grommunio Administration

Official documentation

Version 2026-07-02 · docs.grommunio.com

© grommunio GmbH — CC BY-SA 4.0

grommunio Administration

Everything you need to **deploy, operate and scale grommunio** — from a first appliance boot to multi-node, highly available clusters. grommunio is a complete communication and collaboration suite (e-mail, calendaring, contacts, tasks, notes, video meetings, chat and file management).

Where to start

New to grommunio? Read the **Introduction** for the concepts, then follow the **Quickstart** to stand up an appliance and reach first login — or pick the installation method that fits your environment from the cards below.

Introduction What grommunio is, its components, and how the pieces fit together. →
Quickstart Stand up a grommunio appliance and reach first login in minutes. →
Guided installation The full appliance (ISO/OVA) walkthrough with the CUI setup wizard. →
Manual installation Install the base and core packages by hand for custom integrations. →
Container installation Run grommunio with Docker Compose (gromox-container). →
Administration Day-to-day management in the Admin UI — domains, users, roles, configuration. →
Architecture Components, protocols, the multi-server design and load-balancer examples. →
High availability A reference Pacemaker/DRBD cluster with a floating IP. →
Command-line (CLI) Manage grommunio from the shell — grommunio-admin and the gromox-* tools. →
Migration Move mailboxes from Exchange, Kopano and other systems. →
Knowledge Base Focused how-tos and troubleshooting for specific topics. →

Audience

This manual is for **system administrators** installing and operating grommunio. It assumes:

- An understanding of Linux, e-mail (e.g. SMTP), networking and DNS.
- General familiarity with the Linux operating system.
- Experience managing communication and/or collaboration solutions.

Offline reading

This section is also available to download for offline use:

- Download as PDF
- Download as EPUB

© grommunio GmbH — CC BY-SA 4.0

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.

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

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.

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

© grommunio GmbH — CC BY-SA 4.0

Component	Function	Component Group
gromox-http	HTTP/RPCH protocol handler (RPC-over-HTTP, MAPI-over-HTTP, EWS, optional FastCGI passthrough); also hosts the Information Store / mailbox engine (exmdb_provider)	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.

© grommunio GmbH — CC BY-SA 4.0

- 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 GmbH — CC BY-SA 4.0

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.

> **Tip**
>
> For the full component architecture — the protocol/component flow, the multi-server design and load-balancer (HAProxy/NGINX) examples — see Architecture.

© grommunio GmbH — CC BY-SA 4.0

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.

Minimum requirements

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

- Server or virtual machine (VMware, Xen, KVM or Hyper-V) with at least:
  - 4 CPU cores
  - 6 GB RAM
  - 32 GB system disk for the operating system and base install. The installer overwrites the **entire** target disk (see the caution under *Installation*). Provision **additional** storage for mailbox data — it is the largest sizing factor and grows with the user count and per-mailbox quota.
- 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 reused 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.

© grommunio GmbH — CC BY-SA 4.0

> **⚠Caution**
>
> 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:

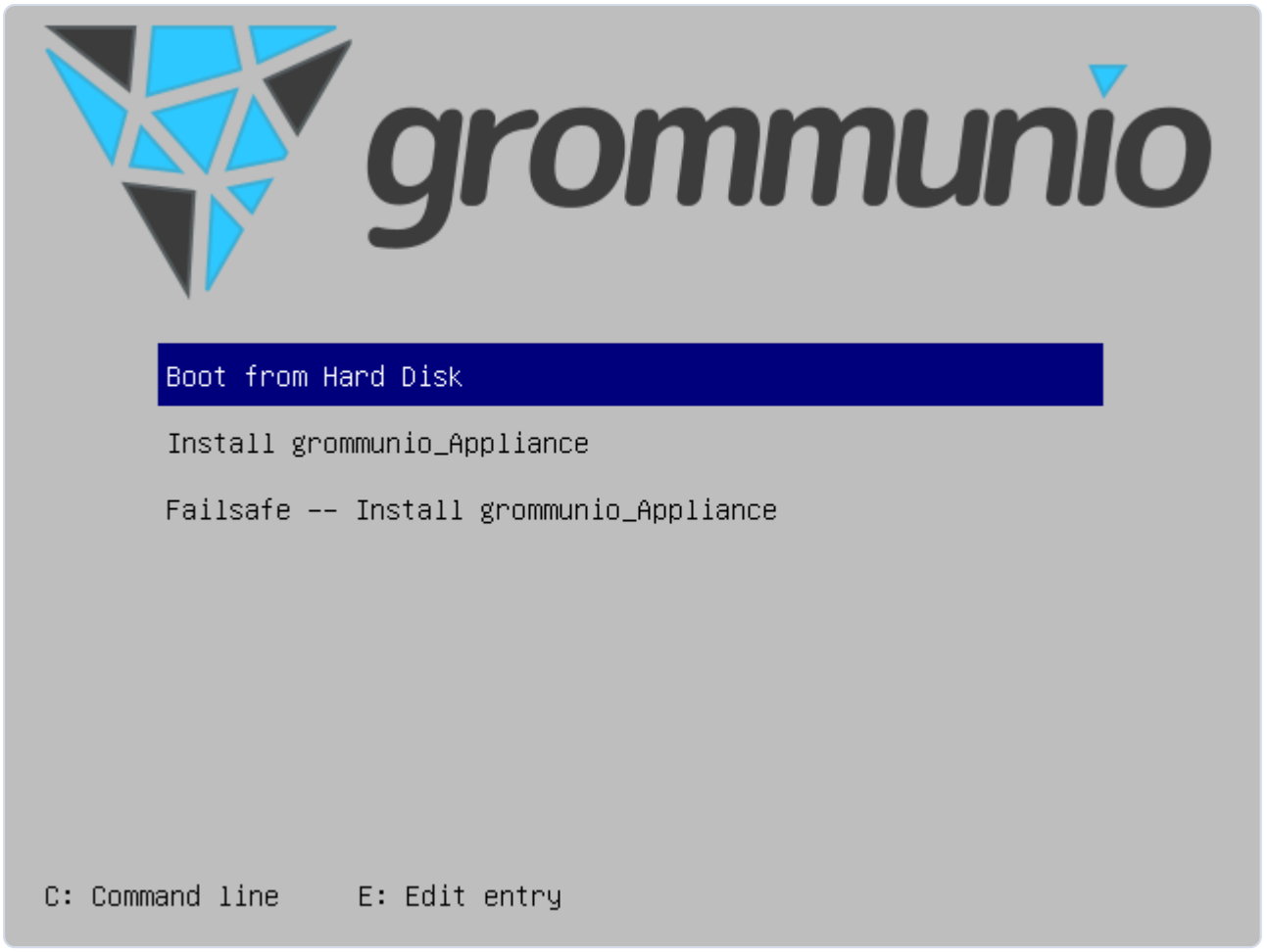
- MX DNS records, for incoming mail delivery.
- At the time of certificate generation by Let's Encrypt, the accessibility of port 80 to all of the defined DNS records is a requirement.

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.

> **⚠Caution**
>
> Note that the installer asks for confirmation to delete and overwrite the installation target!

© grommunio GmbH — CC BY-SA 4.0



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

Setup

After installation, the appliance displays the grommunio console user interface (CUI). For more detailed instructions of the setup process, refer to grommunio Appliance configuration with CUI/setup.

> ⚠**Caution**
>
> 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.

© grommunio GmbH — CC BY-SA 4.0

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>* navigate 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*.

> **Caution**
>
> 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.

> **Caution**
>
> 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.)

© grommunio GmbH — CC BY-SA 4.0

> **⚠Caution**
>
> It is recommended after successful installation 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.

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).

> **⚠Caution**
>
> 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"`

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 provided on a best-effort basis and are not supported. 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

© grommunio GmbH — CC BY-SA 4.0

Certificates

grommunio-setup offers four ways to provision the TLS certificate used by all services:

- **Self-signed certificate** — the simplest option (the default); clients must trust it on first connect. Best for demos and validation, not production.
- **Own CA + certificate** — generate a local certificate authority and sign certificates from it; useful for larger multi-instance validation setups.
- **Import an existing certificate** — bring your own PEM certificate/key pair (a SAN or wildcard certificate is recommended). The most flexible option for publicly trusted CAs.
- **Let's Encrypt** — free, automatic issuance and renewal; requires port 80 reachable from the Internet for every domain during validation (and renewal). Recommended for most simple installations.

Certificates are placed in `/etc/grommunio/ssl` and referenced automatically by the appliance services. See TLS configuration for the detailed walkthrough of each option.

Firewall

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

- 25 (smtp)
- 80 (http)
- 110 (pop3)
- 143 (imap)
- 443 (https)
- 465 (smtps — implicit-TLS mail submission)
- 587 (submission — STARTTLS mail submission)
- 993 (imaps)
- 995 (pop3s)
- 8080 (admin, unencrypted — used during initial provisioning)
- 8443 (admin https)

> **Note**
>
> The Admin API is served unencrypted on port 8080 during initial provisioning. Once setup has finished, switch it to TLS so the Admin UI is reachable over HTTPS on port 8443 — see Admin API TLS configuration.

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).

© grommunio GmbH — CC BY-SA 4.0

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.

Verify the installation

Once grommunio-setup finishes, you have a configured — but still empty — system. Confirm it is working:

- Open the **Admin UI** at `https://<FQDN>:8443/` and sign in as **`admin`** with the password you set (or the one generated by grommunio-setup, recorded in `/var/log/grommunio-setup.log`).
- Open **grommunio Web** at `https://<FQDN>/` — the user webmail and groupware interface.

If both load over HTTPS and the Admin UI signs in, the appliance is ready.

Next steps

A fresh appliance has no mail domains or users yet. Continue with:

- Administration — create your first mail **domain** and **user**, then manage roles, public folders and settings.
- Operations — day-2 tasks, updates, and switching the Admin API to TLS.
- Migration — import mailboxes from Exchange, Kopano and other systems.

> **ⓘ Before going to production**
>
> Define a **backup** strategy covering the mail stores, databases and configuration — see Operations → Backup & Disaster Recovery — and review hardening beyond the firewall (TLS for the Admin API, fail2ban, 2FA/SSO).

© grommunio GmbH — CC BY-SA 4.0

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.

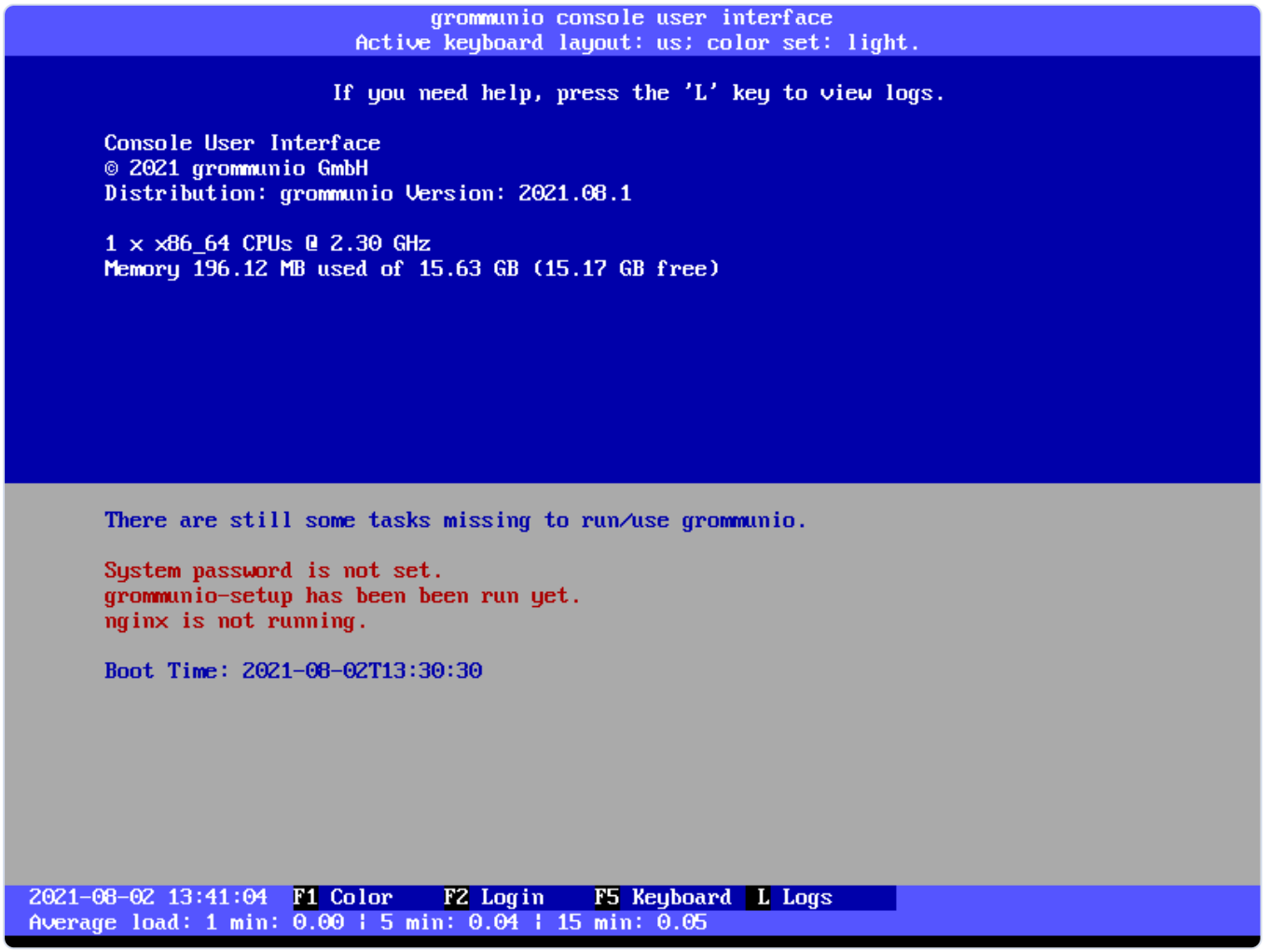
To deploy grommunio via ISO, you need to make the installation media available to your installation target. The ISO is a generic, bootable installation medium which works in most scenarios. To deploy the ISO with bare metal, the ISO can be imaged to USB drives for simplified installation.

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.

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.

© grommunio GmbH — CC BY-SA 4.0



Main screen

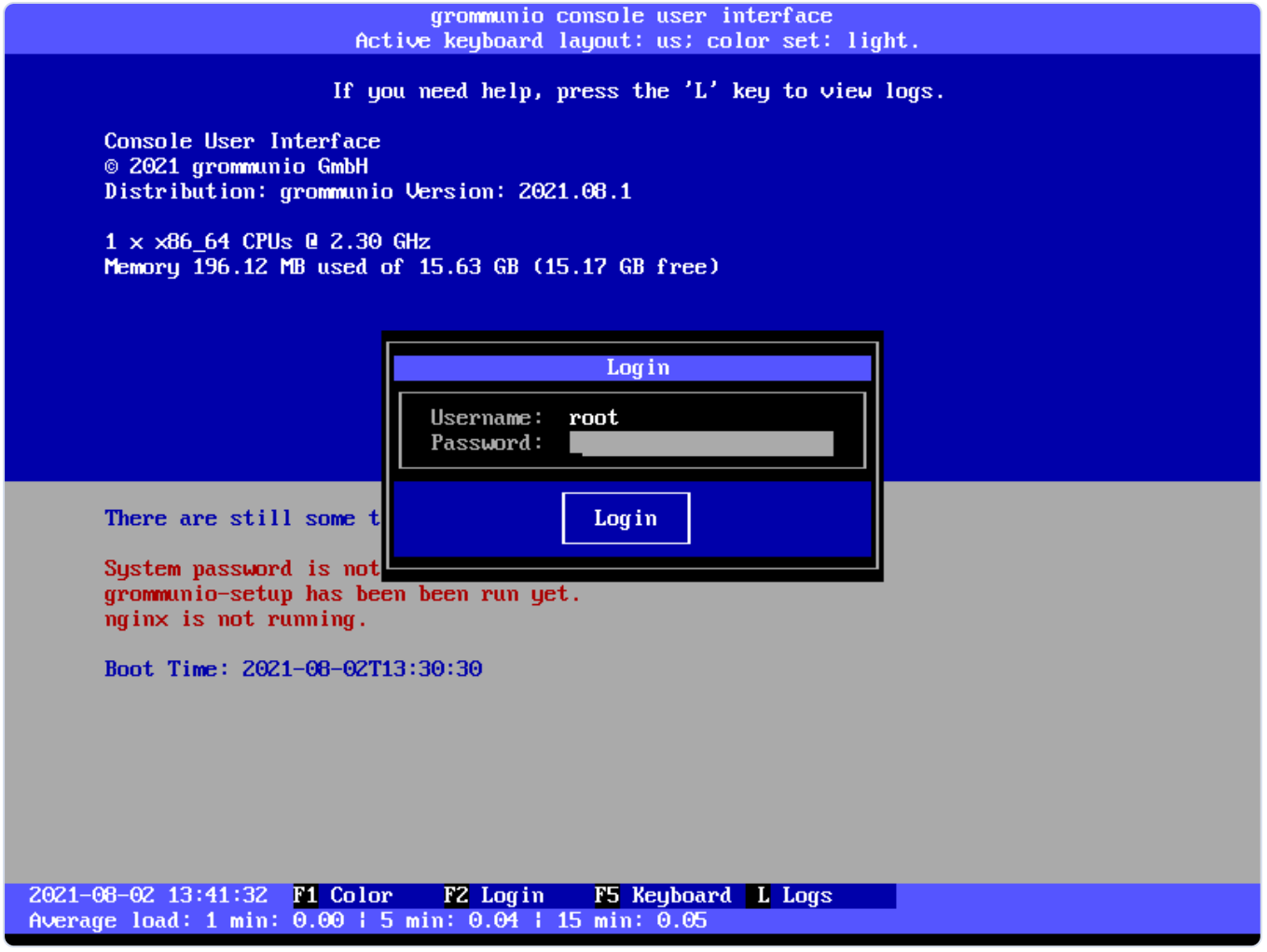
After starting `grommunio-cui`, you are in the main screen. Upon login, you are able to make system configuration changes.

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

© grommunio GmbH — CC BY-SA 4.0

Login



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

> **⚠Caution**
>
> The initial root password is unset (empty). When asked for password at first login, just enter an empty password.

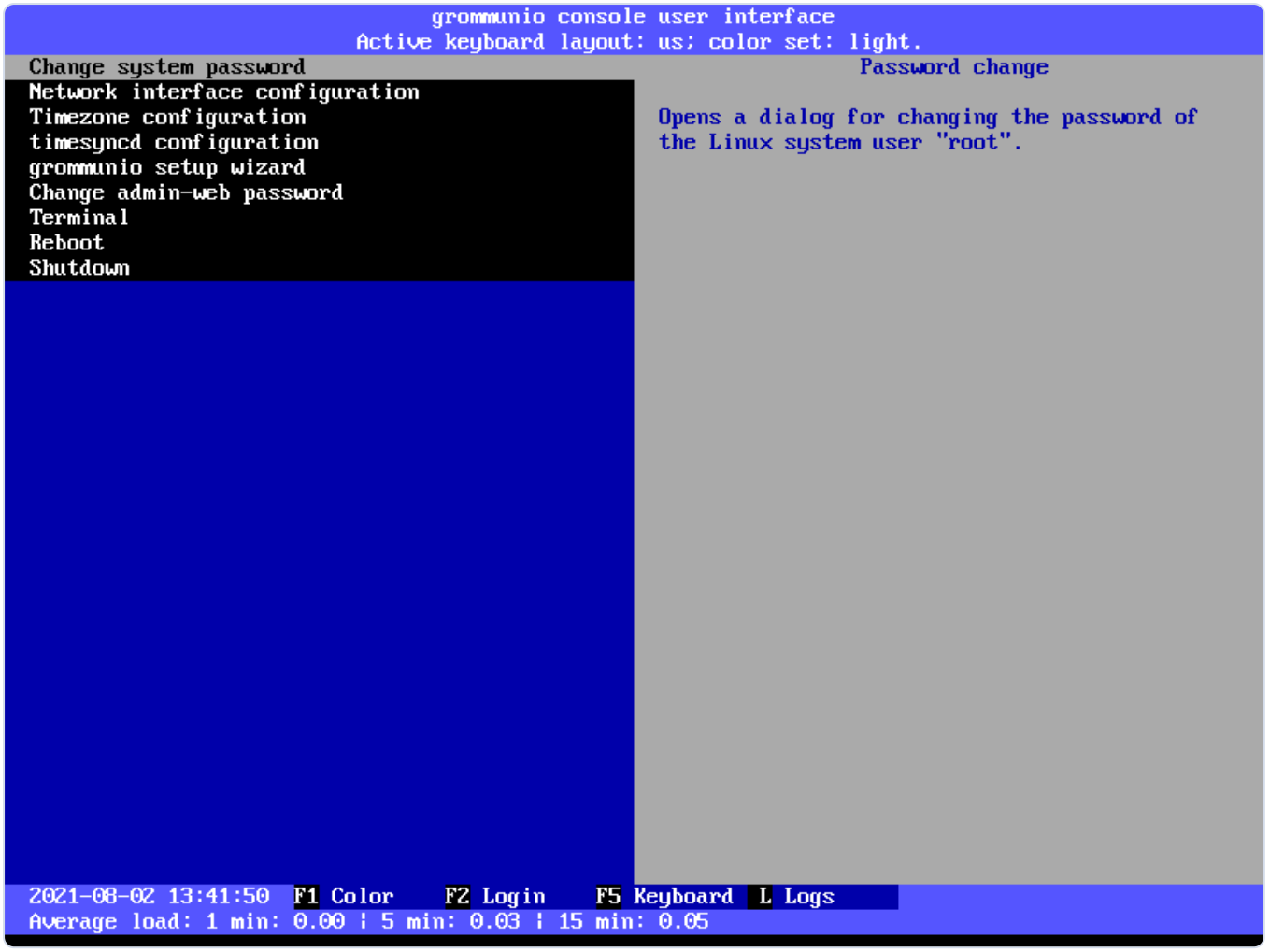
Main configuration screen

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

- Change system password
- Network configuration
- Timezone configuration
- Timesync configuration
- grommunio setup wizard

© grommunio GmbH — CC BY-SA 4.0

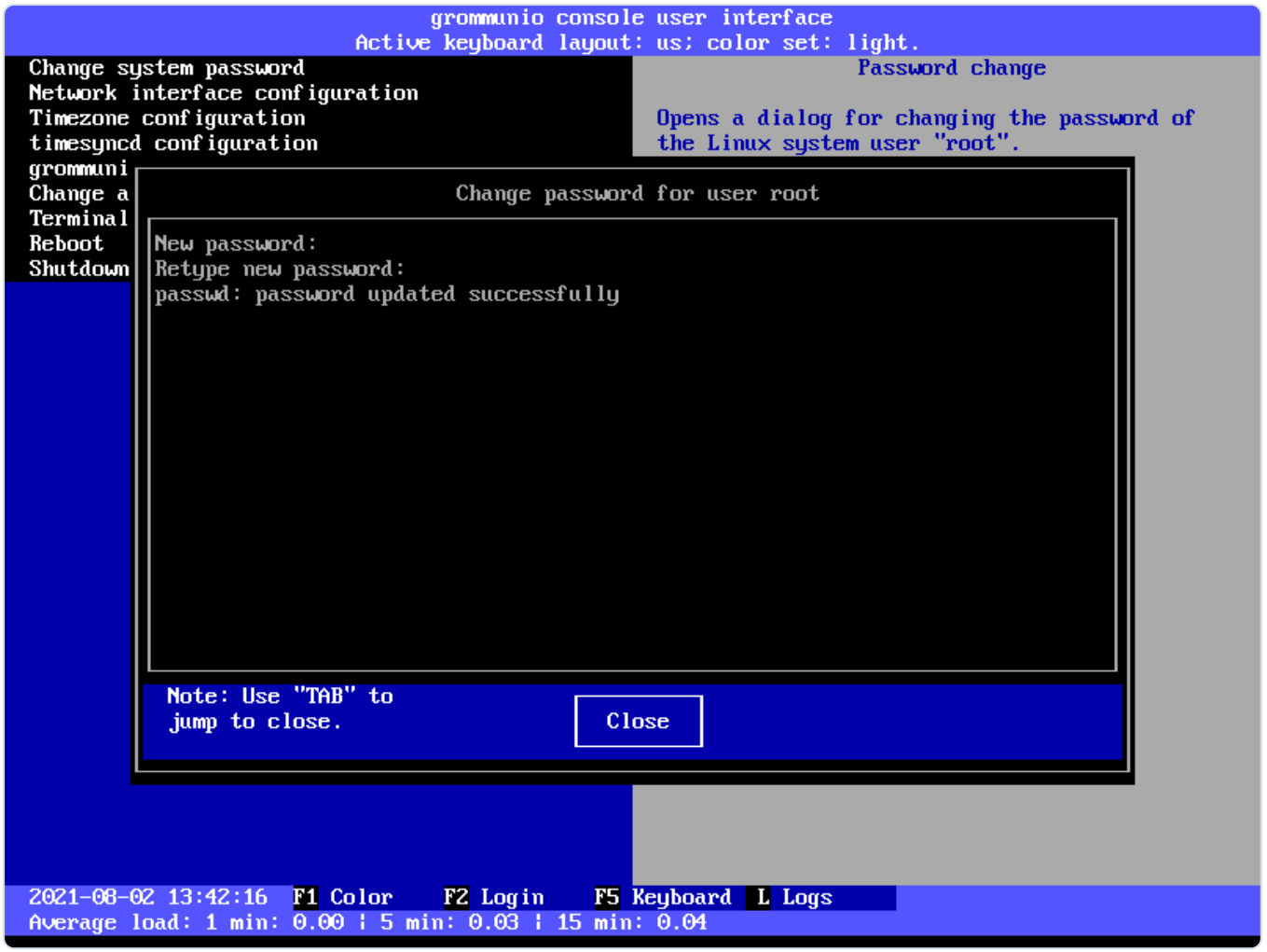
- Change Admin Web UI password
- Terminal
- Reboot
- Shutdown



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.

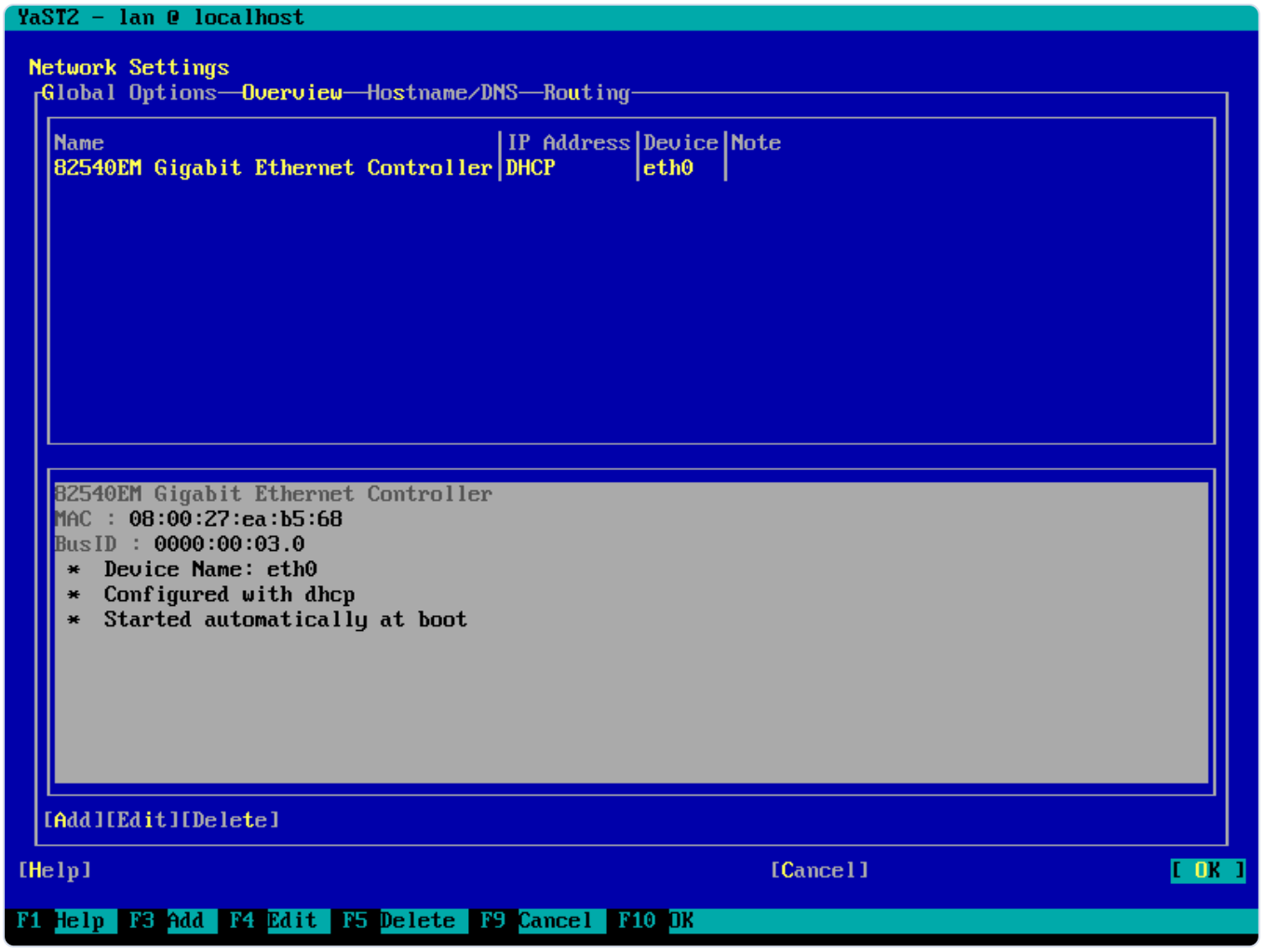
© grommunio GmbH — CC BY-SA 4.0



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-SP6/html/SLES-all/cha-network.html#sec-network-yast

© grommunio GmbH — CC BY-SA 4.0



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

> **⚠Caution**
>
> 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.

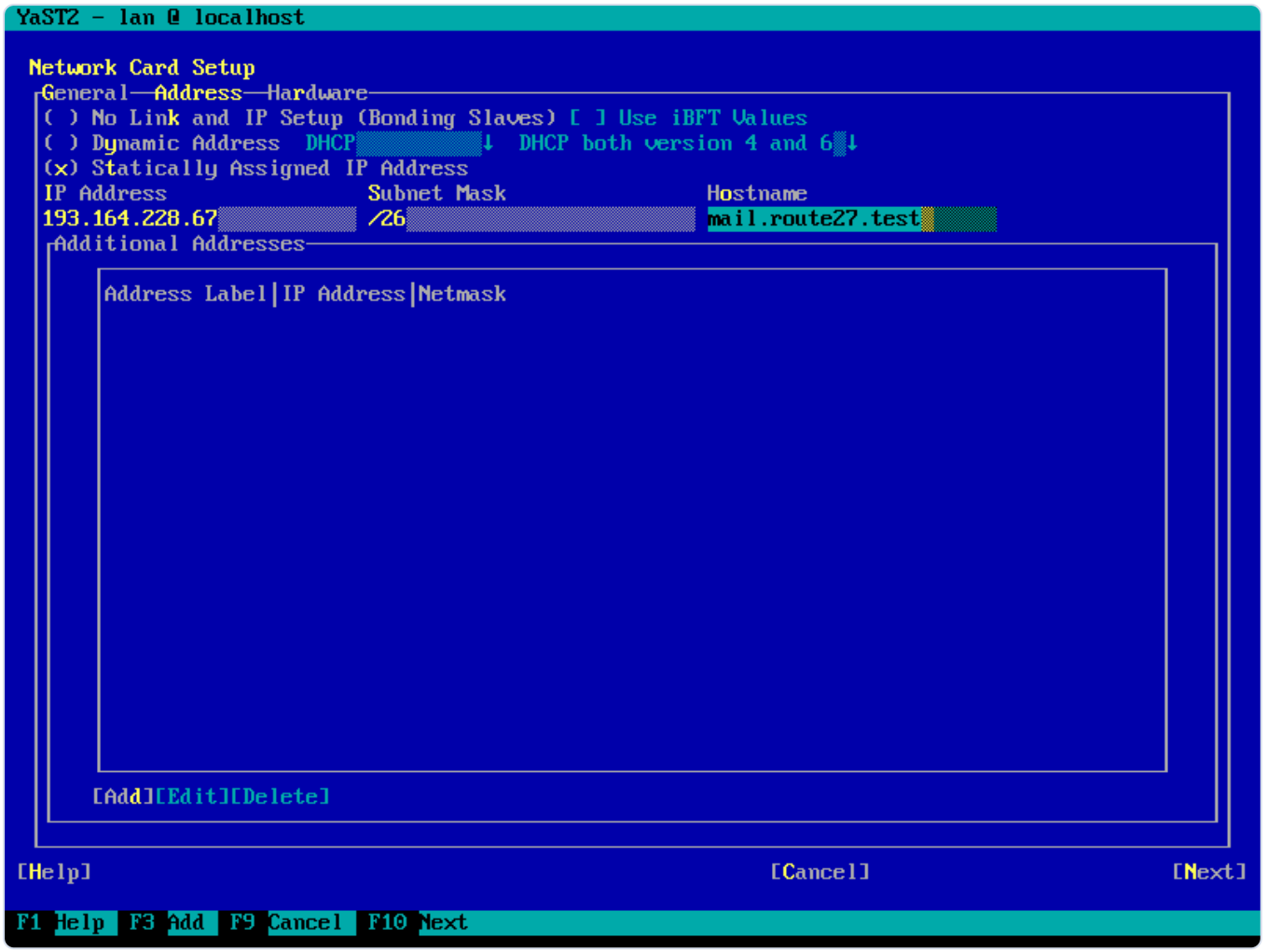
Hostname & FQDN setup

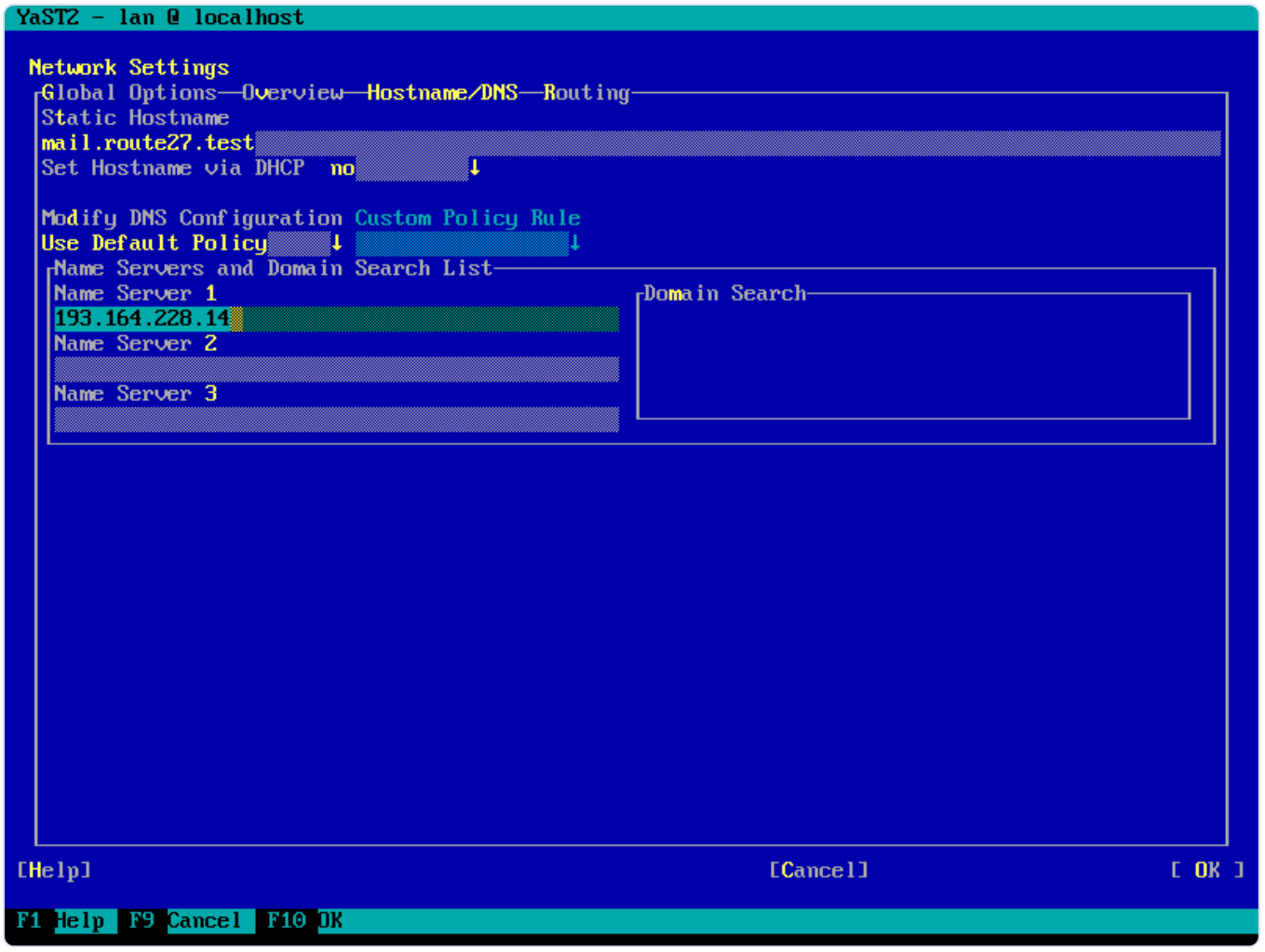
It is a requirement to setup the system hostname and domainname correctly.

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`.

© grommunio GmbH — CC BY-SA 4.0

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.



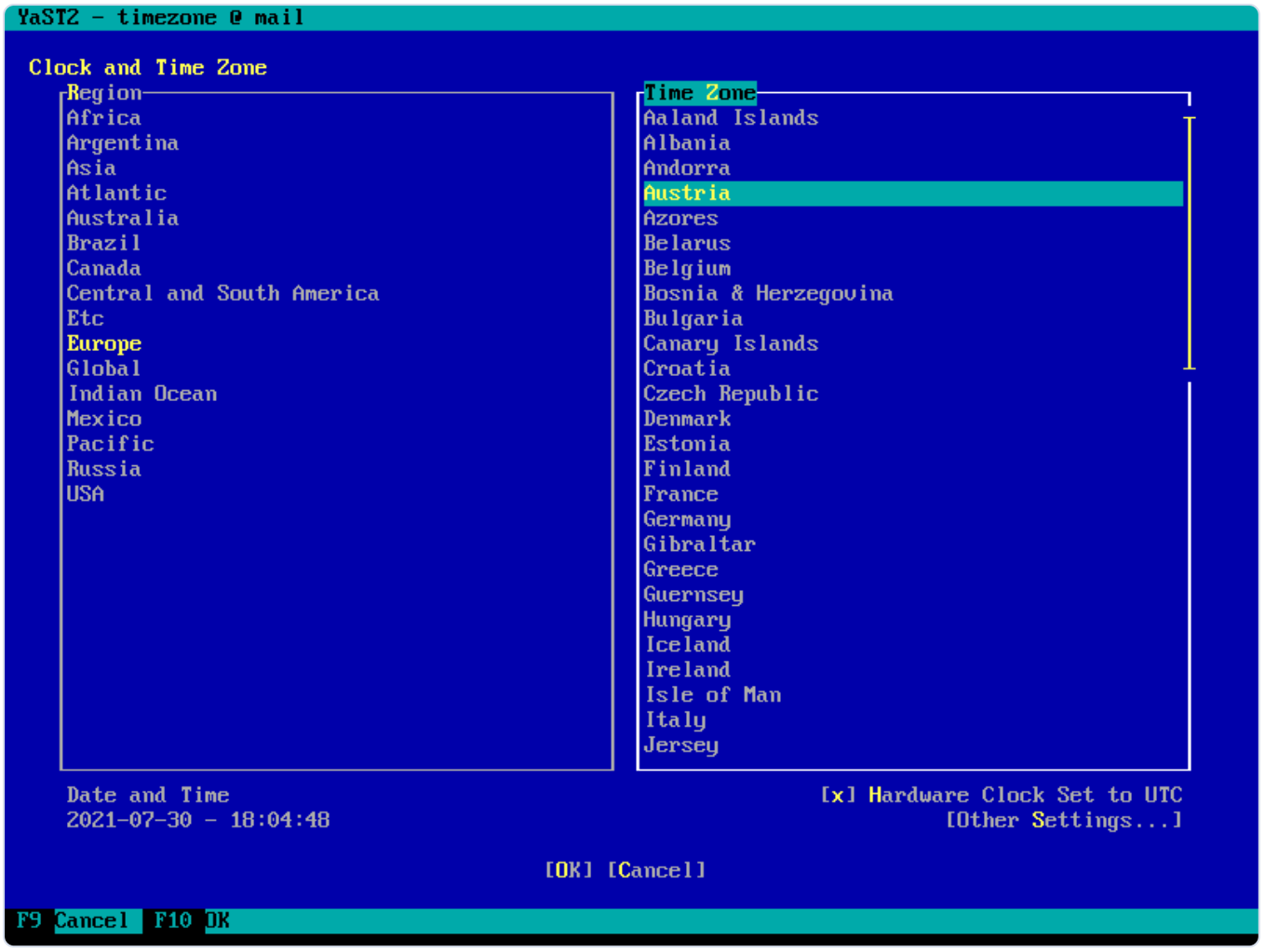


> **⚠Caution**
>
> To verify the settings, the command `hostname` should return the FQDN of the system.

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.

