBSD is a registered trademark of The NetBSD Foundation, Inc.
- Nextcloud is a trademark of Nextcloud GmbH.
- OpenID is a trademark (registered in numerous countries) of the OpenID Foundation.
- ownCloud is a trademark of ownCloud GmbH.
- Percona, XtraDB, Percona XtraDB, XtraBackup, Percona XtraBackup, Percona Server for MySQL, and Percona Live, plus the distinctive visual icons and logos associated with these marks are trademarks or registered trademarks of Percona LLC.
- React, React Native, Facebook, Instragram and Meta are registered trademarks of Meta Platforms, Inc. (”Meta”).
- Red Hat, CentOS, RHEL, Red Hat Enterprise Linux are trademarks, or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.
- Seafile is a trademark of Seafile Ltd.
- Splunk is a registered trademark of Splunk, Inc. in the United States and other countries.
- SUSE, openSUSE, YaST, Apparmor and their respective Logos are trademarks of SUSE LINUX AG.
- The MariaDB® mark is a trademark of MariaDB Corporation Ab. The mariadb.org, MariaDB Foundation and MariaDB Server marks are exclusively licensed to the MariaDB Foundation.
- Ubuntu and Canonical are registered trademarks of Canonical Ltd.
- UNIX is registered trademark of The Open Group in the United States and other countries.
- VMware vSphere, VMware vSAN, VMware vShield, VMware vCloud, VMware vRealize, VMware ESXi, VMware are registered trademarks of Broadcom, Inc. in the United States and other countries.
- grommunio, the grommunio Logo and gromox are registered trademarks of grommunio GmbH.

All other trademarks, trade names, service marks, and companies referenced herein belong to their respective companies, foundations, or development communities.

grommunio

grommunio GmbH, Donau-City-Str. 7, Door 2 / 30th floor, DC Tower, 1220 Vienna, Austria
Email: info@grommunio.com | Web: https://grommunio.com