grommunio Migration Guide

Official documentation

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

grommunio Migration Docs

grommunio is a comprehensive communication and collaboration solution that includes e-mail, calendaring, contacts, tasks, notes, video meetings, chat and file management.

Audience

This document is for administrators working with grommunio. It helps them to execute migrations from various sources to grommunio. This document is primarily focused on the migration in the sense of data migration from other sources, rather than migration in terms of configuration.

Offline reading

This section is also available to download for offline use:

- Download as PDF
- Download as EPUB

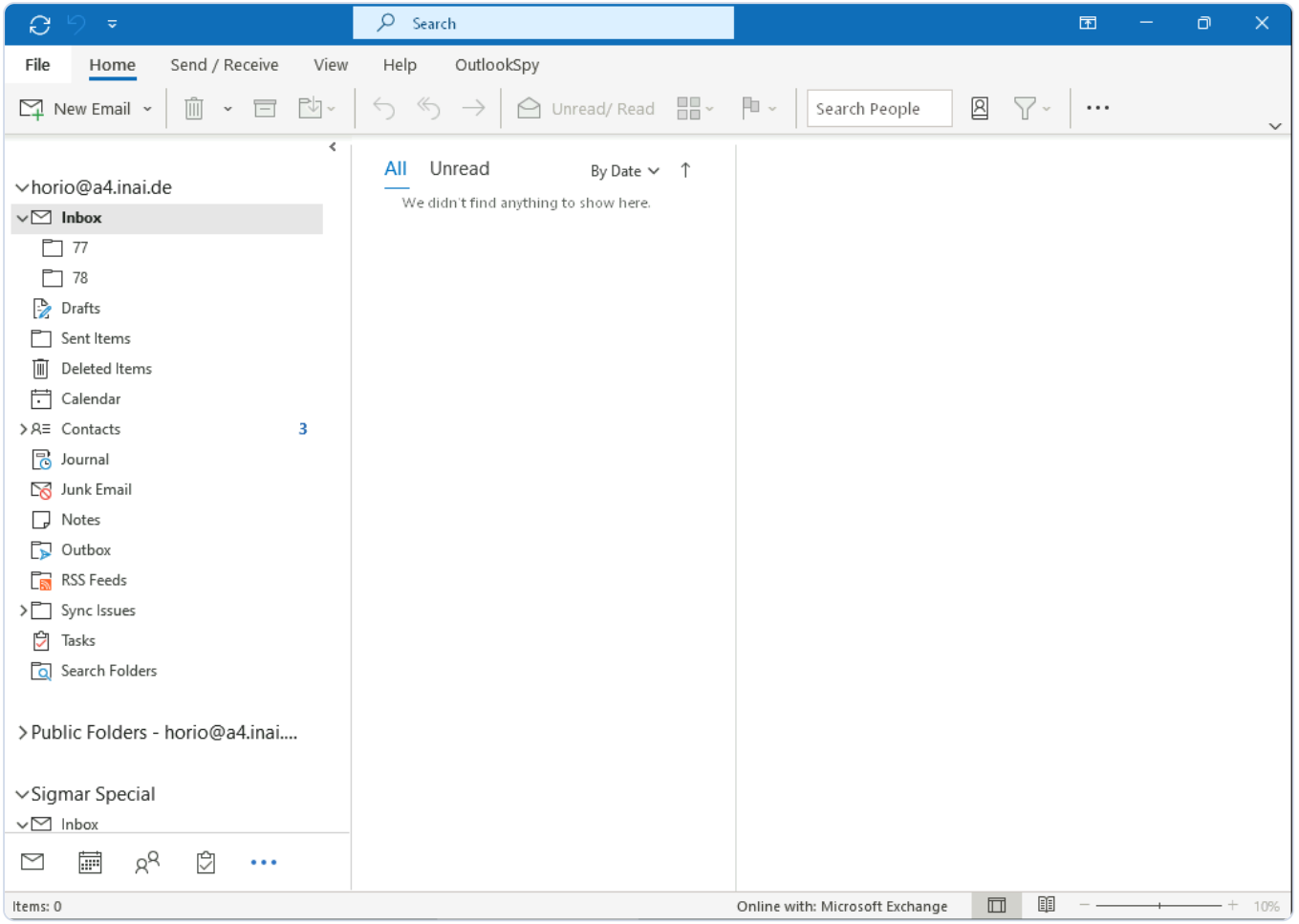
Microsoft Exchange

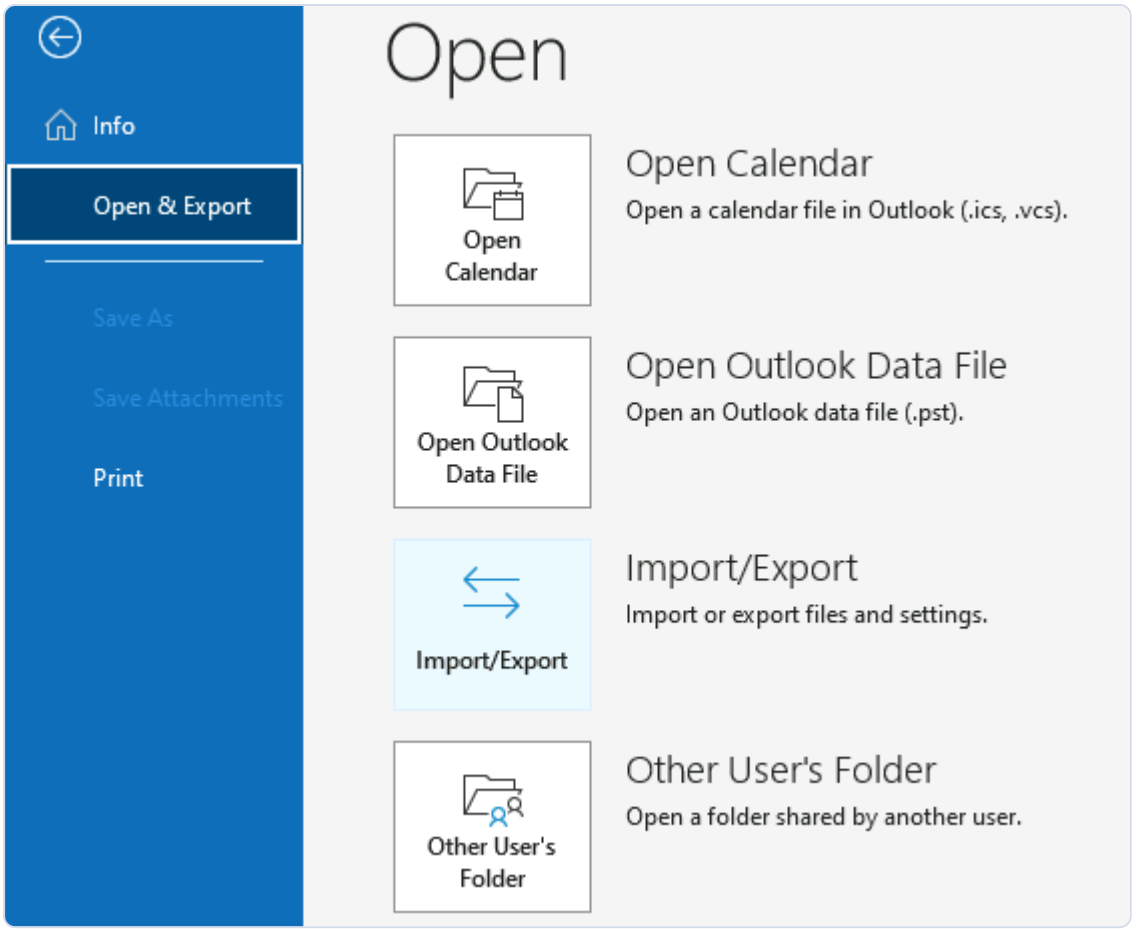
PFF (cf. summary from the Forensics Wiki) is a format exportable from Outlook and Exchange. Outlook makes use of this format for different scenarios, and calls them different names (*.pst*, *.ost*), but it is just one file type.