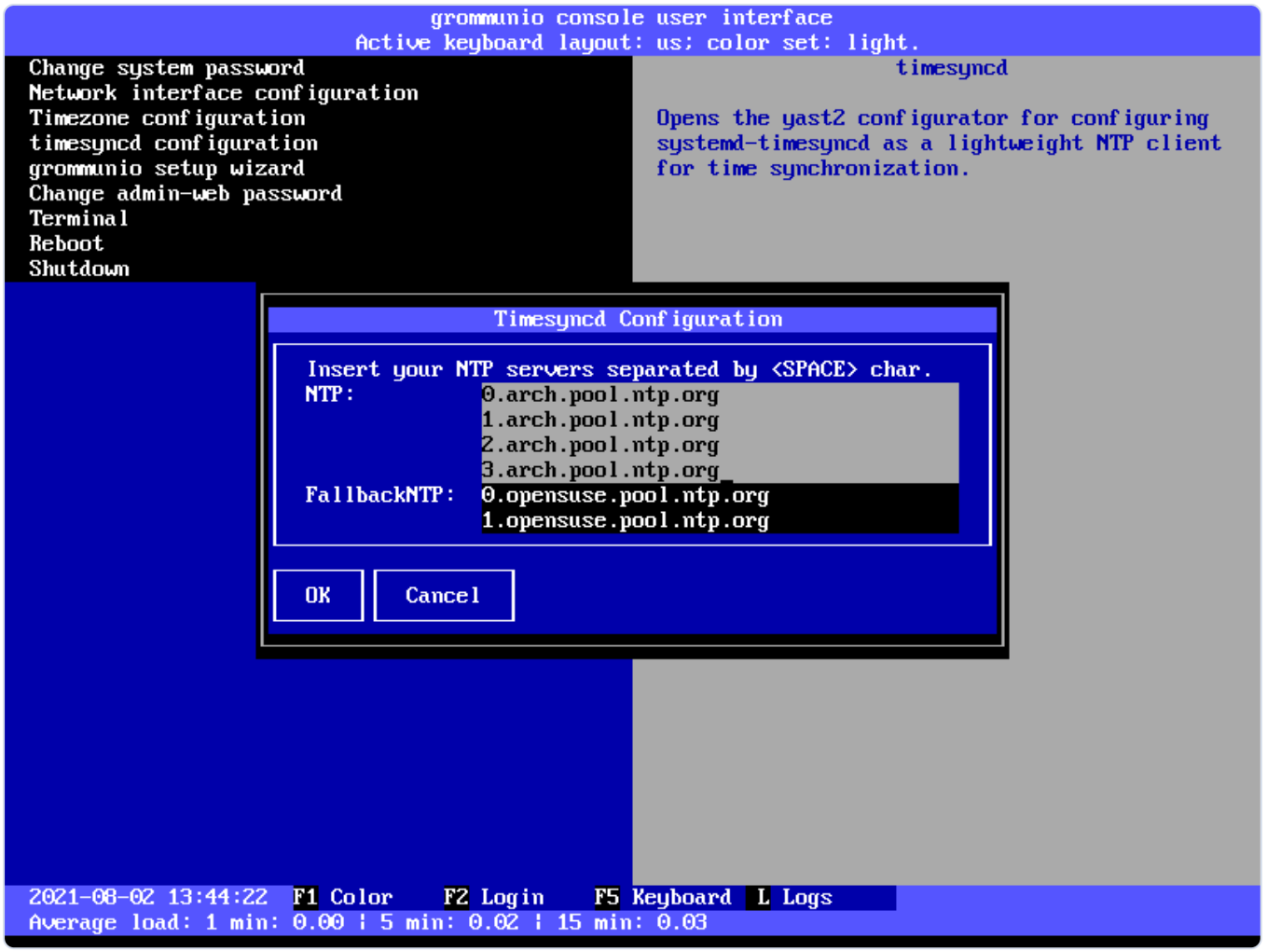
© grommunio GmbH — CC BY-SA 4.0



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.).

© grommunio GmbH — CC BY-SA 4.0



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

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.

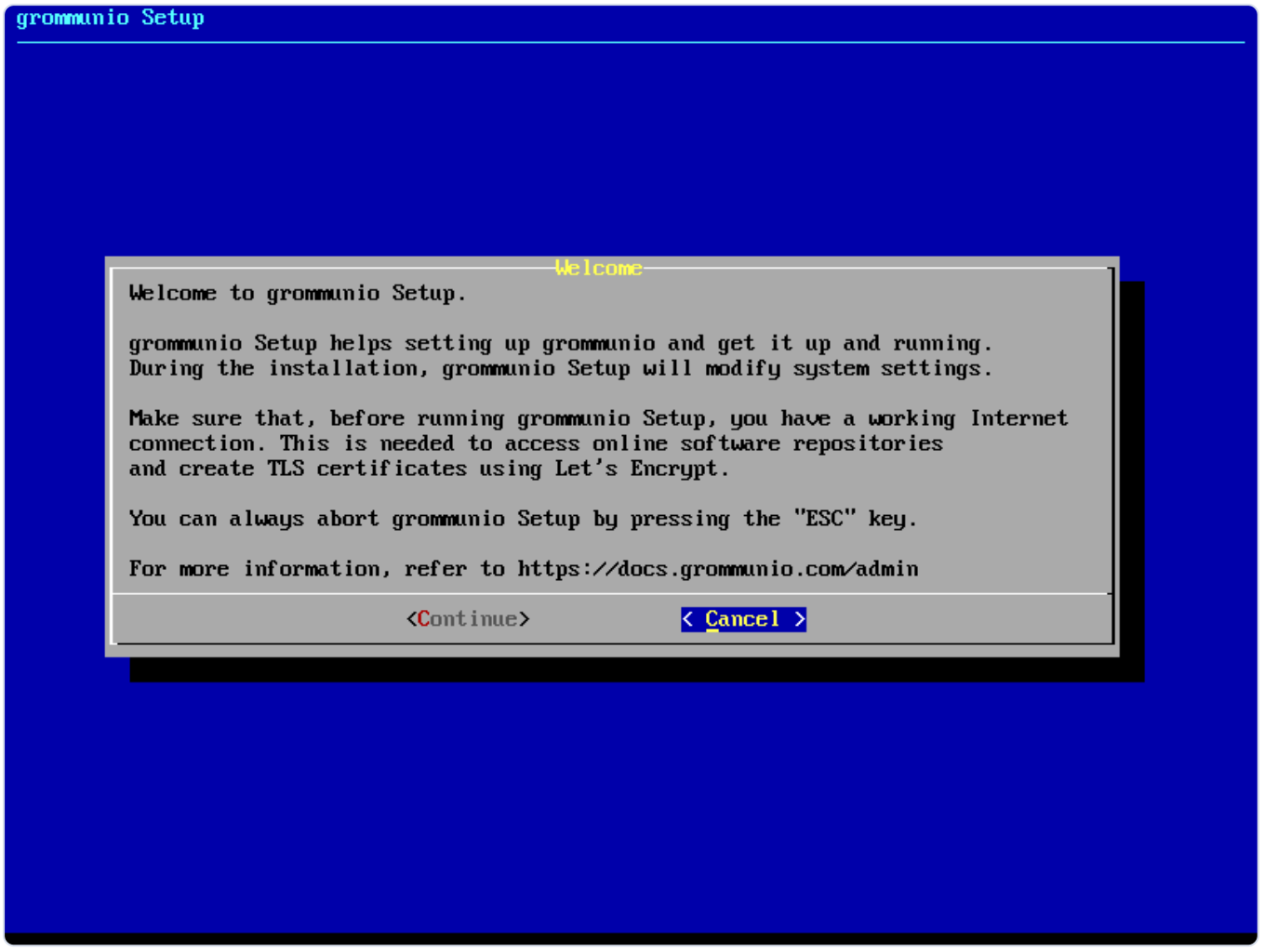
The menu entry `grommunio setup wizard` initiates the `grommunio-setup` program which walks you through the initial setup of grommunio.

© grommunio GmbH — CC BY-SA 4.0

> **⚠Caution**
>
> 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.

Welcome screen

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

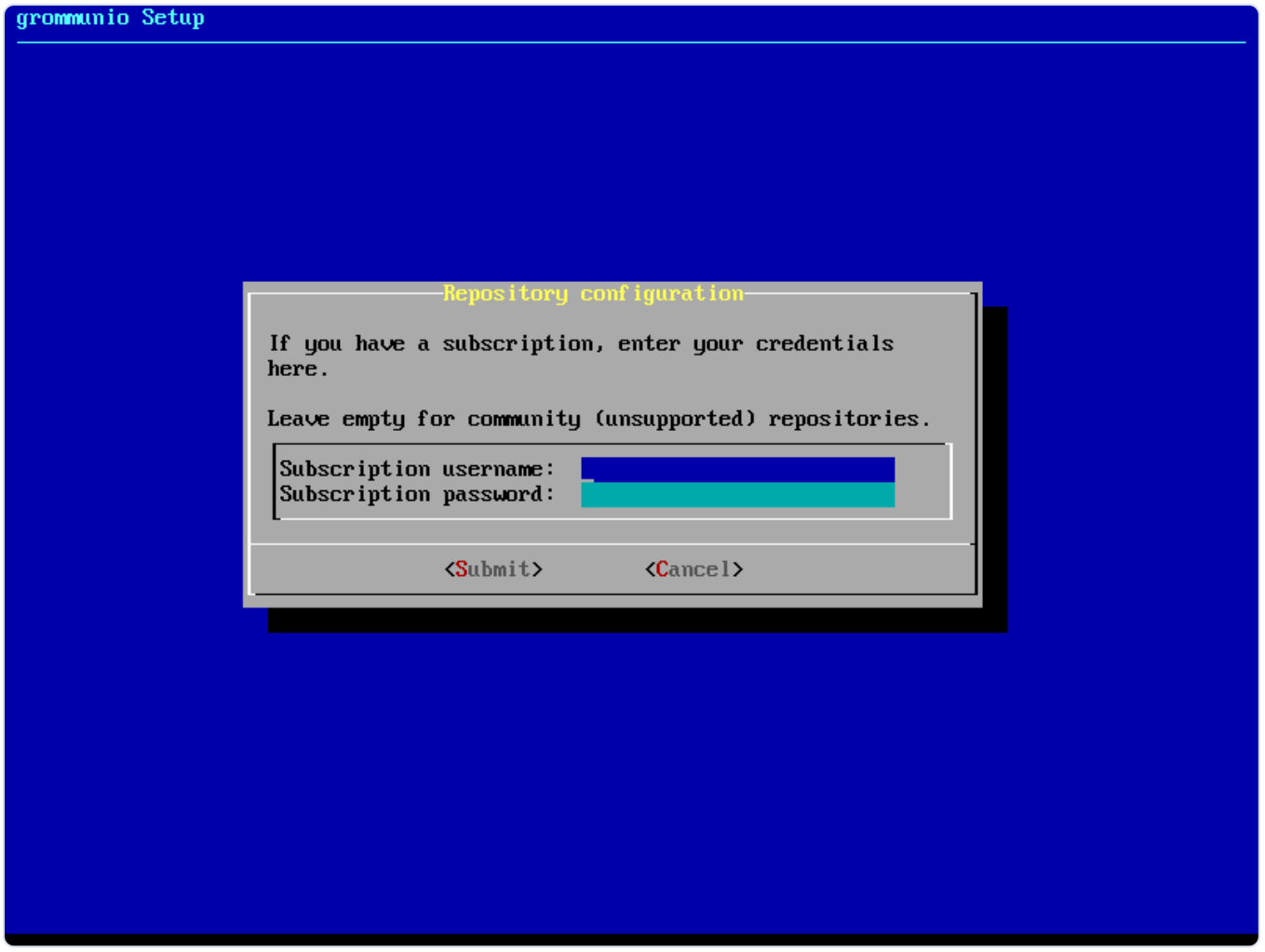


Repository setup

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.

© grommunio GmbH — CC BY-SA 4.0

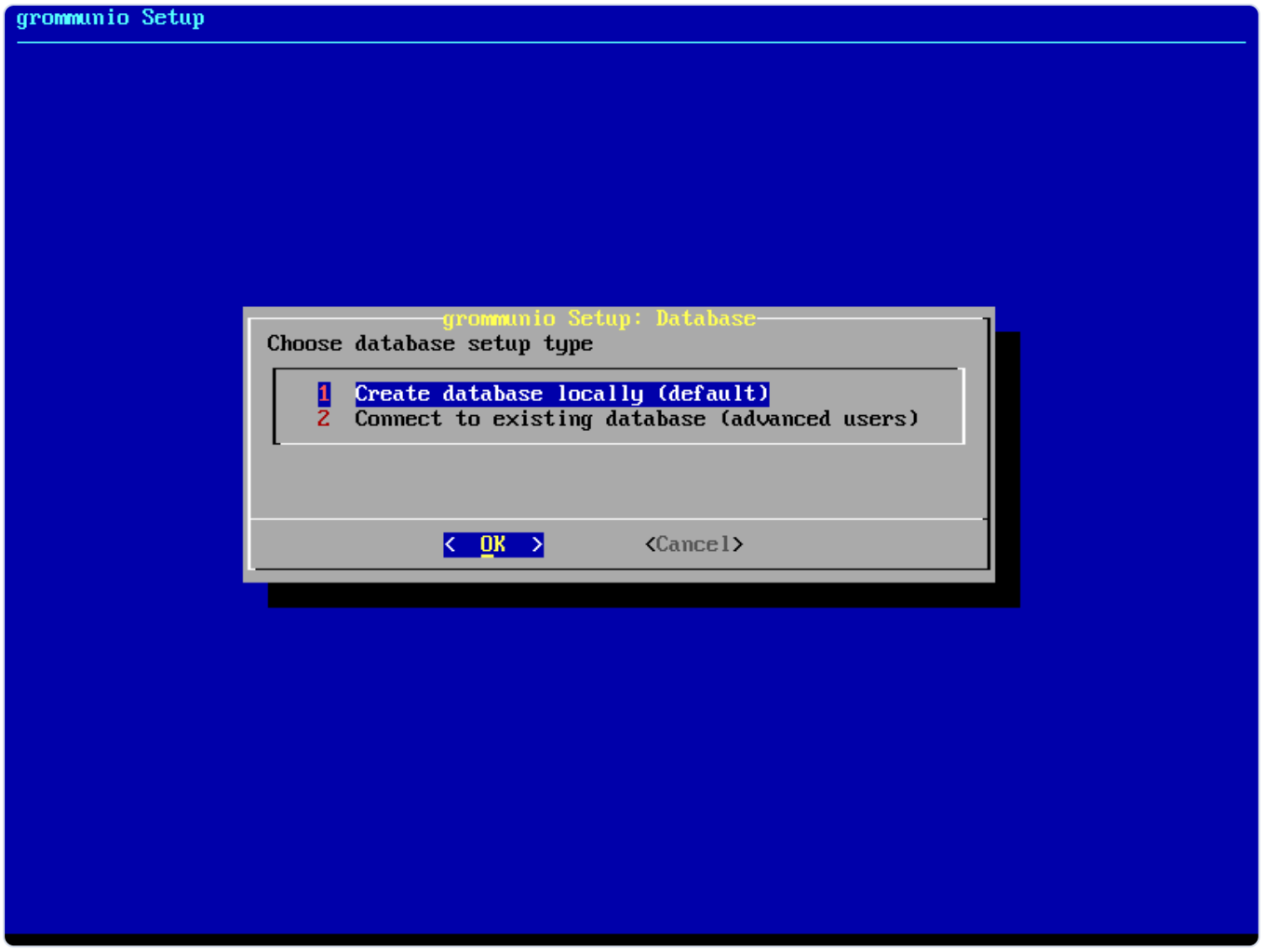
> **Note**
>
> Community repositories are delivered on a best-effort basis and are not supported. While grommunio welcomes community members to use grommunio, the software distribution available with the subscription repositories include production-relevant benefits. Subscription repositories (available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



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.

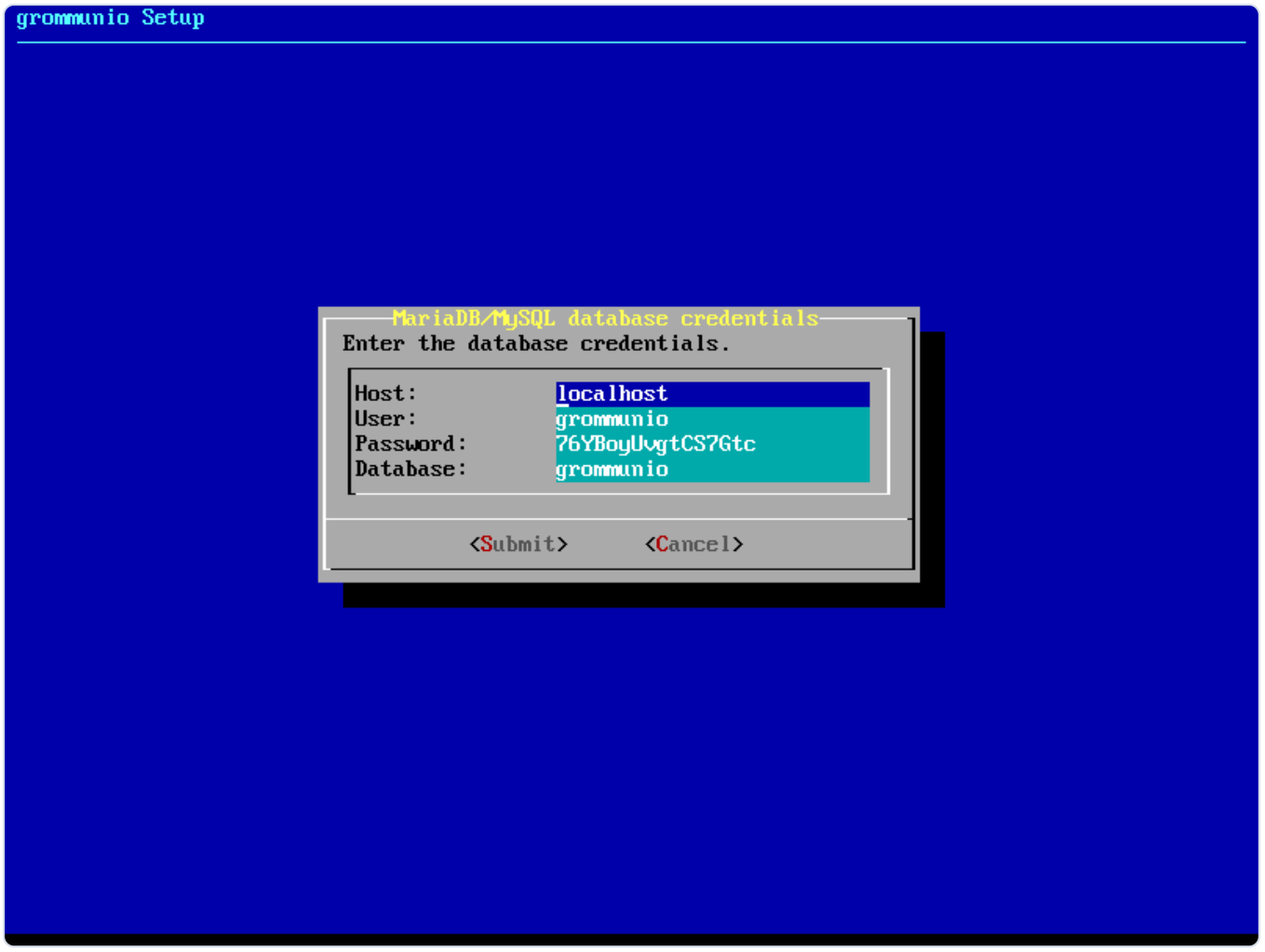
© grommunio GmbH — CC BY-SA 4.0



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.

© grommunio GmbH — CC BY-SA 4.0



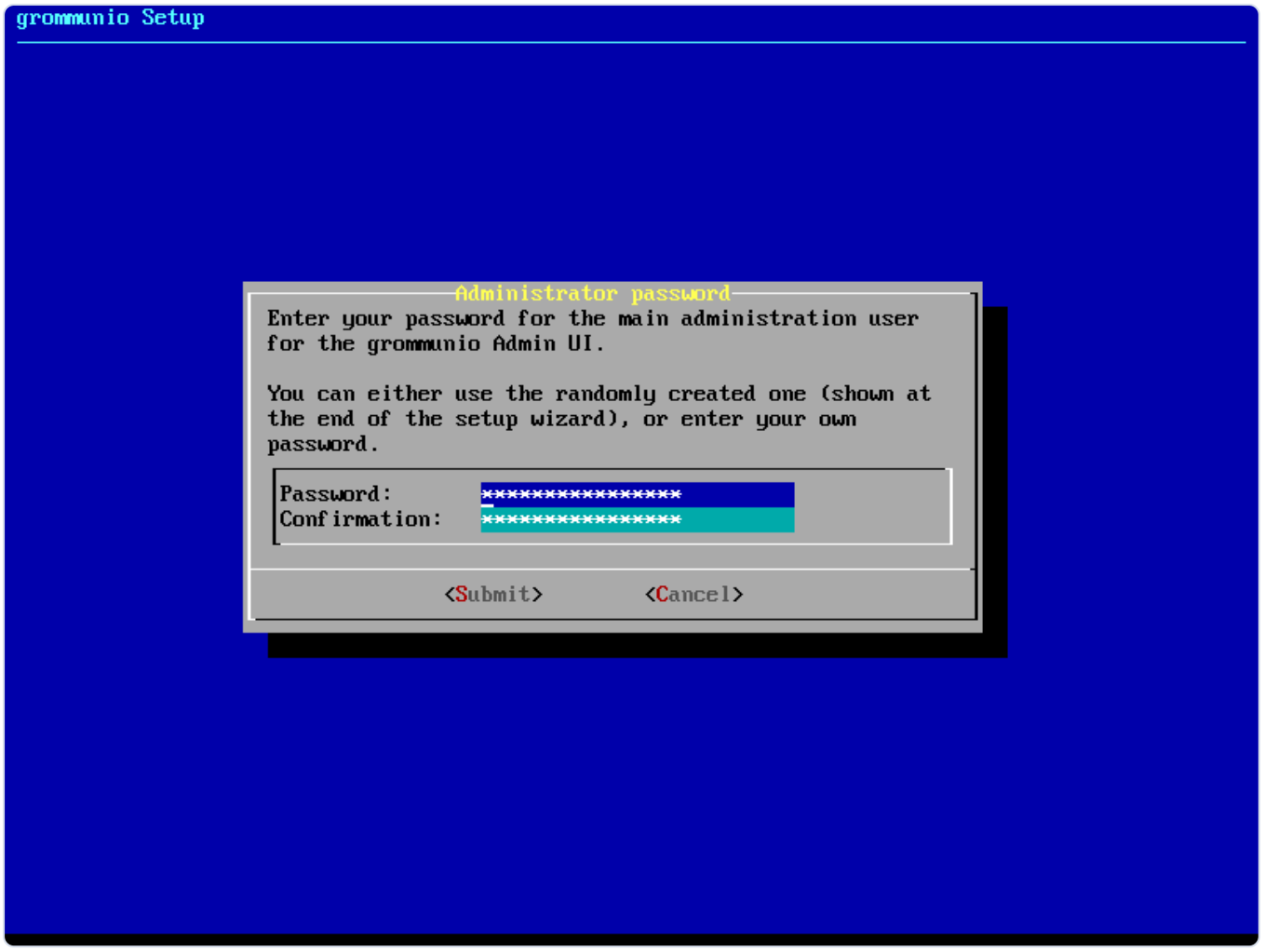
Administration User

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.

> **⚠Caution**
>
> 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`.

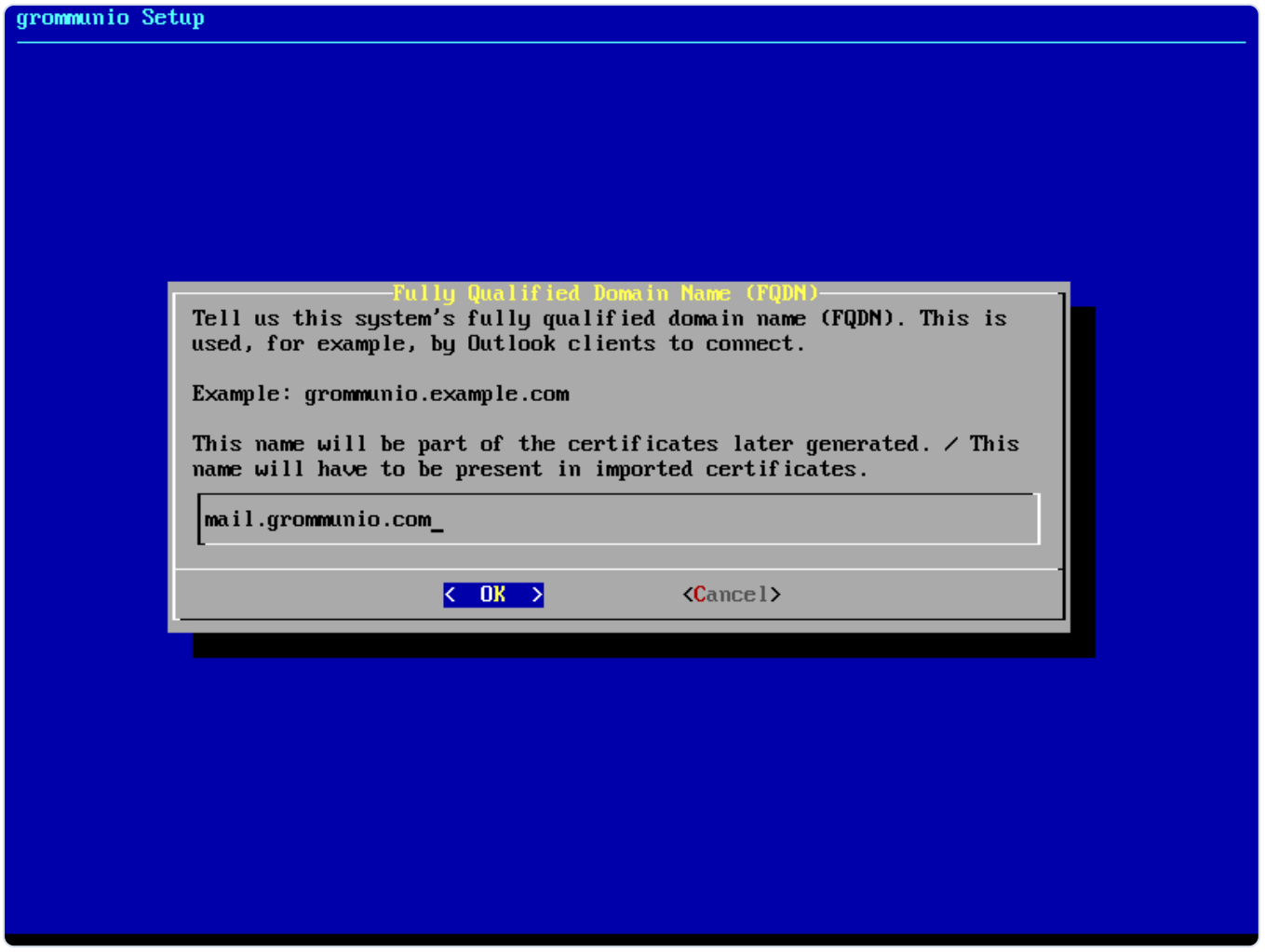
© grommunio GmbH — CC BY-SA 4.0



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`.

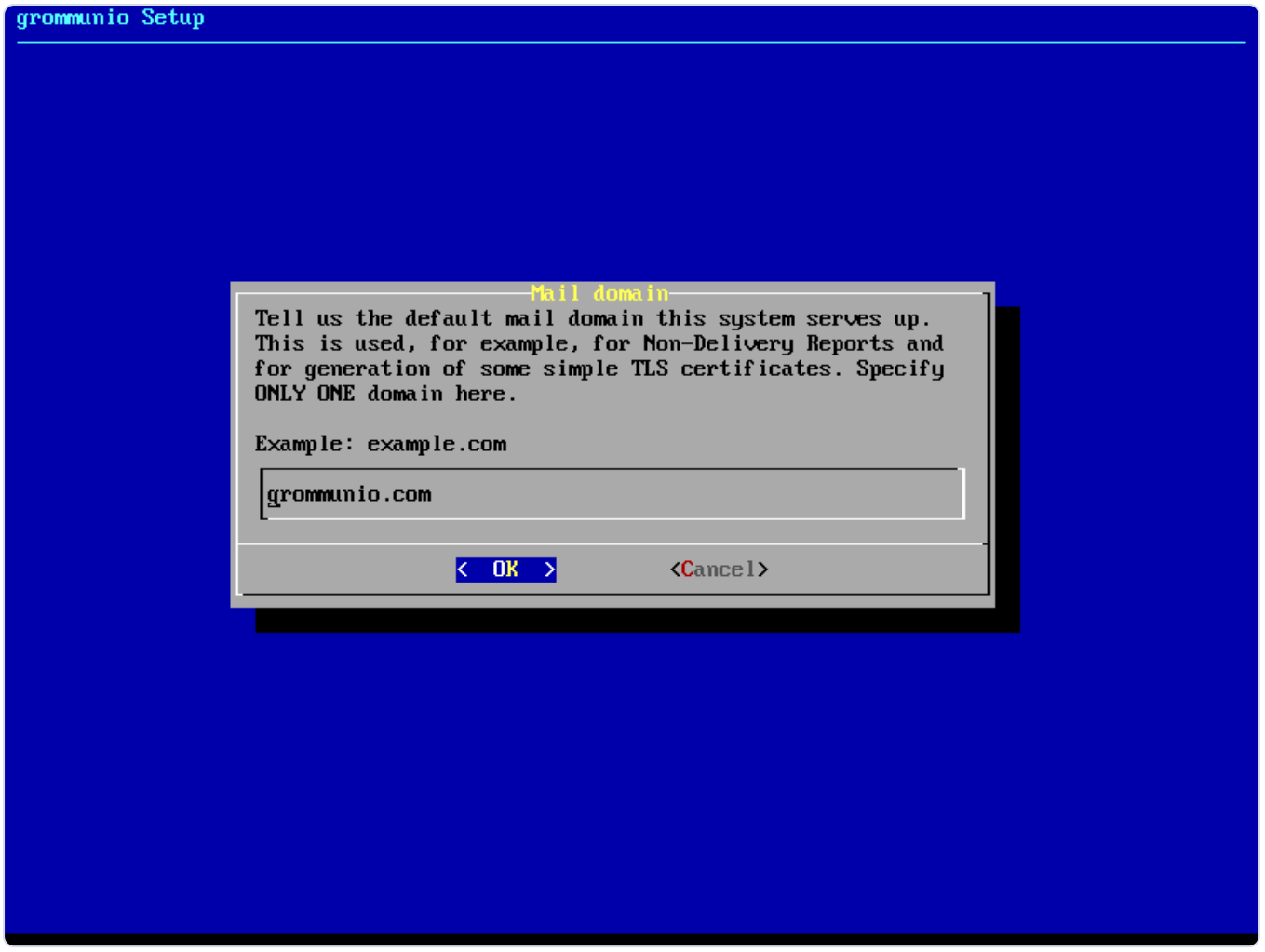
© grommunio GmbH — CC BY-SA 4.0



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.

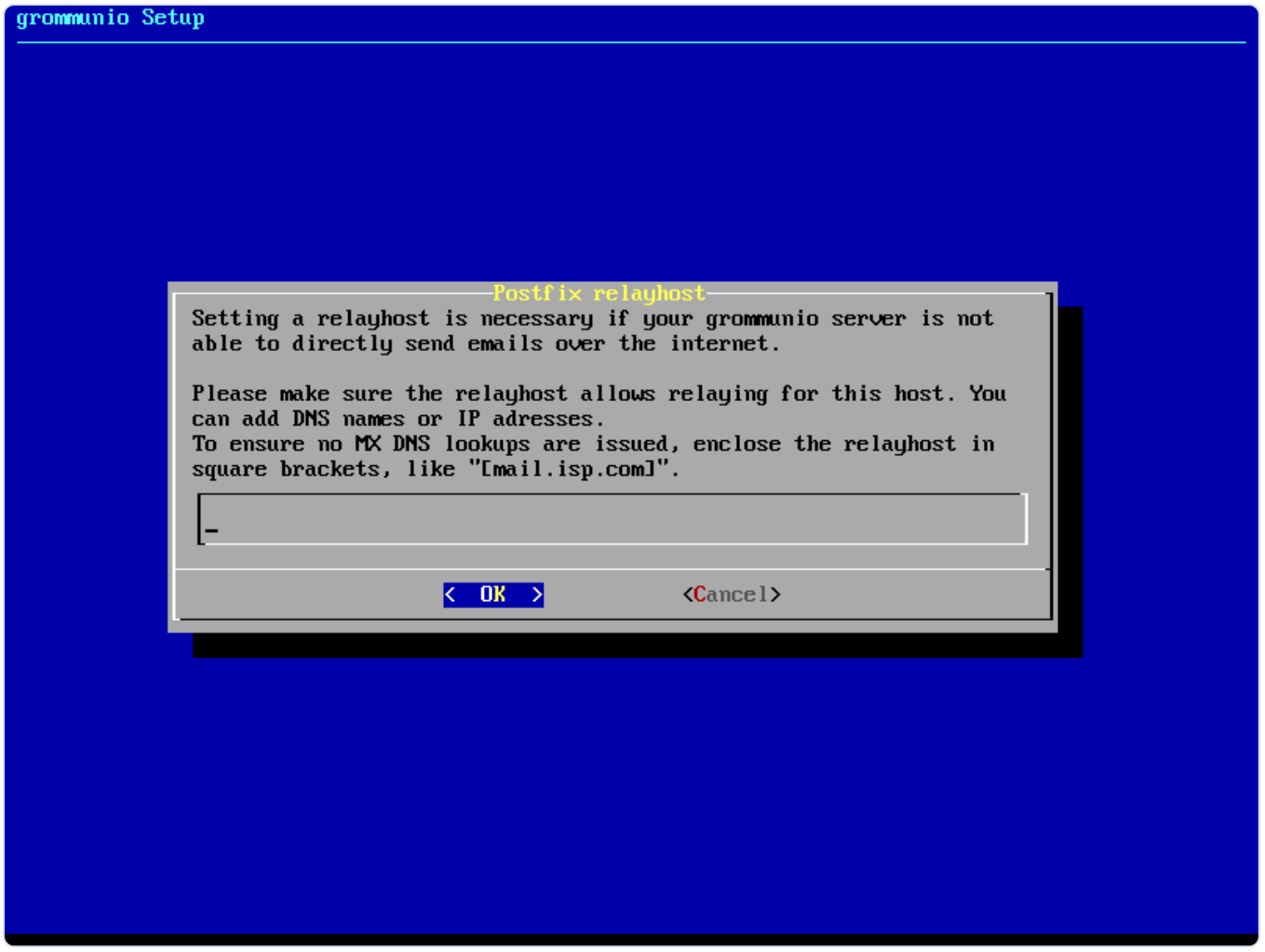
© grommunio GmbH — CC BY-SA 4.0



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]".

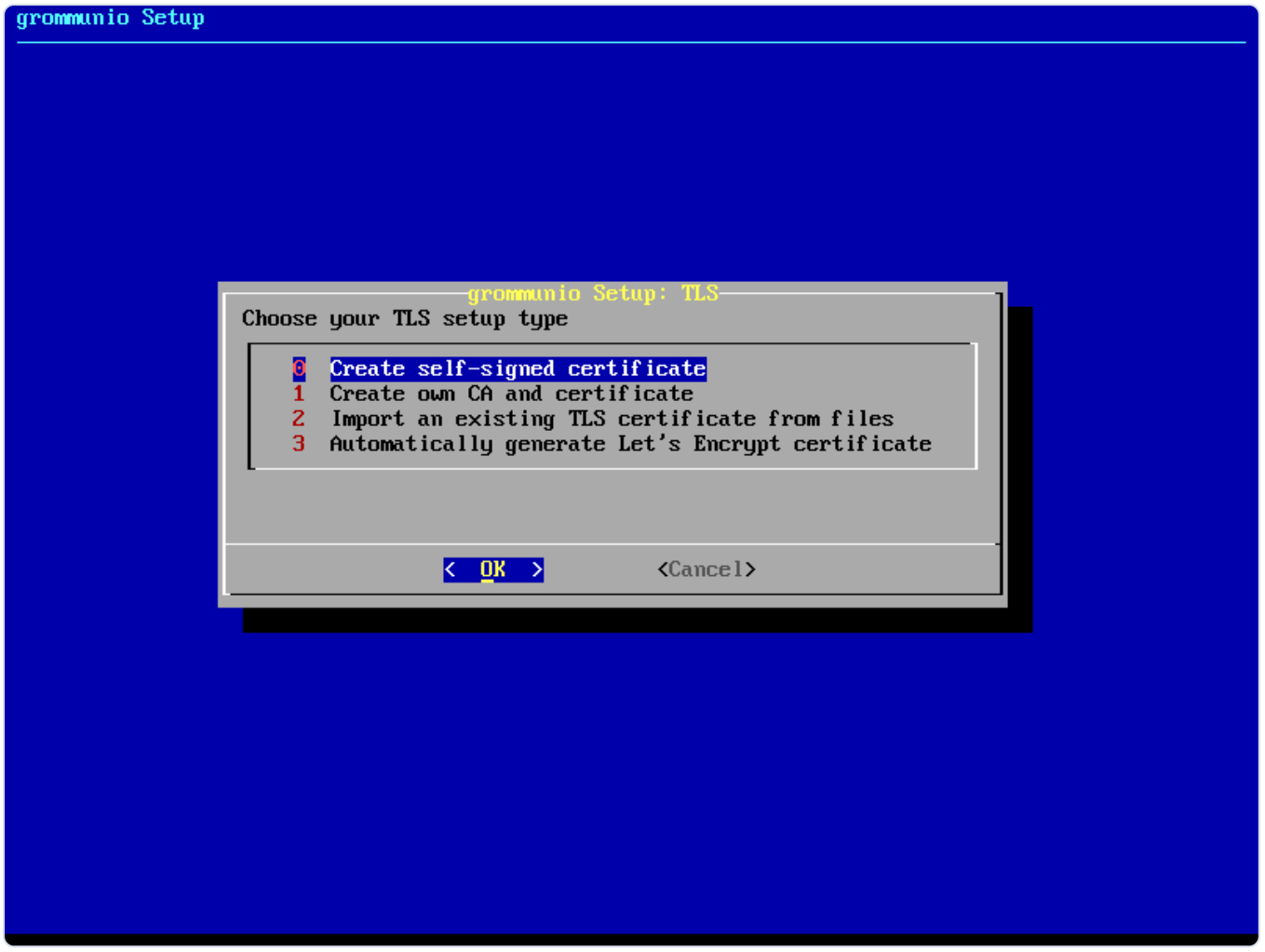
© grommunio GmbH — CC BY-SA 4.0



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:

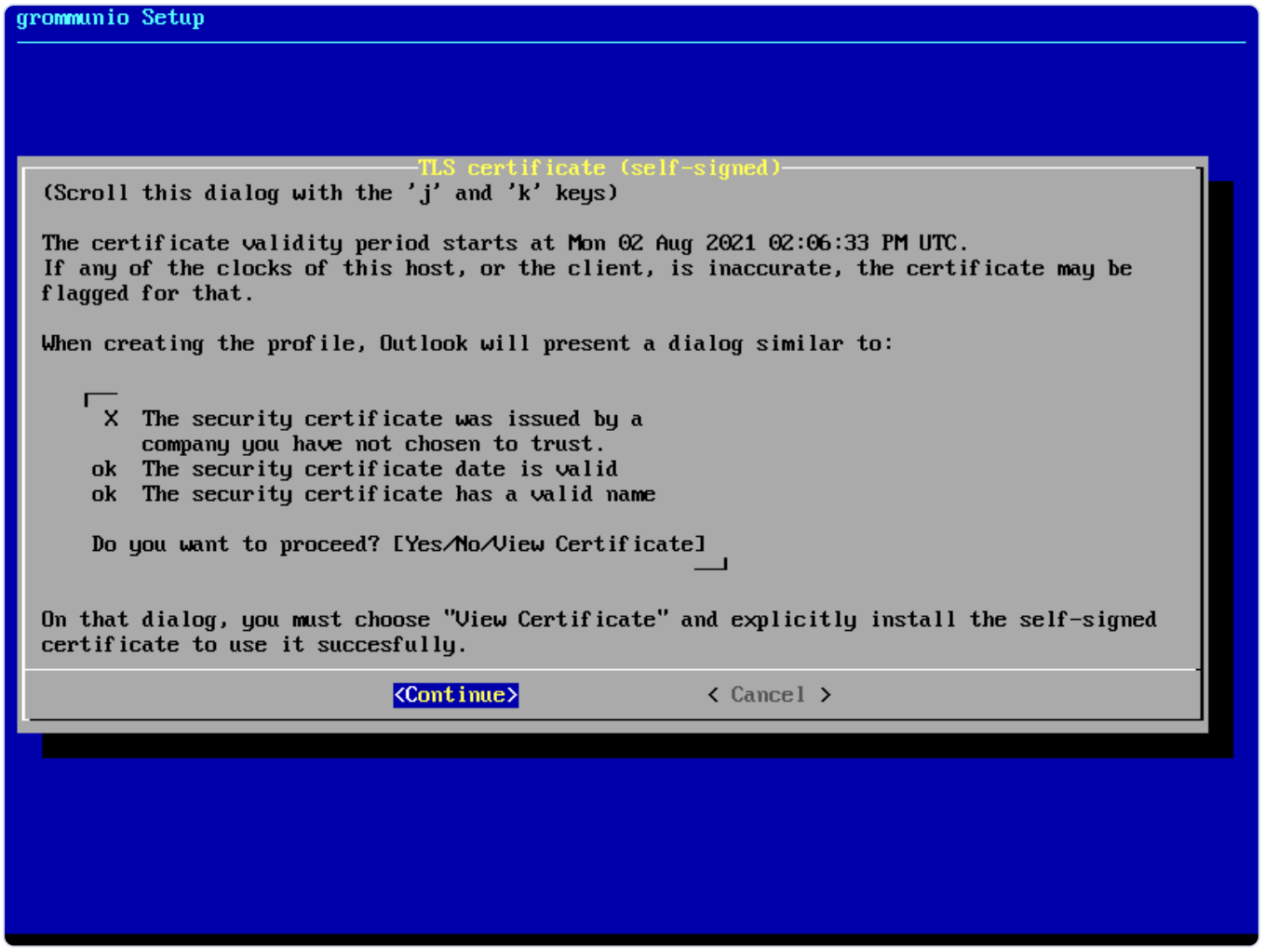
© grommunio GmbH — CC BY-SA 4.0



0: **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.

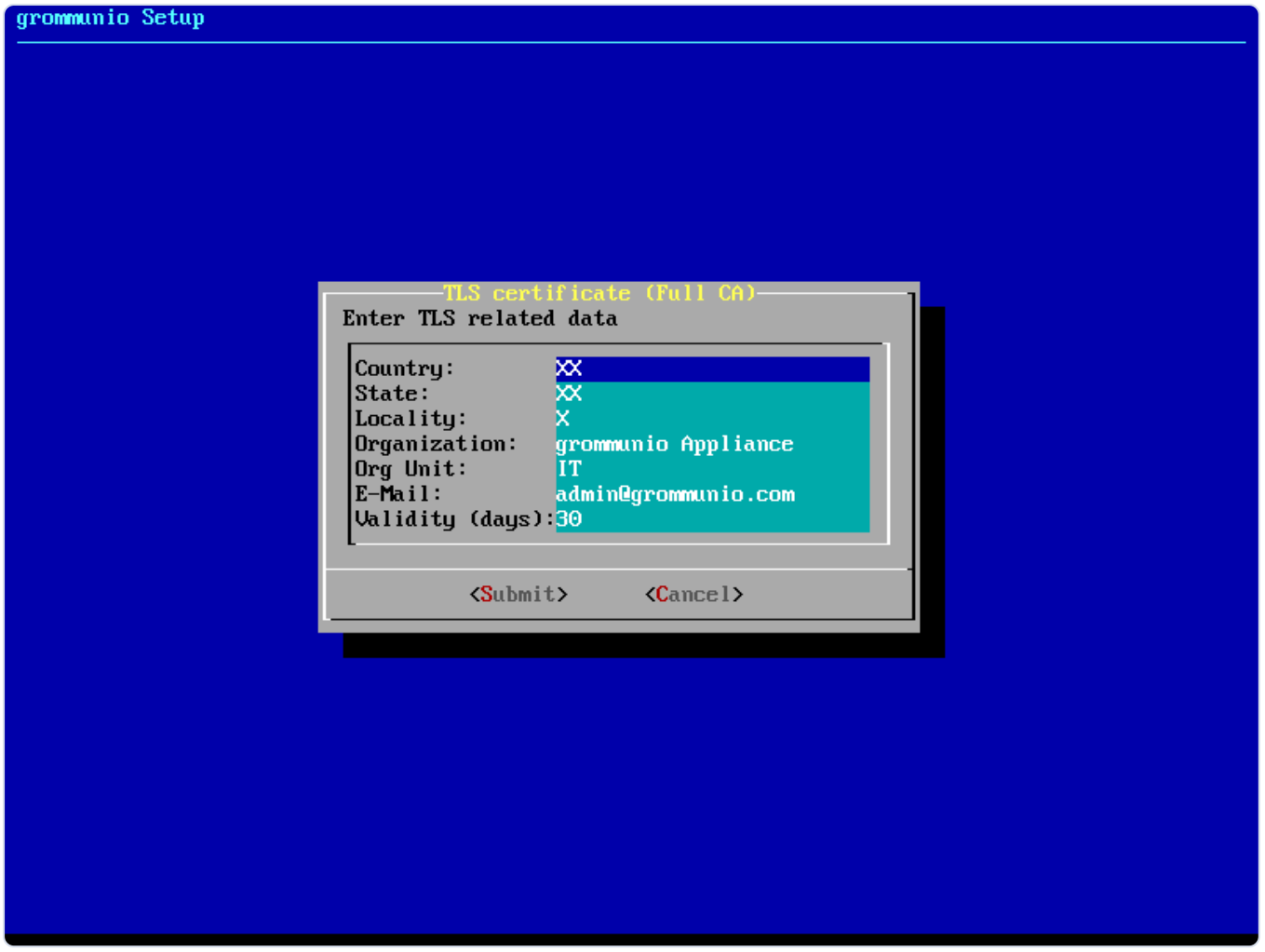
© grommunio GmbH — CC BY-SA 4.0



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.

