grommunio Man Pages

Official documentation

Version 2026.07.02 · docs.grommunio.com

© grommunio GmbH — CC BY-SA 4.0

grommunio Man Pages

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

Audience

This collection of man pages is made available for easy browsing online. They are also present in an installed system. The manpages are for system administrators operating grommunio and who wish to look up topics by name of system process name, filename or concept. Manual pages focus on giving an exhaustive lists of options in an almost formal description.

It is recommended for readers to have at least the following recommended experience to install and operate grommunio:

- Understanding of the technologies and standards: Linux, e-mail (e.g. SMTP), Networking, DNS.
- General concept of the Linux operating system.
- Overall experience with management of 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

gromox(7)

Name

gromox — Overview of the Gromox groupware server

Description

Gromox is a groupware server capable of serving as a replacement for Microsoft Exchange. Connectivity options include RPC/HTTP (Outlook Anywhere), IMAP, POP3, an SMTP-speaking LDA, and a PHP module with a Z-MAPI function subset.

Gromox relies on other components to provide a sensibly complete mail system, such as Postfix as a mail transfer agent, and grommunio-admin for user management. A web interface is available with grommunio-web. The grommunio distribution ships these essentials and has a ready-to-run installation of Gromox. system.

Manual page listing

Gromox documentation consists of at least a dozen manual pages ("manpages") on its individual components. We have grouped these according to their principal function.

Overview and definitions

- gromox(7) — This page, an overview of the Gromox groupware server.
- mapi(7gx) — Definition for "Messaging Application Programming Interface"
- gromox-selinux(5) — SELinux policy for Gromox

Information Store subsystem

- autoconfig(7) — AutoConfig protocols
- autodiscover(4gx) — AutoDiscover HTTP Service Protocol handler (responder).
- autodiscover(7) — AutoDiscover protocols
- exmdb_provider(4gx) — Gromox Information Store
- http(8gx) — Protocol handler for HTTP and RPCH
- istore(8gx) — Information Store launcher
- mod_cache(4gx) — Handler for serving objects from a local filesystem

© grommunio GmbH — CC BY-SA 4.0

- mod_fastcgi(4gx) — Handler for proxying requests to FastCGI servers
- mod_rewrite(4gx) — Handler for altering HTTP request URIs before processing
- timer(8gx) — deferred command executor

Exchange subsystem and its components

- exchange_emsmdb(4gx) — Handler for the Wire Format Protocol (Outlook/Exchange RPCs) and Remote Operations Protocol
- exchange_nsp(4gx) — Handler for the Name Service Provider Interface Protocol
- exchange_rfr(4gx) — Handler for the Address Book Name Service Provider Interface Referral Protocol
- ews(4gx) — Handler for Exchange Web Services requests
- mh_emsmdb(4gx) — Handler for MAPIHTTP-wrapped EMSMDB requests
- mh_nsp(4gx) — Handler for MAPIHTTP-wrapped NSPI requests

PHP-MAPI subsystem

- zcore(8gx) — Bridge for PHP-MAPI requests

Mail retrieval agent subsystem (MRA)

- imap(8gx) — IMAP server
- event_proxy(4gx) — Event sender
- event_stub(4gx) — Event receiver
- midb_agent(4gx) — Client for midb(8gx)
- pop3(8gx) — POP3 server

Local delivery agent (LDA)

- alias_resolve(4gx) — Alias resolution for delivery(8gx) using MySQL
- delivery(8gx) — Backend for local delivery
- delivery-queue(8gx) — LMTP/SMTP frontend for local delivery
- user_filter(4gx) — User logon limiter

Auxiliary services

- gromox-cleaner.service(8) — Soft-deleted message/attachment removal

© grommunio GmbH — CC BY-SA 4.0

- pam_gromox(4gx) — a PAM plugin to authenticate with Gromox
- event(8gx) — Folder change notification daemon
- midb(8gx) — Message Index database daemon