- *.pst* files can be generated with Outlook interactively
- *.ost* files can be taken from *C:Users...*
- *.pst* files can be also generated from an Exchange Server's PowerShell in a mostly unattended fashion

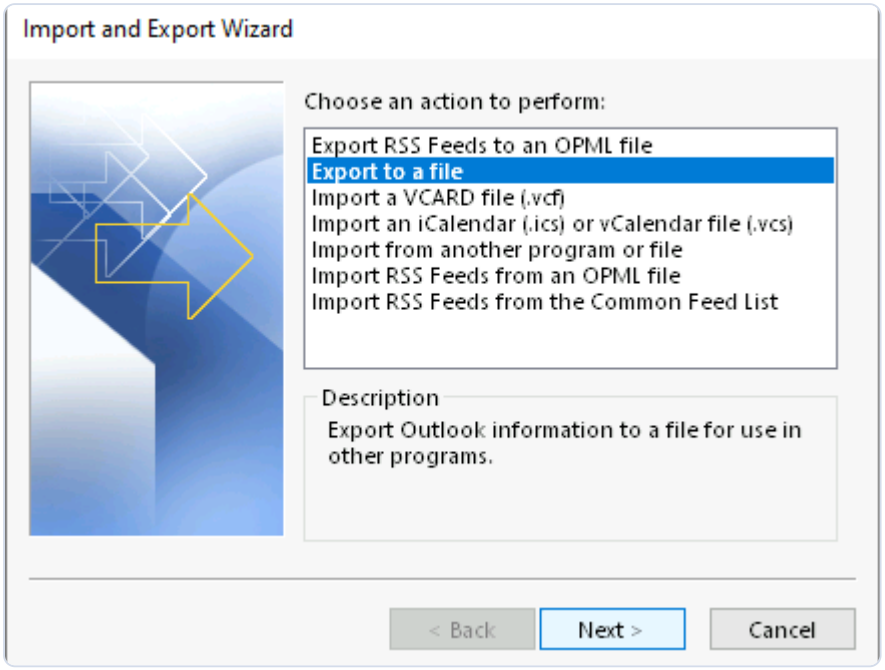
Outlook interactive export

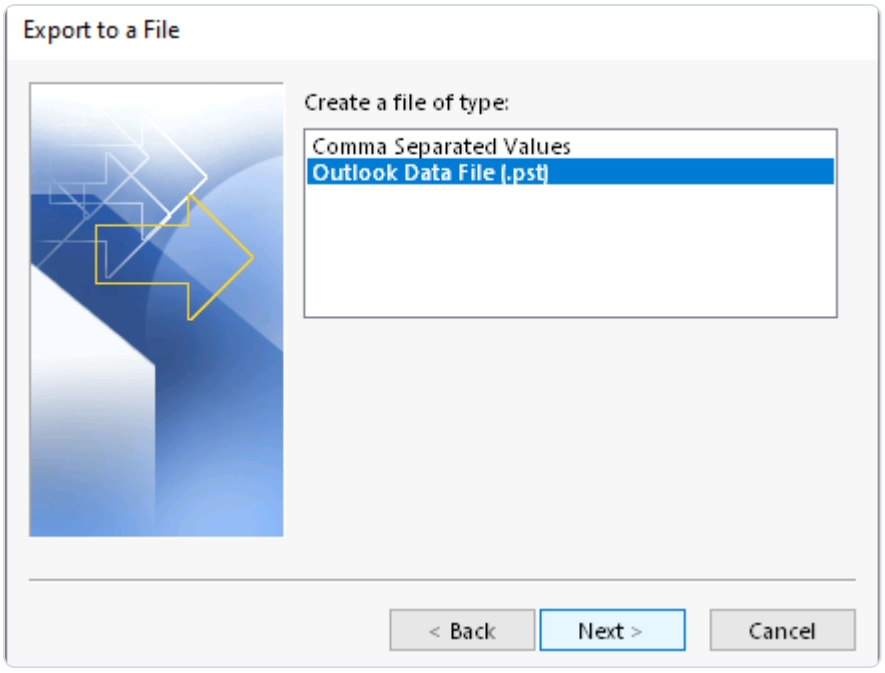
Once the Outlook main window is open, go to “File”, “Open & Export”, “Import/Export”:

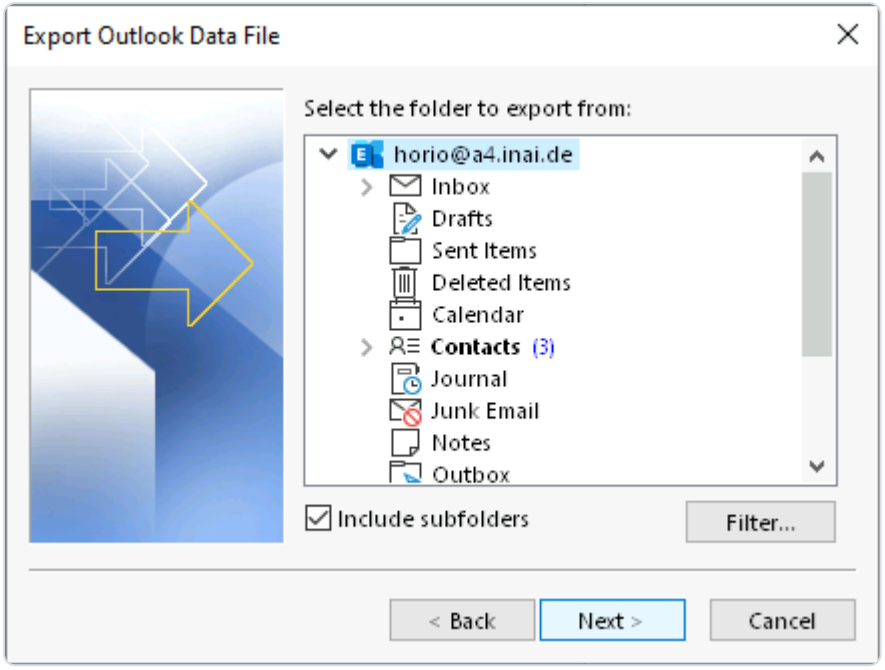


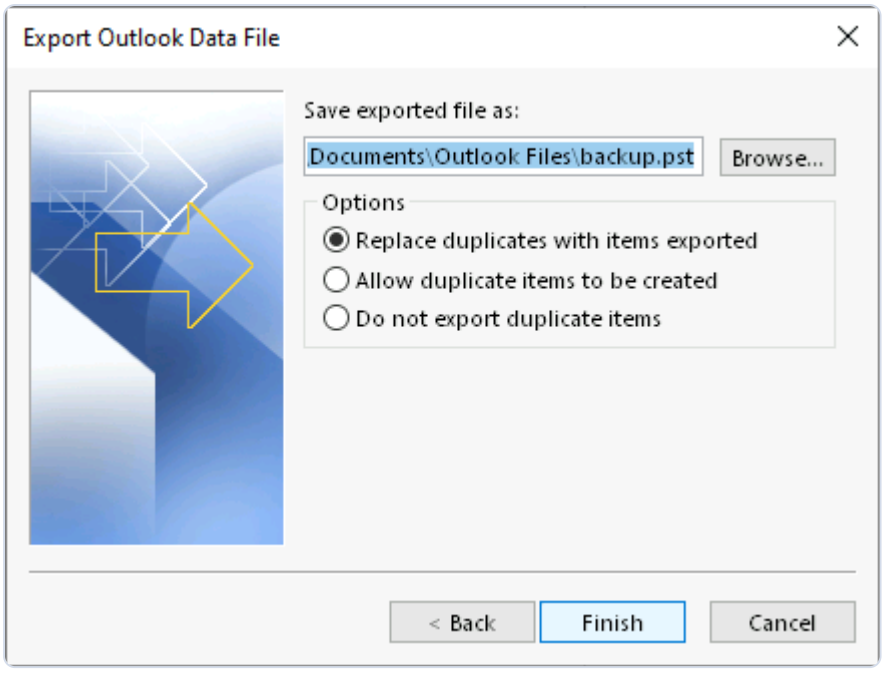


Then follow the usual dialog chain.









⚠Caution

Before attempting to copy PFF files, ensure the file(s) is/are not open anywhere anymore. Even after closing Outlook, Outlook may still execute in the background for some seconds, *in particular* when the MAPI profile used Exchange *Cached Mode*. Various failure modes trying to access active PFF files have been observed, such as:

1. Under the `cmd.exe` shell, the command `type stillactive.pst >new.pst` produces new.pst with just 512 bytes before aborting with the message `The process cannot access the file because another process has locked a portion of the file`.

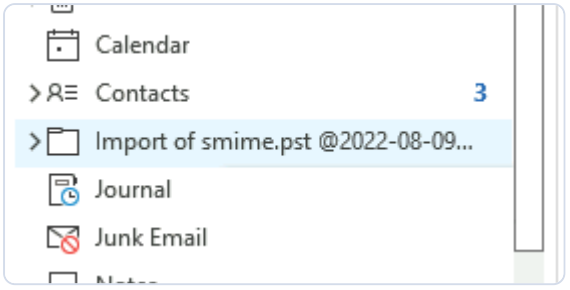
2. Under the `cmd.exe` shell, the command `scp stillactive.pst a@b.com:` can produce the file on the target, but all bytes are ASCII NUL bytes. (So observed with Powershell-OpenSSH v8.x; fixed in 9.x). A log message `Domain error` is output by scp.