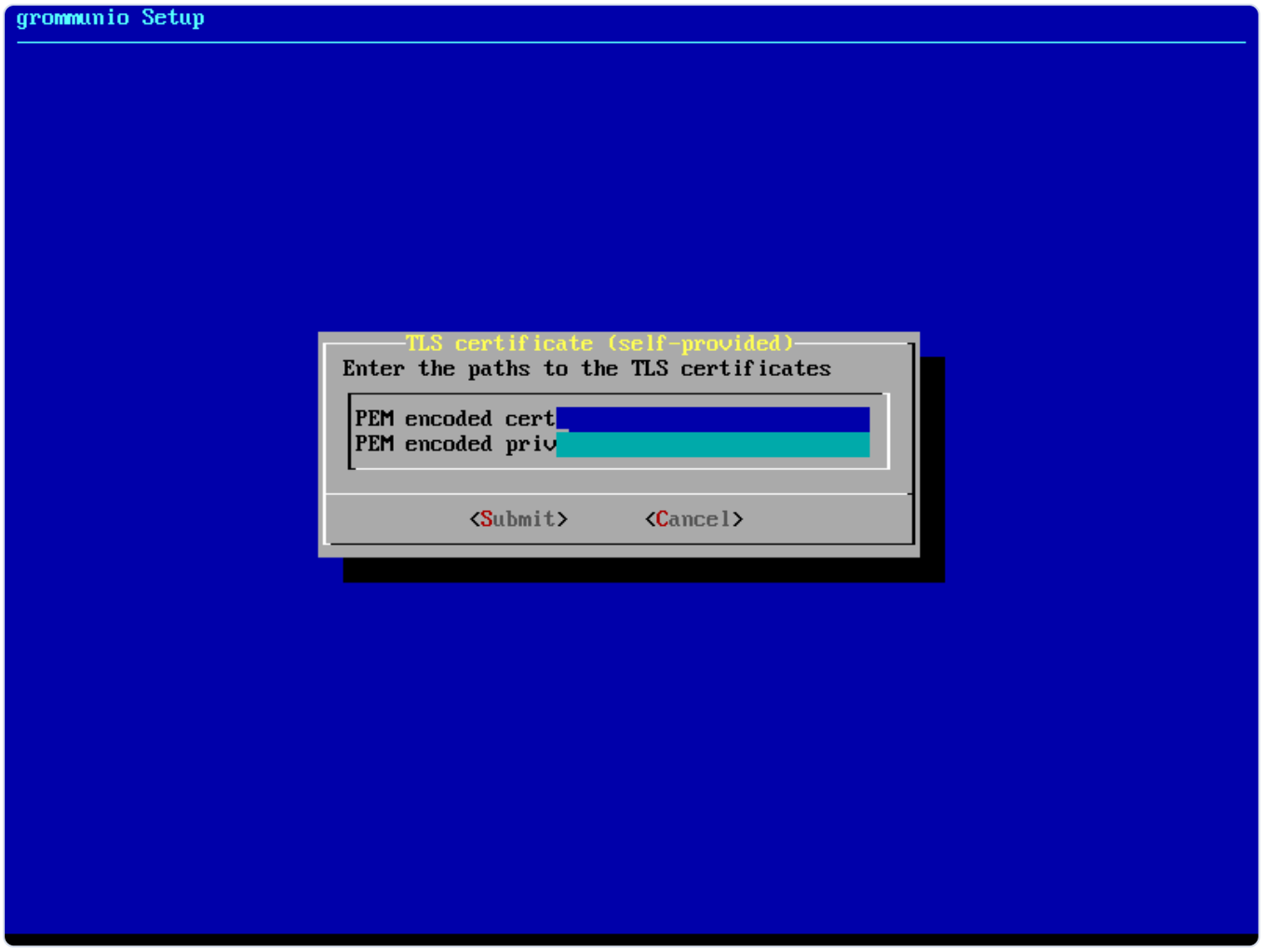
© grommunio GmbH — CC BY-SA 4.0



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.

© grommunio GmbH — CC BY-SA 4.0



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.

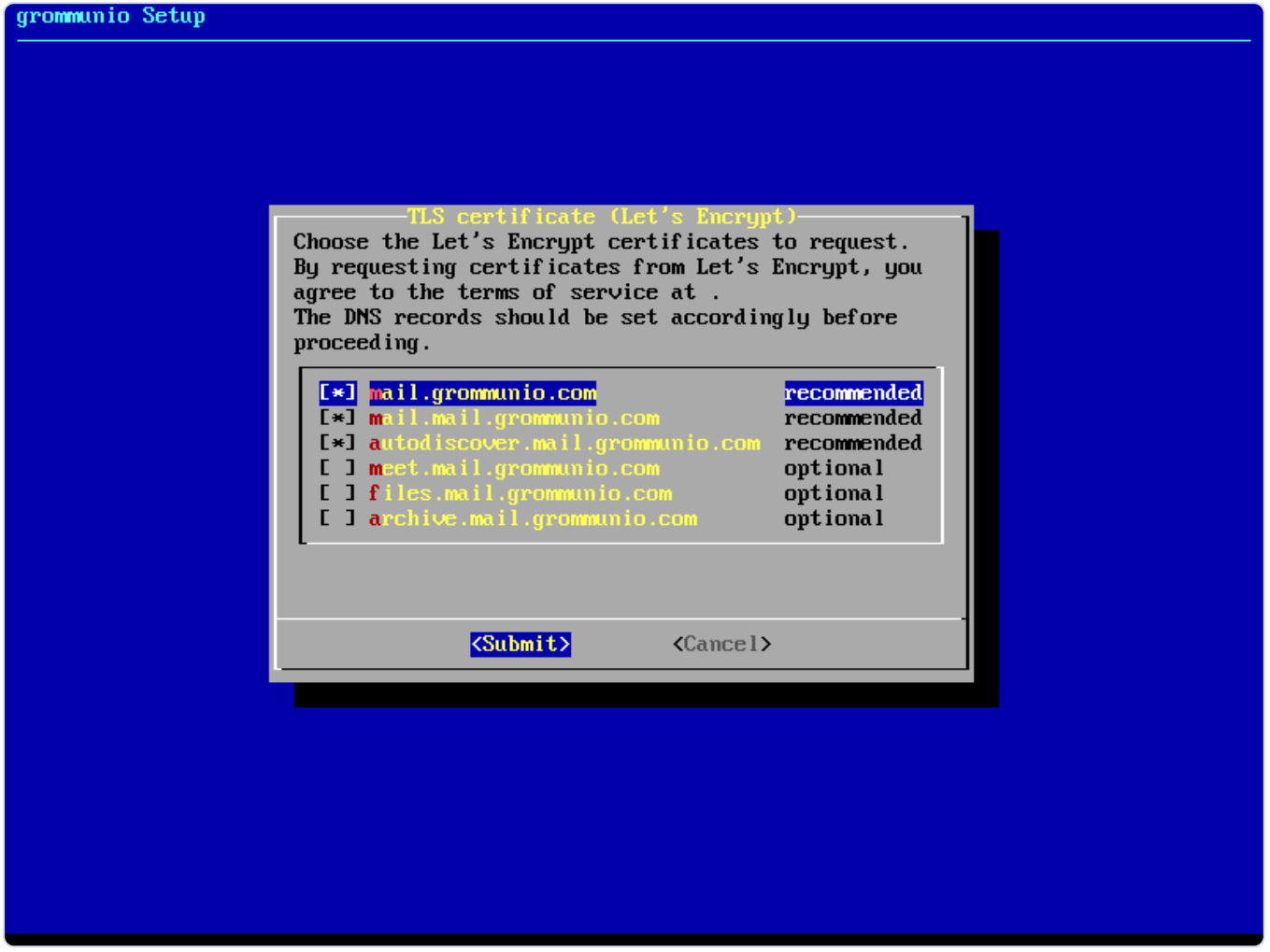
3.a: **Generation of certificates with Let's Encrypt for Multi-Domains**

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

© grommunio GmbH — CC BY-SA 4.0

```
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.

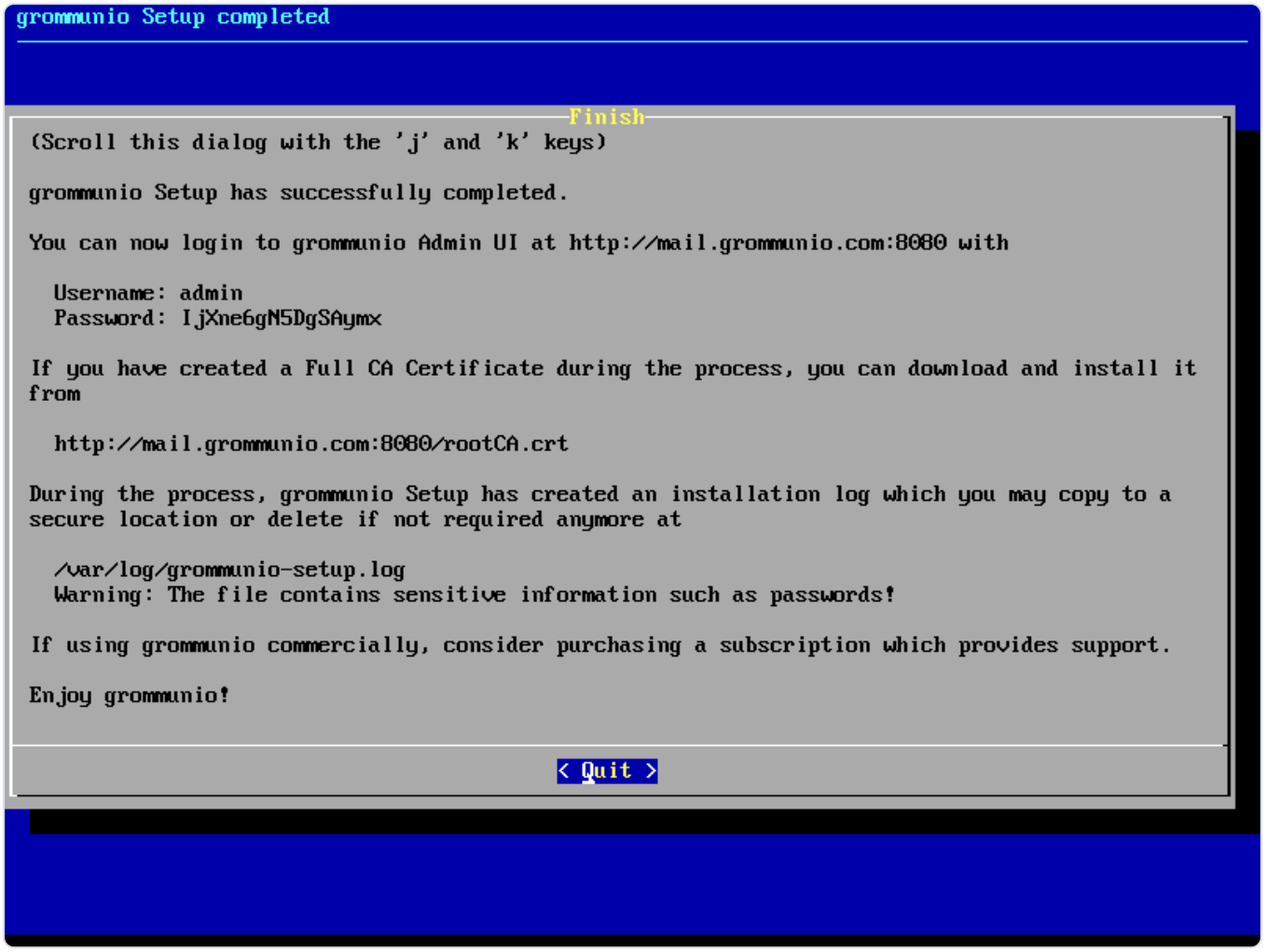


Any certificates so generated are placed in `/etc/grommunio/ssl` and are automatically referenced by any services of the appliance.

© grommunio GmbH — CC BY-SA 4.0

Setup finalization

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

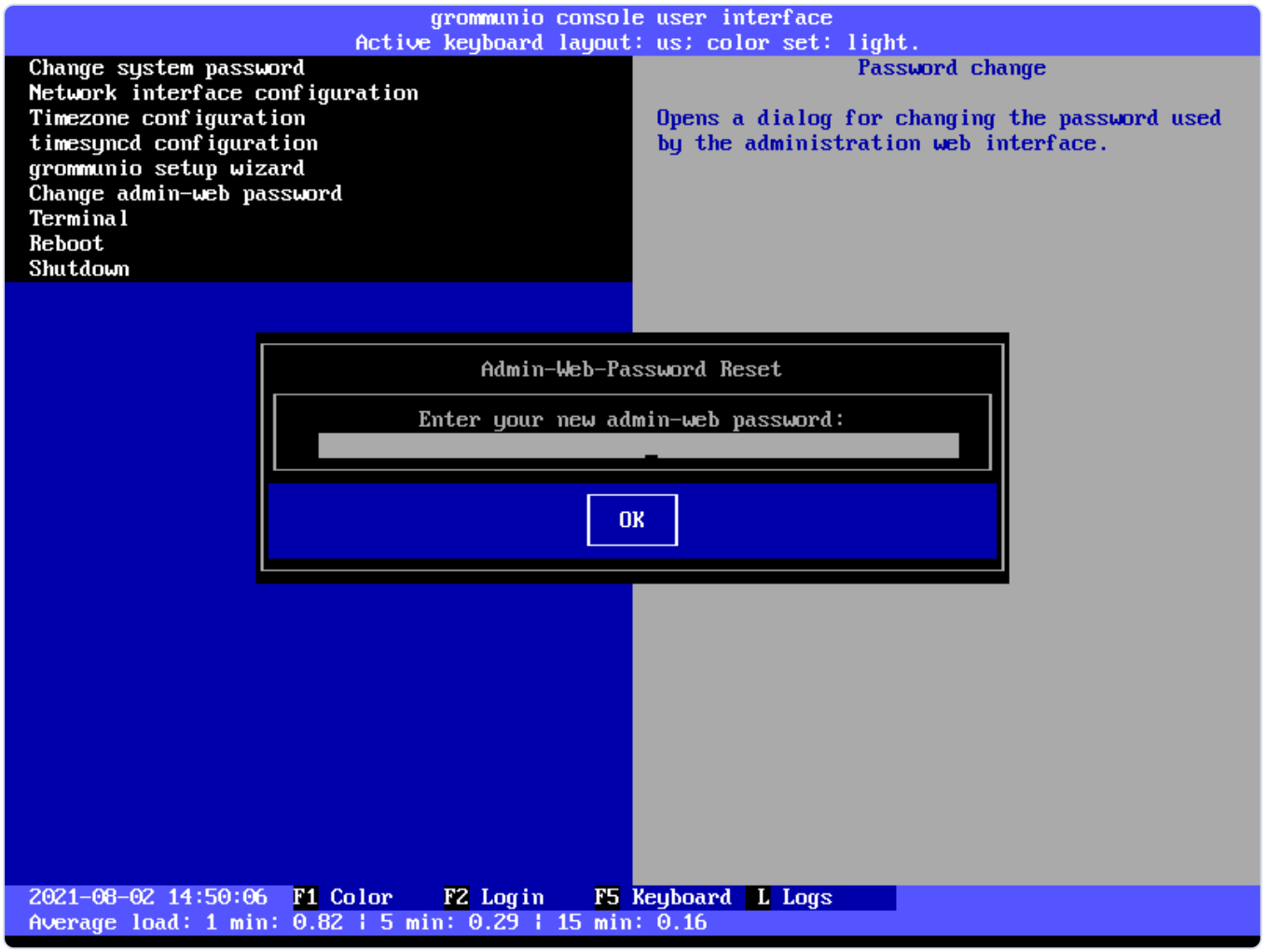


> **⚠Caution**
>
> All installation/setup relevant information is stored at /var/log/grommunio-setup.log. This file includes the passwords used for initialization which you may copy to a secure location or delete if not required anymore.

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`.

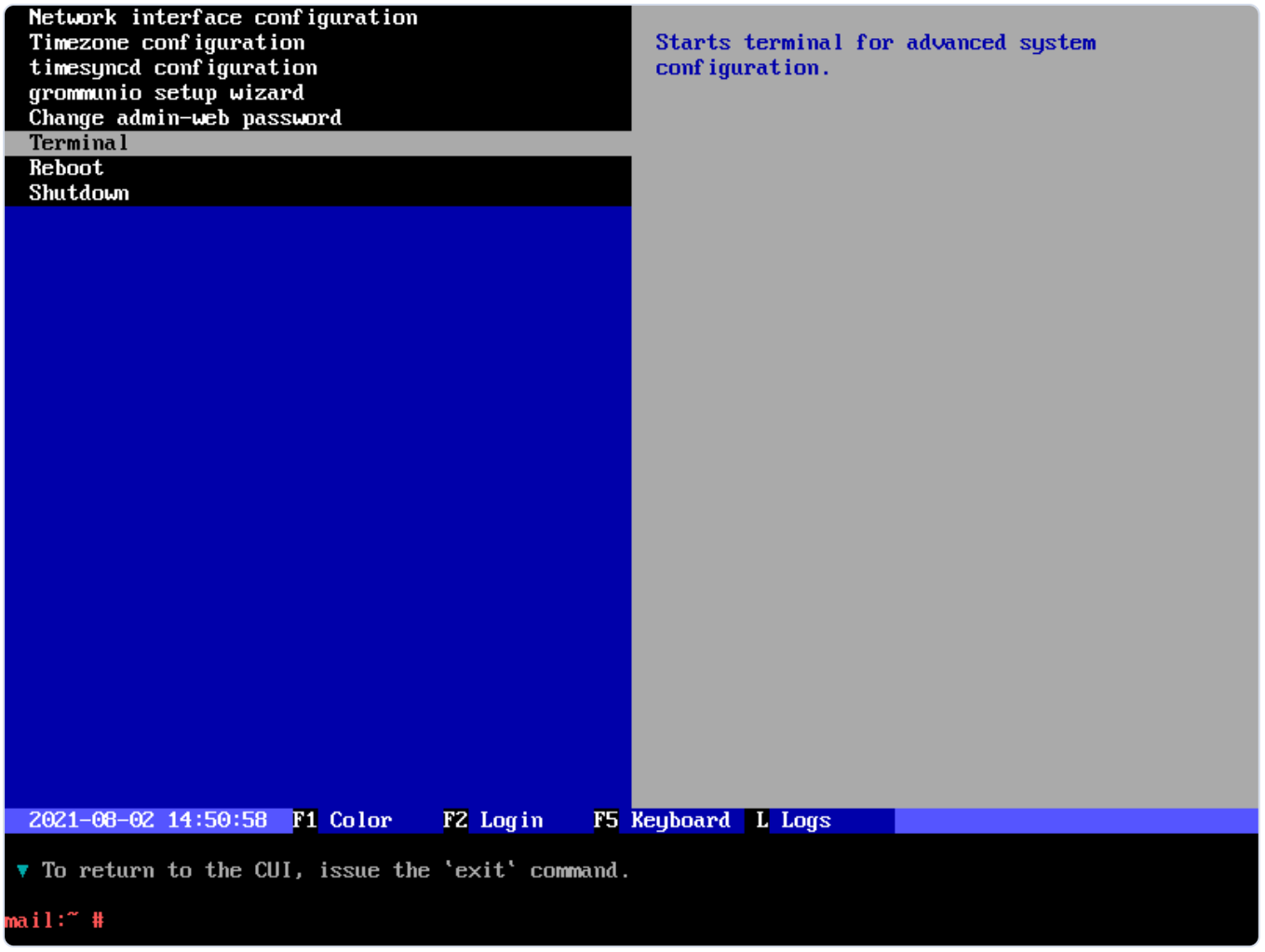
© grommunio GmbH — CC BY-SA 4.0



Terminal

The option `Terminal` enables a classic 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.

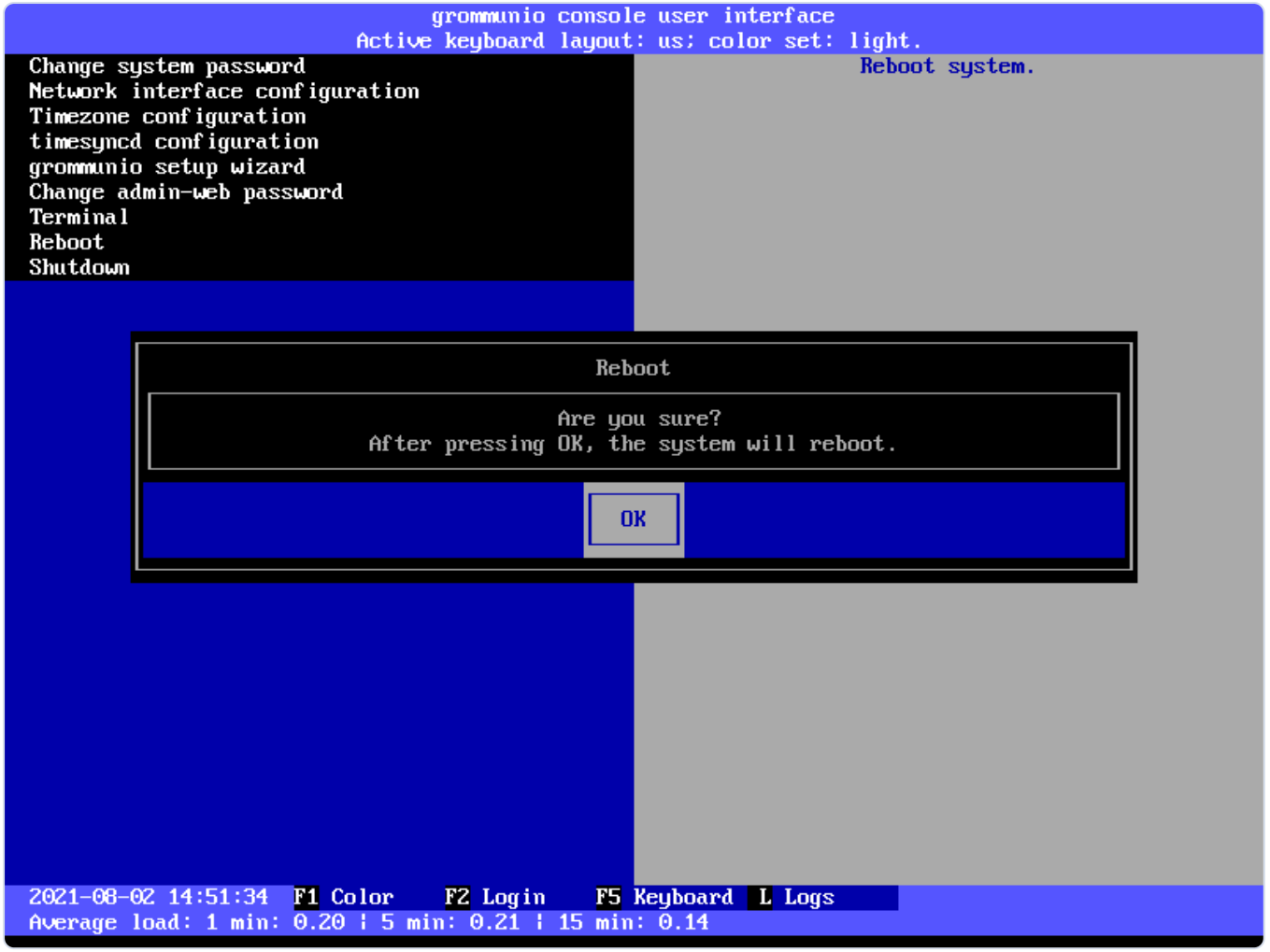
© grommunio GmbH — CC BY-SA 4.0



> **⚠Caution**
>
> 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.

© grommunio GmbH — CC BY-SA 4.0

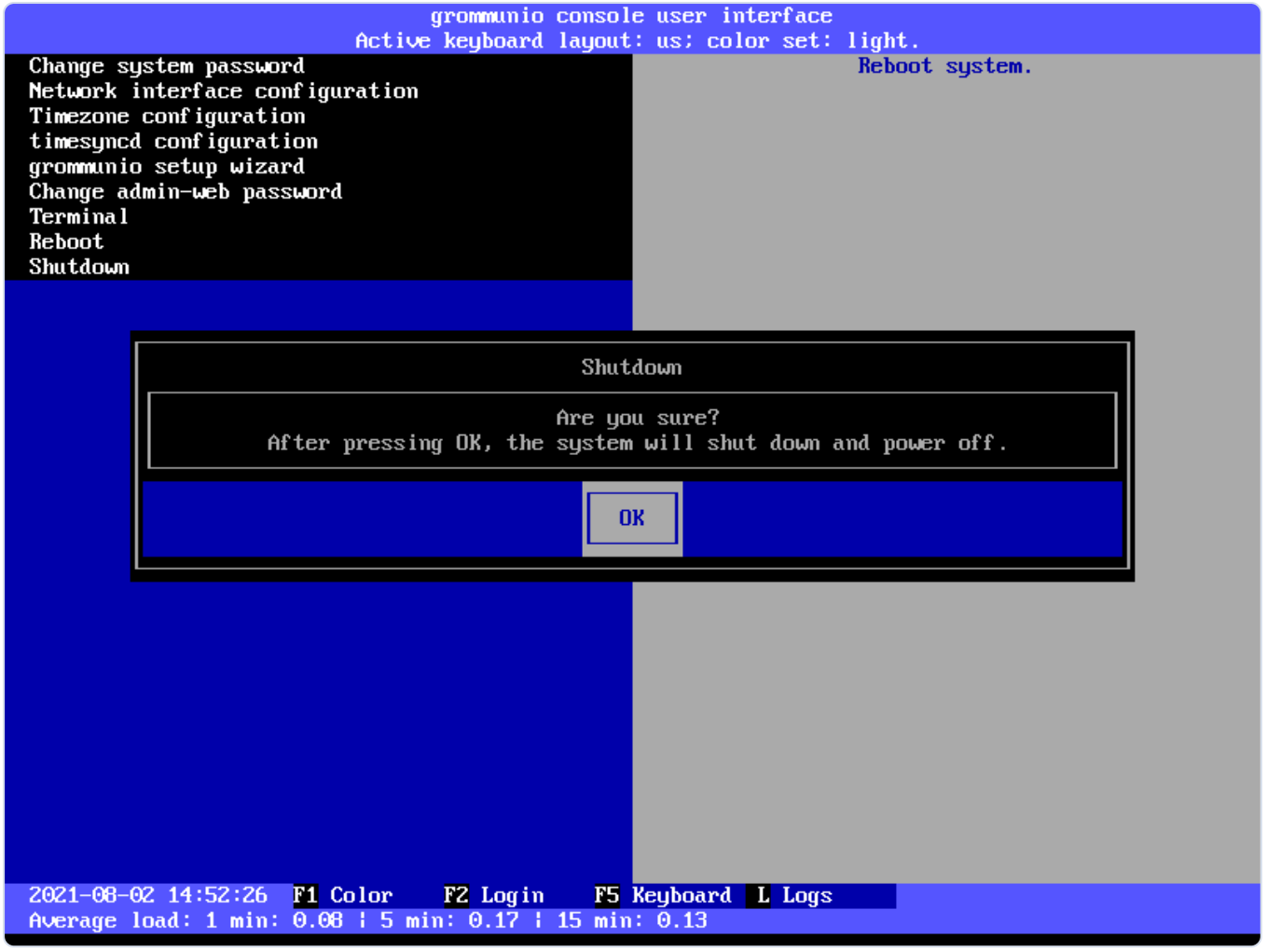
Reboot



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

© grommunio GmbH — CC BY-SA 4.0

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.

© grommunio GmbH — CC BY-SA 4.0

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.

> **Caution**
>
> 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

Establish networking

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

© grommunio GmbH — CC BY-SA 4.0

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

2 links listed.

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.]

© grommunio GmbH — CC BY-SA 4.0

```
localhost:~ # ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 100
    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 def
    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

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.

© grommunio GmbH — CC BY-SA 4.0

> **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.)

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.

zypp

openSUSE uses yum-style `.repo` files for declaring repositories. For openSUSE Leap 16.0, 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_16.0
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.]

© grommunio GmbH — CC BY-SA 4.0

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

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 10
baseurl=https://download.grommunio.com/community/EL10/
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`.

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_13
Suites: Debian_13
Components: main
Signed-By: /usr/share/keyrings/download.grommunio.com.gpg
```

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

(This equally works for *Ubuntu_24.04*, 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.]

© grommunio GmbH — CC BY-SA 4.0

```
# curl https://download.grommunio.com/RPM-GPG-KEY-grommunio >/etc/apt/trusted.gpg.d/downlo
  % 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 noble InRelease
Hit:2 http://gb.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://gb.archive.ubuntu.com/ubuntu noble-backports InRelease
Get:4 https://download.grommunio.com/community/Ubuntu_24.04 Ubuntu_24.04 InRelease [4,692
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:6 https://download.grommunio.com/community/Ubuntu_24.04 Ubuntu_24.04/main amd64 Packag
Get:7 https://download.grommunio.com/community/Ubuntu_24.04 Ubuntu_24.04/main i386 Package
Fetched 16.4 kB in 1s (23.8 kB/s)
Reading package lists... Done
```

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):

© grommunio GmbH — CC BY-SA 4.0

- 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.

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
  libgdbm_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.]

© grommunio GmbH — CC BY-SA 4.0

```
(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/syste
```

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.

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).

© grommunio GmbH — CC BY-SA 4.0

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
```

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.]

© grommunio GmbH — CC BY-SA 4.0

```
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: python38-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.servi
Created symlink /etc/systemd/system/multi-user.target.wants/mariadb.service → /usr/lib/sys
```

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-MariaDB 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 'fre
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

© grommunio GmbH — CC BY-SA 4.0

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.

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
  libjsoncpp25 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`.

© grommunio GmbH — CC BY-SA 4.0

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

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.]

© grommunio GmbH — CC BY-SA 4.0

```
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
```

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
```

© grommunio GmbH — CC BY-SA 4.0

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, set the listen addresses in `/etc/gromox/gromox.cfg` (to move gromox-http off nginx's 80/443):

```
http_listen = [::]:10080
http_listen_tls = [::]:10443
```

and the TLS certificate paths in `/etc/gromox/http.cfg`:

```
http_support_tls=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:

© grommunio GmbH — CC BY-SA 4.0

```
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: /
```

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
```

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_tls is set here. In `/etc/gromox/imap.cfg`, declare:

```
imap_support_tls=true
imap_certificate_path=zzz.pem
imap_private_key_path=zzz.key
imap_force_tls=true
```

In `/etc/gromox/pop3.cfg`:

© grommunio GmbH — CC BY-SA 4.0

```
pop3_support_tls=true
pop3_certificate_path=zzz.pem
pop3_private_key_path=zzz.key
pop3_force_tls=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
> A001 CAPABILITY
< * CAPABILITY IMAP4rev1 XLIST SPECIAL-USE UNSELECT UIDPLUS IDLE AUTH=LOGIN STARTTLS
< A001 OK CAPABILITY completed
…
```

Expected output for POP3:

© grommunio GmbH — CC BY-SA 4.0

```
*   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
```

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

© grommunio GmbH — CC BY-SA 4.0

`/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
…
< 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!
```

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 GmbH — CC BY-SA 4.0

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
```

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
```

© grommunio GmbH — CC BY-SA 4.0

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
```

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-s
```

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.

8443/api/v1/login

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

© grommunio GmbH — CC BY-SA 4.0

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

Expected output:

```
{"grommunioAuthJwt":"eyJ0…"}
```

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.

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.

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/200
<Response xmlns=…
```

© grommunio GmbH — CC BY-SA 4.0

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

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/
```

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*

© grommunio GmbH — CC BY-SA 4.0

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
```

Postfix

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

```
lda_listen = [::]: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'
```

© grommunio GmbH — CC BY-SA 4.0

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
```

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
```

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.

© grommunio GmbH — CC BY-SA 4.0

Container Installation

Besides the grommunio Appliance and a manual installation, grommunio can be run in containers. The official, authoritative project for this is **grommunio/gromox-container** — a **Docker Compose** deployment that packages the groupware into a small number of containers managed by **supervisord**, based on **openSUSE Leap 16.0** with **MariaDB 11**.

> **Authoritative source**
>
> This chapter summarizes the deployment; the container project itself is the source of truth and ships sane example configuration. Always review the project's README and adjust the configuration — especially **all default passwords** — before production use.

Architecture

The stack is split into a core container plus optional add-on containers, each backed by its own MariaDB database, all on a private Docker network:

- **gromox-core** — the main container running all core groupware services (nginx, Postfix, the `gromox-*` daemons, Admin API, antispam, Redis, …) under supervisord.
- **gromox-archive** *(optional)* — e-mail archiving with full-text search.
- **gromox-office** *(optional)* — document editing and file sync.
- **gromox-db / chat-db / files-db / office-db / archive-db** — MariaDB databases for the respective components.

All service containers run **unprivileged** — no `privileged: true`, no `SYS_ADMIN`, no special runtimes. Internal ports are remapped above 1024 and the host ports are mapped back to the standard ones (see Port mapping).

Quick start

You need a host with **Docker** and the **Docker Compose** plugin, a fully qualified domain name (FQDN) pointing at the host, and the standard mail/web ports free.

1. **Get the project**

```
git clone https://github.com/grommunio/gromox-container.git
cd gromox-container
```

© grommunio GmbH — CC BY-SA 4.0

2. **Configure the environment** — copy the example and edit it with your domain, passwords and feature flags:

```
cp var.env.example var.env
$EDITOR var.env
```

   At minimum set `FQDN`, `DOMAIN` and `ADMIN_PASS` (see Configuration).

3. **Prepare the bind-mounted volumes** (or run `./pre-launch.sh`, which does this for you):

```
mkdir -p variables_data gromox_letsencrypt
cp var.env variables_data/var.env
```

4. **Start the stack:**

```
docker compose up -d
```

   The first start takes several minutes while each container initializes its databases, generates certificates and configures its services.

Accessing the services

Once the stack is up, sign in to the Admin UI with the user `admin` and the password from `ADMIN_PASS`:

Service	URL
Webmail (grommunio Web)	`https://<FQDN>`
Admin UI	`https://<FQDN>:8443`
Exchange ActiveSync	`https://<FQDN>/Microsoft-Server-ActiveSync`
CalDAV / CardDAV	`https://<FQDN>/dav`
Files *(if enabled)*	`https://<FQDN>/files`
Archive *(if enabled)*	`https://<FQDN>/archive`

Services in the core container

`gromox-core` runs the full set of groupware services under supervisord:

© grommunio GmbH — CC BY-SA 4.0

Service	Description
`nginx`	Reverse proxy (web, admin, sync, DAV)
`postfix`	Mail transport agent
`gromox-http`	HTTP / MAPI / AutoDiscover
`gromox-imap` / `gromox-pop3`	IMAP / POP3 servers
`gromox-delivery` / `gromox-delivery-queue`	Local delivery + queue
`gromox-zcore`	PHP-MAPI core
`gromox-midb`	Message index database
`gromox-event` / `gromox-timer`	Event bus / scheduled tasks
`grommunio-admin-api`	Admin REST API (uWSGI)
`grommunio-antispam`	Spam filtering (rspamd)
`grommunio-chat` *(optional)*	Messaging
`php-fpm`	PHP FastCGI
`redis`	Cache / session store
`saslauthd`	SMTP authentication
`crond`	Scheduled tasks (certbot renewal, …)

The optional **gromox-archive** container adds `grommunio-archive`, `grommunio-archive-smtp` and a Sphinx `searchd`; **gromox-office** adds the document-service and converter daemons plus RabbitMQ.

Configuration

All configuration is done through environment variables in `var.env`.

Required

Variable	Description	Example
`FQDN`	Server fully qualified domain name	`mail.example.com`
`DOMAIN`	Mail domain	`example.com`
`ADMIN_PASS`	grommunio admin password	`SecurePassword123`
`MYSQL_HOST`	Database host for the core	`gromox-db`
`MYSQL_USER` / `MYSQL_PASS` / `MYSQL_DB`	Core DB credentials / name	`grommunio`

© grommunio GmbH — CC BY-SA 4.0

TLS certificates

Variable	Description	Default
`SSL_INSTALL_TYPE`	`0` = self-signed, `2` = Let's Encrypt	`0`
`SSL_BUNDLE` / `SSL_KEY`	Paths to your own certificate + key	
`SSL_EMAIL`	E-mail for Let's Encrypt notifications	`admin@DOMAIN`

With Let's Encrypt (`SSL_INSTALL_TYPE=2`), certbot runs on first start and a cron job renews every 12 hours; port 80 must be reachable from the internet. If Let's Encrypt provisioning fails, the stack falls back to a self-signed certificate so services still come up.

Optional components

Each add-on is toggled with an `ENABLE_*` flag and configured with its own database credentials:

Component	Enable flag	Extra variables
Chat	*(starts if `CHAT_CONFIG` exists)*	`CHAT_MYSQL_*`, `CHAT_ADMIN_PASS`, `CHAT_CONFIG`
Files	`ENABLE_FILES=true`	`FILES_MYSQL_*`, `FILES_ADMIN_PASS`
Office	`ENABLE_OFFICE=true`	`OFFICE_HOST`, `OFFICE_MYSQL_*`
Archive	`ENABLE_ARCHIVE=true`	`ARCHIVE_HOST`, `ARCHIVE_MYSQL_*`, `ARCHIVE_INDEXER_*_INTERVAL`
Keycloak SSO	`ENABLE_KEYCLOAK=true`	`KEYCLOAK_REALM`, `KEYCLOAK_URL`, `KEYCLOAK_CLIENT_ID`, `KEYCLOAK_CLIENT_SECRET`

Advanced

Variable	Description	Default
`RELAYHOST`	SMTP relay host for outbound mail	*(empty)*
`ORGANIZATION`	Organization name for AutoDiscover	
`TIMEZONE`	Container timezone	`Europe/Vienna`
`FORCE_RECONFIG`	Re-run the setup scripts on next restart	`false`
`CLEAR_DBS`	Drop and recreate databases on startup (needs `MYSQL_ROOT_PASS`)	`false`
`X500`	X.500 organization identifier (auto-generated if empty)	

© grommunio GmbH — CC BY-SA 4.0

> ⚠**Change the defaults**
>
> The example database passwords in `docker-compose.yml` and `var.env` are for getting started only. Change **every** password (database, admin, chat/files/ office/archive) before exposing the deployment.

Port mapping

Internal ports are all above 1024 (so containers need no privileges); the host maps them back to the standard ports:

Host port	Container port	Service
25	2525	SMTP (Postfix inbound)
80	8080	HTTP (Let's Encrypt challenge, HTTP→HTTPS redirect)
443	8443	HTTPS (webmail, sync, DAV, files, office, archive)
465	2465	SMTPS (implicit TLS)
587	2587	Submission (authenticated)
993	2993	IMAPS
143	2143	IMAP (STARTTLS)
995	2995	POP3S
110	2110	POP3 (STARTTLS)
8443	9443	Admin Web UI

Persistent data (volumes)

Named volumes hold all state, so containers can be rebuilt without data loss. The most important ones to **back up**:

Volume	Mount point	Purpose
`gromox`	`/var/lib/gromox`	Mailbox data
`gromox_config`	`/etc/gromox`	Gromox configuration
`cert_data`	`/etc/grommunio-common/ssl`	TLS certificates (shared)
`gromox_mysql_data`	MariaDB data	Core database
`*_mysql_data`	MariaDB data	Chat / Files / Office / Archive databases
`gromox_letsencrypt`	`/etc/letsencrypt`	Let's Encrypt state

© grommunio GmbH — CC BY-SA 4.0

Operations

The services inside a container are managed with `supervisorctl`:

```
# Status of all services in the core container
docker exec gromox-core supervisorctl status

# Restart a single service
docker exec gromox-core supervisorctl restart nginx

# Per-service logs
docker exec gromox-core tail -f /var/log/supervisor-nginx.log
docker exec gromox-core cat /var/log/grommunio-setup.log   # initial setup log
```

Reconfiguring after var.env changes

The setup scripts run only once (guarded by marker files). To re-apply changes:

```
echo "FORCE_RECONFIG=true" >> variables_data/var.env
docker compose restart gromox-core
```

> ⚠ **Caution**
>
> Reconfiguration re-runs the full setup, which may drop and recreate the **optional** databases (chat, files, office, archive). The core gromox database is preserved.

Backup

Back up the database and data volumes, for example:

```
docker run --rm -v gromox_mysql_data:/data -v "$(pwd)/backup:/backup" \
  busybox tar czf /backup/gromox-db.tar.gz /data
docker run --rm -v gromox:/data -v "$(pwd)/backup:/backup" \
  busybox tar czf /backup/gromox-data.tar.gz /data
docker run --rm -v gromox_config:/data -v "$(pwd)/backup:/backup" \
  busybox tar czf /backup/gromox-config.tar.gz /data
```

Updating

© grommunio GmbH — CC BY-SA 4.0