System administration

- gromox-abktconv(8) — Utility for converting between ABKT and JSON
- gromox-abktpull(8) — Utility to extract ABKT templates from LDIF
- gromox-compress(8) — Utility to recompress Gromox content files
- gromox-dbop(8) — User database maintenance utility
- gromox-dscli(8) — Autodiscover command line utility
- gromox-mailq(8) — SMTP queue lister
- gromox-mbck(8) — Mailbox check and repair utility
- gromox-mbop(8) — Mailbox operations utility
- gromox-mbsize(8) — Mailbox size analysis
- gromox-mkmidb(8) — Tool for creating a blank message index database
- gromox-mkprivate(8) — Tool for creating a blank private store
- gromox-mkpublic(8) — Tool for creating a blank public store

Mail import, export and conversion

- gromox-eml2mbox(8) — Utility for converting RFC5322 Internet Mail messages into a RFC4155 mbox-format mailbox
- gromox-import(8) — Utility for analysis of/importing messages from a Gromox Message Transfer datastream
- gromox-export(8) — Utility for exporting messages
- gromox-exm2eml(8) — Wrapper for gromox-export selecting RFC5322 Internet Mail output
- gromox-exm2ical(8) — Wrapper for gromox-export selecting RFC5545/5546 iCalendar output
- gromox-exm2mt(8) — Wrapper for gromox-export selecting Gromox Mailbox Transfer stream output
- gromox-exm2tnef(8) — Wrapper for gromox-export selecting TNEF output
- gromox-exm2vcf(8) — Wrapper for gromox-export selecting RFC6540 vCard output

© grommunio GmbH — CC BY-SA 4.0

- gromox-ical2mt(8) — Utility for analysis of/importing RFC5545/5546 iCalendar objects
- gromox-kdb2mt(8) — Utility for analysis of/importing Zarafa/Kopano SQL-stored mailboxes
- gromox-oxm2mt(8) — Utility for analysis and import of Outlook .msg files
- gromox-pff2mt(8) — Utility for analysis/import of PFF/PST/OST files
- gromox-snapshot(8) — Helper to create btrfs snapshots of mailboxes
- gromox-tnef2mt(8) — Utility for analysis/import of MS-OXTNEF objects
- gromox-vcf2mt(8) — Utility for analysis/import of vCard objects
- kdb-uidextract(8) — Helper for creating a gromox-kdb2mt ACL map
- kdb-uidextract-limited(8) — Helper for creating a gromox-kdb2mt ACL map

Components

- authmgr(4gx) — Demultiplexer for authentication requests
- dnsbl_filter(4gx) — DNS Blacklist filtering
- ldap_adaptor(4gx) — LDAP connector for authentication
- mysql_adaptor(4gx) — MySQL/MariaDB connector for user metadata and authentication
- timer_agent(4gx) — Client for timer(8gx)

Historic commands that have been removed

The following is a list of programs that no longer exist. It is intended solely to capture keyword searches within the documentation for said obsolete/removed commands.

- gromox-mt2exm: renamed to gromox-import

Language bindings

- mapi(4gx) — PHP module providing MAPI functions

Configuration files

Program configuration files reside within /etc/gromox. The format for .cfg files is: one "key=value" pair per line. Empty lines are ignored, as are lines beginning with a '#' character. Lines can have a maximum length of 1024. Each key=value line is logically split at the equals sign, and whitespace is trimmed around key and value. Comments at the end of a value are not supported. Escape sequences are not supported. Each config key is one of four types of how the config file reader should apply an early generalized transform:

© grommunio GmbH — CC BY-SA 4.0

- A quantity, e.g. bytes for a buffer, a limit on the number of users, etc. With 1-unit granularity. See section "Quantity syntax" below.
- A time span, e.g. a duration or maximum lifetime for some object, with 1s or 1ns granularity (whatever the program code wants for any particular key). See section "Time span syntax" below.
- Boolean value. The empty string, the strings "no", "off", "false", and a string that completely parses as "0" via strtoul, is interpreted as false, with everything else considered true.
- Untransformed string.

The format for .ini files is: one "key=value" pair per line. Empty lines are ignored, as are lines beginning with a ';' character. Only used by some ailing PHP code. No generalized transforms.