3. PFF files contain a CRC-32 checksum, which can readily change while the file is in use. Attempts to read the file from underneath Windows (e.g. at the storage or hardware level), or attempting to use a PFF file that was not cleanly closed may result in gromox-pff2mt rejecting the input.

gromox-pff2mt import

On the grommunio system, PFF files can be imported on the command-line with gromox-pff2mt and gromox-mt2exm. These are two commands meant to be chained together by way of a pipe; tend to the linked manual pages to read about the invocation syntax.

```
08:23 a4:~ $ gromox-pff2mt ../pst/smime.pst | gromox-mt2exm -u horio@a4.inai.de
pff: Reading ../pst/smime.pst...
pff: Building list of named properties...
pff: Processing "Outlook-Datendatei"...
pff: Processing ""...
pff: Processing "SPAM Search Folder 2"...
pff: Processing "Oberste Ebene der Outlook-Datendatei"...
pff: Processing "Gelöschte Elemente"...
pff: Processing "Outbox"...
pff: Processing "ESET Antispam"...
pff: Processing "Suchpfad"...
pff: Processing "Suchpfad"...
pff: Processing "Outlook-Datendatei($686167db)/0/Outbox"...
pff: Processing "ESET Antispam"...
08:23 a4:~ $
```



Automated bulk migration with `exchange2grommunio.ps1`

For migrating many mailboxes, grommunio ships a PowerShell script — `exchange2grommunio.ps1` — that automates the whole flow end to end. For each Exchange mailbox it exports a `.pst`, makes it available to the grommunio server over a shared folder, and imports it there via SSH — optionally creating the grommunio mailboxes as well.

How it works

For every selected mailbox the script:

1. Exports the mailbox to a `.pst` with `New-MailboxExportRequest` onto a shared folder that the Exchange subsystem can write to.
2. Reaches the grommunio server over SSH (using `plink.exe`), where the same folder is mounted (`mount.cifs`), and imports the `.pst`.
3. Optionally deletes the `.pst` afterwards to save space, and records the result.

Successful imports are written to `exchange2grommunio.done` and failures to `exchange2grommunio.failed`, so a run can be resumed or retried.

Prerequisites

- grommunio is attached to the **same LDAP/AD** as Exchange, so users resolve on both sides (test LDAP before importing).
- The script is run from an **elevated Exchange Management Shell** (Windows Server 2012 R2 or newer, PowerShell 2.0+; 3.0+ is recommended so Linux command output is captured in the logs).
- `plink.exe` (from PuTTY) sits in the same directory as the script; optionally `pageant.exe` for SSH public-key authentication.
- A **shared folder** (UNC path) that the Exchange subsystem can write to and that the grommunio server can mount (`cifs-utils` installed), with enough space for the `.pst` files.
- SSH access to the grommunio server, with its host key already accepted by `plink.exe`.

Key settings

All settings live at the top of the script (or a separate configuration file). The most important ones:

Variable	Purpose
`$GrommunioServer`	FQDN of the target grommunio server
`$WinSharedFolder` / `$LinuxSharedFolder`	The shared `.pst` folder, as seen from Windows (UNC) and from Linux (mount point)
`$LinuxUser`	Shell user on the grommunio side (e.g. `root`)
`$LinuxUserPWD` / `$LinuxUserSSHKey` / `$UsePageant`	SSH auth: password, private key (`.ppk`), or key via Pageant
`$WindowsUser` / `$WindowsPassword`	Windows account used to mount the share on Linux
`$AutoMount`	Mount the Windows share on the grommunio server automatically
`$DeletePST`	Delete each `.pst` after a successful import
`$MailboxLanguage`	Language for newly created mailboxes (see `/usr/share/grommunio-admin-api/res/storelangs.json`)
`$Organization`	`-o <id>` when created mailboxes have aliases across domains
`$MigrationPriority`	`New-MailboxExportRequest` priority (`Normal` is usually faster than `High`)
`$LogFile`	Summary log; per-mailbox export/import logs go to `<share>\logs\`

Selecting which mailboxes to migrate

- **Allow-list:** set `$ImportMboxes` (an array of addresses) or, if it is empty, the script reads addresses from the `exchange2grommunio.import` file on the share. Leaving both empty migrates **all** mailboxes.
- **Deny-list:** `$IgnoreMboxes` / the `exchange2grommunio.ignore` file exclude mailboxes (always honored).