```
git pull
docker compose build
docker compose up -d
```

> **Resetting wipes data**
>
> `docker compose down -v` removes the containers **and all volumes** — every mailbox and database is lost. Only use it for a deliberate fresh start.

Security

The container images apply current TLS hardening out of the box:

- **Postfix** (25/587): the legacy and **DH-family** ciphers are excluded (`aNULL, eNULL, EXP, MD5, RC4, DES, 3DES, DHE, EDH, kDHE, kEDH, ADH`) while the forward-secret **ECDHE** suites are kept (`smtpd_tls_eecdh_grade = strong`); SSLv2/SSLv3 and TLS 1.0/1.1 are disabled. Excluding only the DH family mitigates CVE-2002-20001 (a DHE-only key-exchange denial of service) without sacrificing forward secrecy.
- **Gromox** http / imap / pop3: `tls_min_proto = tls1.2` — SSLv3 and TLS 1.0/1.1 are disabled, with TLS 1.2 as the floor. (993/995 are client-facing with no proxy in front, so a TLS 1.3 floor would lock out TLS-1.2-only mail clients such as older Thunderbird, mobile and Outlook builds.)
- All containers run **unprivileged**.

For production, additionally place the deployment behind a firewall, use Let's Encrypt or your own CA-signed certificates, and keep the images up to date.

Troubleshooting