Many programs have a **config_file_path** directive with which the search path for further config files can be specified. For example, http(8gx) defaults to config_file_path=/etc/gromox/http:/etc/gromox, so the mysql_adaptor(4gx) component as loaded by http will first try /etc/gromox/http/mysql_adaptor.cfg, then /etc/gromox/mysql_adaptor.cfg. This allows having one file that is shared between multiple programs as well as being able to override on a per program-basis.

Listing of config files per component

A list of components and the config files they potentially use.

- alias_resolve(4gx) inside delivery(8gx): /etc/gromox/alias_resolve.cfg, /etc/gromox/mysql_adaptor.cfg
- authmgr(4gx) inside delivery(8gx), delivery-queue(8gx), http(8gx), imap(8gx), midb(8gx), pam_gromox(4gx), pop3(8gx), zcore(8gx): /etc/gromox/gromox.cfg
- autodiscover(4gx) inside php-fpm(8): /etc/gromox/autodiscover.ini, /etc/gromox/mysql_adaptor.cfg
- delivery(8gx): /etc/gromox/alias_resolve.cfg, /etc/gromox/exmdb_local.cfg, /etc/gromox/ldap_adaptor.cfg, /etc/gromox/mlist_expand.cfg, /etc/gromox/mysql_adaptor.cfg
- delivery-queue(8gx): /etc/gromox/gromox.cfg, /etc/gromox/midb_agent.cfg, /etc/gromox/ldap_adaptor.cfg, etc/gromox/mysql_adaptor.cfg
- event(8gx): /etc/gromox/event.cfg
- exchange_emsmdb(4gx) inside http(8gx): /etc/gromox/exchange_emsmdb.cfg
- exchange_nsp(4gx) inside http(8gx): /etc/gromox/exchange_nsp.cfg
- exchange_rfr(4gx) inside http(8gx): no config file
- exmdb_provider(4gx) inside http(8gx): /etc/gromox/exmdb_provider.cfg

© grommunio GmbH — CC BY-SA 4.0

- http(8gx): /etc/gromox/cache.txt, /etc/gromox/exchange_emsmdb.cfg, /etc/gromox/exchange_nsp.cfg, etc/gromox/exmdb_provider.cfg, /etc/gromox/fastcgi.txt, /etc/gromox/rewrite.txt
- imap(8gx): /etc/gromox/event_proxy.cfg, etc/gromox/event_stub.cfg, /etc/gromox/gromox.cfg, /etc/gromox/imap.cfg, etc/gromox/ldap_adaptor.cfg, /etc/gromox/mysql_adaptor.cfg
- midb_agent(4gx) inside delivery-queue(8gx), imap(8gx), pop3(8gx): /etc/gromox/midb_agent.cfg
- mlist_expand(4gx) inside delivery(8gx): /etc/gromox/mlist_expand.cfg
- mod_cache(4gx) inside http(8gx): /etc/gromox/http.cfg, /etc/gromox/cache.txt
- mod_fastcgi(4gx) inside http(8gx): /etc/gromox/http.cfg, /etc/gromox/fastcgi.txt
- mod_rewrite(4gx) inside http(8gx): /etc/gromox/http.cfg, /etc/gromox/rewrite.txt
- mh_emsmdb(4gx) inside http(8gx): no config file
- mh_nsp(4gx) inside http(8gx): no config file
- pop3(8gx): /etc/gromox/authmgr.cfg, /etc/gromox/event_proxy.cfg, /etc/gromox/imap.cfg, /etc/gromox/ldap_adaptor.cfg, /etc/gromox/mysql_adaptor.cfg
- timer(8gx): /etc/gromox/timer.cfg
- timer_agent(4gx) inside http(8gx), zcore(8gx): /etc/gromox/timer_agent.cfg
- user_filter(4gx) inside http(8gx), imap(8gx), pop3(8gx): /etc/gromox/gromox.cfg
- zcore(8gx): /etc/gromox/authmgr.cfg, /etc/gromox/zcore.cfg, /etc/gromox/ldap_adaptor.cfg, /etc/gromox/mysql_adaptor.cfg, /etc/gromox/timer_agent.cfg