> 🚀 **Generate the import list from grommunio**
>
> You can produce the list of existing grommunio mailboxes directly:
>
> ```
> gromox-mbop foreach.mb echo-username > /mnt/pst/exchange2grommunio.import
> ```
>
> To re-run only the failures, copy the failed list over the import list (`cp exchange2grommunio.failed exchange2grommunio.import`).

Creating mailboxes during migration

Two switches decide whether the script also provisions the grommunio mailboxes (this requires a working LDAP configuration):

`$CreateGrommunioMailbox`	`$OnlyCreateGrommunioMailbox`	**Behaviour**
`$false`	`$false`	Migrate into **existing** mailboxes, one by one
`$true`	`$false`	**Create then migrate** each mailbox in turn
`$true`	`$true`	**Two-pass**: create all mailboxes first, then run again with both set to `$false` to migrate the data

The two-pass mode is recommended for large migrations: users can start working in grommunio immediately with empty mailboxes (new mail arrives normally) while the historical data is imported in the background.

Running it

Launch the script from the elevated Exchange Management Shell:

```
.\exchange2grommunio.ps1
```

- For an **unattended** run, set `$WaitAfterImport = $false` and `$StopOnError = $false` so it does not block waiting for the administrator on an error.
- To **stop gracefully**, create the stop-marker file (`exchange2grommunio.STOP`) on the share — the script finishes the current mailbox and then exits.

The underlying import on the grommunio side uses the same tooling as the manual path (gromox-pff2mt → gromox-mt2exm), so the import notes above apply equally.

Generic Migration

This chapter covers overall migration to grommunio with generic and standardized protocols. These instructions are intentionally named `generic`, as these migration scenarios apply to multiple providers, installations and other communication software installations.

Individual emails

With the gromox-eml2mt, gromox-ical2mt, gromox-vcf2mt and gromox-mt2exm command-line utilities, grommunio has utilities with which individual emails, calendars or contact card files can be read and imported. Tend to the linked manual pages to read about the invocation syntax.

Migration via IMAP

As a user, you can do IMAP-to-IMAP transfers. This can be done interactively with a MUA such as Thunderbird or Alpine by having both the original and the Gromox IMAP accounts added and moving mails. Alternatively, the *imapsync* command-line utility may be used to do so non-interactively.

Kopano

Migrating Kopano is a multi-step process which also depends on the configuration of the backend used by Kopano.

If Kopano uses LDAP, the high-level view of the migration is as follows:

- Configure grommunio appropriately to LDAP (settings user filters, etc.)
- Create stores in grommunio
- Migrating user data (which this article covers mainly)
- Switch mail-routing

This migration focusses mainly on the migration of the dataset and does not imply an active LDAP configuration.

> **⚠Caution**
>
> This guide is not conclusive and is provided for convenience reasons. There might be aspects of your Kopano installation which is not covered by this manual. Please refer to your partner or feel free to contact us, specifically Professional Services for extended inquiries if you are missing something relevant to your migration.

Preparation

When migrating Kopano, being well-prepared matters. For this to happen, we need to make sure that the relevant metadata for migration is prepared, ideally in a list format which we can use to create our users in the grommunio installation:

```
kopano-admin -l | sed -e '1,4d' -e '/^$/d' | awk '{ print $1 }' | sort | while read user;
```

This (long) command executed on the Kopano system will create us a list of users with the important metadata of users which we will require in a format which can be used for further scripting.

With this list now, we can create the used domains in grommunio:

```
MAX_USERS_DOMAIN=250

cat kopano-users.txt | awk -F\; '{ print $3 }' | awk -F@ '{ print $2 }' | sort | uniq | se
    grommunio-admin domain create -u ${MAX_USERS_DOMAIN} $DOMAIN
done
```

On the grommunio system, Kopano databases can be imported on the command-line with gromox-kdb2mt and gromox-mt2exm. These are two commands meant to be chained together by way of a pipe; tend to the linked manual pages to read about the invocation syntax.