```
docker compose ps              # container + database health
docker compose logs gromox-core
# a service stuck in FATAL — read its error log:
docker exec gromox-core cat /var/log/supervisor-<service>-err.log
```

Common causes: a missing `FQDN` / `DOMAIN` in `var.env` (Postfix fails), missing `OFFICE_MYSQL_*` (document service fails), or grommunio Files rejecting a self-signed certificate — use Let's Encrypt, or add the host to Files' trusted domains. See the project's README troubleshooting section for details.

© grommunio GmbH — CC BY-SA 4.0

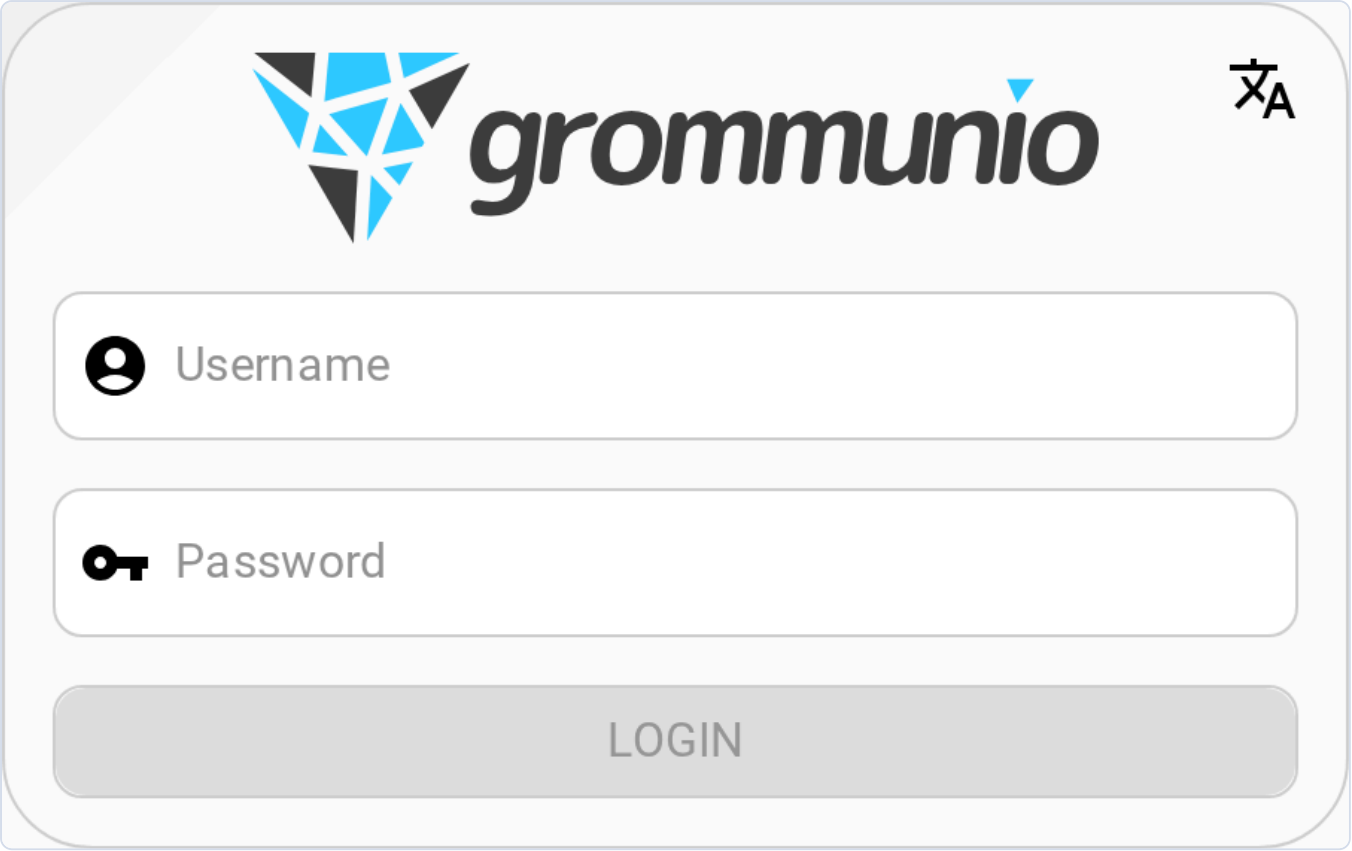
Administration

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

grommunio admin UI (AUI)

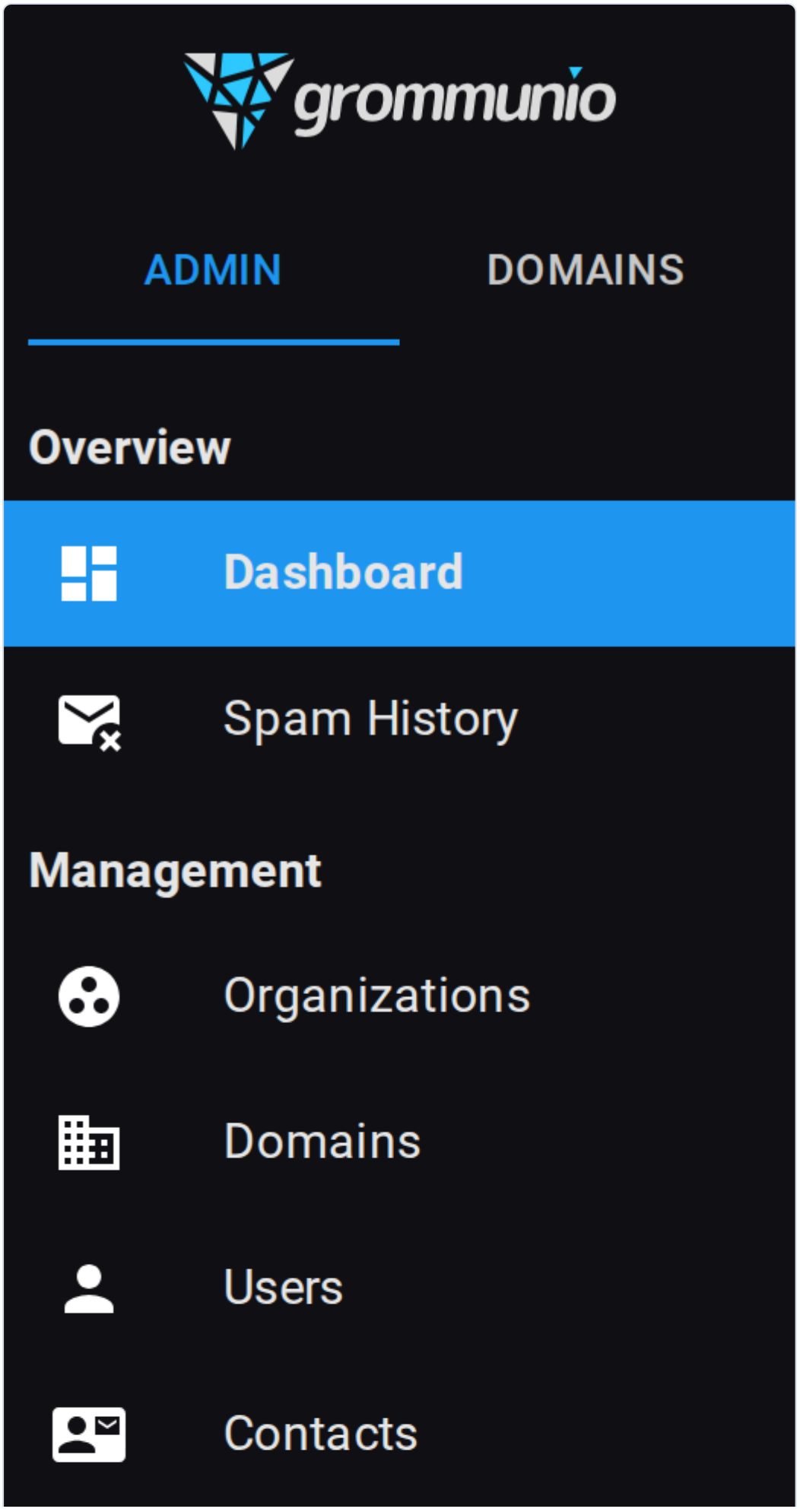
After successfully installing the grommunio Appliance, you can access the UI through your browser on port 8080 (8443 with https soon).

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

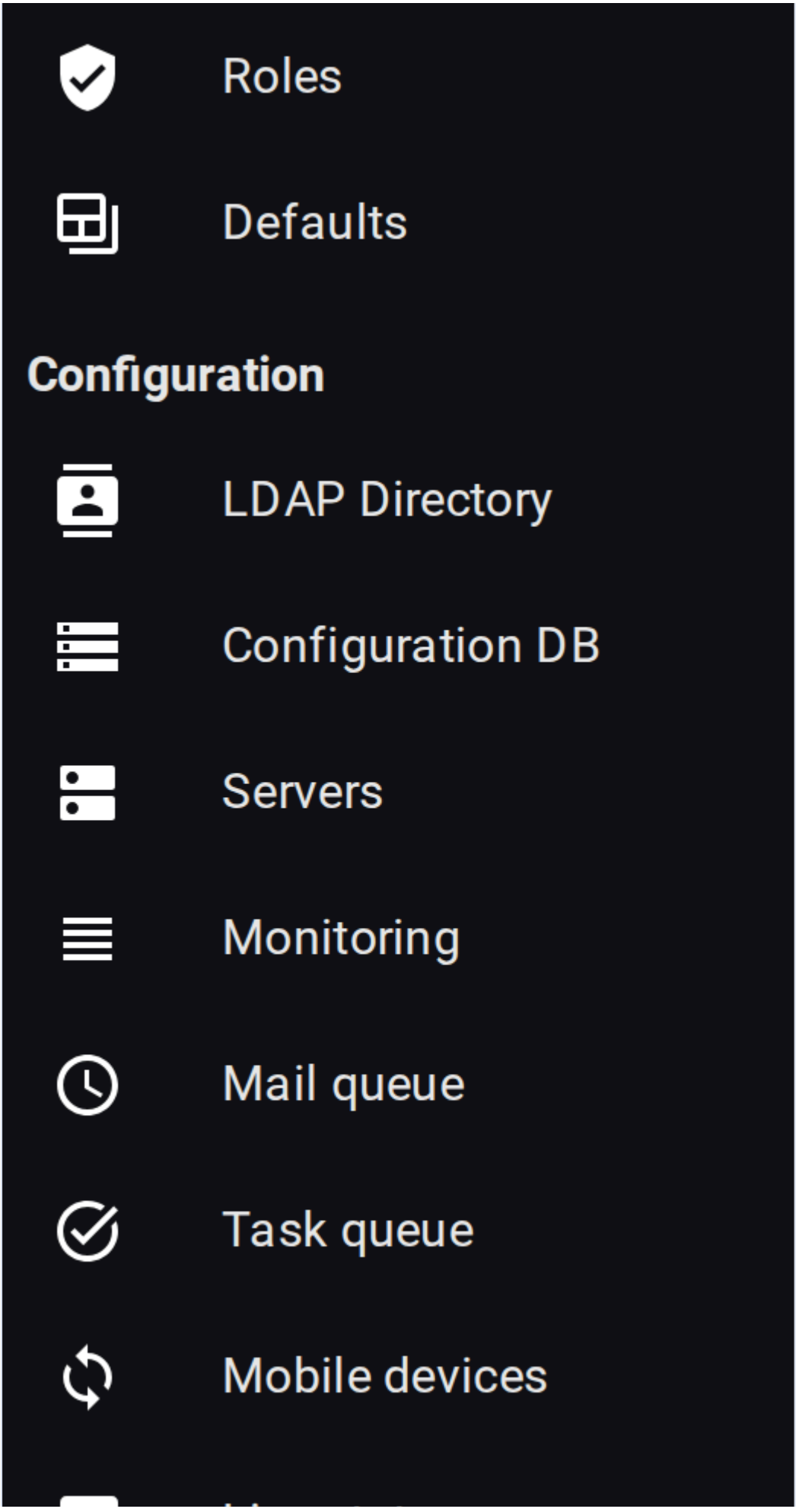


To navigate through the UI, simply use the drawer on the left side of the page.

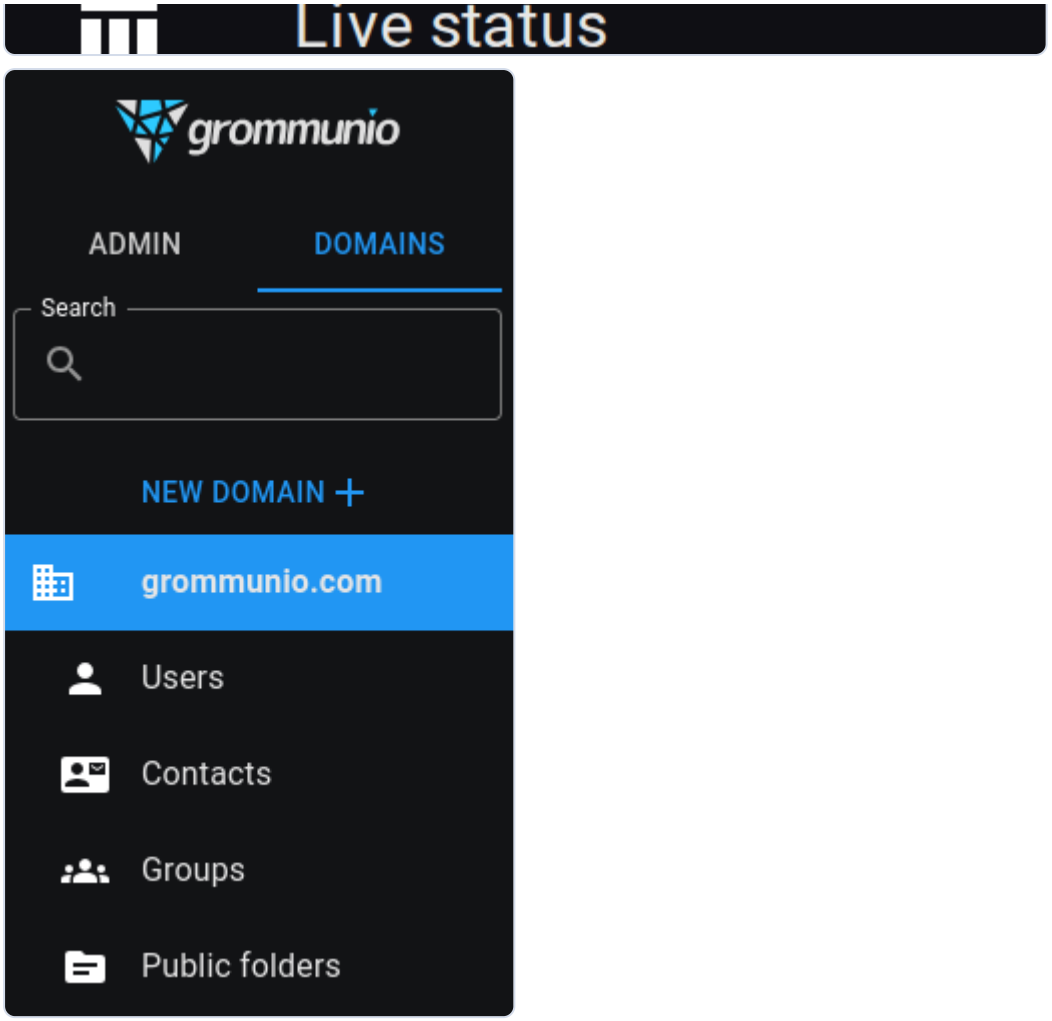
© grommunio GmbH — CC BY-SA 4.0



© grommunio GmbH — CC BY-SA 4.0



© grommunio GmbH — CC BY-SA 4.0



Dashboard

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

Antispam

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

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.

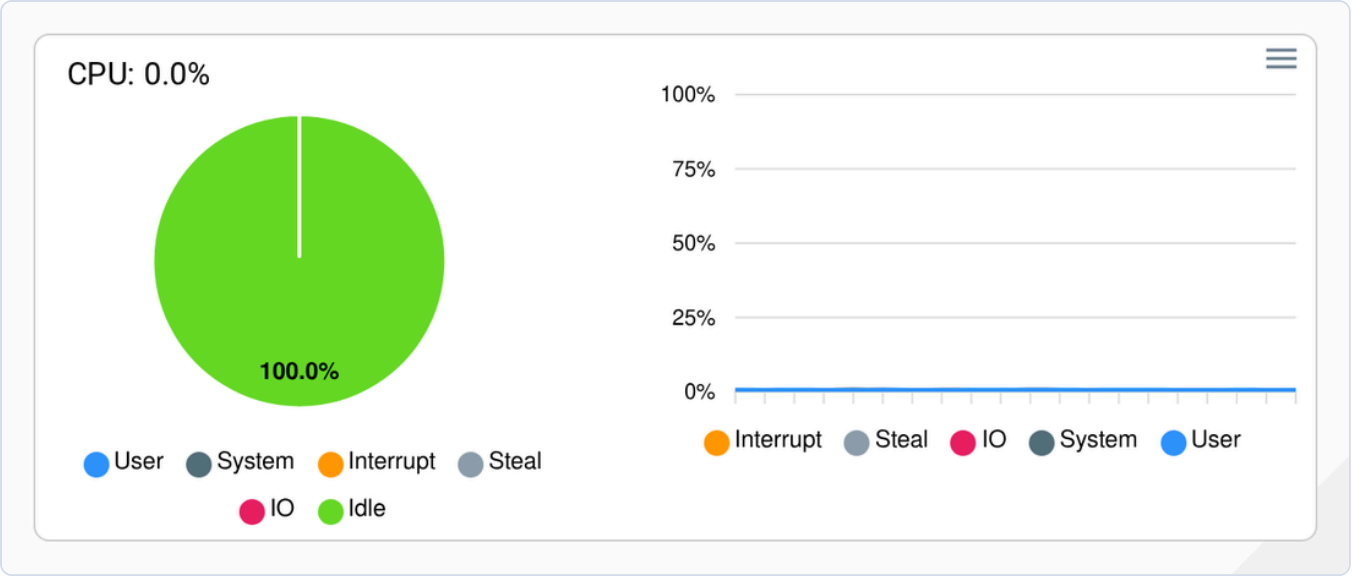
© grommunio GmbH — CC BY-SA 4.0

Service	State | Autostart	Actions
grommunio-antispam	active enabled	
gromox-delivery	active enabled	
gromox-event	active enabled	
gromox-http	inactive enabled	
gromox-imap	active enabled	
gromox-midb	active enabled	
gromox-pop3	active enabled	
gromox-delivery-queue	active enabled	
gromox-timer	active enabled	
gromox-zcore	inactive enabled	
nginx	active enabled	
php-fpm	active enabled	
postfix	active enabled	
redis@grommunio	active enabled	

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

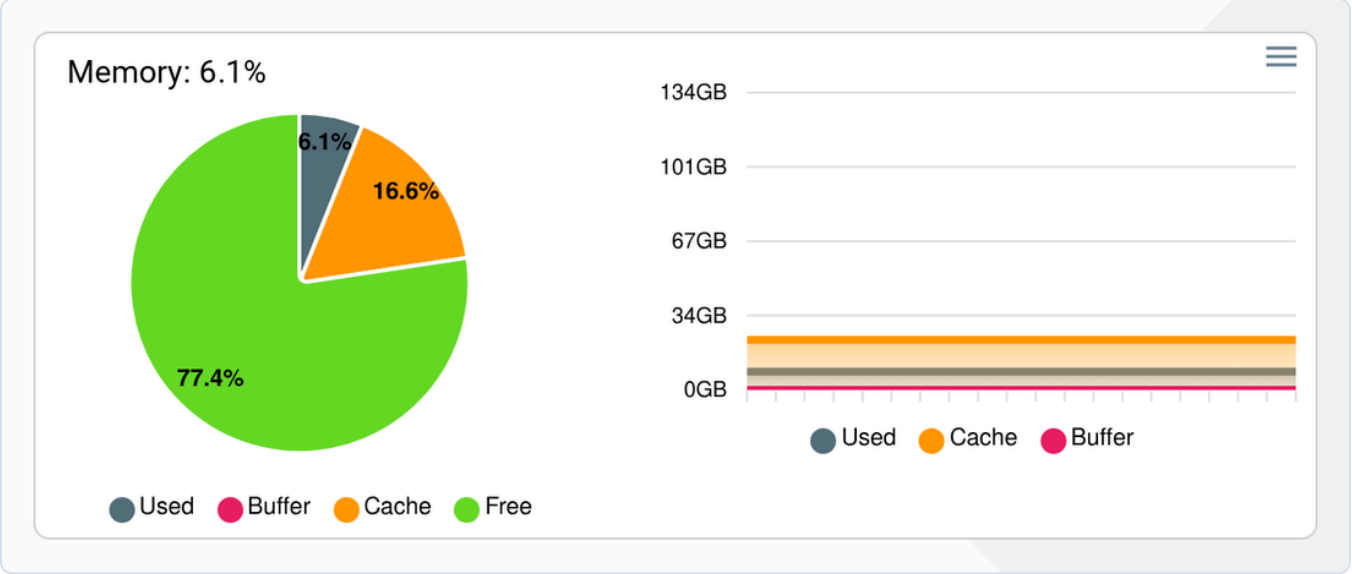
© grommunio GmbH — CC BY-SA 4.0

CPU



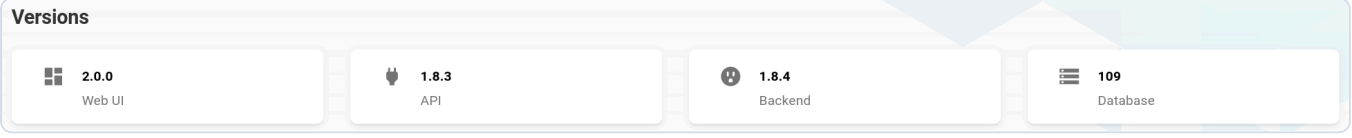
A live and history display of the CPU usage.

Memory



A live and history display of the memory usage.

Versions



A display of installed component versions.

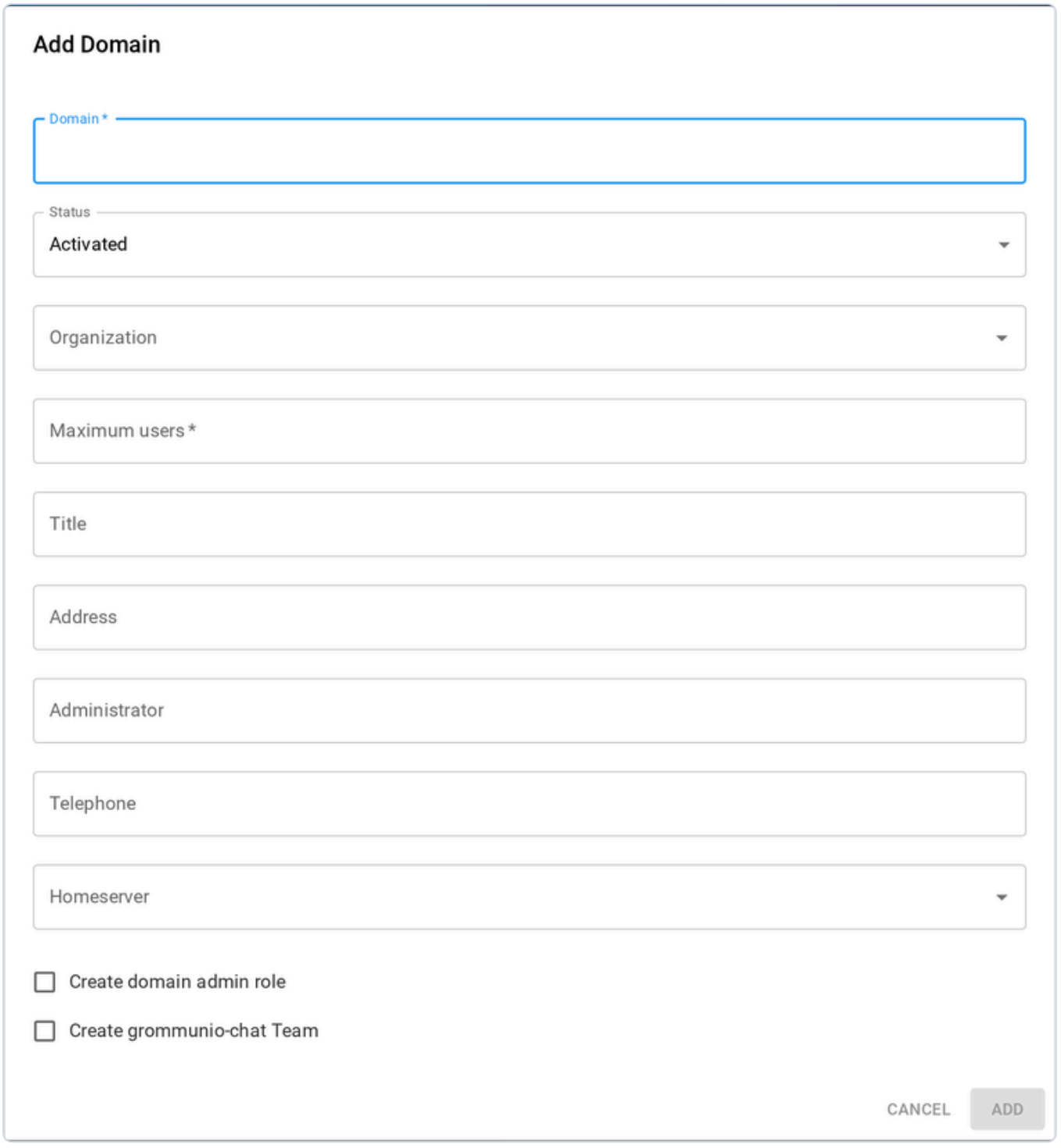
© grommunio GmbH — CC BY-SA 4.0

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*.

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:

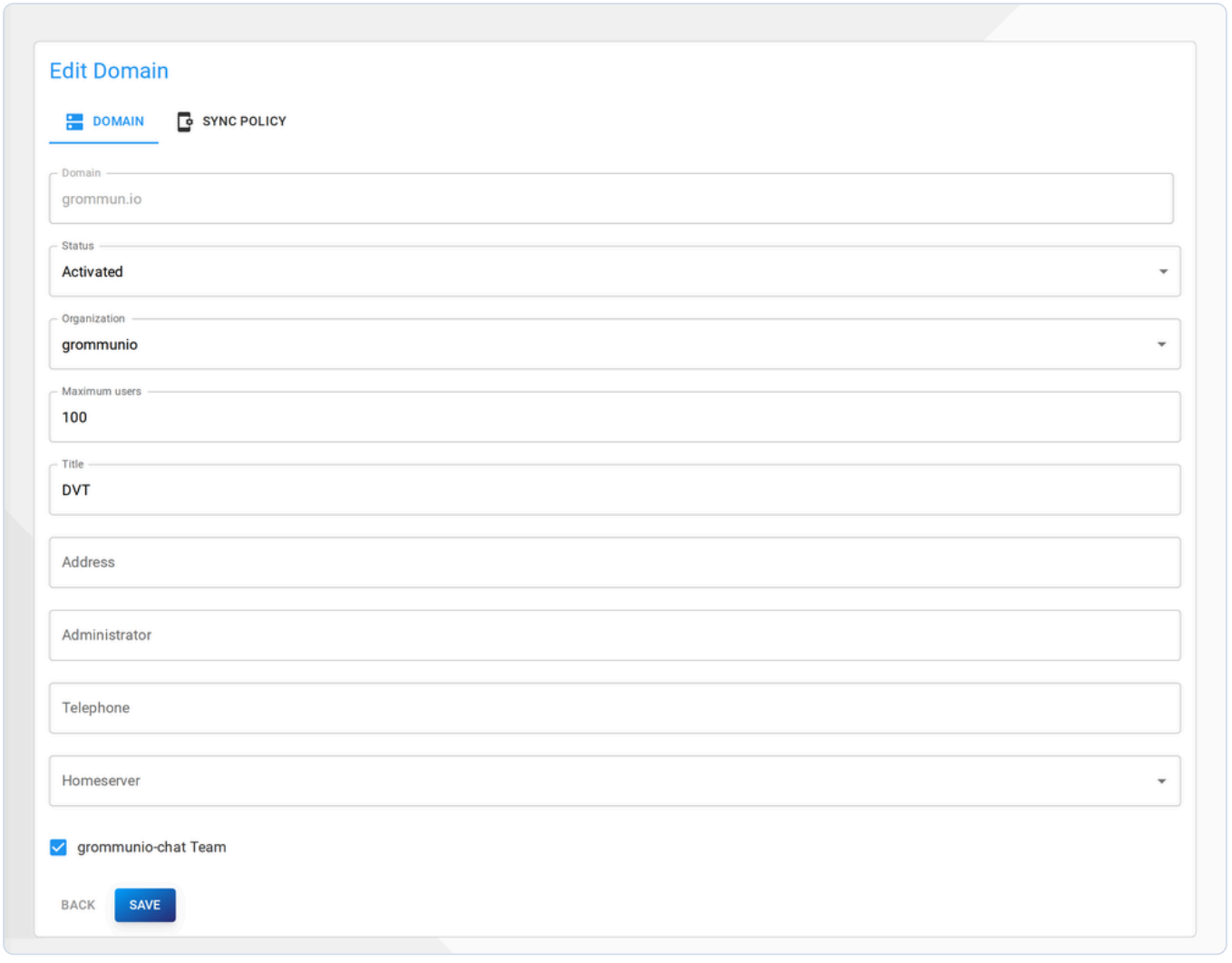
© grommunio GmbH — CC BY-SA 4.0

- 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.

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.

© grommunio GmbH — CC BY-SA 4.0

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.

Deleting a domain

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

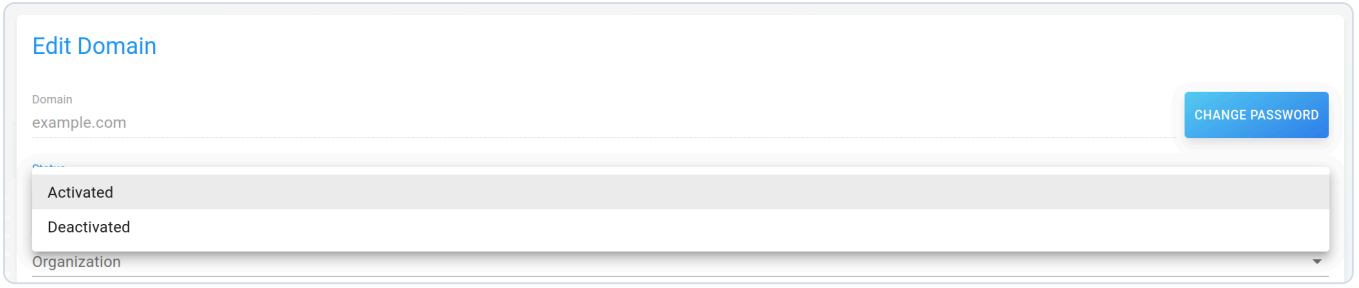
The following flags can be set:

- 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

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.



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).

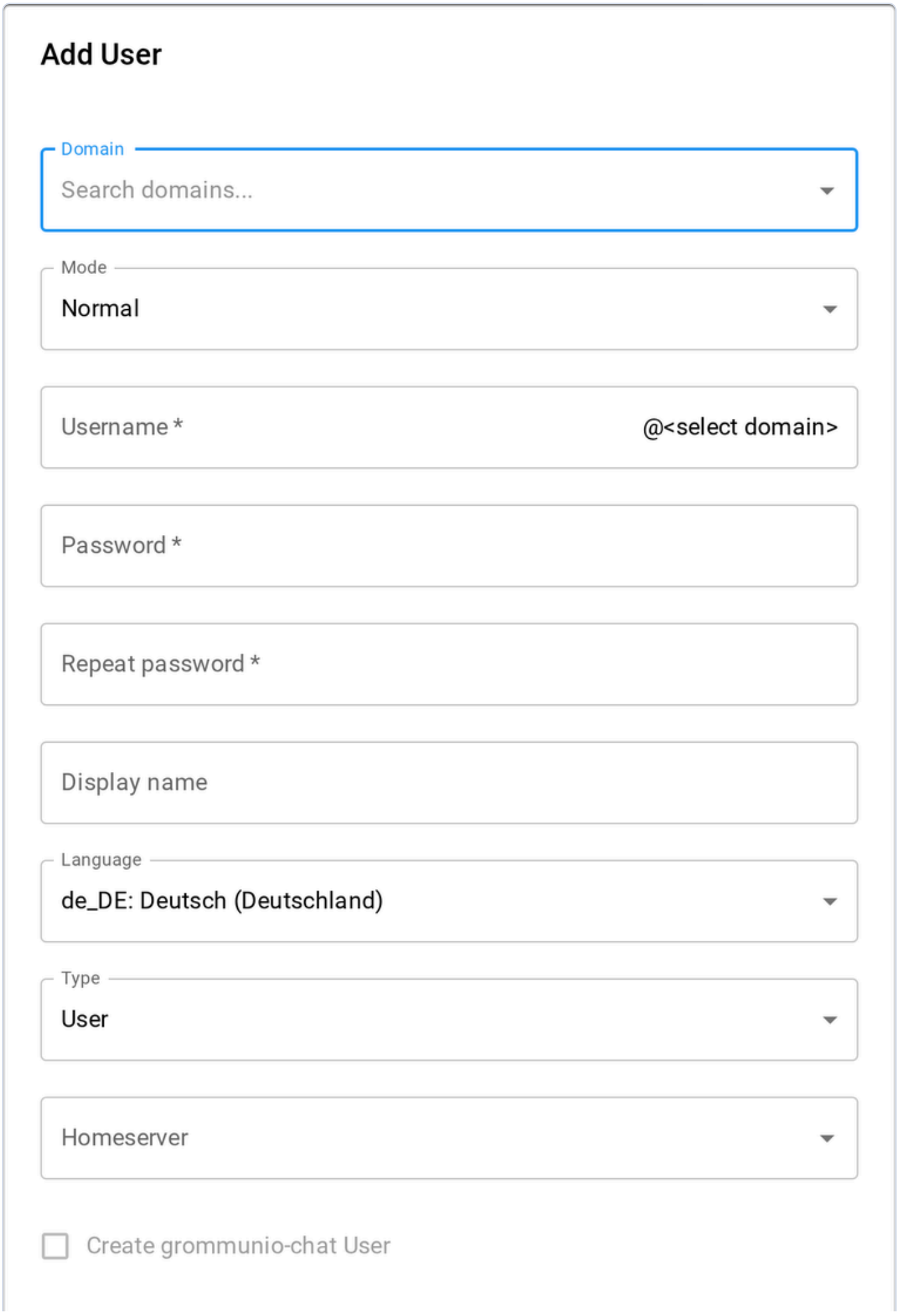
Click on a domain to expand available sub-pages and click on *Users*, which will redirect you to the list of users of this domain. If you just installed grommunio or added the domain, the list will be empty.

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

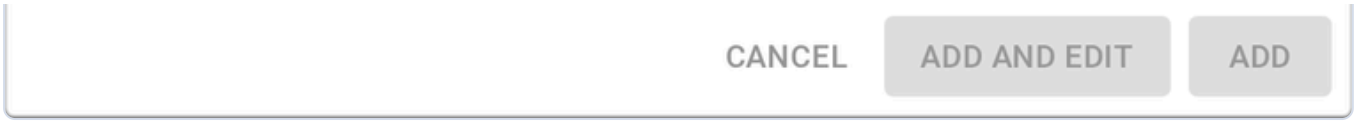
Adding a user

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

© grommunio GmbH — CC BY-SA 4.0



© grommunio GmbH — CC BY-SA 4.0



The following properties can be set:

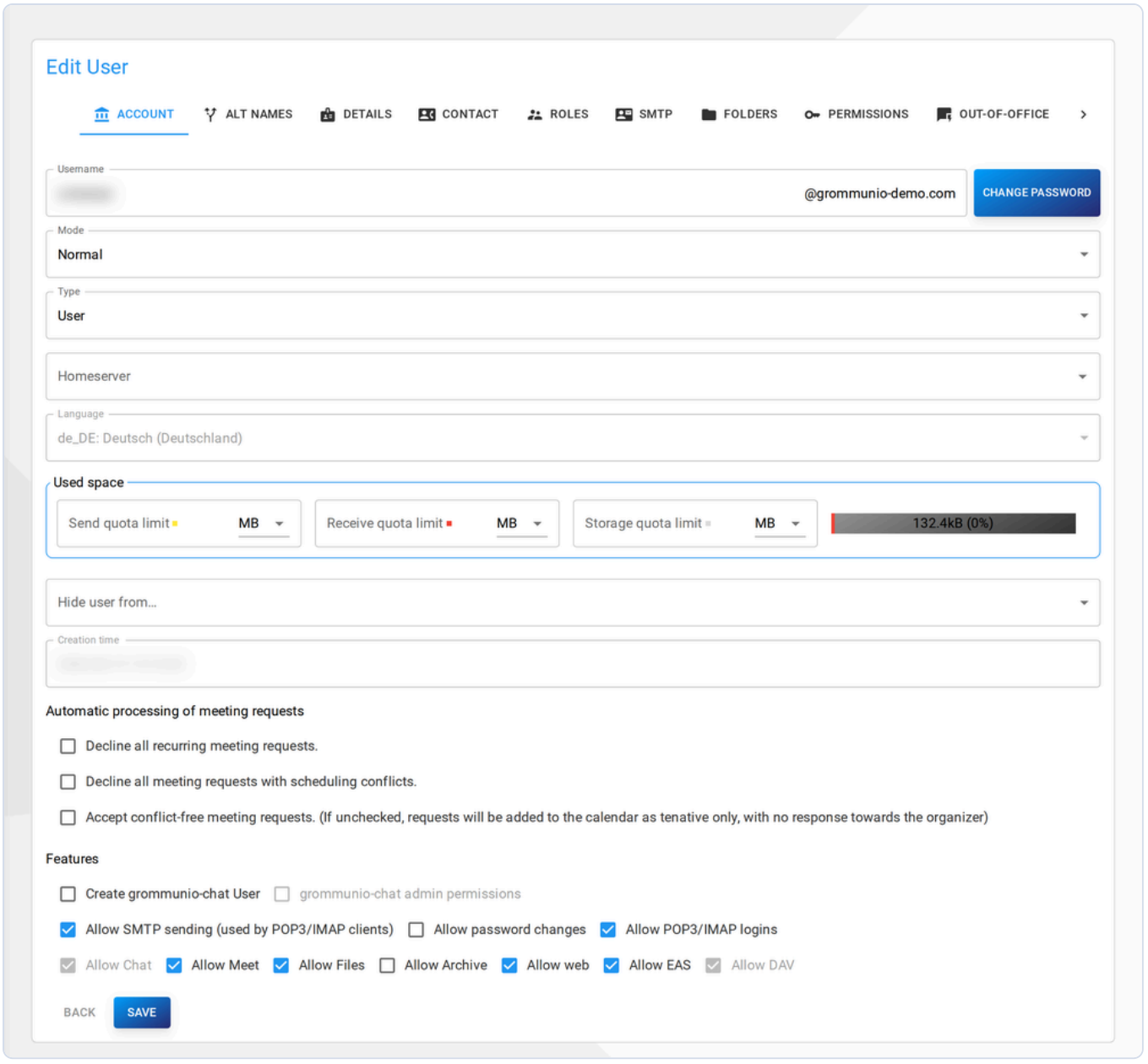
- Mode: Normal or shared user
- Username (required): Username of the user
- Password (required): Password of the user
- Display name: Name to be displayed for this user
- Storage quota limit: Storage limit of the user
- Type: Type of user
- Homeserver: The server on which the user's data is stored

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.

Editing a user

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

© grommunio GmbH — CC BY-SA 4.0



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)

Account

© grommunio GmbH — CC BY-SA 4.0

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.

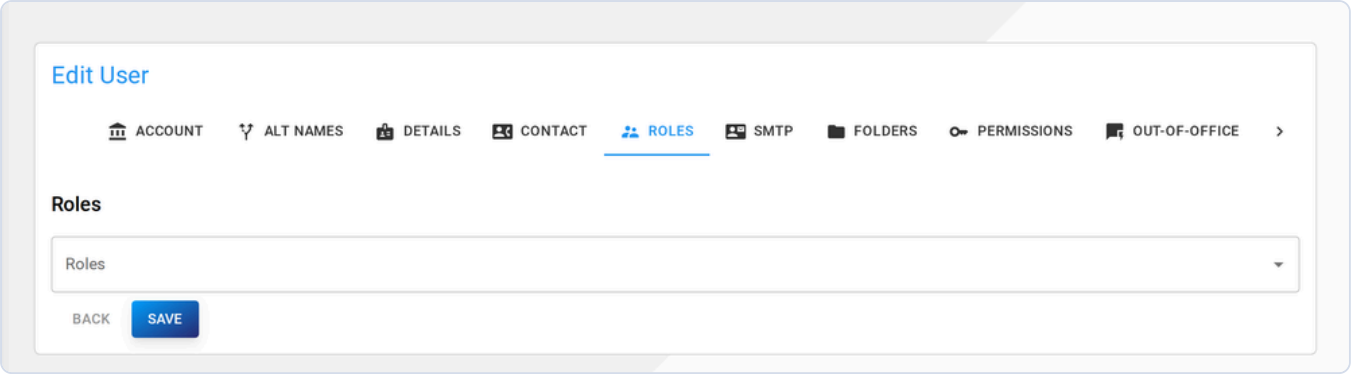
User & Contact

© grommunio GmbH — CC BY-SA 4.0

Common MAPI props. These are self-explanatory.

Roles

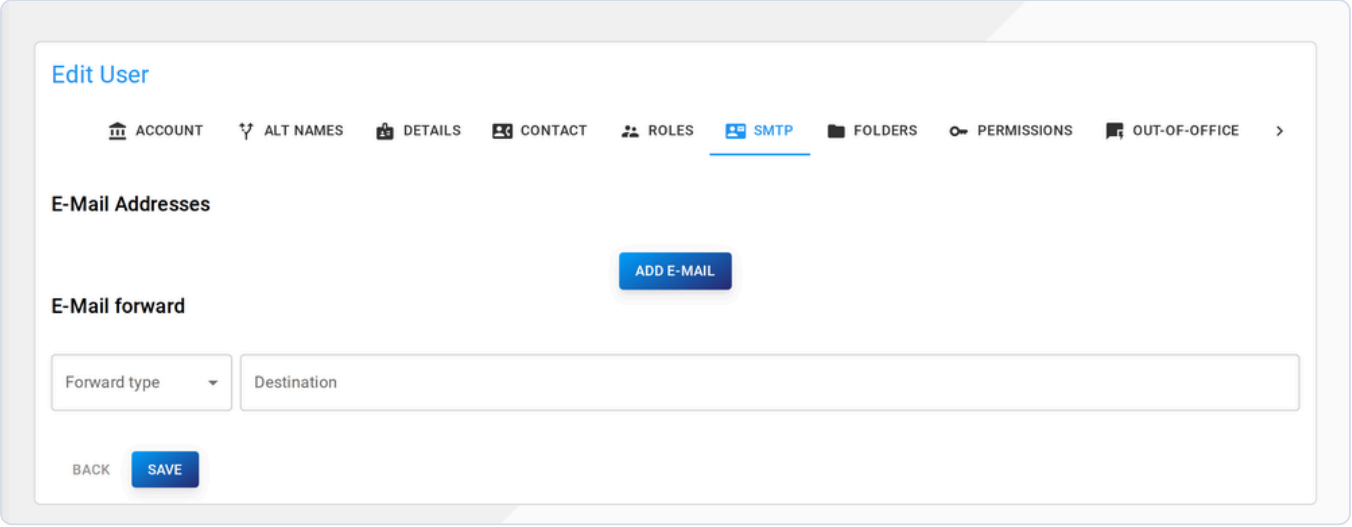
Roles of the user, which can be edited with the autocompleting textfield



SMTP

User aliases: The textfield(s) can be used to set aliases for the current user. Use the "Add E-Mail" button to add, or the trashcan icon to delete an alias.

E-Mail forward: This can be used to enforce a redirect or message cloning action irrespective of the Inbox Rules configured by a mailbox owner. The mail transfer agent used in your mail system must support this and must be configured accordingly to evaluate the SQL table where the forward info has been stored (see below).



Configuration fragments to implement CC mode (forward_type 0) for e.g. Postfix:

1. File `/etc/postfix/main.cf`: setting `recipient_bcc_maps = mysql:/etc/postfix/grommunio-bcc-forwards.cf`
2. File `grommunio-bcc-forwards.cf`: setting `user = ...`, `password = ...`, `hosts = localhost`, `dbname = grommunio`, `query = SELECT destination FROM forwards WHERE username='%s' AND forward_type = 0`

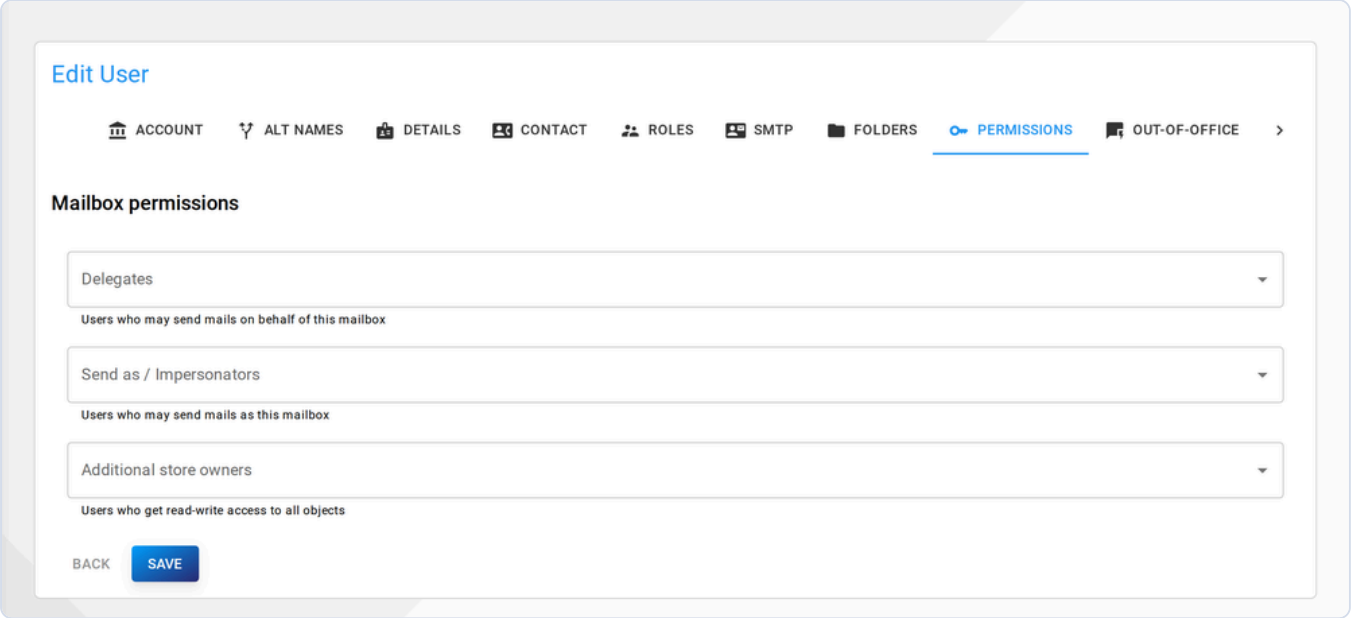
© grommunio GmbH — CC BY-SA 4.0

Redirect mode (forward_type 1) is left as an exercise to the administrator.

Permissions

This dialog allows giving other user identities certain permissions at the mailbox level.

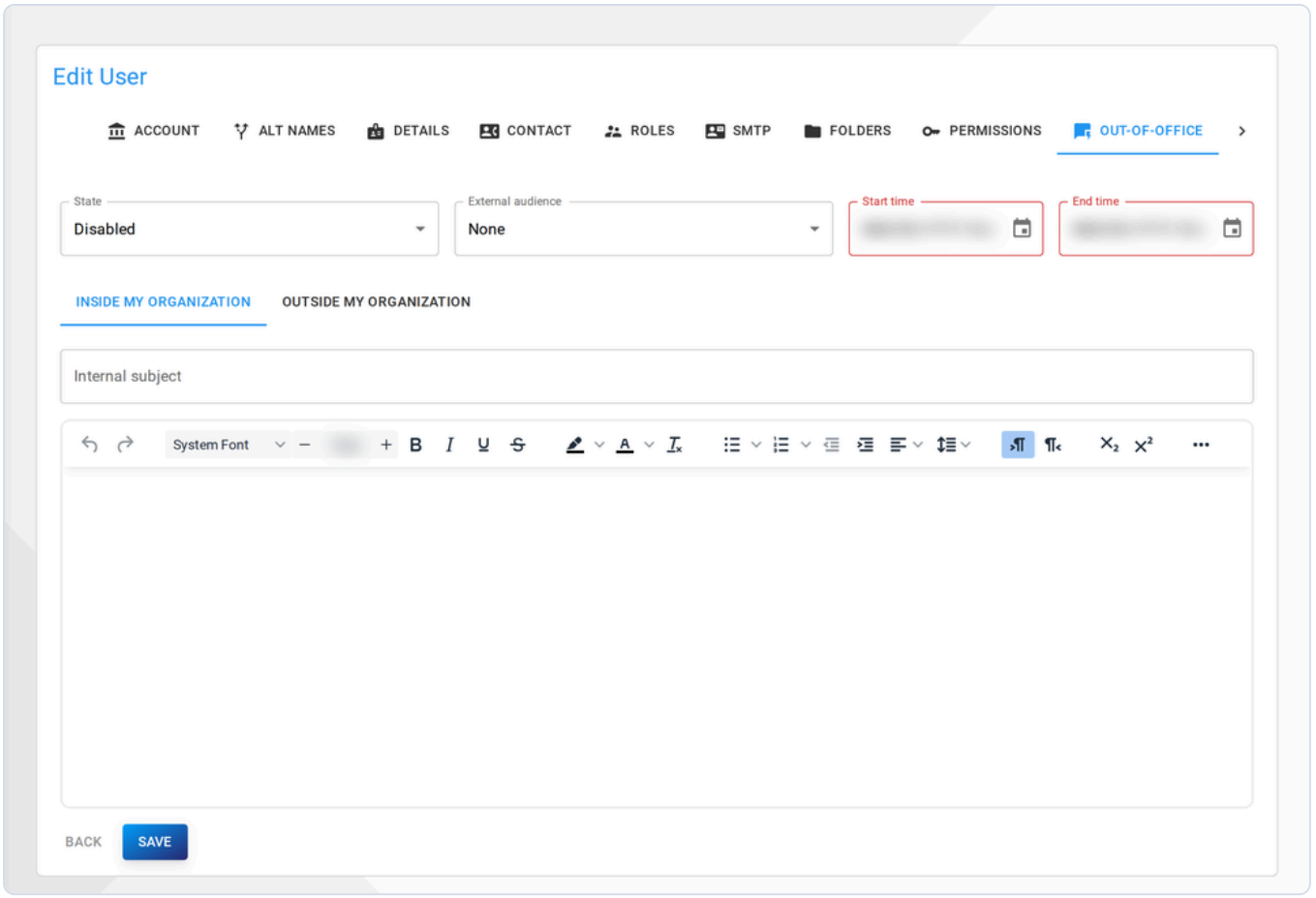
- Delegates: Users in the delegate list may exercise the "send on behalf" feature, i.e. send messages with a `From:` line containing the delegator's identity. *The use of delegation is recorded in messages.*
- Send As (also known as Impersonation): Users in the send-as list may send messages with a `From:` line containing the delegator's identity. It is similar to delegation, but the use of delegation is *not* recorded in messages. Send-As overrides and masks Send-On-Behalf, because SA/SOB permission is only known server-side and MAPI clients have no way to choose between the two.
- “Full permissions”: Users in this list will be treated like the mailbox owner and not be subject to permission checks when reading or writing folder or message objects.



OOF

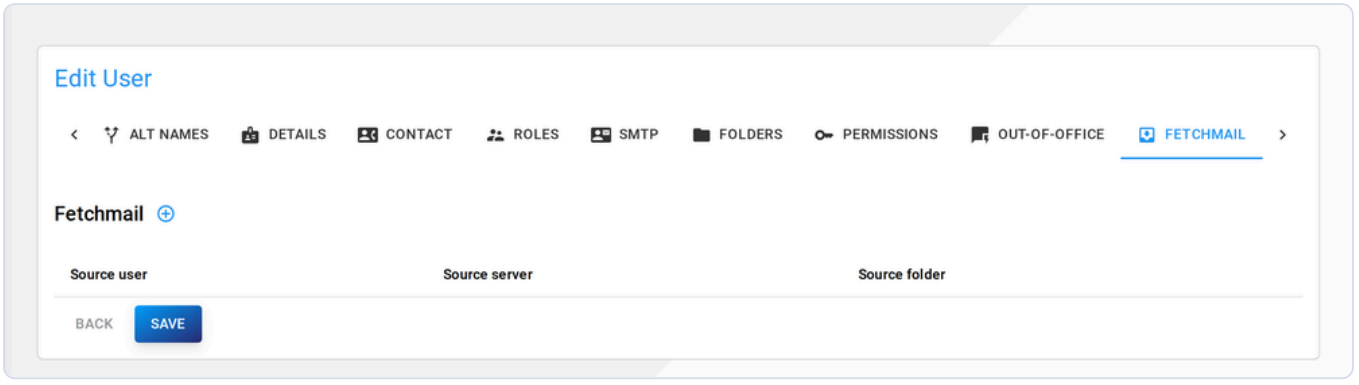
Out of office settings (auto-reply messages).

© grommunio GmbH — CC BY-SA 4.0



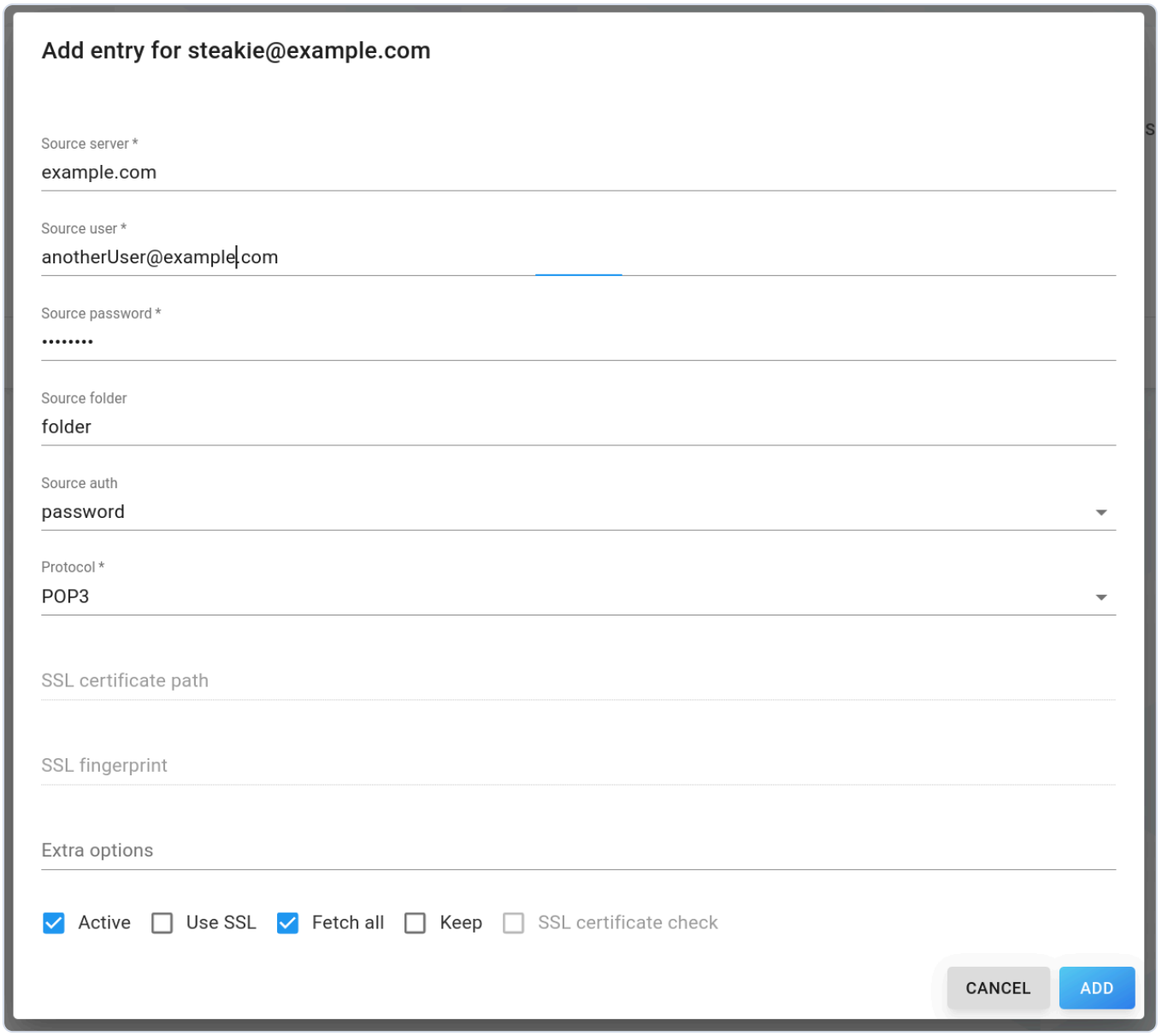
Fetchmail

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:

© grommunio GmbH — CC BY-SA 4.0



- 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

© grommunio GmbH — CC BY-SA 4.0

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

> **⚠Caution**
>
> Any changes will only be saved after clicking *Save* on the bottom of the page.

Mobile Devices

Synchronized mobile devices of this user

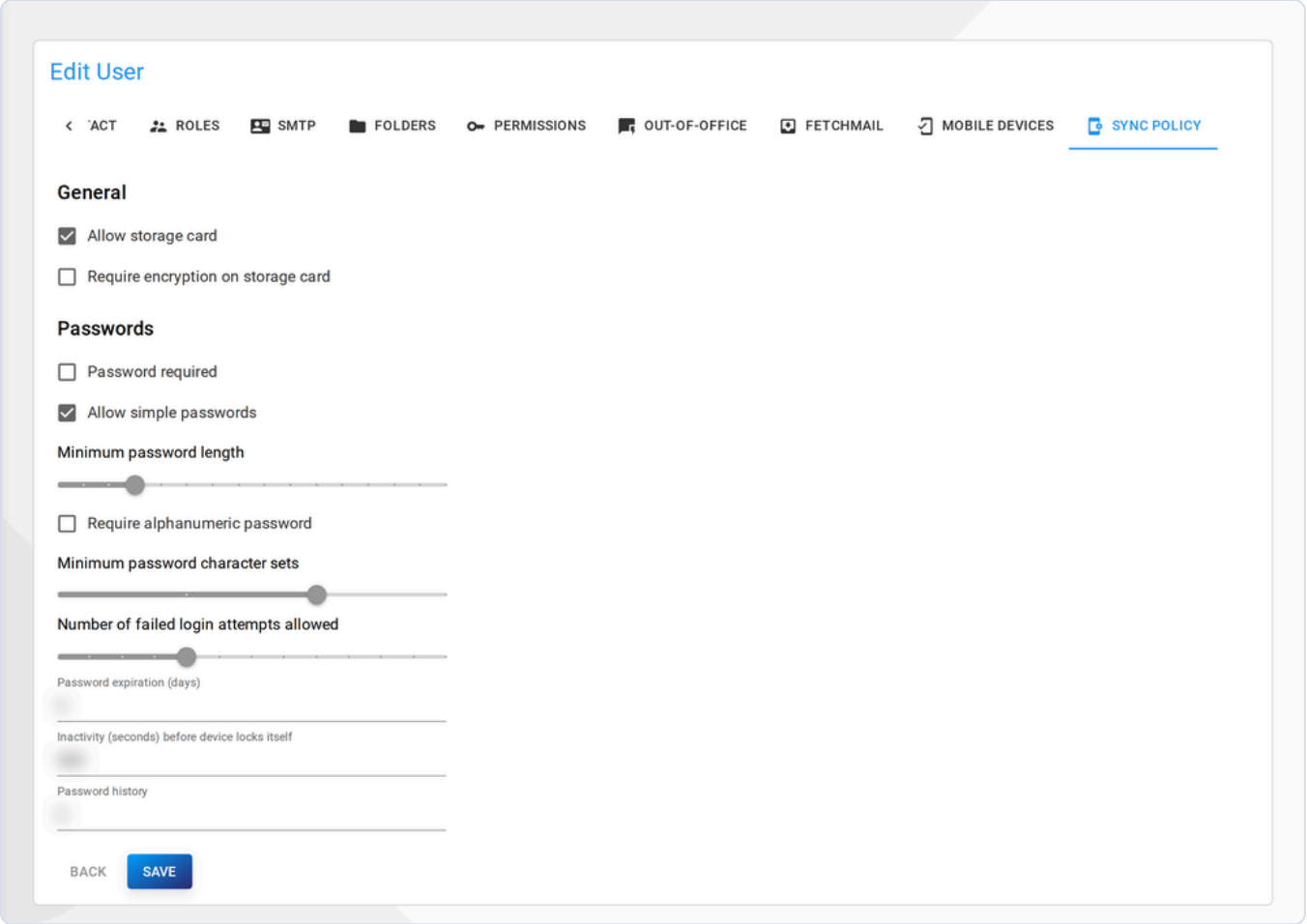
Actions:

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

Sync policy

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

© grommunio GmbH — CC BY-SA 4.0



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

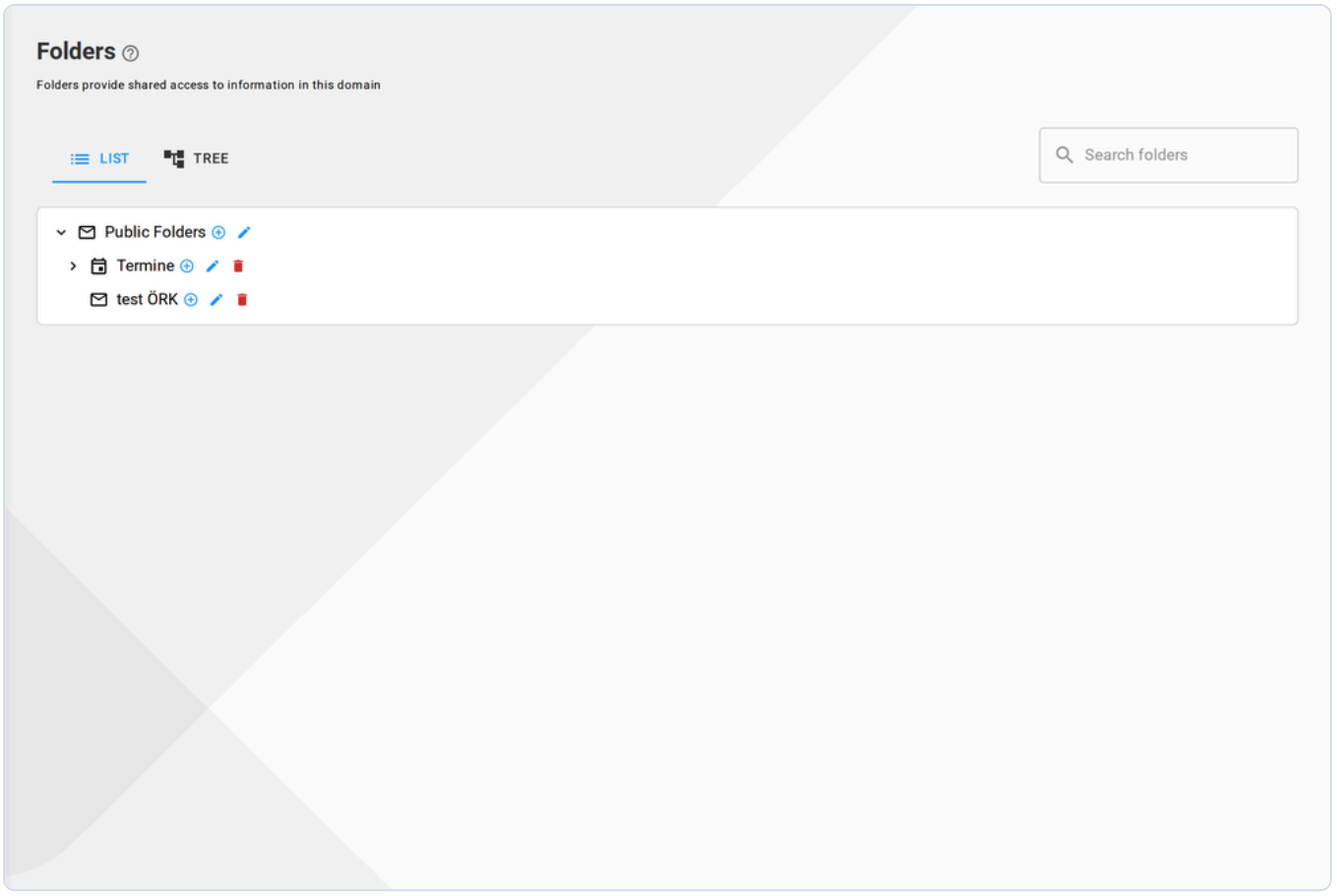
Click *Confirm* to confirm or *Cancel* to cancel.

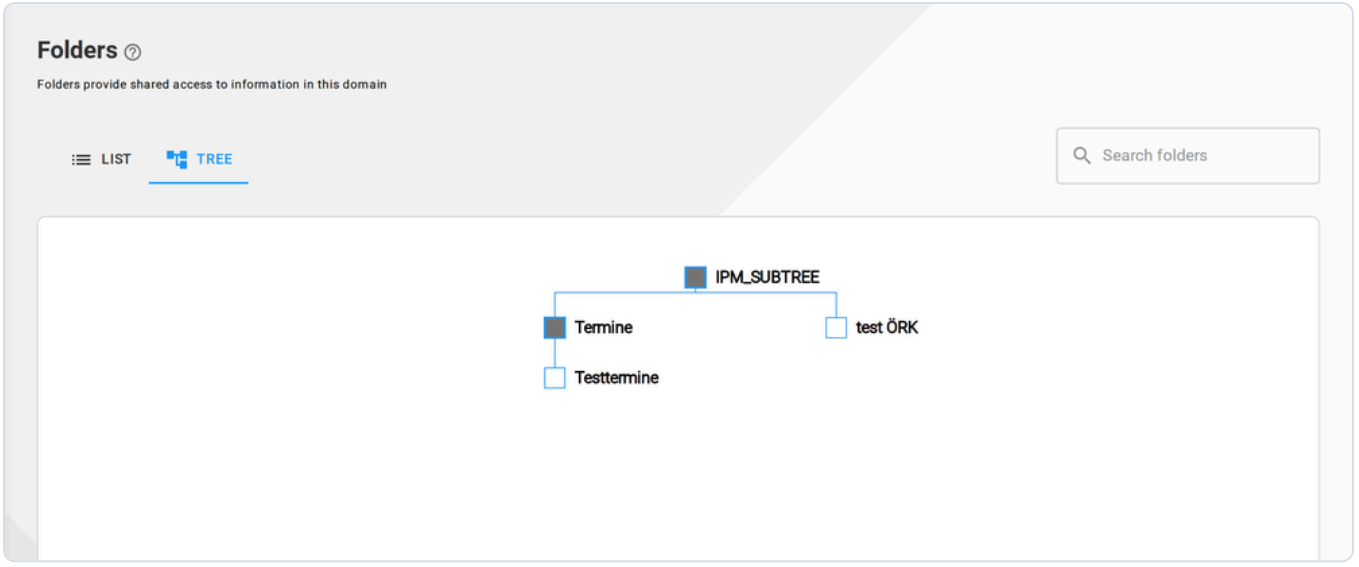
Public folders

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.

© grommunio GmbH — CC BY-SA 4.0

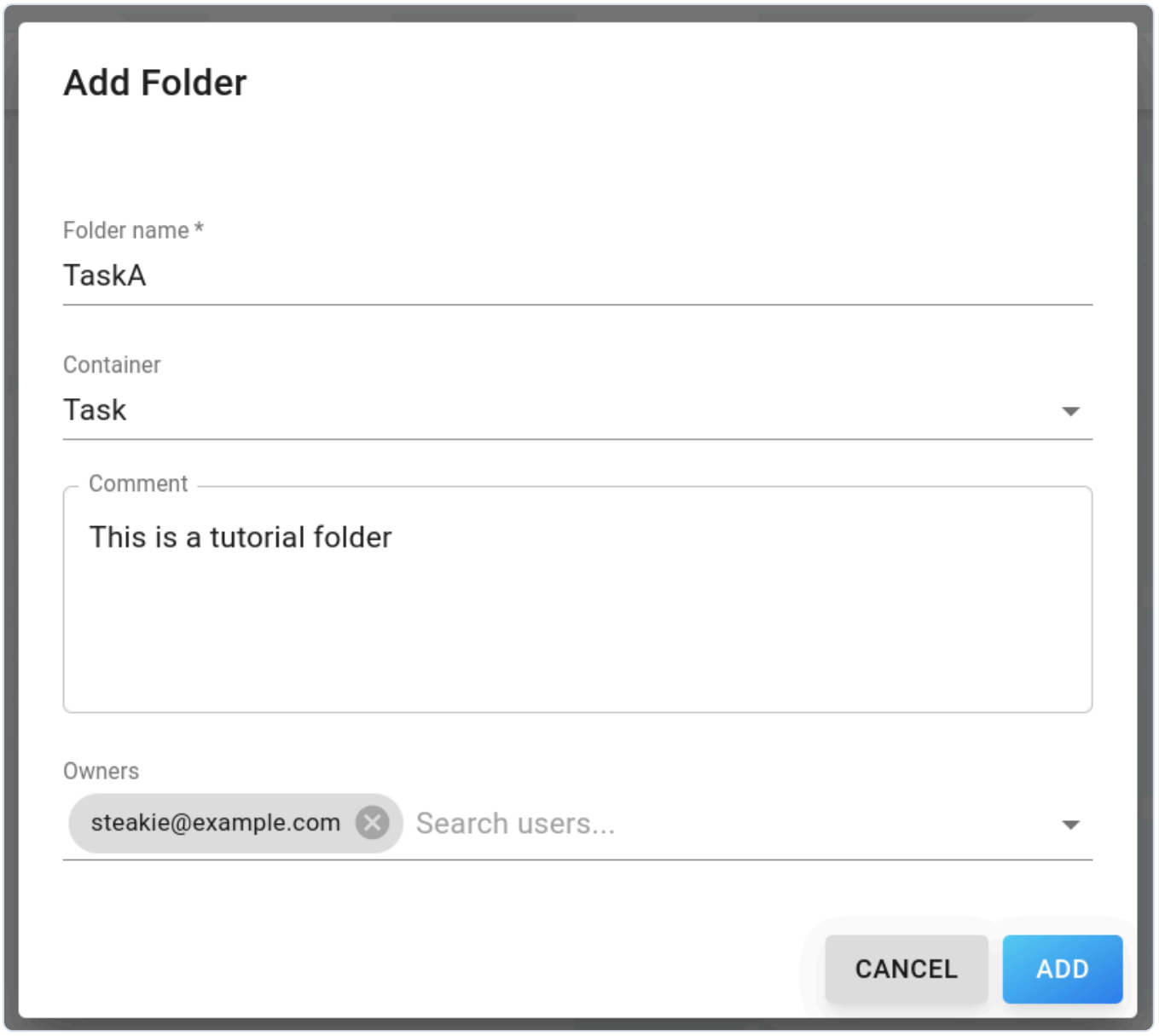




Adding a folder

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)*.

© grommunio GmbH — CC BY-SA 4.0



The following properties can be set:

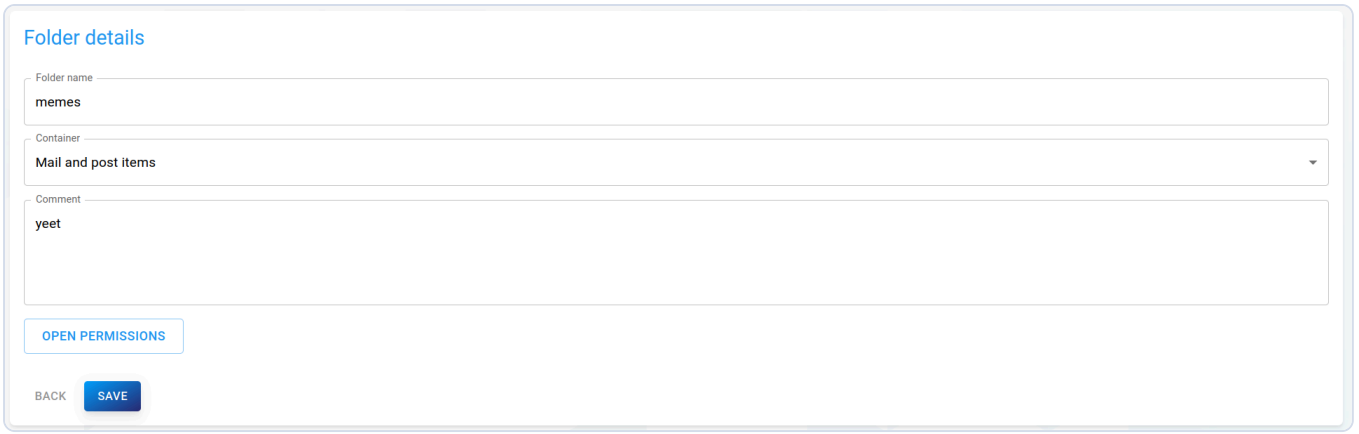
- Folder name (required): Name of folder
- Container: Type of folder container
- Comment: Comment
- Owners: Owners of this folder (Multi-select of users in the database)

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

Editing a folder

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

© grommunio GmbH — CC BY-SA 4.0

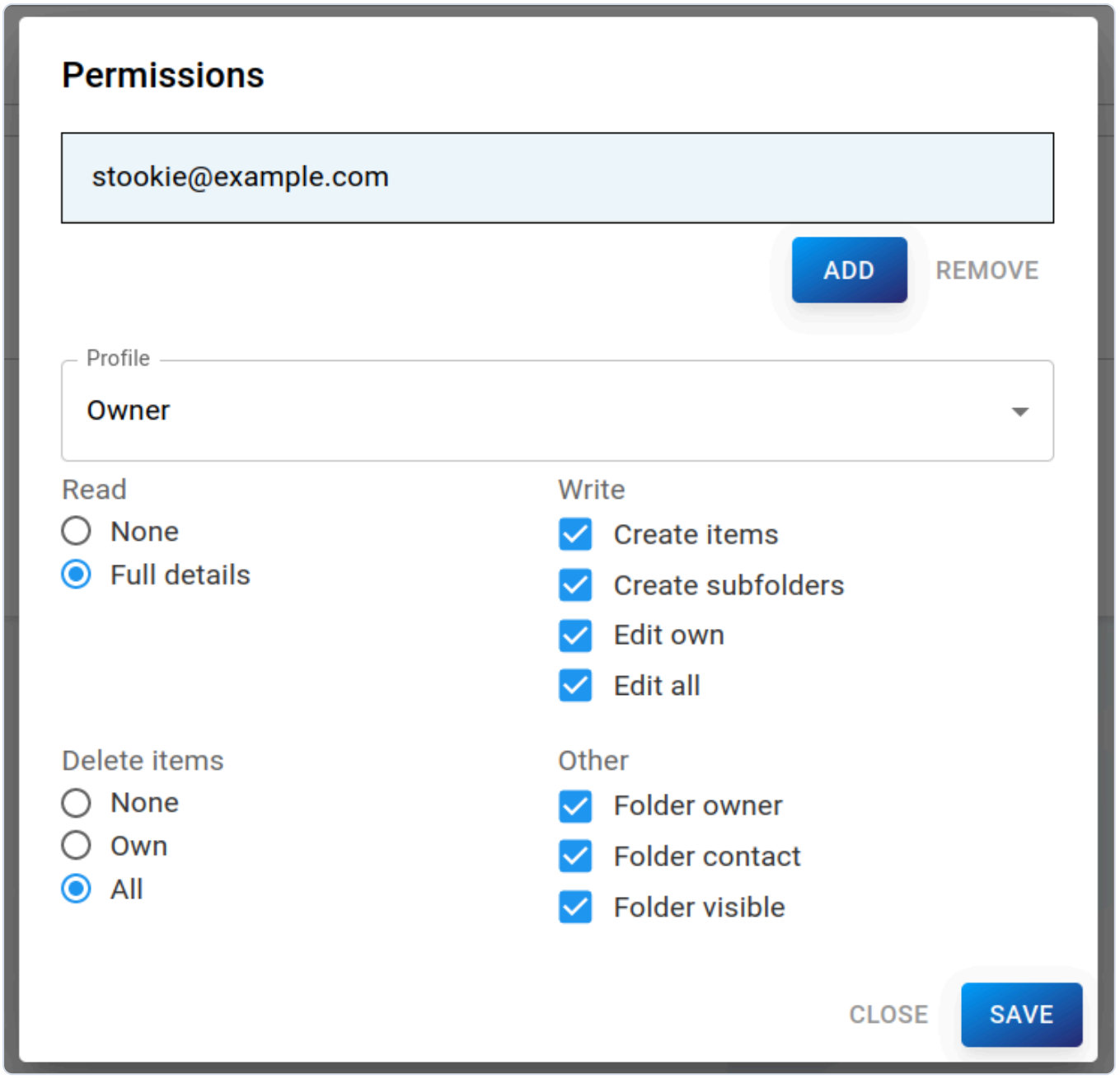


Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

Folder permissions

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

© grommunio GmbH — CC BY-SA 4.0



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.

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.

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.

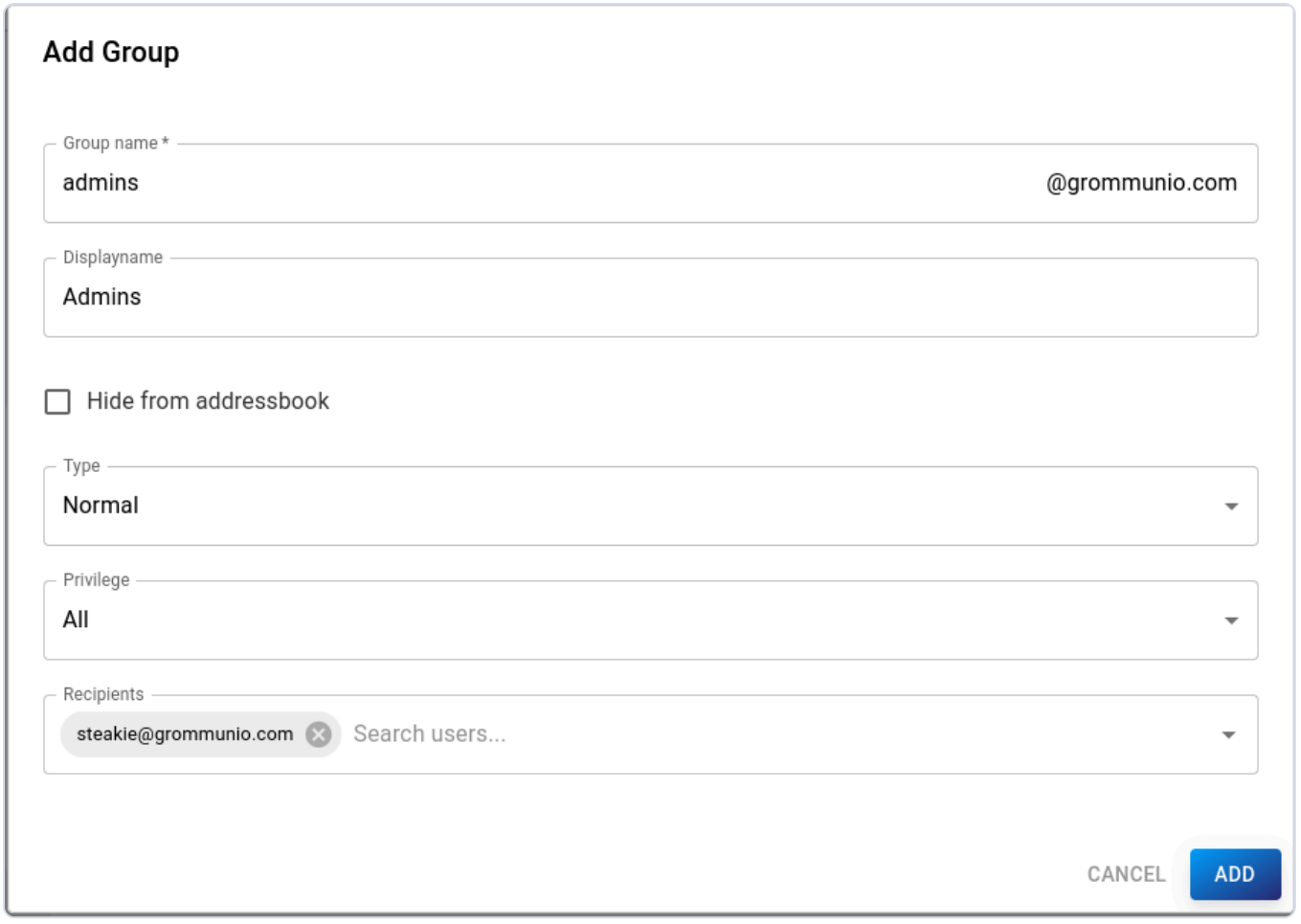
Click on a domain to expand available sub-pages and click on *Groups*, which will redirect you to the list of groups of this domain. If you have just installed grommunio or added the domain, the list will

© grommunio GmbH — CC BY-SA 4.0

be empty.

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

© grommunio GmbH — CC BY-SA 4.0

    - 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.

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.

Roles

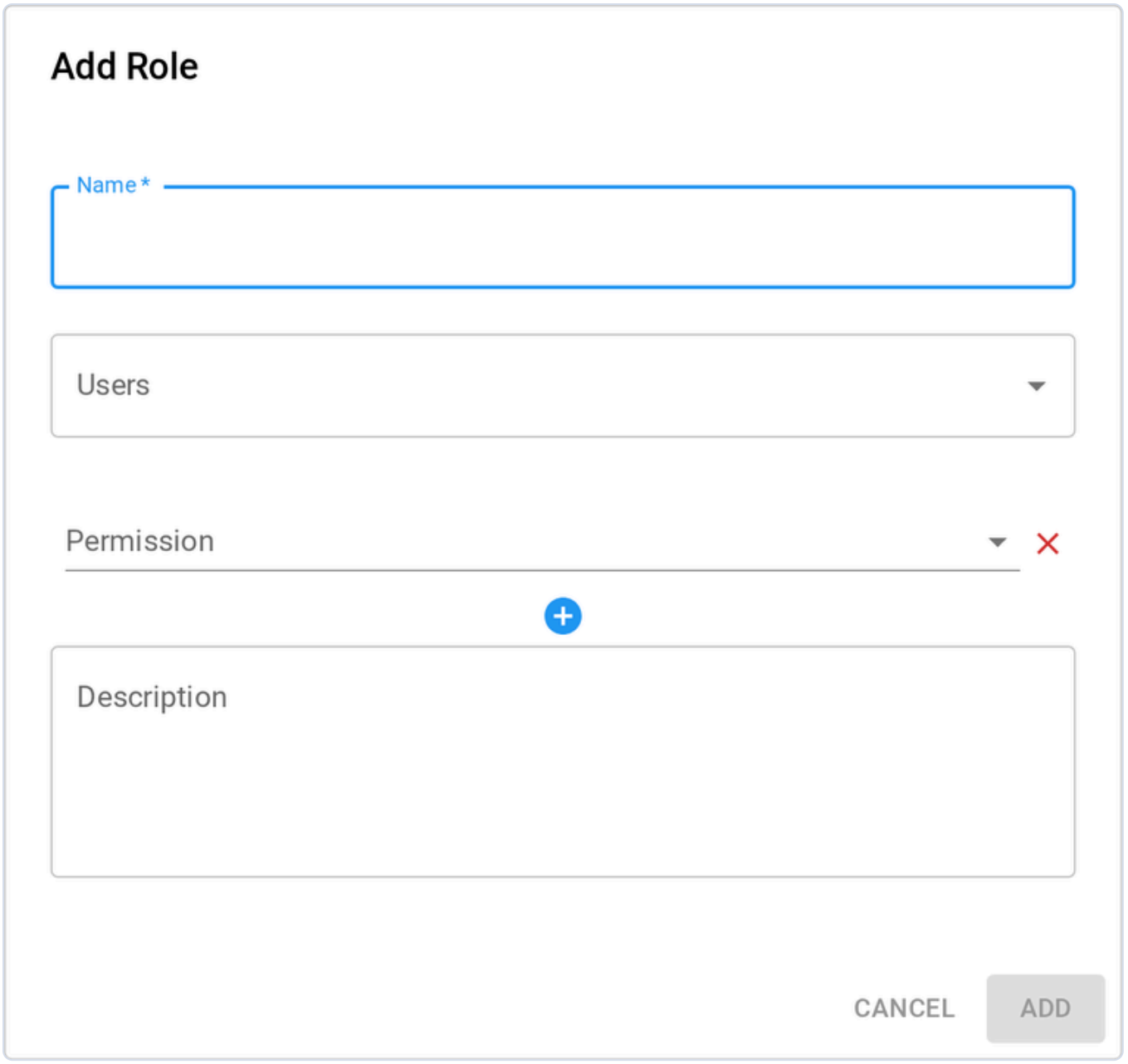
Click on *Roles* in the drawer, which will redirect you to the list view of existing roles. If you have just set up grommunio, the table will be empty.

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.

Adding a role

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

© grommunio GmbH — CC BY-SA 4.0



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

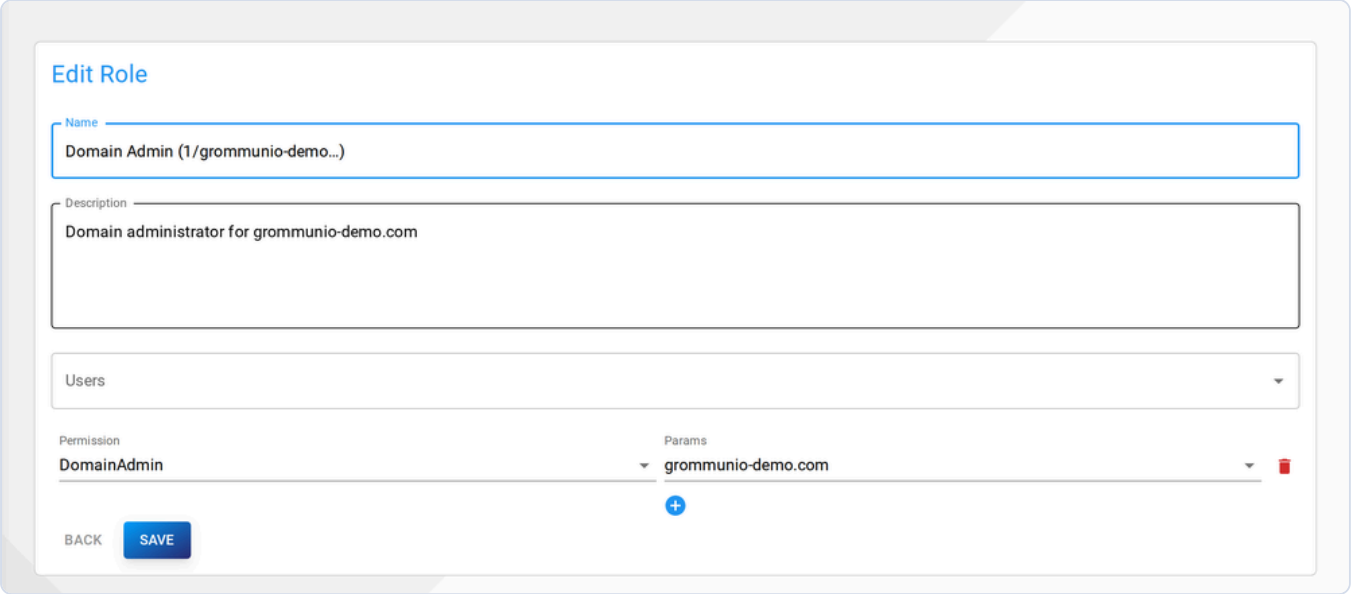
© grommunio GmbH — CC BY-SA 4.0

- Description: Role description

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

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.

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.

Organizations

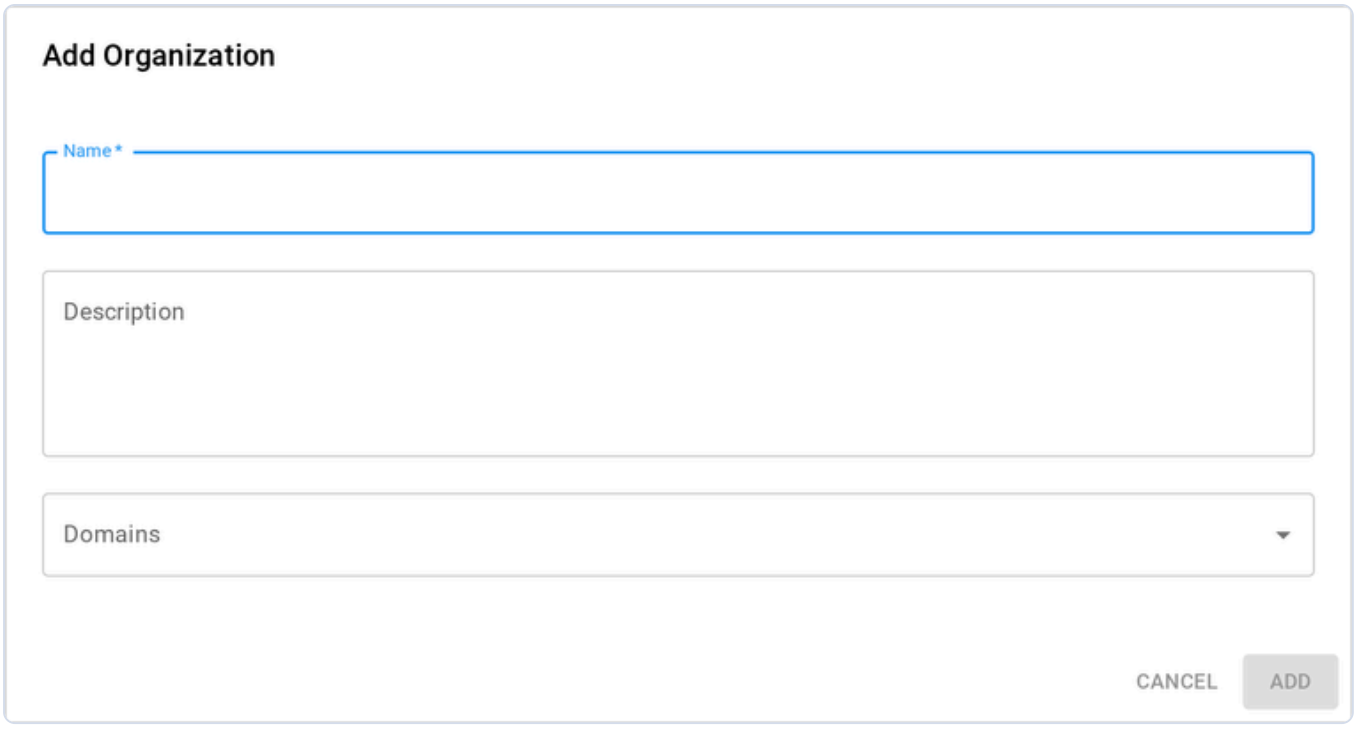
Click on *Organizations* in the drawer, which will redirect you to the list view of existing organizations. If you have just set up grommunio, the table will be empty.

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.

Adding an organization

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