Listing of components per config file

- /etc/gromox/alias_resolve.cfg: used by the alias_resolve(4gx) component, accessed process-wise by the delivery(8gx) process.
- /etc/gromox/authmgr.cfg: used by the authmgr(4gx) and pam_gromox(4gx) components, accessed process-wise by delivery(8gx), delivery-queue(8gx), http(8gx), imap(8gx), midb(8gx), pop3(8gx), zcore(8gx), and arbitrary PAM applications.
- /etc/gromox/autodiscover.ini: used by the autodiscover(4gx) component, accessed process-wise by php-fpm(8).
- /etc/gromox/event.cfg: used by the event(8gx) process.
- /etc/gromox/event_proxy.cfg: used by the event_proxy(4gx) component, accessed process-wise by imap(8gx), midb(8gx), pop3(8gx).

© grommunio GmbH — CC BY-SA 4.0

- /etc/gromox/event_stub.cfg: used by the event_stub(4gx) component, accessed process-wise by imap(8gx).
- /etc/gromox/exchange_emsmdb.cfg: used by the exchange_emsmdb(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/exchange_nsp.cfg: used by the exchange_nsp(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/exmdb_local.cfg: used by the exmdb_local(4gx) component, accessed process-wise by delivery(8gx).
- /etc/gromox/exmdb_provider.cfg: used by the exmdb_provider(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/gromox.cfg: An effort to consolidate all the invididual .cfg files you see around here. This is a work-in-progress. See the gromox.cfg(5) manpage.
- /etc/gromox/http.cfg: used by the mod_cache(4gx), mod_fastcgi(4gx), mod_rewrite(4gx) components, and the http(8gx) process.
- /etc/gromox/imap.cfg: used by the imap(8gx) process.
- /etc/gromox/ldap_adaptor.cfg: used by the ldap_adaptor(4gx) component, accessed process-wise by delivery(8gx), delivery-queue(8gx), http(8gx), imap(8gx), midb(8gx), pop3(8gx), zcore(8gx), and arbitrary PAM applications.
- /etc/gromox/midb_agent.cfg: used by the midb_agent(4gx) component, accessed process-wise by delivery-queue(8gx), imap(8gx), pop3(8gx).
- /etc/gromox/mlist_expand.cfg: used by the mlist_expand(4gx) component, accessed process-wise by delivery(8gx).
- /etc/gromox/mysql_adaptor.cfg: used by the alias_resolve(4gx), mysql_adaptor(4gx) components, accessed process-wise by delivery(8gx), delivery-queue(8gx), http(8gx), imap(8gx), midb(8gx), pop3(8gx), zcore(8gx), and arbitrary PAM applications.
- /etc/gromox/midb.cfg: used by the midb(8gx) process.
- /etc/gromox/mod_cache.txt: used by the mod_cache(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/mod_fastcgi.txt: used by the mod_fastcgi(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/mod_rewrite.txt: used by the mod_rewrite(4gx) component, accessed process-wise by http(8gx).
- /etc/gromox/mysql_adaptor.cfg: used by the autodiscover(4gx) component, http(8gx), imap(8gx), pop3(8gx), zcore(8gx) processes.

© grommunio GmbH — CC BY-SA 4.0

- /etc/gromox/pam.cfg: used by the pam_gromox(4gx) component, accessed process-wise by arbitrary PAM applications.
- /etc/gromox/pop3.cfg: used by the pop3(8gx) process.
- /etc/gromox/timer.cfg: used by the timer(8gx) process.
- /etc/gromox/timer_agent.cfg: used by the timer_agent(4gx) component, accessed process-wise by http(8gx), zcore(8gx).
- /etc/gromox/zcore.cfg: used by the zcore(8gx) process.

Databases

- User information is held in a MariaDB/MySQL database. This database can be accessed by multiple Gromox servers, and so enables distributed Gromox operation. The MariaDB system itself provides the necessary utilities for distributing or replicating this database.
- Per-user e-mail messages are stored in a SQLite database (e.g. /var/lib/gromox/user/m1/1/1/exchange.sqlite3), as is a message index (e.g. /var/lib/gromox/user/m1/1/1/midb.sqlite3). These are normally only used by one system, but can be shared through network filesystems provided that file locking is properly implemented in the filesystem driver. Normal file mechanisms can be used to backup or transfer the database to another Gromox host.

Host addresses

Gromox exclusively uses the AF_INET6 socket family with the Berkeley/BSD/POSIX socket API. What this means is that, whenever an IP address is logged, it will have the form specified in RFC 4291 §2.2/§2.5.5. Furthermore, whenever some configuration file directive (field) requires an IP address (i.e. you cannot or do not want to use a hostname), the RFC 4291 form **must** be used for both IPv6 and IPv4.

Listening sockets

- /run/gromox/zcore.sock — zcore(8gx)
- *:24 — delivery-queue(8gx) LMTP/SMTP service (when Postfix is on 25)
- :*25 — Normally, your own MTA (postfix(1), exim(8), whatever the case may be). delivery-queue(8gx) will only be on 25 in developer setups that wish to cut and skip Postfix/etc. to get a simpler test setup.
- *:80 — http(8gx) HTTP service
- *:110 — pop3(8gx) POP3 service

© grommunio GmbH — CC BY-SA 4.0

- *:143 — imap(8gx) IMAP service
- *:443 — http(8gx) HTTP over implicit TLS
- *:993 — imap(8gx) IMAP over implicit TLS
- *:995 — pop3(8gx) POP3 over implicit TLS
- [::1]:5000 — exmdb_provider(4gx) component inside http(8gx)
- [::1]:5555 — midb(8gx) service
- [::1]:6666 — timer(8gx) service
- [::1]:33333 — event(8gx) service

Files

- /usr/share/gromox/cpid.txt: mapping between character set IDs and names
- /usr/share/gromox/folder_names.txt: Translations for essential folders in a message store.
- /usr/share/gromox/lang_charset.txt: mapping from language code to character set
- /usr/share/gromox/lcid.txt: mapping between locale IDs and names
- /usr/share/gromox/mime_extension.txt: mapping between file extensions and MIME types (if you also have /etc/mime.types, it will be used too, but it is optional)
- /var/lib/gromox: basic root directory of all variadic data for Gromox
- /var/lib/gromox/queue: directory for delivery-queue(8) temporary files
- /var/lib/gromox/user/account@domain: individual mailbox container
  The directory name/path has only few requirements. The users.maildir column in the gromox MariaDB/MySQL database needs to reflect that location. Some user management tools generate extra directory levels, e.g. /user/m1/1/0.
- .../a@d/exmdb/exchange.sqlite3: mail store with almost everything (no mail bodies)
- .../a@d/cid/: attachments and message bodies (PR_BODY, PR_HTML, PR_RTF_COMPRESSED).
- .../a@d/eml/*mid_string*: RFC5322 representation for a message.
  mid_string has no required form. Typically, there is *timestamp.seqid.hostname* which represents EMLs captured by delivery(8gx) on ingestion, and *timestamp.seqid*.midb for EMLs generated by midb(8gx) out of MAPI messages.
- .../a@d/ext/*mid_string*: Digest for the RFC5322 file.
  This JSON-encoded file contains e.g. indexing information for individual MIME parts of the RFC5322 representation. Generated by midb(8gx).

© grommunio GmbH — CC BY-SA 4.0

fail2ban integration

Daemons emit a mostly consistent log messages on authentication failures that can be matched with (PCRE):

/rhost=\(\S+)?\\S* user=(\S+) .*(auth|login.*|logon) rejected:/

Operation texts can be "HTTP auth rejected" (http), "zs_logon rejected" (zcore), "zs_logon_token rejected" (zcore), "LOGIN phase0 rejected" (imap), "LOGIN phase1 rejected" (imap), "LOGIN phase2 rejeceted" (imap), "login rejected" (pop3).

Quantity syntax

The syntax for strings that are parsed to obtain some kind of number may have one SI unit prefix letter affixed to the number to select a power-of-1024(!) multiplier. Formally,

quantum := number [letter]

"number" can be whatever strtol(3) accepts. If a period is detected, parsing switches to strtod(3).

letter := 'k' | 'M' | 'G' | 'T' | 'P' | 'E' | 'Z' | 'Y' | 'R' | 'Q'

Examples: *4k* is 4*1024^1 = *4096*, *1.5M* is 1.5*1024^2 = 1572864.