© grommunio GmbH — CC BY-SA 4.0



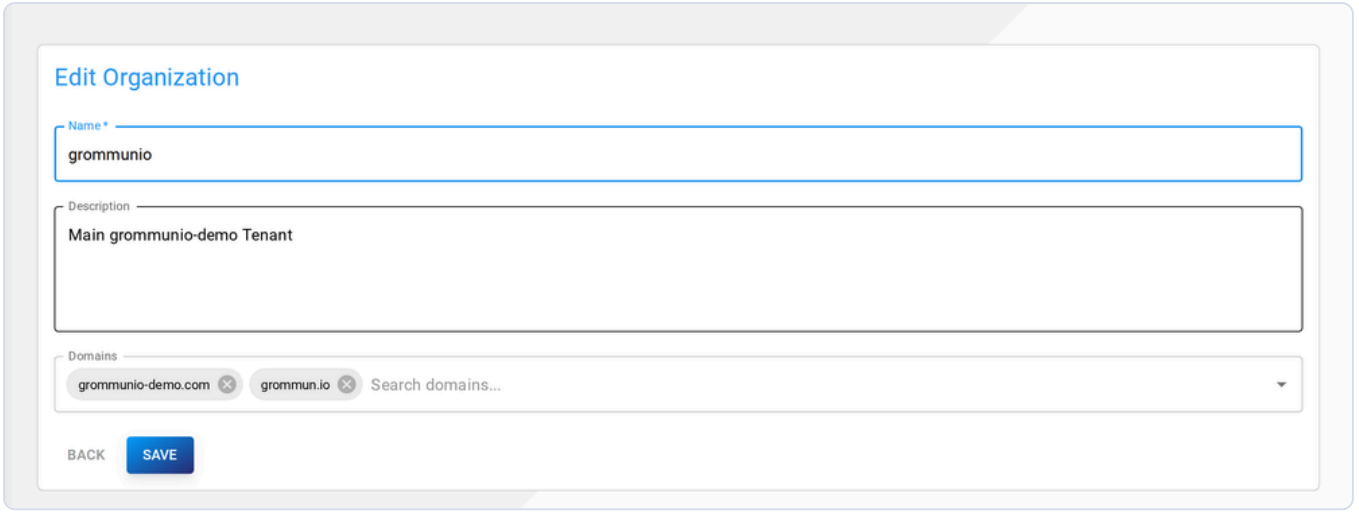
The following properties can be set:

- 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.

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.

© grommunio GmbH — CC BY-SA 4.0

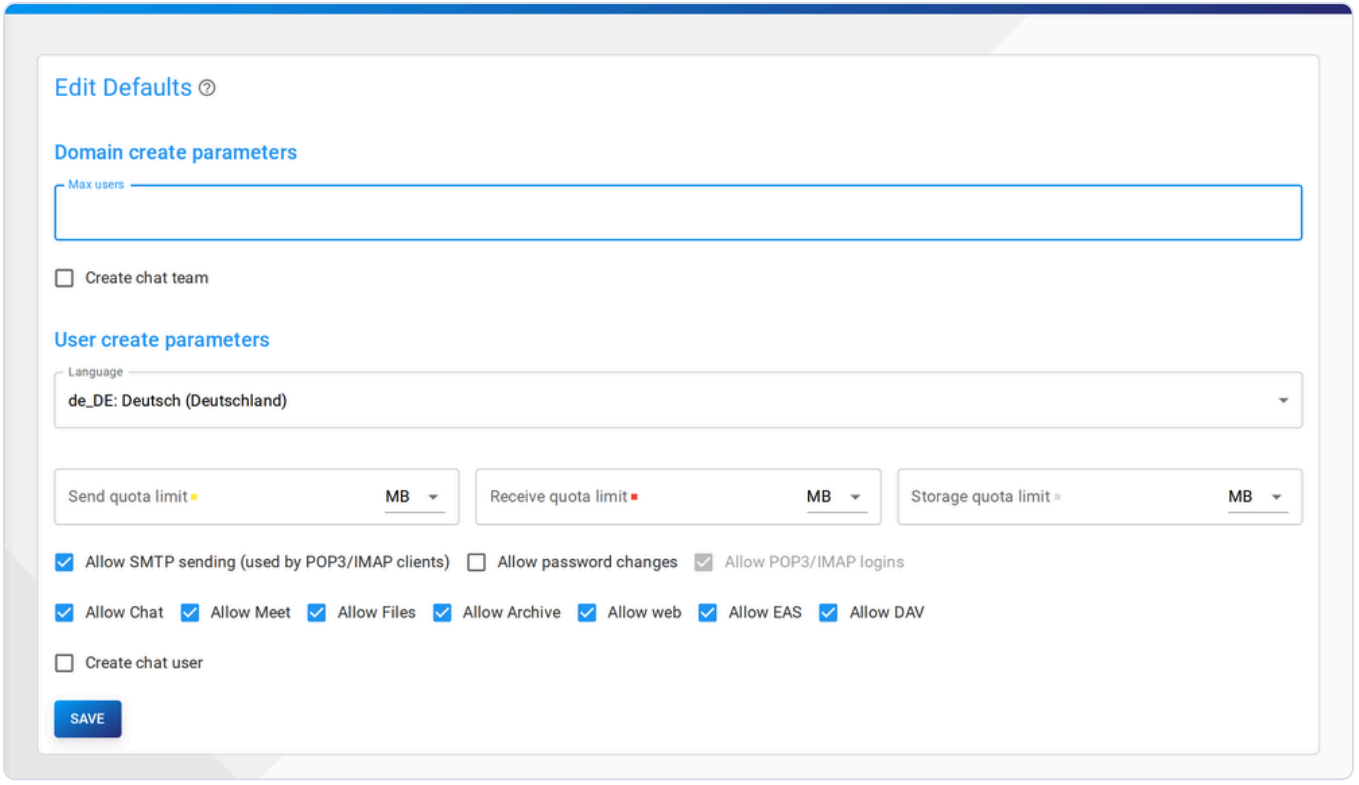
Deleting an organization

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

Defaults

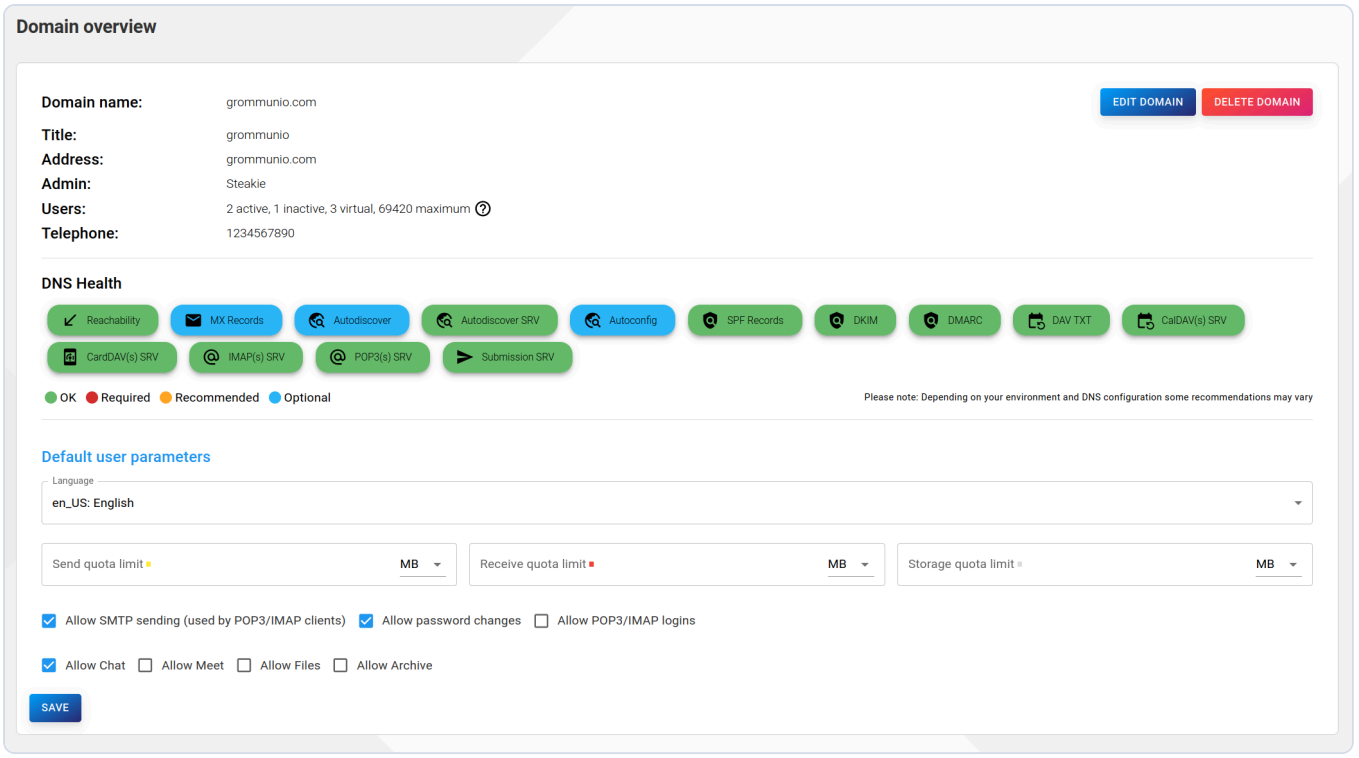
To simplify the creation of domains and especially users, it is possible to set default create parameters. If set, the input masks for adding a domain or user will automatically be filled with these values.

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:

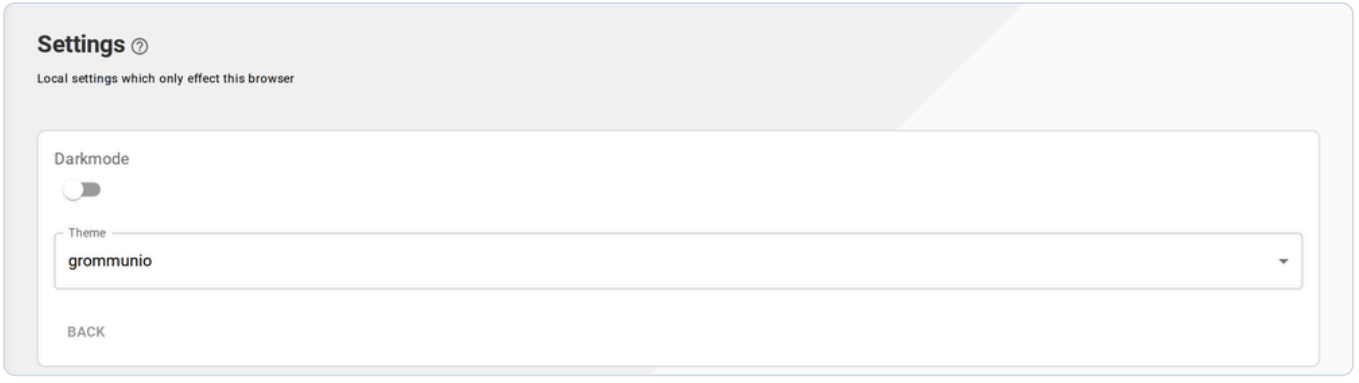
© grommunio GmbH — CC BY-SA 4.0



Settings

Two kinds of settings are reachable from the top bar:

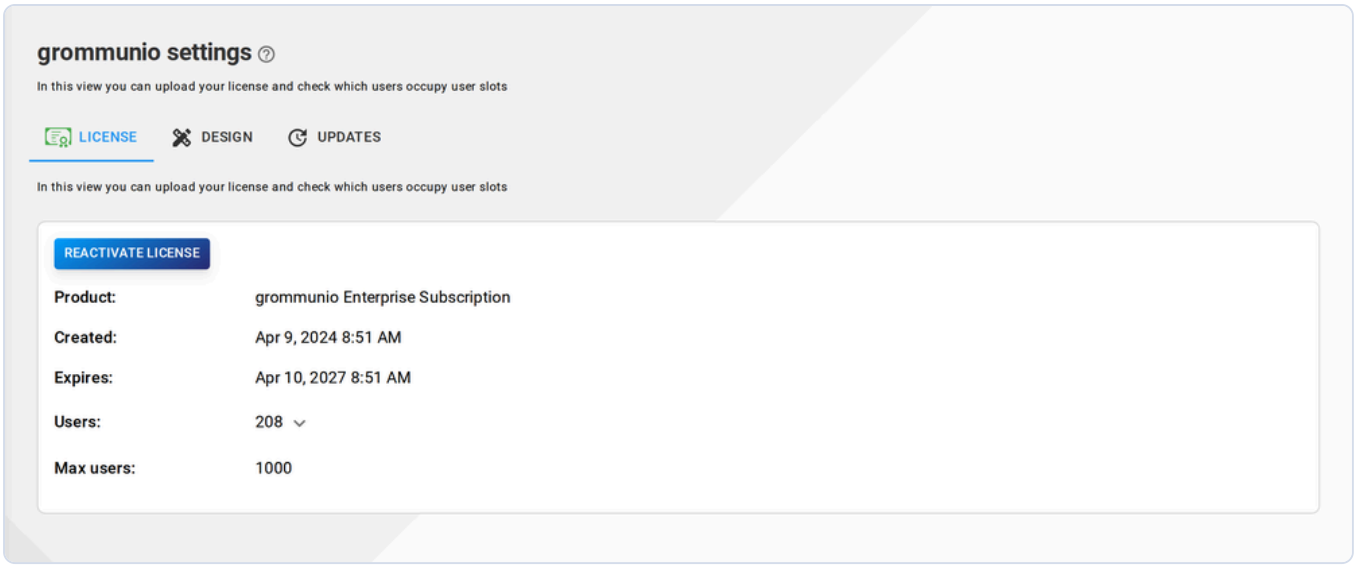
- **Local settings** — per-browser preferences (**dark mode** and the colour **theme**) on the *Settings* page. The language and light/dark toggle are also in the top bar.
- **grommunio settings** — the server-wide configuration page, with **License**, **Design** and **Updates** tabs (described below).



License

On the *License* tab you upload your license (click *Upload* and select your purchased license; *Reactivate license* re-applies it). The following properties are shown:

© grommunio GmbH — CC BY-SA 4.0



- Product — type of grommunio subscription (Community, Enterprise, …)
- Created — date the license was issued
- Expires — last day the license is valid
- Users / Max users — current and maximum number of users

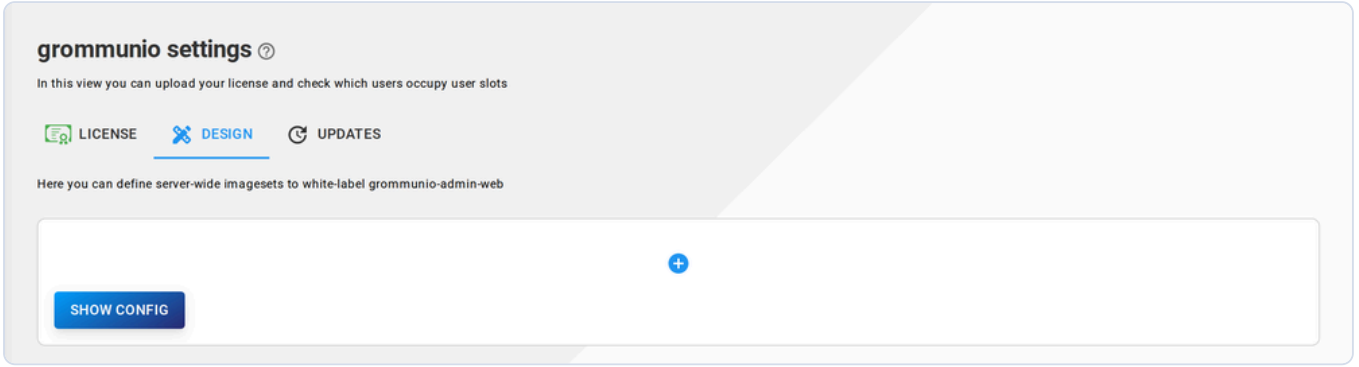
Expanding the users count shows which users occupy the license's user slots.

Design

The *Design* tab **white-labels** the Admin UI server-wide. Click the plus icon to add an image set for a hostname; each key is a URL to an image file:

- `logo` — logo in the login form
- `logoLight` — logo in the expanded drawer
- `icon` — icon in the collapsed drawer
- `background` / `backgroundDark` — background image in light / dark mode

You only need to override the images you want to change. *Show config* displays the resulting `customImages` object to copy into `/etc/grommunio-admin-common/config.json`.

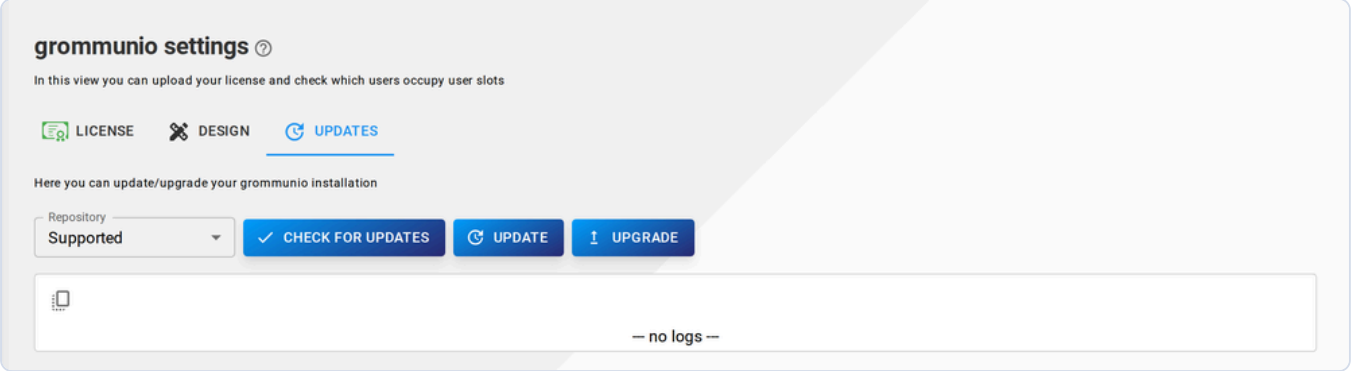


Updates

The *Updates* tab updates and upgrades the grommunio installation from the Admin UI:

© grommunio GmbH — CC BY-SA 4.0

- Choose the **repository** — *Community* (public) or *Supported* (license required).
- **Check for updates**, then **Update** or **Upgrade** the packages with the respective buttons.



Application links and server-side configuration

The external **application links** in the top-bar app launcher are set server-side in `/etc/grommunio-admin-common/config.json` (each a URL, empty by default): `rspamdWebAddress`, `mailWebAddress` (grommunio Web), `chatWebAddress` (Chat), `videoWebAddress` (Meet), `fileWebAddress` (Files) and `archiveWebAddress` (Archive).

The same file holds further behaviour:

- `tokenRefreshInterval` — token refresh interval in seconds (default `86400`, 24 h)
- `defaultDarkMode` / `defaultTheme` — default appearance (themes: grommunio, green, purple, magenta, teal, orange, brown, bluegrey)
- `loadAntispamData` — load antispam data on the dashboard (default `true`)
- `searchAttributes` — possible LDAP search attributes (default: all attributes)

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.

> **⚠Caution**
>
> 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.

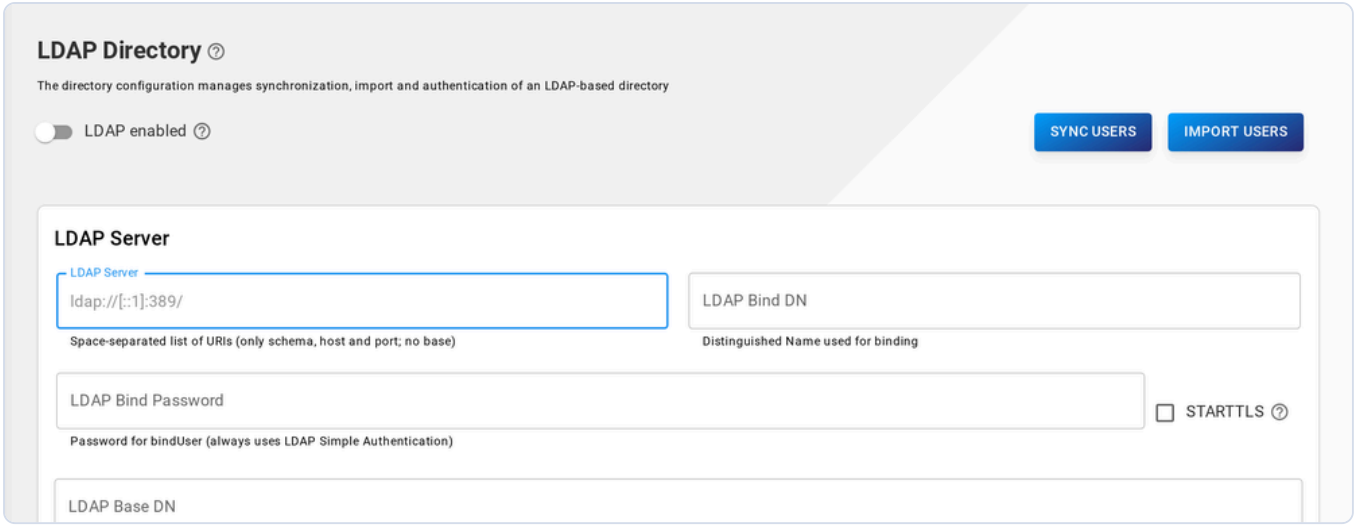
After applying a new LDAP configuration, the services are intentionally not automatically restarted as this would result into possibly inconvenient downtime if existing internal users are already used by the authentication manager (authmgr). Services can either be restarted through admin UI in the dashboard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

© grommunio GmbH — CC BY-SA 4.0

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

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.



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.

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

Authentication manager

Primary authentication mechanism

- Always MySQL (default): MySQL authentication
- Always LDAP: LDAP authentication

© grommunio GmbH — CC BY-SA 4.0

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

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

LDAP Search Attributes

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

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

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.

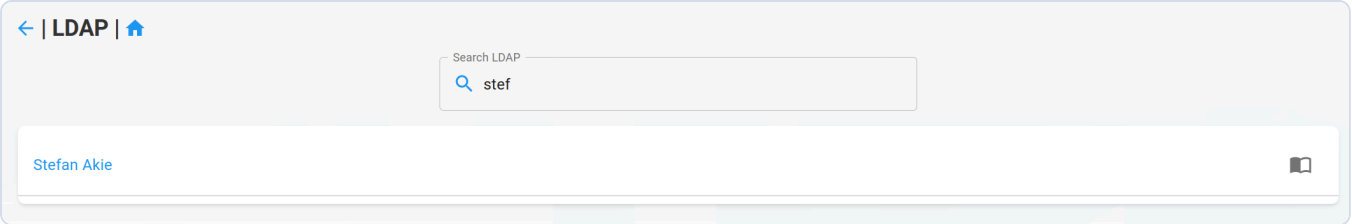
Domain user import and synchronisation

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:

© grommunio GmbH — CC BY-SA 4.0

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 username 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).

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.

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*.

© grommunio GmbH — CC BY-SA 4.0

DB Configuration

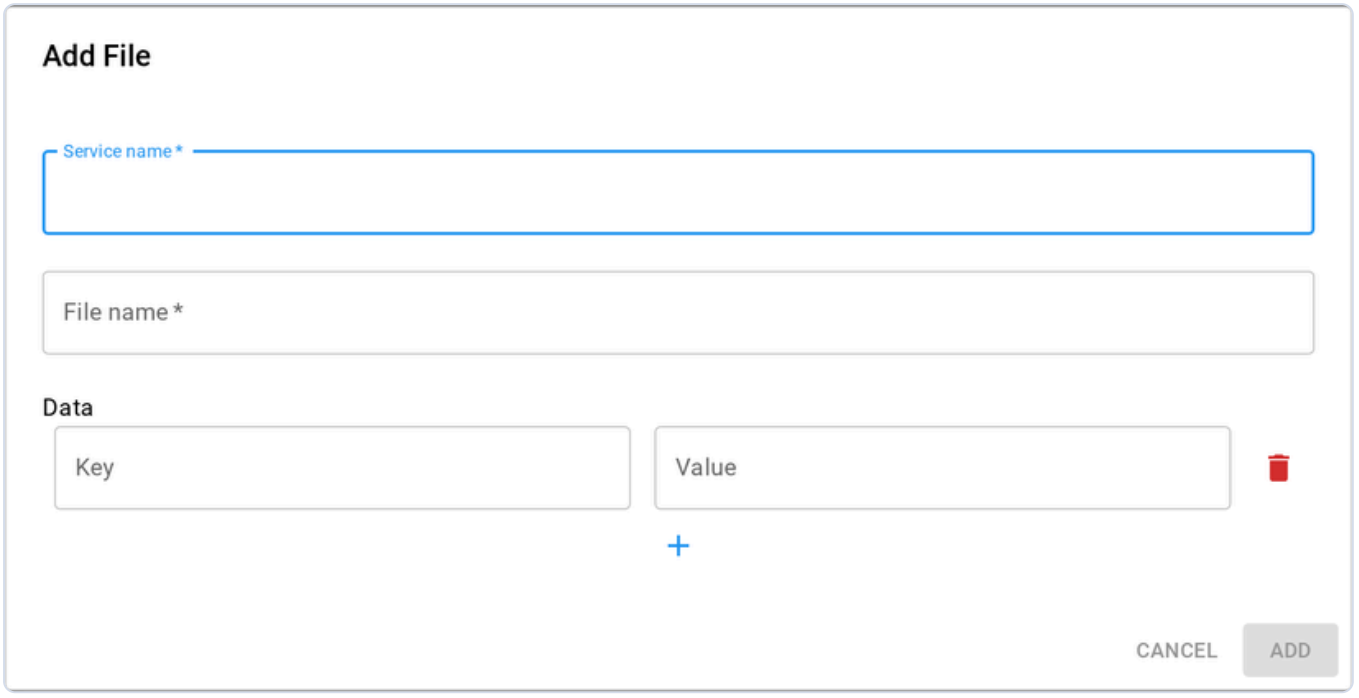
It is possible to create config files in the database to manage services. Every config file manages exactly one file and includes lines of *(key, value)* pairs.

This creates a hierarchical structure:

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

Adding a file

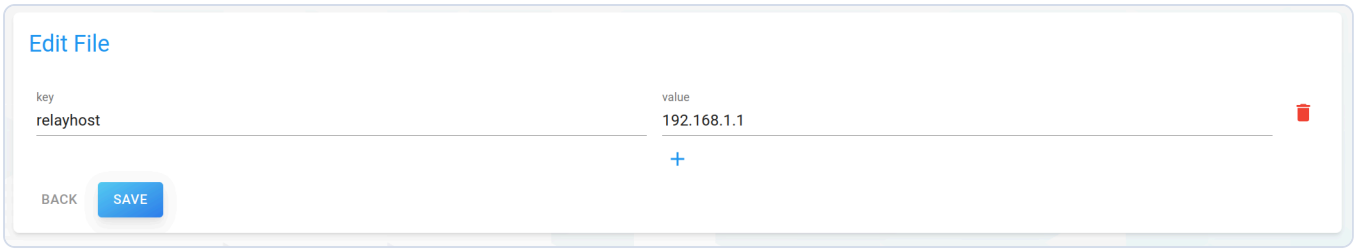
A useful example would be to configure a relayhost in postfix:



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.

© grommunio GmbH — CC BY-SA 4.0



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

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.

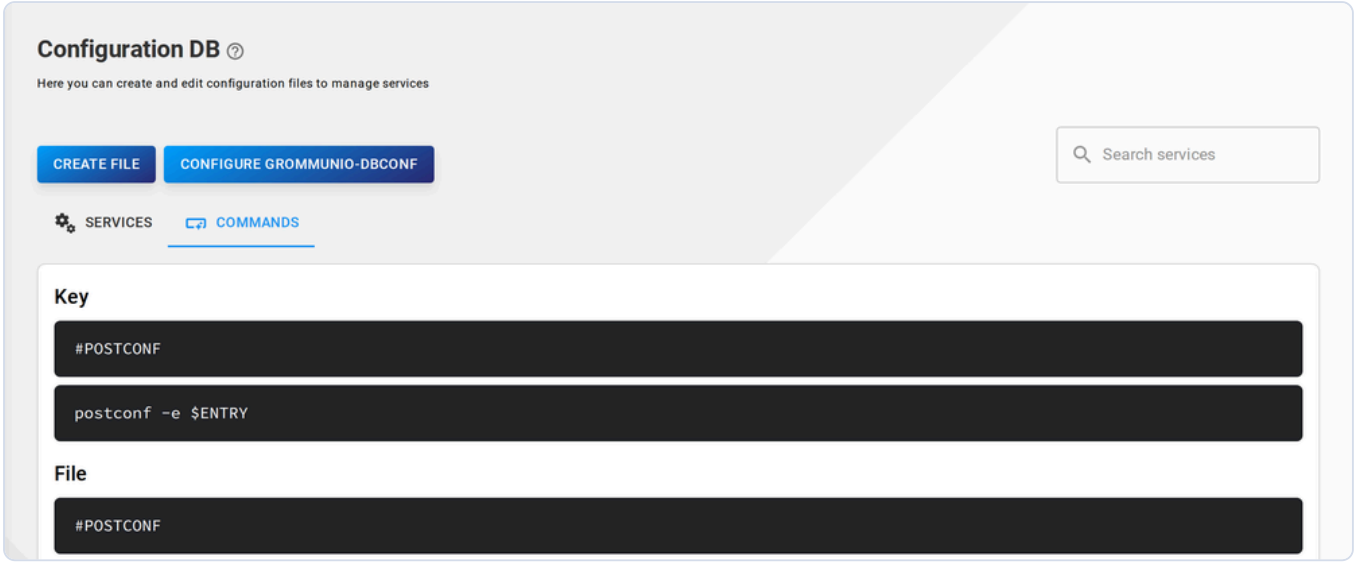
Configuring grommunio-dbconf

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

Adding a grommunio-dbconf file

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

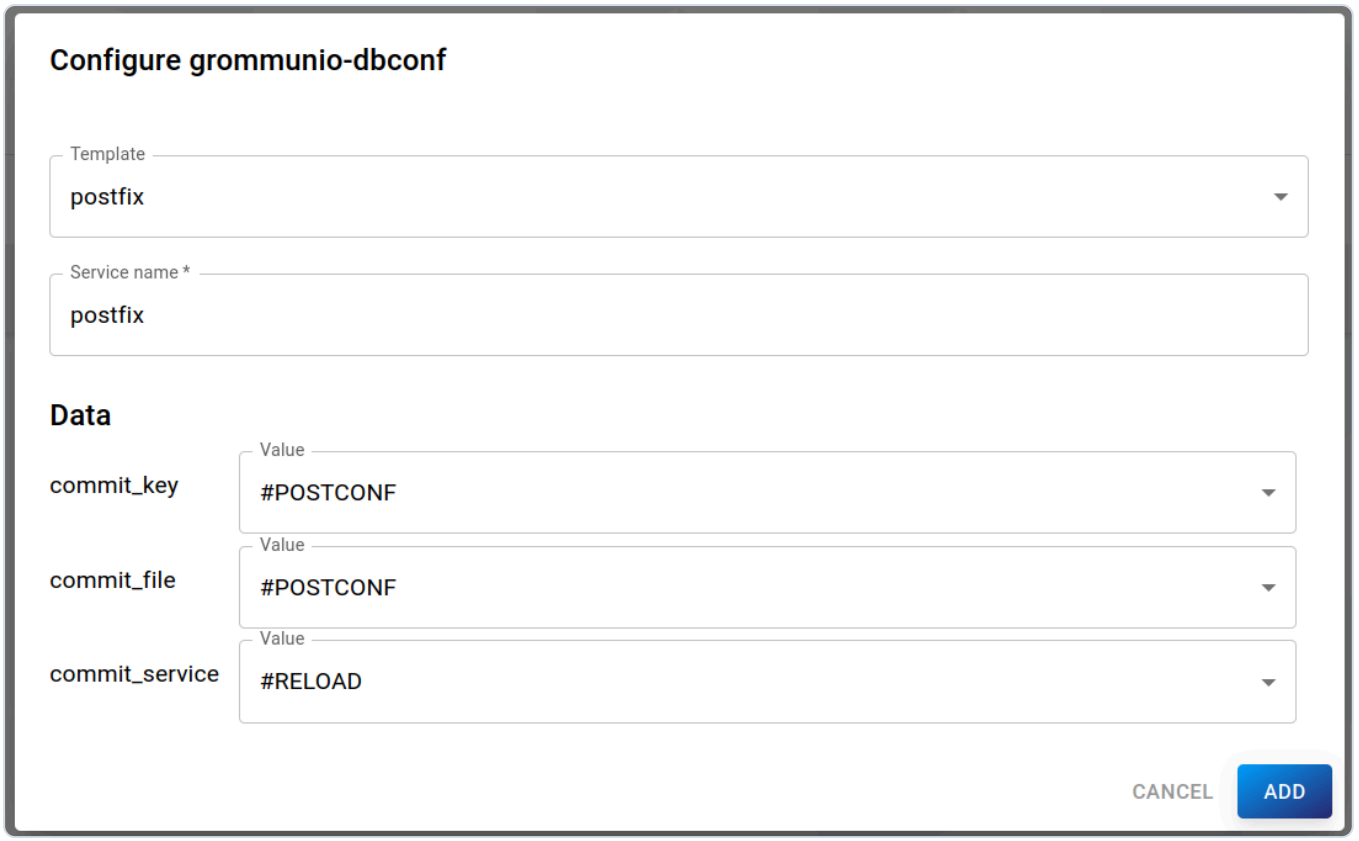
There are pre-made commands to set for either key, file or service changes. Those can be found on the *Commands* tab.



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:

© grommunio GmbH — CC BY-SA 4.0

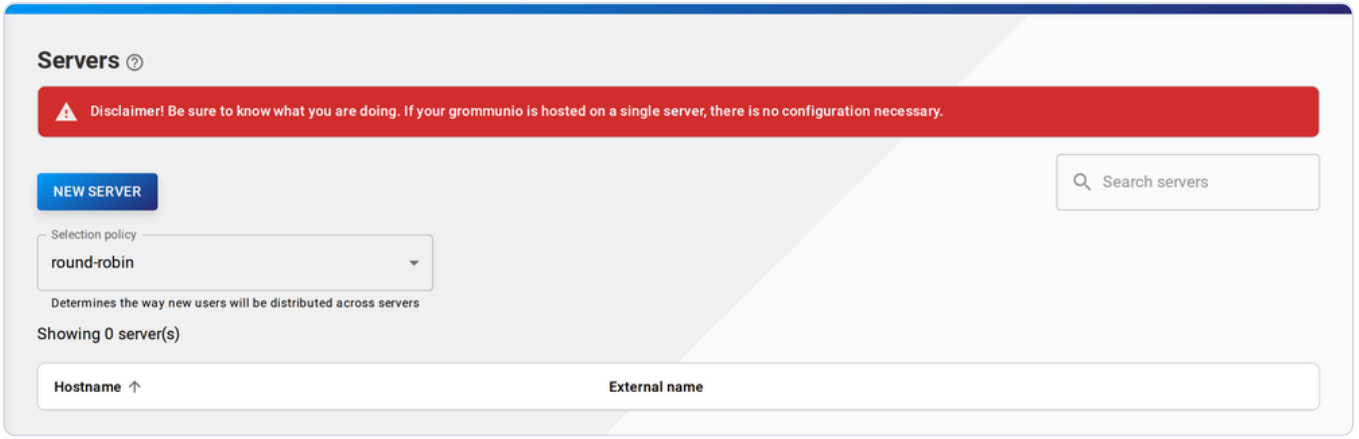


This will, among else, restart the service if the service config changes.

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:

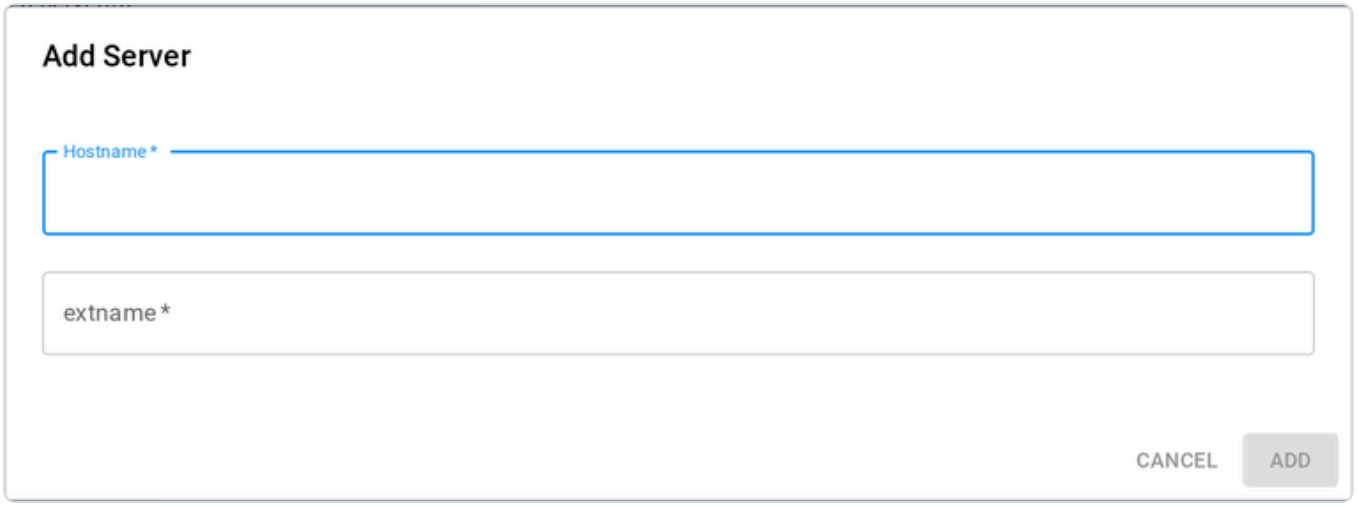
- round-robin: Always use the server on which a user has *not* been added for the longest time (in a circle-like manner).
- balanced: Put new user on server with least workload
- first: Always use the first server
- last: Always use the last server
- random: Pick a random server



© grommunio GmbH — CC BY-SA 4.0

Adding a server

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

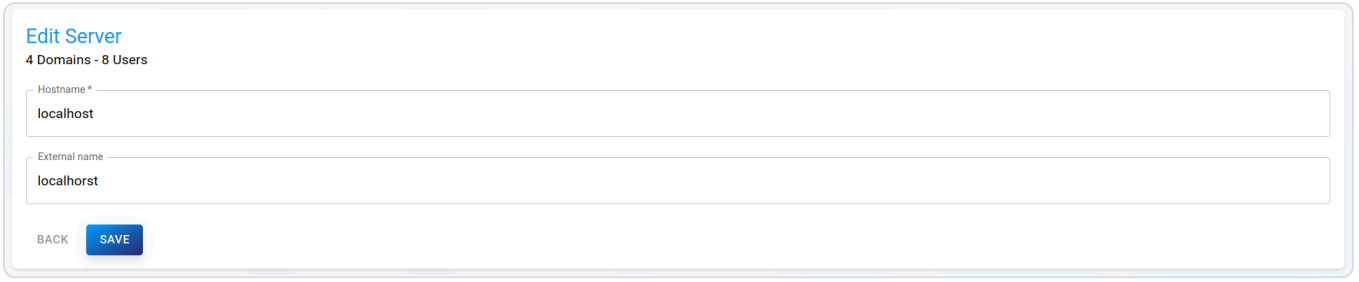


The following properties can be set:

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

Editing a server

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

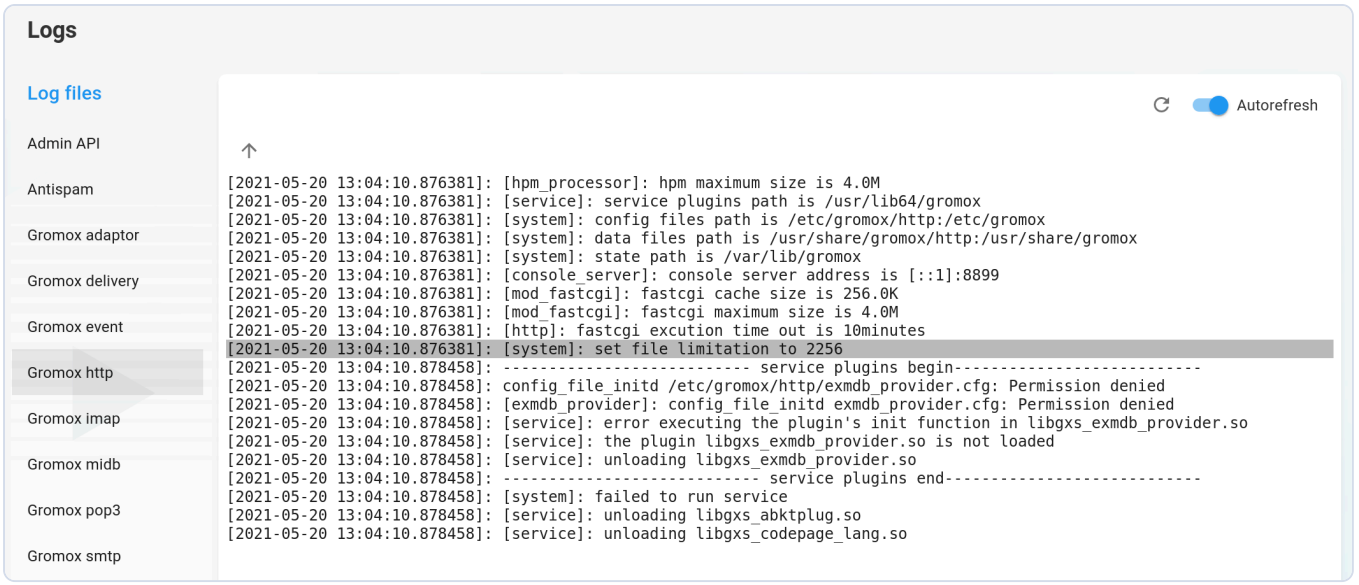


Simply change attributes to your needs, then click *Save* on the bottom to save your changes.

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.

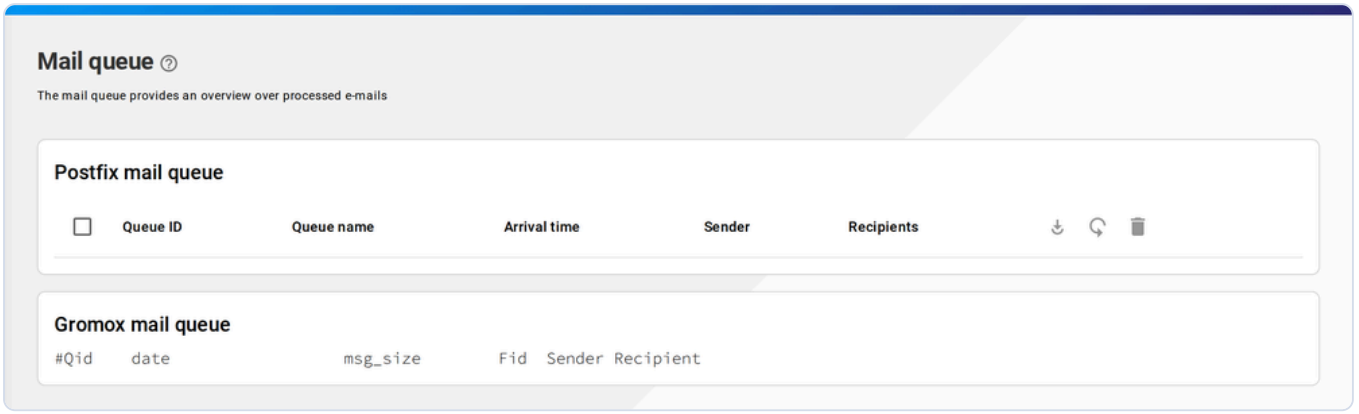
© grommunio GmbH — CC BY-SA 4.0



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.

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.

Actions

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

The actions are:

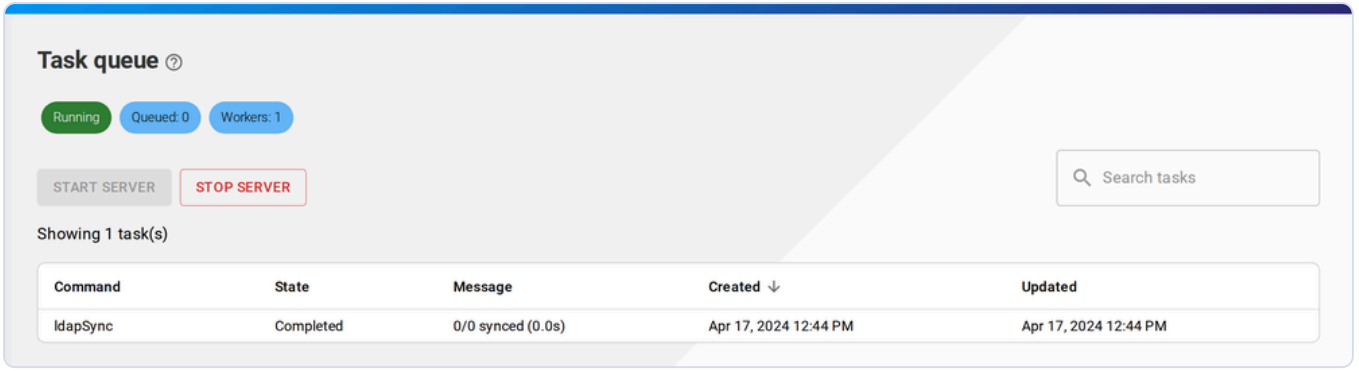
- Flush: Try to continue mail processing
- Requeue: Remove mail from queue and add queue the same mail as new entry
- Delete: Permanently remove mail from queue

© grommunio GmbH — CC BY-SA 4.0

Tasks

Click on *Tasks* 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.

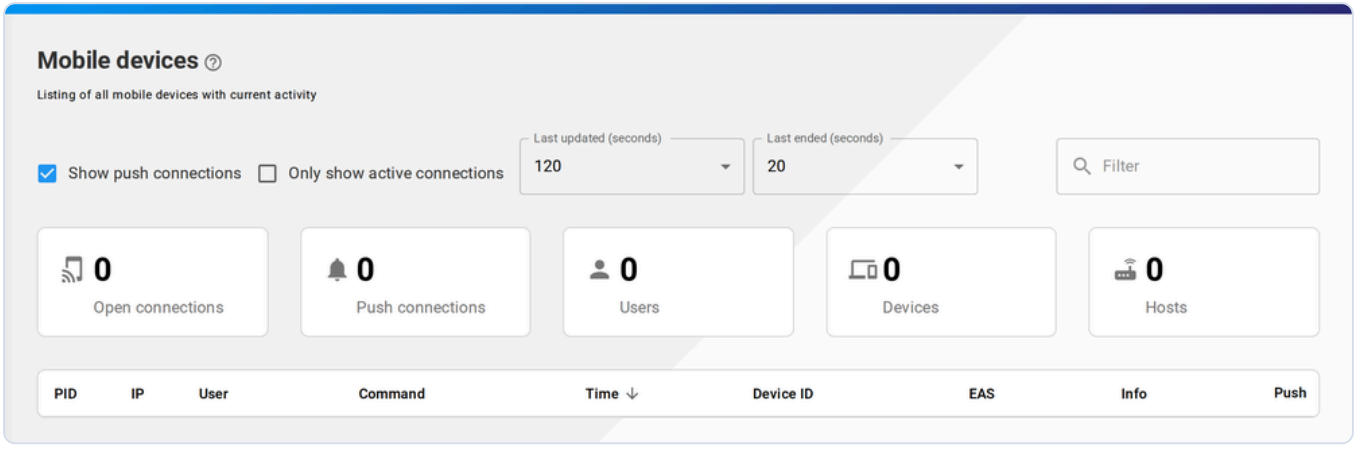


In case the internal task processor is not running, it can be started manually by clicking the *Start server* button.

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

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.



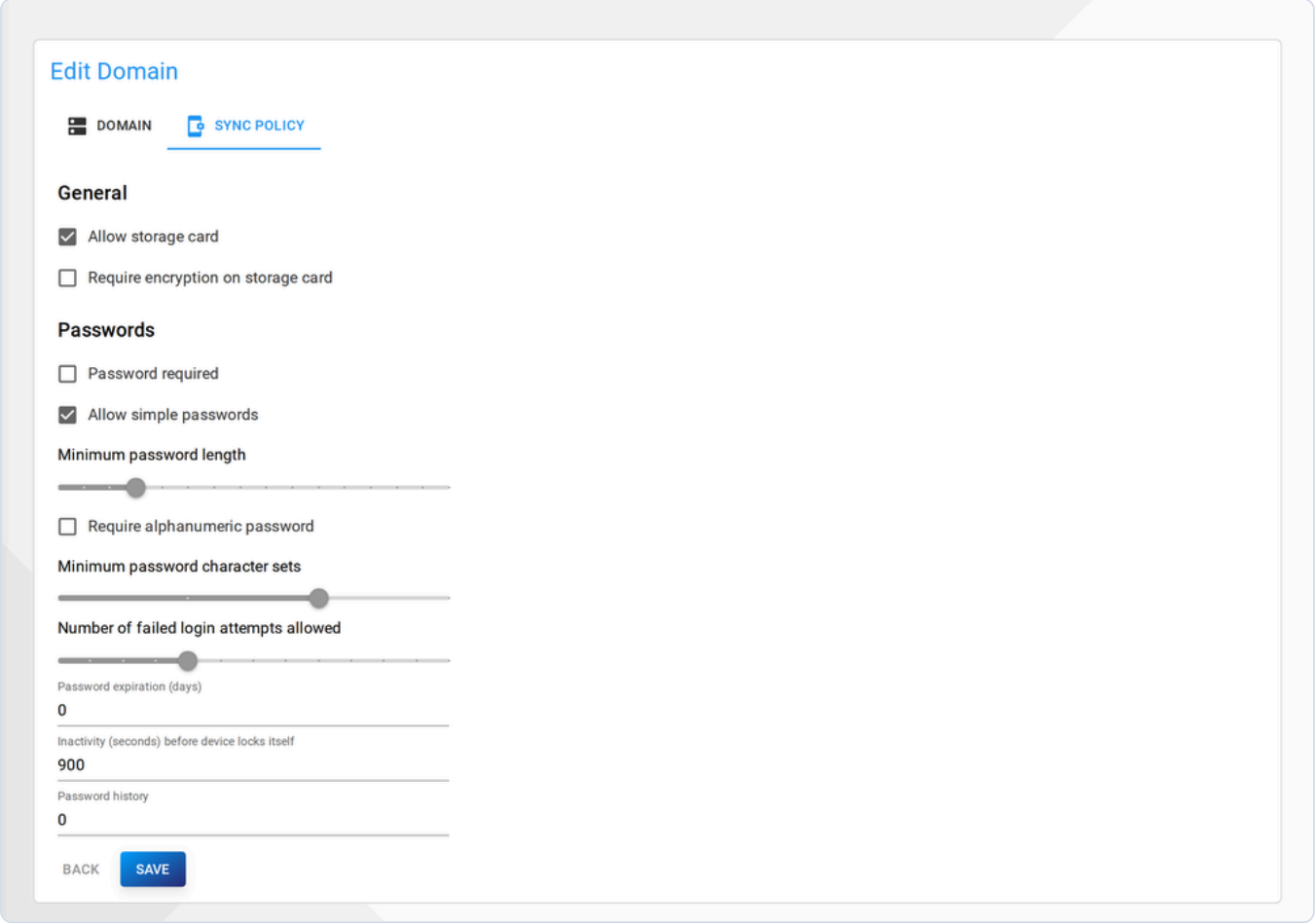
The view will update the devices every 2 seconds. On the top, you can specify filters for the table, like text-based search or activity of devices.

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

© grommunio GmbH — CC BY-SA 4.0

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.



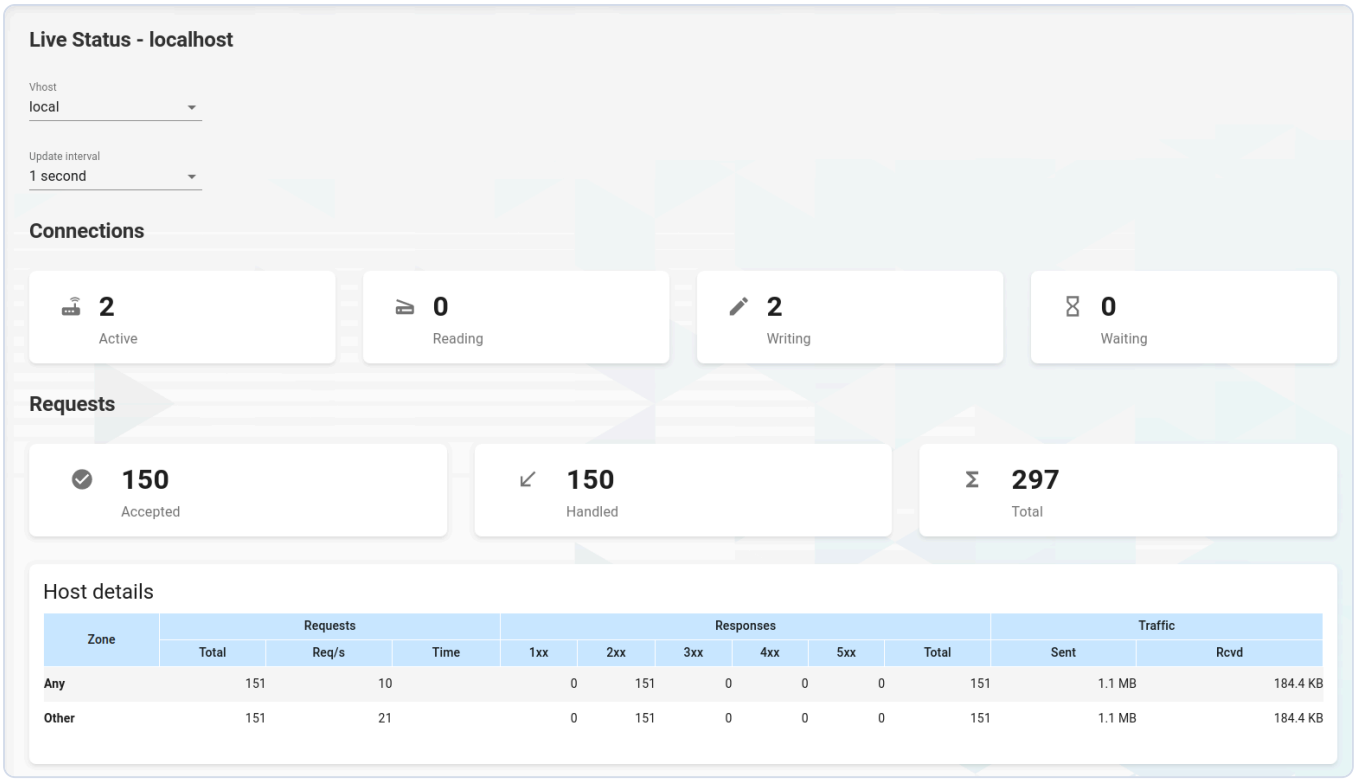
Blue checkboxes, sliders or textfields indicate deviations from the default policy, grey ones match it.

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.

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.

© grommunio GmbH — CC BY-SA 4.0



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

grommunio admin CLI (ACLI)

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.
fetchmail	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.

© grommunio GmbH — CC BY-SA 4.0

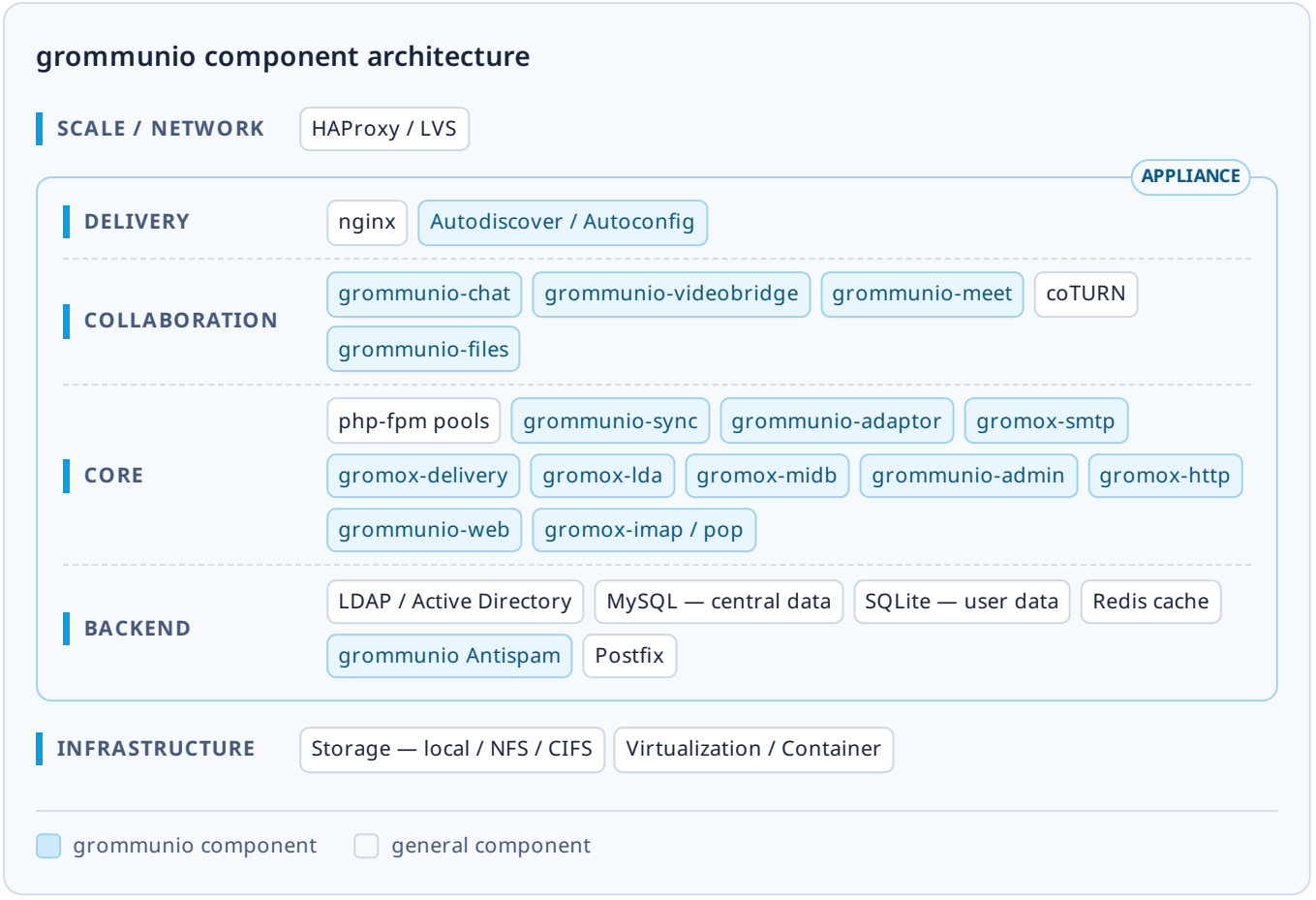
passwd	Internal user password management. See grommunio-admin passwd.
run	Run the REST API. See grommunio-admin run.
shell	Start interactive shell. See grommunio-admin shell.
taginfo	Print information about MAPI property tags. See grommunio-admin taginfo.
user	User management. See grommunio-admin user.
version	Show version information. See grommunio-admin version.

© grommunio GmbH — CC BY-SA 4.0

Architecture

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).

© grommunio GmbH — CC BY-SA 4.0

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.

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).

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.

© grommunio GmbH — CC BY-SA 4.0

> **⚠Caution**
>
> 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.

> **⚠Caution**
>
> 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.

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

© grommunio GmbH — CC BY-SA 4.0