Note: This syntax specifies no unit, nor unit symbols. Whether "4k" means 4096 bytes, 4096 nibbles or 4096 users depends on the enclosing context (e.g. configuration directive) that uses this syntax.

Time span syntax

The syntax for strings that are to be interpreted as a duration, period/interval must be of the form:

duration := quantum [quantum]*

quantum := number unit

"number" can be whatever strtol(3) accepts. If a period is detected, parsing switches to strtod(3).

unit := "ns" | "nsec" | "µs" | "µsec" | "ms" | "msec" | "s" | "sec" | "second" | "seconds" | "min" | "minute" | "minutes" | "h" | "hour" | "hours" | "d" | "day" | "days" | "week" | "weeks" | "month" | "months" | "y" | "year" | "years"

Whitespace is ignored whereever it appears (so use as much as you need). Quanta with the same unit may be used; they are simply added together. Per this syntax, numbers can be positive or negative, integral or fractional (be mindful of precision limits of computers' floating-point math). Unit names must be used as specified; any other names, especially non-English ones, are not accepted. In places where program code does not care about subsecond accuracy, time spans will have the subsecond parts truncated, i.e. "1.5s" becomes 1.0 seconds, and "1.5min" stays the 90 seconds that it is.

© grommunio GmbH — CC BY-SA 4.0

Examples: *1d1h1m1s*, *3.5 hours*, *1 hour 1 hour* (2 hours), *1 hour 60 minutes* (2 hours)

Note that Gromox may impose additional restrictions on specific configuration directives after the basic parse to enforce certain minimum and maximum values.

© grommunio GmbH — CC BY-SA 4.0

gromox.cfg(5)

Name

gromox.cfg — Central configuration file for Gromox services

Description

The common configuration file utilized by all services. Historically, each service had separate configuration files, but the number of directives that needed explicit configuration were reduced over time to the point that there were too many distinct files for little value, and so, gromox.cfg came to be.

Configuration directives

This manpage does not describe all possible directives. Please also consult the manpages of gromox daemons and command-line utilities for more.

> `autoreply_silence_window`
> If an autoreply message (other than a bounce report, e.g. out-of-office) is to be generated, that return message will be suppressed if another autoreply for the given {From, To} address pair was produced within the given time period previously. (Unlike the response_audit_capacity directive, autoreply pairs are stored persistently, in the message database.)
> Default: 1day

> `backfill_transport_headers`
> Try to fill the PR_TRANSPORT_MESSAGE_HEADERS property when messages are submitted. Turning this option on requires extra storage (usually between 0.5KB and 1KB per sent message). Transport headers for sent messages can be generated on the fly from the regular MAPI message data structures e.g. via gromox-export(8) for analysis, so unless there is a need to retain the results from older generator versions, you should leave this off.
> Default: off

> `bounce_postmaster`
> This directive defines the Envelope-From and From addresses for system-generated bounce messages such as Non-Delivery-Reports (autoresponse messages about e.g. mailbox being full, or a target email address being invalid). RFC-5321-5322-consuming components like SMTP, IMAP and EAS want to have an SMTP-looking address in this field. Preferably, it should point to the administrator's mailbox (or an alias thereto).
> When the value set here has no domain part and ends in just '@', the domain is taken from a re-resolution of the system hostname (so as to cope with an unqualified hostname in /etc/hosts that

© grommunio GmbH — CC BY-SA 4.0

many system installations use).
Default: postmaster@

`lda_junk_rules`
A semicolon-separated list of header/value matches that make delivery(8) place incoming messages into the junk folder rather than Inbox. Each entry in this list has the form *Header=val* (`:` may be used in place of `=`). Header names and values are compared case-insensitively. Values are checked both against the full header value and individual comma/semicolon separated tokens; use an empty value to match any header value.
Default: (empty)
Example: X-Spam-Flag=YES; X-MC-Relay=; X-SG-EID=

`exmdb_client_rpc_timeout`
If the execution of an RPC takes longer than the specified time, the client will sever the connection and return an error to the calling program. The value cannot be lower than 4s. The special value 0 disabled RPC timeout checking.
Default: 0

`outgoing_smtp_url`
A string which selects how outgoing messages are handed to a mail transfer agent. The syntax follows the Common Internet Scheme for URIs (RFC 1738 section 3.1), so something like "sendmail://localhost" or "smtp://[::1]:25/" can be used.
The sendmail:// transport selects the local maildrop queue (hostname is ignored). Maildrop is perhaps best known to administrators by the command /usr/sbin/sendmail. This transport is the most robust one, because local maildrop implementations generally accept messages even if the MTA is not running (assuming everything else is fine, e.g. there is enough free disk space). Partial Delivery Failure handling then is also the MTA's responsibility, and the MTA will consistently emit NDR/DSN for unreachable recipients.
Other recognized schemes are smtp:, smtp+tls: (STARTTLS), smtp+unverifiedtls: (STARTTLS but ignore validation issues) and smtps: (always-on/implicit TLS). These are only meant for developers who want to avoid running an MTA for having faster turnarounds. All SMTP transport drivers are **locally unqueued**, which means that, if the SMTP server is unavailable due to network issues, pressing the "Send" button will immediately produce an error in always-online MUAs, and the user has to save the message as a draft and try again later on their own accord. MAPI also has no way to be conveyed Partial Delivery Failure occuring in SMTP conversations, so a multi-recipient message with one bad recipient can lead to three bad characteristics all at once: valid recipients will receive a message copy; the sender got the Send button error; no Non-Delivery Report is generated for users in the local mail system. The proper way to use a custom outgoing SMTP server is to configre the localhost MTA to perform relaying.
Default: sendmail://localhost

`ruleproc_debug`
Make the "TWOSTEP" Client-Side Inbox Rule Processor emit information about the conditions it is evaluating and the actions it is carrying out. The surrounding process also needs to have log level set to at least 6 (debug) to see anything, i.e. delivery.cfg:lda_log_level=6 in case of delivery(8gx), or

© grommunio GmbH — CC BY-SA 4.0

> --loglevel=6 in case of command-line tools like case of gromox-import(8). [The Rule Processor implementation inside exmdb_provider(8gx) logs unconditionally, and its log messages will be seen, provided level 6 is set with http.cfg:http_log_level=6.]
> Default: off

> `reported_server_version`

Default: 15.00.0847.4040

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

mapi(7gx)

Name

mapi — Definition for "Messaging Application Programming Interface"

Definition

MAPI (Messaging API) can have many different meanings. As abstracta:

- The concept of the hierarchial organization into containers, tables, contents and properties. Typical containers are stores and **folders**, typical contents are **messages**, **recipients**, **attachments**.
- More concrete variations, such as the set of "well-known" properties like, for example, subject, modification date, body, etc., and the semantics expected of them.
- Optionally, the profile configuration structure present in some MAPI client implementations, comprised of **profiles**, **providers**, **services**, **profile sections**, **sessions** and their associated properties. A server generally does not know or need to know about these.
- A specification of types and functions to make use of any of these concepts in a programming language.

As an implementation or part thereof:

- (Seemingly the most common meaning of "MAPI"!) The MSMAPI interface for programming in C and/or C++ that comes with the Windows operating system/Office package, with types like "IMAPIFolder" and "IMessage", functions like "MAPILogonEx" or "IMessage::GetProperties", and identifiers like PR_SUBJECT.
- The Office VBA interface for programming in VisualBasic, with types like "Outlook.MailItem", object properties like "MailItem.Subject".
- The collection of wire protocols used for communication with Exchange Server. It has colloquially been referred to as "MAPI/RPC" or "MAPI protocol", but detail-oriented readers will get know the actual parts instead: the Remote Operations (ROP/OXCROPS) protocol, the EMSMDB v1 protocol, the NSPI protocol, the MAPI Extensions For HTTP protocol (MAPIHTTP, MH), and all the serialized representations of MAPI data structures.

Gromox implementation

Gromox does not offer an MSMAPI-style programming API for the C or C++ language environments; the types and functions that are exposed in .h/.hpp files are intended for its own use only and are

© grommunio GmbH — CC BY-SA 4.0

not considered public let alone constant over time.