```
# exmdb_hosts_allow replaces the former exmdb_acl.txt (Gromox 2.8+).
# There is no exmdb host map any more — routing comes from the database (below).
exmdb_hosts_allow=:: ::ffff:10.10.10.20 ::ffff:10.10.10.30 ::ffff:10.10.10.40 ::1
listen_ip=::

# /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=::
```

exmdb routing comes from the database

The former `/etc/gromox/exmdb_list.txt` host map is **obsolete** (since the development phase that followed the Gromox 3.4 release). exmdb routing is now derived from the grommunio-admin **server** definitions — the `users.homedir` abstract path and the server's hostname column — so there is no static exmdb host map to maintain. Likewise, access control moved out of the old `*_acl.txt` files into the `*_hosts_allow` directives shown above.

`midb_list.txt` is still the **midb** multiserver map (directory-prefix → midb host:port); in its absence midb assumes a single local server (`/ ::1 5555`). The `*_listen_ip` directives still work, but recent Gromox prefers the consolidated `gromox.cfg` form (e.g. `midb_listen`).

Caution

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

© grommunio GmbH — CC BY-SA 4.0

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`.

> **⚠Caution**
>
> 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.

© grommunio GmbH — CC BY-SA 4.0

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.

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 standby 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:

© grommunio GmbH — CC BY-SA 4.0

```
-- 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-*).

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 efficient 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

© grommunio GmbH — CC BY-SA 4.0

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:

Protocol and component flow of grommunio Groupware

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.

> **⚠Caution**
>
> 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.

© grommunio GmbH — CC BY-SA 4.0

HAPROXY

> **⚠Caution**
>
> The example below is a working starting point, not a hardened production configuration. In particular the `ssl-default-bind-ciphers` list is permissive (it still contains 3DES `DES-CBC3-SHA` and DHE suites); for production, drop the legacy 3DES/DHE ciphers and keep TLS 1.2+ with ECDHE/AES-GCM and the TLS 1.3 suites. Adjust the node names, ports and ACLs to your environment.

© grommunio GmbH — CC BY-SA 4.0

```
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:AES
 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

 mode http
 maxconn 80000

 bind *:443 ssl crt /etc/haproxy/proxy.pem alpn h2,http/1.1
 no option httpclose
 option forwardfor
```

© grommunio GmbH — CC BY-SA 4.0

```
    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
```

© grommunio GmbH — CC BY-SA 4.0

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

backend be_chatnodes
 stick-table type ip size 10240k expire 60m
 stick on src
 balance roundrobin
 option forwardfor
```

© grommunio GmbH — CC BY-SA 4.0

```
    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

  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
```

© grommunio GmbH — CC BY-SA 4.0

```
    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
    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
```

Postfix (PROXY protocol)

The `be_smtp` and `be_submission` backends forward to Postfix with `send-proxy`, so the load balancer prepends a PROXY-protocol header and the backend sees the real client address instead of the proxy's. Postfix must be told to **expect** that header on the proxied services — otherwise it parses the header as SMTP commands and rejects the session. In `master.cf`:

© grommunio GmbH — CC BY-SA 4.0

```
# Port 25, fronted by postscreen
smtp      inet  n  -  n  -  1  postscreen
  -o postscreen_upstream_proxy_protocol=haproxy
smtpd     pass  -  -  n  -  -  smtpd

# Port 587 (submission)
submission inet  n  -  n  -  -  smtpd
  -o smtpd_upstream_proxy_protocol=haproxy
  -o syslog_name=postfix/submission
```

If port 25 is served by a plain `smtpd` (without postscreen), put `-o smtpd_upstream_proxy_protocol=haproxy` on that service instead.

> **⚠Caution**
>
> The PROXY header lets the sender declare its own source address, so accept it **only** from the load balancer — restrict ports 25/587 to the proxy's address(es) with a firewall (and/or `postscreen_access_list`). Without `send-proxy` on the HAProxy side, do **not** set these options, or Postfix will reject ordinary connections.

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 scalability related configuration directives, for example (but not limited to) *worker_processes*.

The optimal value depends on many factors including 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.

© grommunio GmbH — CC BY-SA 4.0

```
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;
}

upstream be_adminnodes {
 server admin01.grommunio.at:8443;
}

upstream be_archivenodes {
 server archive01.grommunio.at:443;
}

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;
}
```

© grommunio GmbH — CC BY-SA 4.0

```
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 {
 listen 443 ssl http2;
 listen [::]:443 ssl http2;
 # listen 443 quic reuseport;
 # listen [::]:443 quic reuseport;
```

© grommunio GmbH — CC BY-SA 4.0

```
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-Serve
location ~* /oab { proxy_pass https://be_gromoxnodes/OAB; }
location ~* /Rpc { proxy_pass https://be_gromoxnodes/Rpc; }
```

© grommunio GmbH — CC BY-SA 4.0

```
 location ~* /web { proxy_pass https://be_webnodes/web; }

 location / { proxy_pass https://be_gromoxnodes/; }
}
```

Node-aware load balancing

The above example does not take into account largely distributed installations. With the example below, and a few automations done, haproxy can be instructed to select the appropriate backend node directly, eliminating the requirement for backend nodes to distribute the requests by doing RPC redirects and/or cross-node traffic. For this to work, a special *mailbox.map* file needs to be generated. This can be done in various ways, including interfacing the grommunio-admin-api.

On low level - by accessing the grommunio database directly - this can be done by this query for example:

```
mysql -N -e "
SELECT u.username, s.hostname
FROM users u
JOIN servers s ON s.id = u.homeserver
UNION ALL
SELECT a.aliasname, s.hostname
FROM aliases a
JOIN users u ON u.username = a.mainname
JOIN servers s ON s.id = u.homeserver;
" > /etc/haproxy/mailbox.map
```

Please note that for this to work, the appropriate relation of mailboxes to servers must be configured. This example requires the haproxy installation to be able to be resolved appropriately (when "extname" hostnames are not set to be publicly available throughout the haproxy configuration.

This configuration takes 2 assumptions:

- All backend nodes also serve web requests (grommunio-(web|sync), etc.).
- Gromox nodes are appropriately configured as per multi-server documentation.

© grommunio GmbH — CC BY-SA 4.0

```
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:AES
 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

 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 }

 # --- extract mailbox for direct routing ---
 acl web_login path_beg /web/?login
 http-request set-var(txn.mbox) req.body_param(username) if web_login
```

© grommunio GmbH — CC BY-SA 4.0

```
    http-request set-var(txn.mbox) urlp(MailboxId) if autodiscover or ews or mapi or rpc

    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
```

© grommunio GmbH — CC BY-SA 4.0

```
    option tcplog
    bind :995 name pop3s
    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
    # pick correct gromox node when mailbox is known
    use-server %[var(txn.mbox),map(/etc/haproxy/mailbox.map)] if { var(txn.mbox) -m found }

  backend be_chatnodes
    stick-table type ip size 10240k expire 60m
    stick on src
    balance roundrobin
    option forwardfor
    option http-server-close
```

© grommunio GmbH — CC BY-SA 4.0

```
    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
    # route login requests to the mailbox's home node (if web servers share names with gromox
    use-server %[var(txn.mbox),map(/etc/haproxy/mailbox.map)] if { var(txn.mbox) -m found }

  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
```

© grommunio GmbH — CC BY-SA 4.0

```
    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
    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
```

How this works:

- Frontend fe_http captures the mailbox from AutoDiscover/EWS/MAPI requests or from the grommunio-web login POST.
- Backend be_gromoxnodes uses use-server with mailbox.map to pick the correct gromox node when a mapping exists; otherwise it falls back to round-robin.

© grommunio GmbH — CC BY-SA 4.0

- Backend be_webnodes can apply the same lookup if the web servers share server names with the gromox nodes (otherwise keep round-robin).
- This keeps gromox requests and grommunio-web sessions on the mailbox's home server, avoiding unnecessary redirects or proxy hops.

> **⚠Caution**
>
> Please note that this configuration example does not cover pure TCP connections (such as e.g. submission) nor does it take OIDC into account.

With OIDC/keycloak, a different approach is recommended - we recommend here to set a cookie at login time and evaluate this in haproxy to match the usermap, e.g. extending the IdP's nginx by reading the appropriate header and setting a cookie containing the username:

```
proxy_set_header Authorization $http_authorization;

# After successful login, extract username from ID token (needs lua/openresty or auth_requ
add_header Set-Cookie "kc_username=$jwt_claim_preferred_username; Path=/; HttpOnly";
```

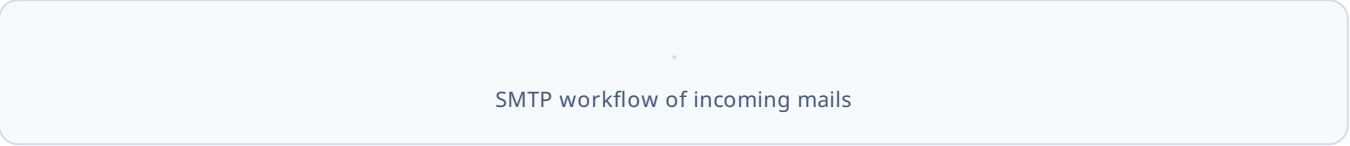
SMTP

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.

Incoming



SMTP workflow of incoming mails

Mails are processed as follows (applies to incoming and outgoing):

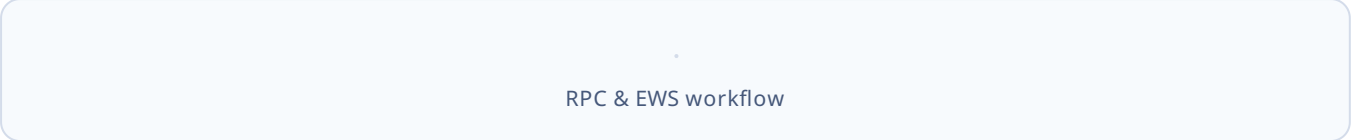
1. The included Postfix MTA receives messages and passes them to grommunio-antispam via the *Milter* mail filter protocol.
2. grommunio-antispam checks the message for spam. If configured, grommunio-antispam (optionally) passes the message to an anti-virus processing service.
3. The response from the anti-virus check is read back by antispam.

© grommunio GmbH — CC BY-SA 4.0

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 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 workflow of outgoing mails

RPC/HTTP, MAPI/HTTP & EWS workflow



RPC & EWS workflow

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://learn.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.

Exchange ActiveSync (EAS)

Exchange ActiveSync (EAS) workflow

© grommunio GmbH — CC BY-SA 4.0

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 referred 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://learn.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.

POP3

POP3 workflow

IMAP

IMAP workflow

CalDAV / CardDAV (grommunio-dav)

grommunio-dav provides standards-based **CalDAV** (calendars, tasks) and **CardDAV** (contacts) access for clients that speak DAV — such as Apple Calendar/Contacts on iOS and macOS, Mozilla Thunderbird and Evolution. Clients reach it over HTTPS through nginx, which routes DAV requests to grommunio-dav; grommunio-dav uses php-mapi/gromox-zcore to read and write the mailbox in the information store (exmdb). Client setup is assisted by AutoConfig and the `.well-known` redirects served by gromox-http.

© grommunio GmbH — CC BY-SA 4.0

CalDAV / CardDAV (grommunio-dav) workflow

Authentication

Authentication workflow

© grommunio GmbH — CC BY-SA 4.0

High availability (Pacemaker, DRBD & a floating IP)

This guide describes a **reference architecture** for running grommunio in a highly available (HA) two-node active/standby cluster with a third witness node. Storage is replicated at the block level with **DRBD**, the cluster is managed by **Pacemaker/Corosync**, and clients reach the active node through a **floating virtual IP (VIP)**. On failover the whole stack — storage, IP and services — moves to the surviving node together.

> **Reference architecture**
>
> The hostnames, IP addresses, network interface and domain below are **examples**. Replace them with your own values. DRBD, fencing and network specifics depend on your hardware and environment — treat your live installation as the source of truth and adapt the snippets accordingly.

Architecture overview

- **Two data nodes** (`node1`, `node2`) replicate a block device to each other with DRBD. At any time one is *primary* (active), the other *secondary*.
- The active node promotes DRBD, mounts the replicated volume as XFS at `/grodata`, **bind-mounts** the service data directories from `/grodata/*` onto the standard `/var/lib/*` paths, takes the **floating VIP**, and starts all grommunio/Gromox services.
- A **witness node** (`node3`) participates only in quorum voting. It holds no data and runs no grommunio services.
- Clients resolve `grommunio.example.com` via DNS to the VIP, which always lives on the active node.

Highly available grommunio cluster: clients reach a floating VIP on the active node, which promotes DRBD, mounts /grodata, bind-mounts data directories and runs the service stack; a second node holds the DRBD secondary and a third node provides quorum.

Nodes and roles

Role	Example host	Purpose
Data node 1	`node1`	DRBD peer; eligible to run the active stack
Data node 2	`node2`	DRBD peer; eligible to run the active stack
Witness / quorum	`node3`	Quorum vote only — no data, no services

© grommunio GmbH — CC BY-SA 4.0

Role	Example host	Purpose
Floating VIP	`10.0.0.10/24`	The service address clients connect to

Prerequisites

- Three nodes on the **same OS and grommunio package versions** (two data nodes plus one witness).
- A dedicated **block device** on each data node for DRBD (the same size on both).
- A spare IP address for the **VIP** on the cluster network interface (`eth0` in the examples — use your actual NIC name, e.g. `ens192`).
- Reliable name resolution (`/etc/hosts` entries for all nodes), time synchronization (chrony), and SSH connectivity between nodes.
- Cluster packages installed on all nodes: `pacemaker`, `corosync`, a CRM shell (`crmsh`), `drbd-utils` and `resource-agents`.

> ⚠ **Install grommunio, then hand control to the cluster**
>
> Install grommunio on **both** data nodes, but let **Pacemaker** start and stop the services — they must not also be started independently by systemd. Set the managed units to the appropriate `enabled` / `disabled` state for your operating model so the two layers do not fight over the services.

Cluster stack: Corosync & Pacemaker

Corosync (`/etc/corosync/corosync.conf`) defines the cluster name and the three member nodes; Pacemaker manages the resources on top.

A few cluster-wide properties matter for this design:

- **Quorum** with three nodes tolerates the loss of any one node (including the witness) without losing quorum.
- The witness is kept service-free with **location constraints** that score the VIP, the `/grodata` mount and the service group at `-inf` on `node3`.

> ⚠ **Configure fencing for production**
>
> The example configuration sets `stonith-enabled=false` for brevity. A production HA cluster **must** have working **STONITH/fencing**; without it a split-brain can promote DRBD on both nodes and corrupt data. Configure a fencing device (IPMI/iLO/iDRAC, a PDU, or SBD with a watchdog) before going live.

© grommunio GmbH — CC BY-SA 4.0

Storage: DRBD and `/grodata`

DRBD resource

A single DRBD resource (`grodata`, device `/dev/drbd0`) replicates the backing disk between the two data nodes. After the initial full synchronization, DRBD is handed to Pacemaker, which manages it as a **promotable clone** (`clone-max=2`, `promoted-max=1`) so exactly one node is primary at a time.

```
# Inspect replication state before and after any change
drbdadm status
cat /proc/drbd
```

> **Replication protocol**
>
> Protocol **C** (synchronous) gives the strongest data-safety guarantee. If you intend to switch to protocol **B**, do so only **after** the initial sync has completed, with a documented sync state, and inside a controlled maintenance window — checking `drbdadm status`, `cat /proc/drbd` and the Pacemaker resource state first.

Mount and bind-mount concept

On the active node `/dev/drbd0` is XFS-mounted at `/grodata`. Each service's data directory is then **bind-mounted** from `/grodata` onto its standard `/var/lib/*` path, so the persistent data follows the DRBD volume on failover.

Source under `/grodata`	Bind-mount target	Purpose
`/grodata/mysql`	`/var/lib/mysql`	MariaDB data directory
`/grodata/redis`	`/var/lib/redis`	Redis persistence
`/grodata/gromox`	`/var/lib/gromox`	Gromox mail store / store data
`/grodata/grommunio-web`	`/var/lib/grommunio-web`	Web data, sessions, index
`/grodata/grommunio-antispam`	`/var/lib/grommunio-antispam`	Antispam data
`/grodata/grommunio-dav`	`/var/lib/grommunio-dav`	DAV data
`/grodata/grommunio-admin-api`	`/var/lib/grommunio-admin-api`	Admin-API data

In the cluster the bind-mounts are modeled as `Filesystem` primitives with `fstype=none` and `options=bind`, collected in a group (`grodata_binds`) so they all follow the DRBD mount.

© grommunio GmbH — CC BY-SA 4.0

Pacemaker resources

Resource summary

Resource	Agent	Role
`groCluster`	`ocf:heartbeat:IPaddr2`	The floating VIP
`ms_grodata`	`ocf:linbit:drbd` (promotable)	DRBD primary/secondary
`grodata_mount`	`ocf:heartbeat:Filesystem`	XFS mount of `/dev/drbd0` at `/grodata`
`grodata_binds`	group of `Filesystem` (bind)	The seven bind-mounts
`grommunio_svc`	group of `systemd:*`	The ordered grommunio/Gromox service stack

Service start order (`grommunio_svc`)

The service group starts (and stops, in reverse) in a fixed order so dependencies come up first:

#	Resource	Unit
1	`mariadb`	`systemd:mariadb`
2	`redis-grommunio`	`systemd:redis@grommunio.service`
3	`php-fpm`	`systemd:php-fpm`
4	`gromox-http`	`systemd:gromox-http`
5	`gromox-midb`	`systemd:gromox-midb`
6	`gromox-zcore`	`systemd:gromox-zcore`
7	`gromox-event`	`systemd:gromox-event`
8	`gromox-timer`	`systemd:gromox-timer`
9	`gromox-imap`	`systemd:gromox-imap`
10	`gromox-pop3`	`systemd:gromox-pop3`
11	`gromox-delivery-queue`	`systemd:gromox-delivery-queue`
12	`gromox-delivery`	`systemd:gromox-delivery`
13	`grommunio-antispam`	`systemd:grommunio-antispam`
14	`grommunio-admin-api`	`systemd:grommunio-admin-api`
15	`nginx`	`systemd:nginx`

Ordering and colocation

The constraints enforce one logical chain — **promote DRBD → mount `/grodata` → provide the bind-mounts → bring up the VIP → start the services** — and keep everything colocated on the DRBD-primary node:

© grommunio GmbH — CC BY-SA 4.0

Constraint	Effect
`drbd_before_grodata`	`ms_grodata` must be promoted before `grodata_mount` starts
`grodata_before_binds`	`/grodata` mounts before the bind-mounts
`binds_before_grommunio_svc`	Bind-mounts start before the services
`grodata_before_ip` / `ip_before_grommunio_svc`	The VIP is tied to the `/grodata` stack and starts before `grommunio_svc`
`grodata_on_drbd` (colocation)	`/grodata` runs on the DRBD-promoted node
`grommunio_svc_on_*` (colocation)	Services run together with the VIP, `/grodata` and the bind-mounts
`no_*_on_witness` (location, `-inf`)	The VIP, mount and services are excluded from the witness

The full configuration is in the example below.

Postfix (MTA)

In this reference, **Postfix runs under systemd and is *not* a cluster resource** — it is configured separately on each node. Postfix integrates with grommunio via MySQL lookup maps and a milter:

Parameter	Example value / path
`myhostname`	`grommunio.example.com`
`virtual_mailbox_domains`	`mysql:/etc/postfix/grommunio-virtual-mailbox-domains.cf`
`virtual_mailbox_maps`	`mysql:/etc/postfix/grommunio-virtual-mailbox-maps.cf`
`virtual_alias_maps`	`mysql:/etc/postfix/grommunio-virtual-mailbox-alias-maps.cf`
`recipient_bcc_maps`	`mysql:/etc/postfix/grommunio-bcc-forwards.cf`
`virtual_transport`	`smtp:[::1]:24`
`smtpd_milters`	`inet:localhost:11332` (when grommunio-antispam is active)

You can run Postfix in any of three supported ways; choose per your operating model:

- **systemd-only on each node** (as above). A standby node can still send local system mail even though it holds no active Postfix role.
- **As a Pacemaker resource** added to the HA group, so it fails over with the rest of the stack.
- **As a clone**, running on both nodes simultaneously.

After any change, reconcile the documentation with `postconf -n` from the live host.

© grommunio GmbH — CC BY-SA 4.0

Operations runbook

Status and health checks

```
# Cluster
crm_mon -1
crm status
crm configure show

# Floating IP
ip a | grep 10.0.0.10
crm resource status groCluster

# DRBD and the mount
drbdadm status
cat /proc/drbd
mount | grep /grodata
df -h /grodata

# Services and logs
systemctl --failed
journalctl -u corosync -u pacemaker
journalctl -fu gromox-http
journalctl -fu grommunio-admin-api
```

Clean up and restart resources

```
# Clear failure state (all resources, or a single one)
crm resource cleanup
crm resource cleanup <RESOURCE>

# Restart an individual service
crm resource restart gromox-http
crm resource restart grommunio-admin-api
crm resource restart nginx
```

Node standby / online

```
crm node standby node1     # take a node out of resource placement
crm node online  node1     # make it eligible again
crm node online  node2     # keep the secondary ready for failover
```

© grommunio GmbH — CC BY-SA 4.0

Planned failover

1. Check cluster health: `crm_mon -1` — quorum present, no failed resources.
2. Ensure the target node is online and not in standby (`crm node online node2`).
3. Verify the DRBD sync state: `drbdadm status` and `cat /proc/drbd`.
4. Put the active node into standby, or move the resources to the target node in a controlled way.
5. Confirm on the target node: DRBD promoted, `/grodata` mounted, bind-mounts active, VIP up, services started.
6. Validate application-side: web login, Admin API, IMAP/SMTP, mail queue.

Postfix operations

```
postconf -n
systemctl status postfix
journalctl -fu postfix
mailq
postqueue -f
postsuper -d ALL        # flush the queue — only with operational sign-off
```

Backup and restore

> **DRBD is replication, not backup**
>
> DRBD faithfully replicates every change — including accidental deletions and corruption. You **need an independent backup strategy** in addition to the cluster.

Configuration to back up: `/etc/corosync/*`; the Pacemaker CIB (`crm configure show > cib.txt`, optionally `cibadmin --query > cib.xml`); `/etc/drbd.d/*`; `/etc/fstab` and the `/grodata` mount layout; `/etc/gromox/*`; `/etc/grommunio-common/*` (including TLS material); `/etc/grommunio-admin-api/*`; `/etc/nginx/*`; PHP/php-fpm configuration; `/etc/postfix/*`; and host/network files (`/etc/hosts`, `/etc/hostname`, NetworkManager connections, `sshd_config`).

Data to back up: consistent MariaDB dumps and/or physical backups of `/grodata/mysql`; a file-level backup of `/grodata/gromox`; the remaining `/grodata/*` subdirectories (web, redis, dav, admin-api, antispam); TLS certificates and private keys (with separate access control); and the secrets from your password/secret manager.

Restore principle: provision a node with an identical OS/package base, restore (or re-initialize) the DRBD configuration and backing disk, replay the configuration from backup, restore the data under `/grodata` and verify the bind-mounts, start MariaDB consistently and check the grommunio/Postfix maps, then bring the cluster resources up in a controlled order and run acceptance tests.

© grommunio GmbH — CC BY-SA 4.0

Security & hardening

- Configure **STONITH/fencing** (see the caution above) before production use.
- Keep **TLS private keys** (`/etc/grommunio-common/ssl/server.key`) and the certificate bundle with restrictive file permissions.
- Hold **secrets** (database credentials, etc.) in a secret manager — never in the documentation or a repository.
- Harden SSH and restrict cluster/replication traffic to a trusted network.

Example configuration

A genericized `crm configure show` for the architecture above. Adapt node names, the VIP, the NIC and your DRBD/fencing specifics; this is an example to work from, not a drop-in.

© grommunio GmbH — CC BY-SA 4.0

```
node 1: node1 attributes standby=off
node 2: node2 attributes standby=off
node 3: node3 attributes standby=off

primitive groCluster IPaddr2 \
    params ip=10.0.0.10 cidr_netmask=24 nic=eth0 \
    op monitor interval=15s

primitive grodata ocf:linbit:drbd \
    params drbd_resource=grodata \
    op monitor interval=15s role=Promoted \
    op monitor interval=30s role=Unpromoted

primitive grodata_mount Filesystem \
    params device="/dev/drbd0" directory="/grodata" fstype=xfs \
    op monitor interval=20s

# One bind-mount primitive per service data directory (fstype=none, options=bind)
primitive grodata_mount_bind_mysql Filesystem \
    params device="/grodata/mysql" directory="/var/lib/mysql" fstype=none options=bind \
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_redis Filesystem \
    params device="/grodata/redis" directory="/var/lib/redis" fstype=none options=bind \
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_gromox Filesystem \
    params device="/grodata/gromox" directory="/var/lib/gromox" fstype=none options=bind \
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_grommunio_web Filesystem \
    params device="/grodata/grommunio-web" directory="/var/lib/grommunio-web" fstype=none
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_grommunio_antispam Filesystem \
    params device="/grodata/grommunio-antispam" directory="/var/lib/grommunio-antispam" fs
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_grommunio_dav Filesystem \
    params device="/grodata/grommunio-dav" directory="/var/lib/grommunio-dav" fstype=none
    op monitor interval=20s timeout=40s
primitive grodata_mount_bind_grommunio_admin_api Filesystem \
    params device="/grodata/grommunio-admin-api" directory="/var/lib/grommunio-admin-api"
    op monitor interval=20s timeout=40s

# Service primitives (one per unit; same op timeouts) — see the start-order table
primitive mariadb systemd:mariadb \
    op monitor interval=30s timeout=30s \
    op start interval=0s timeout=60s \
    op stop interval=0s timeout=60s
# … redis-grommunio, php-fpm, gromox-http, gromox-midb, gromox-zcore, gromox-event,
#    gromox-timer, gromox-imap, gromox-pop3, gromox-delivery-queue, gromox-delivery,
#    grommunio-antispam, grommunio-admin-api, nginx (identical pattern)
```

© grommunio GmbH — CC BY-SA 4.0

```
group grodata_binds \
    grodata_mount_bind_mysql grodata_mount_bind_redis grodata_mount_bind_gromox \
    grodata_mount_bind_grommunio_web grodata_mount_bind_grommunio_antispam \
    grodata_mount_bind_grommunio_dav grodata_mount_bind_grommunio_admin_api

group grommunio_svc \
    mariadb redis-grommunio php-fpm gromox-http gromox-midb gromox-zcore \
    gromox-event gromox-timer gromox-imap gromox-pop3 gromox-delivery-queue \
    gromox-delivery grommunio-antispam grommunio-admin-api nginx

clone ms_grodata grodata \
    meta promoted-max=1 promoted-node-max=1 clone-max=2 clone-node-max=1 \
         notify=true promotable=true interleave=true

# Ordering: promote DRBD → mount → binds → VIP → services
order drbd_before_grodata          Mandatory: ms_grodata:promote grodata_mount:start
order grodata_before_binds         Mandatory: grodata_mount:start grodata_binds:start
order grodata_before_ip            Mandatory: grodata_mount groCluster
order binds_before_grommunio_svc   Mandatory: grodata_binds:start grommunio_svc:start
order ip_before_grommunio_svc      Mandatory: groCluster:start grommunio_svc:start

# Colocation: keep the whole stack on the DRBD-primary node
colocation grodata_on_drbd         inf: grodata_mount ms_grodata:Promoted
colocation grodata_binds_on_grodata inf: grodata_binds grodata_mount
colocation clusterip_on_grodata    inf: groCluster grodata_mount
colocation grommunio_svc_on_grodata inf: grommunio_svc grodata_mount
colocation grommunio_svc_on_binds  inf: grommunio_svc grodata_binds
colocation grommunio_svc_on_ip     inf: grommunio_svc groCluster

# Keep data and services off the witness node
location no_grodata_on_witness    groCluster    -inf: node3
location no_mount_on_witness      grodata_mount -inf: node3
location no_grommunio_on_witness  grommunio_svc -inf: node3

property cib-bootstrap-options: \
    have-watchdog=false \
    cluster-infrastructure=corosync \
    cluster-name=grommuniocluster \
    stonith-enabled=false
rsc_defaults build-resource-defaults: \
    resource-stickiness=1
```

⚠Caution

`stonith-enabled=false` is shown only to keep the example short. **Enable fencing before production use.**

© grommunio GmbH — CC BY-SA 4.0

Kerberos single sign-on (SPNEGO)

With Kerberos single sign-on, domain-joined Windows clients — Outlook in particular — authenticate to grommunio **transparently**, without a password prompt. Gromox's HTTP service accepts Kerberos tickets via **SPNEGO** ("Negotiate"), validated against an Active Directory KDC using a dedicated service account, its **Service Principal Names (SPNs)** and an exported **keytab**.

> **Example values**
>
> The walkthrough uses lab values — realm `TESTDOM.LOCAL` (DNS domain `testdom.local`), grommunio host `gromm1`, domain controller `dc1`, and the service account `gromox`. Replace them with your own throughout.

Before you start

- An Active Directory domain with a reachable domain controller (the KDC).
- A grommunio server whose HTTP service will accept Kerberos tickets for an `HTTP/…` SPN.
- Administrative access to the DC (to create the account, register SPNs and export a keytab).

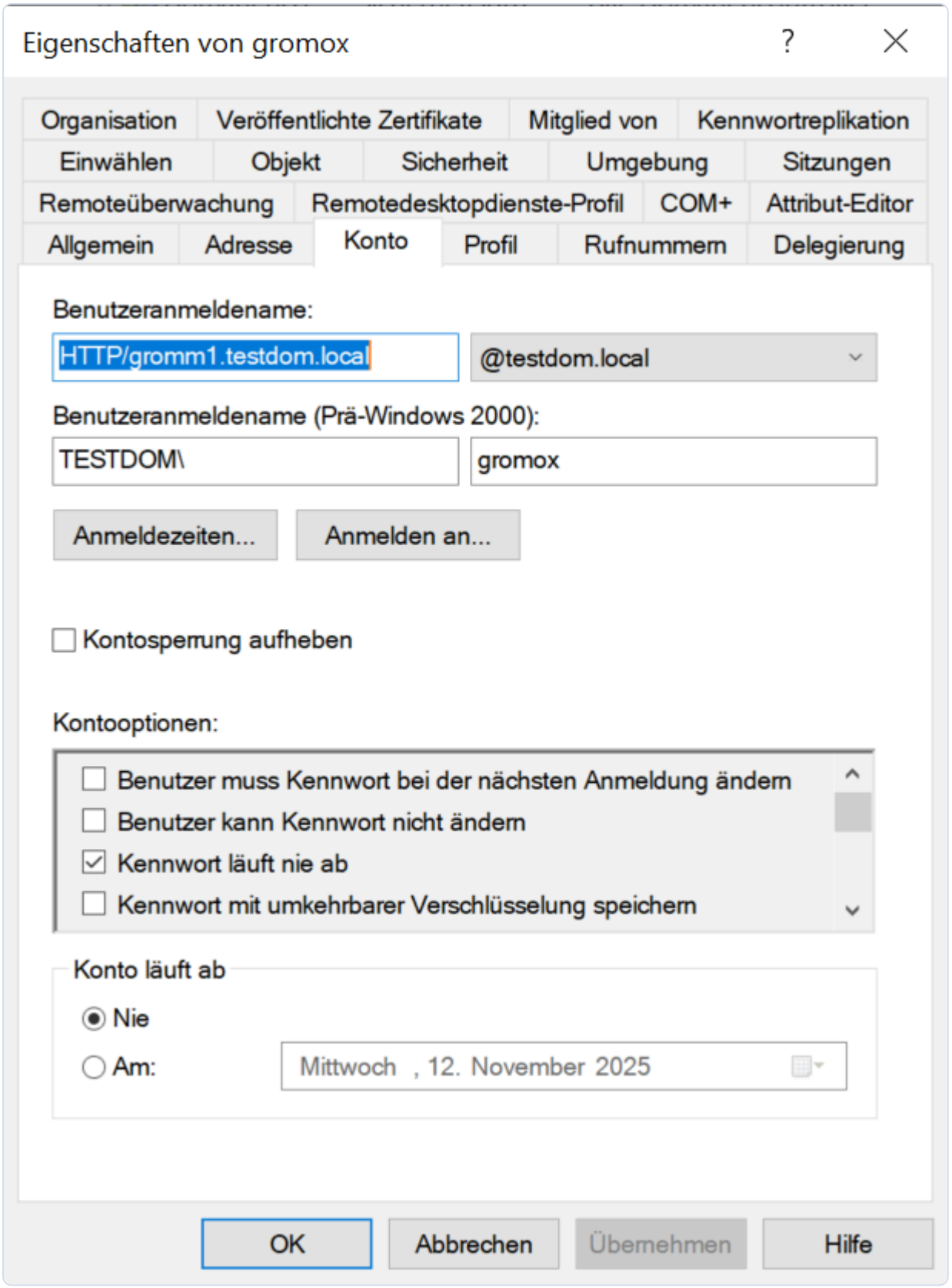
1. Synchronize time

Kerberos does **not tolerate a clock skew of more than 5 minutes**. Synchronize the grommunio server and the domain controller against the same time source (e.g. with `chronyd`) before anything else.

2. Create the service account in Active Directory

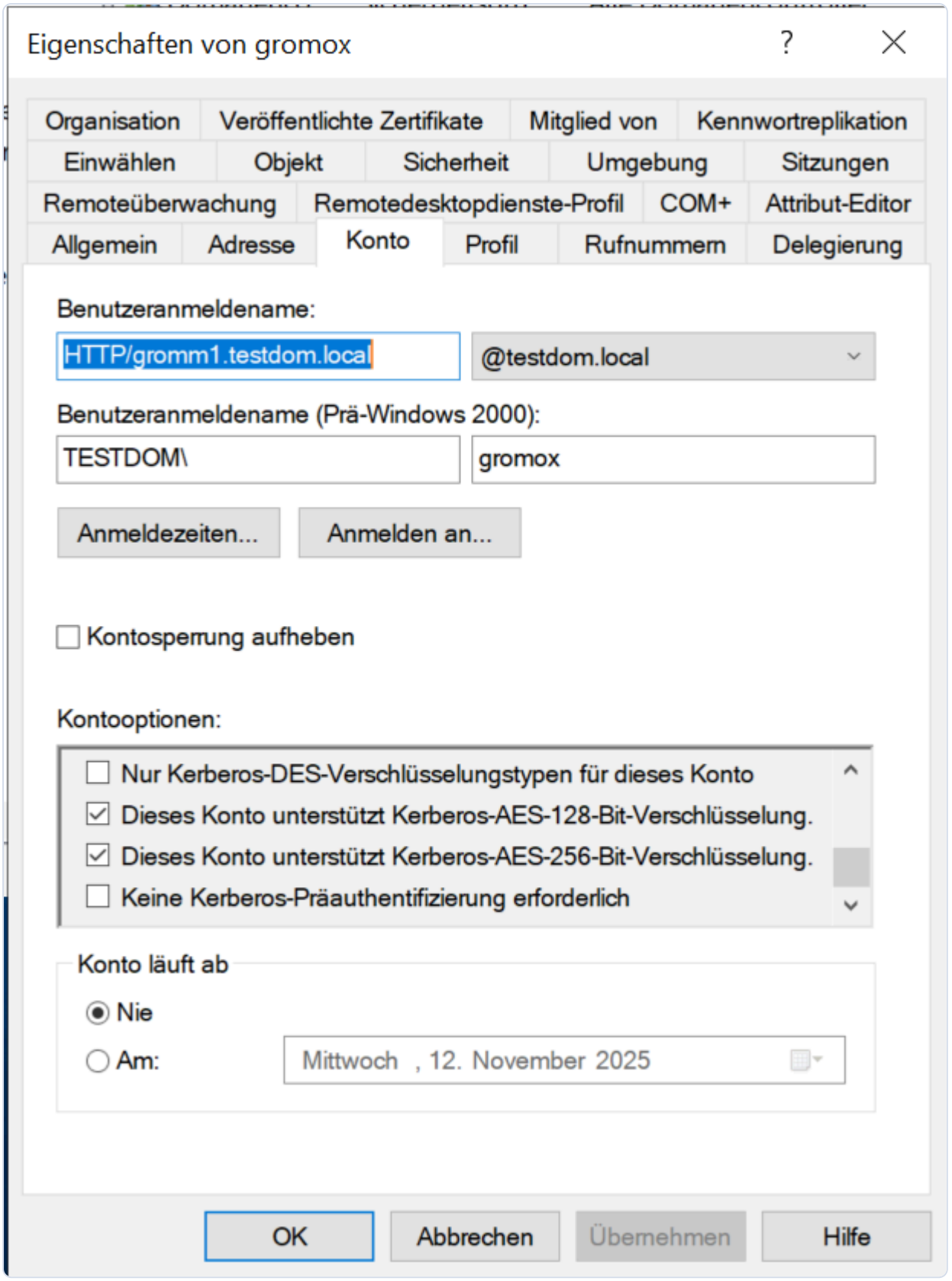
On the domain controller, create a user account that represents the HTTP service (here: `gromox`). On its **Account** tab, set the user logon name to the service's SPN (`HTTP/gromm1.testdom.local`) in the domain realm, and enable **Password never expires**.

© grommunio GmbH — CC BY-SA 4.0



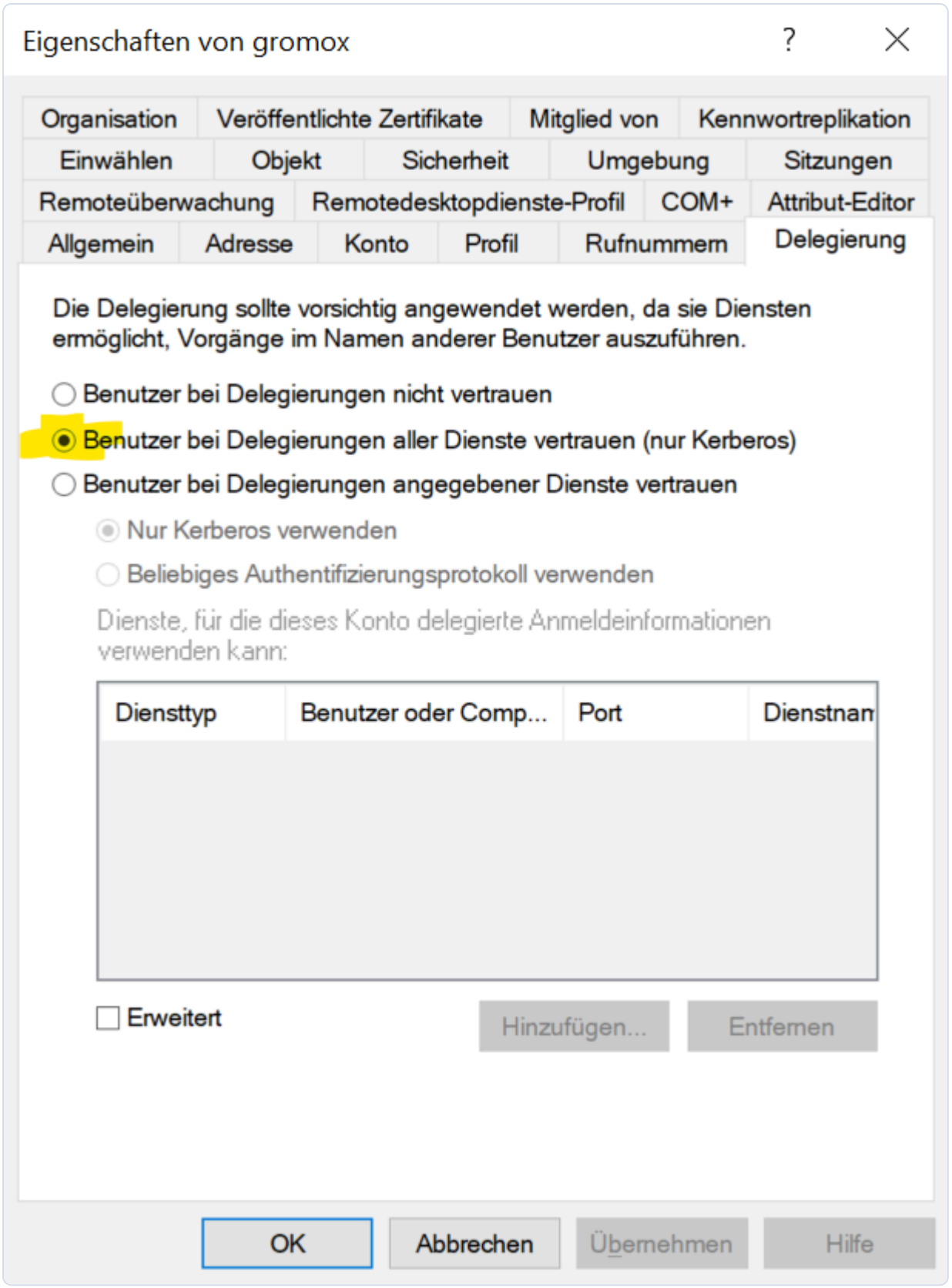
In **Account options**, enable both **"This account supports Kerberos AES 128-bit encryption"** and **"... AES 256-bit encryption"**.

© grommunio GmbH — CC BY-SA 4.0



On the **Delegation** tab, select **"Trust this user for delegation to any service (Kerberos only)"**.

© grommunio GmbH — CC BY-SA 4.0



3. Register the SPNs

Still on the domain controller, register the service principal names against the account. The `HTTP/<host>` and `HTTP/autodiscover.<domain>` SPNs are the essential pieces:

© grommunio GmbH — CC BY-SA 4.0

```
setspn -S HTTP/gromm1.testdom.local@TESTDOM.LOCAL gromox
setspn -S HTTP/autodiscover.testdom.local gromox
```

Verify (and, if needed, remove) the registered SPNs:

```
setspn -L gromox            # list the account's SPNs
setspn -D <SPN> gromox      # delete an SPN
```

```
C:\Users\administrator>setspn -L gromox
Registrierte Dienstprinzipalnamen (SPN) für CN=gromox,CN=Users,DC=testdom,DC=local:
        HTTP/autodiscover.testdom.local
        HTTP/gromm1.testdom.local@TESTDOM.LOCAL
```

4. Export the keytab

Export a Kerberos keytab that maps the SPN to the service account, using AES256-SHA1:

```
ktpass.exe -princ HTTP/gromm1.testdom.local@TESTDOM.LOCAL ^
  -mapUser gromox@TESTDOM.LOCAL -pass <PASSWORD> ^
  -ptype KRB5_NT_PRINCIPAL -crypto AES256-SHA1 -out C:\Temp\krb5.keytab
```

> **Pin the key version number**
>
> You can fix the key version number with `-kvno` (e.g. `-kvno 51`). It must match the KVNO the keytab reports on the grommunio server (see Verification).

5. Copy the keytab to the grommunio server

Transfer `krb5.keytab` to the grommunio server as `/etc/krb5.keytab` (e.g. with WinSCP or `scp`). Give it restrictive permissions — it contains the service account's key material:

```
chown root:root /etc/krb5.keytab
chmod 600 /etc/krb5.keytab
```

© grommunio GmbH — CC BY-SA 4.0

6. Configure Kerberos on the server

Edit `/etc/krb5.conf`:

```
[libdefaults]
    default_realm = TESTDOM.LOCAL
    default_tgs_enctypes = aes256-sha1 aes128-sha1
    default_tkt_enctypes = aes256-sha1 aes128-sha1

[realms]
    TESTDOM.LOCAL = {
        kdc = dc1.TESTDOM.LOCAL
        admin_server = dc1.TESTDOM.LOCAL
        default_domain = TESTDOM.LOCAL
    }

[domain_realm]
    .testdom.local = TESTDOM.LOCAL
    testdom = TESTDOM.LOCAL
```

7. Enable SPNEGO in Gromox

Edit `/etc/gromox/http.cfg` and add:

```
http_auth_spnego=yes

## GSS provider – the built-in one is the default:
#gss_program=internal-gss

## …alternatively, use the Squid GSS helper:
gss_program=/usr/lib/squid/negotiate_kerberos_auth -s GSS_C_NO_NAME

## verbose logging while testing (remove afterwards):
http_debug=2
```

Restart the HTTP service so the changes take effect:

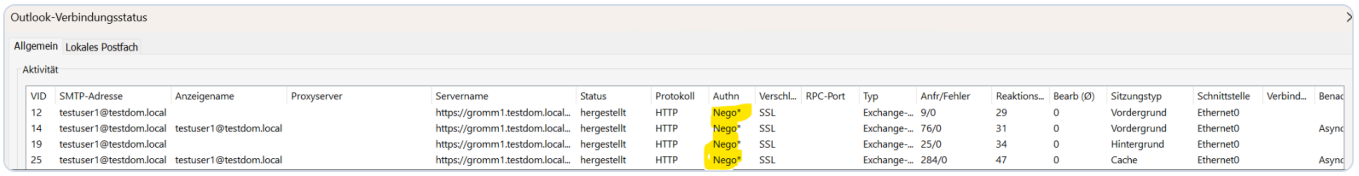
```
systemctl restart gromox-http
```

8. Test from a domain-joined client

1. Log in to a domain-joined Windows PC.
2. Create a **new Outlook profile**.

© grommunio GmbH — CC BY-SA 4.0

3. Outlook may prompt for credentials **once** on first run — enter them but **do not save the password**.
4. On subsequent starts, Outlook must **not** prompt for a password.
5. Open the Outlook **Connection Status** (hold `Ctrl` and click the Outlook tray icon → *Connection Status*). The **Authn** column should read `Nego*`, confirming Kerberos/Negotiate.



Verification & troubleshooting

Inspect the keytab and its key version number — the KVNO must match the grommunio server logs:

```
klist -k /etc/krb5.keytab          # list principals + KVNO
klist -e -k /etc/krb5.keytab       # include encryption types
# e.g. 51 HTTP/gromm1.testdom.local@TESTDOM.LOCAL (aes256-cts-hmac-sha1-96)
```

Exercise the Squid GSS helper directly:

```
# Print a Negotiate token for the local host
/usr/lib/squid/negotiate_kerberos_auth_test $(hostname -f)

# Validate a token against the service principal; paste the token when prompted
/usr/lib/squid/negotiate_kerberos_auth -s HTTP/gromm1.testdom.local@TESTDOM.LOCAL
# → OK token=… user=testuser1@TESTDOM.LOCAL group=…
```

Get a verbose Kerberos trace for a test login:

```
KRB5_TRACE=/dev/stdout kinit -V user@TESTDOM.LOCAL
```

On a Windows client, you can map an additional realm to a KDC if required:

```
ksetup /addkdc TEST.AT dc1.testdom.local
```

© grommunio GmbH — CC BY-SA 4.0

Notes

- The setup works fine with **multiple domain controllers** — just make sure DNS resolves the KDC(s) correctly.
- The two SPNs — `HTTP/<host>` and `HTTP/autodiscover.<domain>` — are the critical piece; most failures come down to a missing or mistyped SPN, a clock skew over 5 minutes, or a KVNO/enctype mismatch between the keytab and the DC.
- See also the authentication overview for how Gromox fits Kerberos alongside its other authentication backends.

© grommunio GmbH — CC BY-SA 4.0

Operations

Configuration

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_certificate.conf /etc/grommunio-admin-common/nginx-s
```

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 :

© grommunio GmbH — CC BY-SA 4.0

```
# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful

# systemctl restart nginx
```

Note that by restarting the webserver, existing connections are terminated.

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 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
```

Updating grommunio

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

© grommunio GmbH — CC BY-SA 4.0

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_16.0/?ssl_verify=n
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:

© grommunio GmbH — CC BY-SA 4.0

```
# 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.
 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 Mi
 Retrieving: grommunio-admin-web-1.0.1.8.6c8842f-lp153.1.1.noarch.rpm ..................
 Checking for file conflicts: ..........................................................
 (1/1) Installing: grommunio-admin-web-1.0.1.8.6c8842f-lp153.1.1.noarch ................
```

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.
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
```

© grommunio GmbH — CC BY-SA 4.0

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:

© grommunio GmbH — CC BY-SA 4.0

```
# 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/<domain>/<localpart>`: 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/php8/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.

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:

© grommunio GmbH — CC BY-SA 4.0

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 --all-databases --single-transaction --routines --triggers --events --add-drop-d
```

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>
```

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.

© grommunio GmbH — CC BY-SA 4.0

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
```

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:

© grommunio GmbH — CC BY-SA 4.0

- `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.

© grommunio GmbH — CC BY-SA 4.0

Troubleshooting

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 strongly 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

> **⚠Caution**
>
> 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".

© grommunio GmbH — CC BY-SA 4.0

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.

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-ActiveSync`	nginx	`/var/log/nginx/grommunio-web-access.log`, `/var/log/nginx/grommunio-web-error.log`
`/Microsoft-Server-ActiveSync`	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`

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

© grommunio GmbH — CC BY-SA 4.0

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.

© grommunio GmbH — CC BY-SA 4.0

Release Notes

grommunio 2026.06.1

- Release type: Major
- Release date: 30th of June 2026
- General availability: Yes

grommunio 2026.06.1 is the largest release to date. Over a year of work reaches across the whole stack — from the C++ mail engine up to the boot ISO — with the guiding principle unchanged: you stay in charge of your own infrastructure. Mail, calendars, files, meetings, identities and, new in this release, your AI, all run on hardware you control, with nothing forced through someone else's cloud.

Highlights

- **grommunio AI** — an assistant inside grommunio Web, with your choice of model and where it runs (cloud or fully local).
- **gromox** — a lighter, faster engine: per-mailbox information-store workers, IMAP4rev2 (RFC 9051), a much-expanded EWS, and a from-scratch Offline Address Book.
- **Shared mailboxes on mobile** — grommunio-sync now does impersonation over Exchange ActiveSync.
- **Rebuilt admin** — the Admin Web interface is now fully TypeScript, with per-domain DKIM key generation from the UI.
- **A refreshed suite** — Meet (managed TURN relay), Files (new generation), Archive 1.4, Keycloak 26.6.4 and Desk 1.2.
- **A fresh platform** — the whole appliance rebuilt on openSUSE Leap 16.0, plus native Debian/Ubuntu `.deb` packages for the entire stack.

grommunio AI

The headline addition is grommunio AI, an assistant that lives inside grommunio Web. It can summarise a single mail or a whole thread, translate messages, and help you write — draft a reply, adjust its length or tone, fix the grammar. Smart Actions turn a mail into a meeting invitation, a task, a contact or a reply; each opens a pre-filled dialog, and nothing happens until you click.

What sets it apart is that **you decide which model runs it, and where**. grommunio AI speaks both the OpenAI-compatible and Anthropic API standards, which between them cover:

- free cloud tiers — Google Gemini (the shipped default), Groq and OpenRouter
- commercial APIs — OpenAI, Anthropic/Claude, Mistral, Azure OpenAI and the rest
- local models you host yourself — Ollama, LM Studio, vLLM, llama.cpp or LocalAI

© grommunio GmbH — CC BY-SA 4.0

Point it at a local Ollama setup and nothing leaves the building — no key, no outbound call, no third party reading your mail. Privacy was built to match: the plugin ships switched off; an admin must enable it, after which every user still opts in individually; all calls happen on the server and the API key stays in the server config where the browser never sees it. The one caveat: pointed at a cloud provider, the text you are working on does go to that provider — which is exactly why the local, nothing-leaves option is a one-line change away.

gromox: a lighter, faster engine

- **Lower memory** — the information store can split itself up, running a small worker process per mailbox under a lightweight director instead of one ever-growing process. Joined by a heap reaper that returns unused memory on a schedule, tighter IMAP-daemon limits, a shorter idle-cache timeout and a stack of plugged leaks, the result is simple: the server holds onto less memory, so more mailboxes fit on the same box.
- **IMAP4rev2 (RFC 9051)** — gromox is now one of the few open-source groupware servers to implement it: advertised behind the ENABLE handshake, with ESEARCH, LIST-EXTENDED, server-side MOVE, saved search results, UTF-8 mailbox names and the cleaner status/response codes.
- **Exchange Web Services** — greatly expanded: tasks, calendar occurrences with proper timezones, real delegate management with permission levels, and the full meeting workflow (invitations, cancellations, responses, and a server-side processor that keeps the organiser's calendar in step), plus streaming notifications and room/people lookups.
- **Offline Address Book** — a brand-new, from-scratch implementation of Microsoft's OAB format and its LZX compression, built and served by gromox itself, with no Microsoft code in it.
- **Better body rendering** — HTML/RTF/plain-text conversion can hand off to Pandoc and Chawan for higher fidelity, and the in-house RTF reader was overhauled for CJK text, right-to-left scripts and nested tables.
- **Notable fixes** — Outlook no longer loses appointments when opening a shared calendar; Outlook 2010 can log in again under OpenSSL 3; AutoDiscover stopped advertising a non-existent OWA endpoint (which had quietly broken Thunderbird setup); recurring meetings keep their timezones straight. The import and export tools were also renamed `gromox-import` and `gromox-export`.

grommunio Web

Besides being the new home of grommunio AI, the web client kept getting polished where it counts — composing, S/MIME, search and theming — the steady work that keeps it the most capable open-source Exchange web client around.

Shared mailboxes on mobile, at last

grommunio-sync now supports impersonation over Exchange ActiveSync, so a shared or functional mailbox can be put on a phone with full rights, signing in with your own account — something Exchange and Microsoft 365 do not offer over ActiveSync. gromox's AutoDiscover understands the combined `sharedmailbox!user` login and enforces it server-side, checking both who you are and whether you hold owner rights on the target, so the phone is set up straight against the shared

© grommunio GmbH — CC BY-SA 4.0

store. As a safety net, a remote wipe on an impersonated session is dialled back to account-only, so nobody wipes a personal device by accident.

A rebuilt admin, and DKIM out of the box

- The Admin Web interface has been rebuilt in **TypeScript** — the whole codebase moved over, file by file, to a strictly-typed setup with no JavaScript left behind, for fewer runtime surprises and a console that is far easier to work on and contribute to.
- **DKIM** is now handled for you: generate a per-domain signing key right from the Admin UI and grommunio hands back the public record to paste into DNS — no more fishing around with `rspamadm` and `openssl`. This release also sorted out default and anonymous permissions on public folders and tightened several directory-integration paths. Once your records are published, grommunio's existing DNS health check confirms whether SPF, DKIM, DMARC, MX and the rest line up.

The rest of the suite

- **grommunio Meet** — a big refresh, including a central managed STUN/TURN relay (`turn.grommun.io`) for self-hosters who can't run their own, with a fallback over port 443 that looks like ordinary HTTPS and nearly always gets through; the whole Jitsi stack moved to a current build on JDK 17, and lobby and breakout rooms are on by default. You can still point it at your own coturn.
- **grommunio Files** — up a full generation: fresher, faster and more secure, with built-in collaborative in-browser editing of office documents, grommunio branding and single sign-on, and in-place self-upgrade of older installs. The self-hosted answer to OneDrive and SharePoint.
- **grommunio Keycloak** — moved to Keycloak 26.6.4, the identity provider behind single sign-on across Web, Admin, Files and Meet from one grommunio login; its configuration now lives under `/etc`, so settings survive upgrades.
- **grommunio Archive 1.4** — overhauled for the Leap 16.0 / PHP 8 base and running as a regular service: a web interface for searching and reviewing mail, full-text search underneath, and an SMTP listener that copies messages as they pass through — on top of archiving and retention rules, legal hold, deduplication, fingerprinting and verification, tagging, export/restore, audit logs, direct IMAP import, and bulk import from sources like Google Workspace and Microsoft 365.
- **grommunio Desk 1.2** — spell checking from your system languages, server-view zoom with reset, a right-click context menu (spelling suggestions, copy/paste), isolated per-server sessions, editable server names and a reload option, clearer title-bar icons, automatic settings migration on an updated Electron base, and assorted fixes (URL validation, macOS shortcuts, the server-view loading state, dialog buttons).
- **grommunio DAV** — can now publish the company Global Address List as a read-only CardDAV address book, and remembers per-folder CalDAV/CardDAV settings, so calendar colours and ordering set in Apple Calendar finally stick. **grommunio Index 1.6** made server-side full-text search more configurable and steadier under load.

© grommunio GmbH — CC BY-SA 4.0

A fresh platform: openSUSE Leap 16.0

The whole appliance was rebuilt on openSUSE Leap 16.0 — a current, maintained base with a newer kernel, toolchain and crypto stack. And it is not only the VM image: every guided install target moved to 16.0 — the VMware/OVA appliance, the install ISO, the offline all-in-one ISO for air-gapped sites, and the container/compose stack (now Leap 16.0 with MariaDB 11 and de-privileged). Networking switched to systemd-networkd, and grommunio-setup is now safe to re-run, so roles can be added or removed on a live system without a clean reinstall — on both 15.6 and 16.0.

Debian, properly this time

With 2026.06.1 there are native `.deb` packages for the whole stack — the gromox engine and every front-end — built the proper Debian way. The setup tooling learned to work out what it is running on and do the right thing: write apt sources, import signing keys into the keyring, and drive apt on Debian 13 and Ubuntu 24.04 and 26.04 LTS, right next to openSUSE and RHEL. The packages are reaching general availability deliberately, one component at a time and with selected partners first, so each one is proven before it goes out to everyone.

Documentation, rebuilt

docs.grommunio.com has been rebuilt from the ground up on a modern static-site stack, with fast search across every page and a clean split into User, Administration and Development sections. New material includes a high-availability clustering guide, a Kerberos single sign-on walkthrough for transparent domain-joined Outlook logins, a fuller Exchange-to-grommunio migration guide, and a rewritten container chapter for Leap 16.0.

Supported Distributions

As of 2026.06.1, grommunio supports installation and operation on:

- openSUSE Leap 16.0 / SLES 16 (appliance base)
- Debian 13
- Ubuntu 26.04 LTS

Update

Existing installations update through the usual grommunio update process; see Updating grommunio.

Acknowledgements

A big thank-you to the customers and partners whose feedback shaped this release, and to everyone who builds grommunio out in the open. Join the conversation in the grommunio community.

grommunio 2025.01.2

- Release type: Minor

© grommunio GmbH — CC BY-SA 4.0

- 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

© grommunio GmbH — CC BY-SA 4.0

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.

© grommunio GmbH — CC BY-SA 4.0

- 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.

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).

© grommunio GmbH — CC BY-SA 4.0

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

© grommunio GmbH — CC BY-SA 4.0

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

© grommunio GmbH — CC BY-SA 4.0

- 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 and many others. Your insights drive our roadmap and make grommunio more robust, secure, and performant with each release.

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 were 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

© grommunio GmbH — CC BY-SA 4.0

- 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 `ews_beta=1` in `/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/.

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.

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)

© grommunio GmbH — CC BY-SA 4.0

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.

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

© grommunio GmbH — CC BY-SA 4.0

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 `ews_beta=1` to `/etc/gromox/ews.cfg`
- Activation of `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

© grommunio GmbH — CC BY-SA 4.0

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 planning 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 and many others

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 Desk, 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

© grommunio GmbH — CC BY-SA 4.0

- 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 navigation 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 management 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 and 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.

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)

© grommunio GmbH — CC BY-SA 4.0

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:

© grommunio GmbH — CC BY-SA 4.0

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.

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 GmbH — CC BY-SA 4.0

- 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 GmbH — CC BY-SA 4.0

- 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.

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 GmbH — CC BY-SA 4.0

- 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 mailboxes)
- 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, compatibility, 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 GmbH — CC BY-SA 4.0

- 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.

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 GmbH — CC BY-SA 4.0

- 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.

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:

```
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
```

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

© grommunio GmbH — CC BY-SA 4.0

- 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

> **⚠Caution**
>
> 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 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.

The 2021.08.1 release also brought large advances across every component. The most notable, by area:

- **grommunio Core (gromox)** — full S/MIME and GPG support across MAPI/HTTP, MAPI/RPC and IMAP/POP/SMTP; auto-attach of shared mailboxes via AutoDiscover with full owner permissions; language-independent folder migration mapping; an Exchange (online/on-premise) migration script; expanded multi-value, variable-length property handling; and language-based store creation (mkprivate/mkpublic).
- **grommunio Admin (API & Web)** — organizations and role-based permissions (including read-only roles), public-folder hierarchy and ACLs, LDAP server pooling and import of aliases, fetchmail management, database-stored configuration (dbconf), a live status page, log and mail-queue viewers, per-user sync-policy management, and a redesigned dashboard.
- **grommunio CUI** — a reworked console interface: YaST-based network and timezone configuration, keyboard-layout switching, a log viewer, journald integration, and numerous usability and safety fixes.
- **grommunio Sync** — a 24% lower memory footprint per EAS device and static-analysis hardening.
- **grommunio Setup** — simplified, integrated setup of Files, Meet, Chat and Archive.
- **grommunio Web** — S/MIME key management, integration of Files, Office and Archive, OpenStreetMap maps for contacts, multi-level and prefix search, and broad performance and editor improvements.

The full per-repository commit history for this release is available on GitHub.

© grommunio GmbH — CC BY-SA 4.0

The official documentation covers the necessary steps for the update procedure.

© grommunio GmbH — CC BY-SA 4.0

Roadmap

This roadmap reflects the state of grommunio as of June 2026. Dates and scope are planning targets and may change; see the Disclaimer below.

Release schedule

Release	Type	Planned date	Status
2026.06.1	Major	30 June 2026	Current stable — generally available
2026.06.2	Minor	11 August 2026	Planned — maintenance update
Next major release	Major	6 July 2027	Planned — details to follow

The **current stable** release, 2026.06.1, carries a three-year support lifecycle, with optional extensions available through grommunio's subscription program. The **next minor** release, 2026.06.2, is a maintenance update focused on bugfixes, security updates and smaller features. The **next major** release is planned for 6 July 2027; its feature scope will be announced closer to the date.

Release strategy

grommunio is committed to delivering quality software in a predictable, customer-friendly way. Releases follow two chains:

- **Major releases** (e.g. 2026.06.1, 2025.01.1) — larger feature sets and potential architectural changes.
- **Minor releases** (e.g. 2026.06.2, 2025.01.2) — bugfixes, security updates and smaller features.

grommunio delivers at least one major release per year, with minor (point) releases in between.

Supported distributions

As of 2026.06.1, grommunio is supported on the following platforms:

Platform	Notes
grommunio Appliance — openSUSE Leap 16.0 base	Current appliance
grommunio Appliance — openSUSE Leap 15.6 base	Supported until 1 August 2026
openSUSE Leap 16.0	
SUSE Linux Enterprise Server (SLES) 16	
Debian 13	

© grommunio GmbH — CC BY-SA 4.0

Platform	Notes
Ubuntu 26.04 LTS	

Extended Support Contracts (ESC)

Additional platforms are available to customers and partners under an **Extended Support Contract**:

Platform	Availability
Enterprise Linux 10 (EL10)	Extended Support Contract
Debian 12	Extended Support Contract
Ubuntu 24.04 LTS	Extended Support Contract

Extended Support Contracts are arranged on an individual basis through grommunio Professional Services, with a dedicated Project Manager scoping and coordinating the engagement. To discuss an ESC, contact your grommunio partner or grommunio.

Disclaimer

While grommunio strives for the highest level of transparency, this roadmap is subject to change based on factors such as customer demand, technological developments and community response. It is prepared with great care, but changes may occur and are communicated through grommunio's established channels — news sections, newsletters and social networks.

© grommunio GmbH — CC BY-SA 4.0