Gromox offers interfacing at the network level, e.g. via exchange_emsmdb(4gx)/http(8gx) for the Exchange protocol formats, or via exmdb_provider(4gx) and zcore(8gx) using its own wire formats.

Gromox offers a somewhat MSMAPI-resembling interface for programming in PHP. This is a set of 119 freestanding functions that implement the concepts of stores, folders, messages, etc.

© grommunio GmbH — CC BY-SA 4.0

alias_resolve(4gx)

Name

alias_resolve — Alias resolution and expansion for delivery(8gx)

Description

alias_resolve is a component of the delivery agent which rewrites the Envelope FROM and RCPT fields of incoming messages, trimming extensions, resolving aliases to the primary email address, resolving contact objects where they occur, and expanding mailing lists to their members.

Configuration directives (gromox.cfg)

`lda_alias_cache_lifetime`
Interval between refreshes of the alias and contact object cache.
Default: 1h

`lda_recipient_delimiter`
The set of characters that separate an email address into localpart and extension. This mirrors the "recipient_delimiter" directive from postconf(5).
Default: (empty)

Signals

When the component is reloaded, i.e. by way of sending SIGHUP to delivery(8gx), the alias cache is refreshed as well, resetting the timer for the next refresh.

Files

- *data_file_path*/mlist_bounce/: response templates for when a mailing list could not be expanded

See also

gromox(7), **delivery**(8gx), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

authmgr(4gx)

Name

authmgr — Demultiplexer for authentication requests

Description

authmgr is a component that, for mail account authentication, dynamically selects the authentication backend per user account.

Configuration directives

The following directives are recognized when they are in /etc/gromox/gromox.cfg:

`auth_fail_delay`
The amount of time to wait after a failed authentication attempt (unknown user or rejected password). When the MySQL database itself is unavailable, the delay is also added, which mitigates a little what could otherwise be an authentication storm. Users which are externally managed (e.g. in LDAP) are exempt from auth_fail_delay in Gromox, since the expectation is that the LDAP Bind operation already inserts a delay of its own. Setting the value to 0 disables the delay. Per-username banning by **user_filter**(4gx) applies regardless of this setting. Subsecond resolution is available.
Default: 1s

`auth_backend_selection`
This controls how authmgr will verify passwords supplied with login operations. See the "Authentication modes" section below for details.
Available: *deny_all*, *allow_all*, *ldap*, *pam*
Default: ldap

Authentication modes

- *deny_all*: every attempt at authentication is rejected. This is at best useful for testing. This also implies auth_fail_delay=0.
- *allow_all*: every attempt at authentication (provided the user exists) is permitted. This may be handy when doing the initial mass-import of mailboxes via external IMAP-to-IMAP synchronization utilities such as imapsync without needing to know user passwords.
- *ldap* (old name: **externid**): authmgr will selectively pick LDAP/MySQL, depending on whether the externid column in the user database has a value or contains just the empty string. The

© grommunio GmbH — CC BY-SA 4.0

particular value is ignored and only meaningful to the importer.

- *pam*: authmgr will selectively pick PAM/MySQL. The PAM service name will be "gromox". Be sure that pam_gromox.so is **not** invoked as part of that PAM service stack, or it will lead to infinite recursion.

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7), **ldap_adaptor**(4gx), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

authtry(8gx)

Name

authtry — Diagnostic utility for debugging authentication

Synopsis

PASS='*xyz!***' /usr/libexec/gromox/authtry -u** *abc@example.com*

PASS='*xyz!***' /usr/libexec/gromox/authtry -L** [**-Z**] [**-H** *ldap://localhost/*] **-u** *cn=abc,o=example*

Description

authtry can be used to debug the user authentication procedure without having to use gdb to attach to, and pause, a running daemon. The password is to be conveyed via an environment variable, PASS. Note that PASS being unset is distinct from PASS being the zero-length string; both modes are supported.

Options

`-H` *uri*
LDAP server to connect to, in URI form. If not specified, the libldap default is used.

`-L`
Bypass the normal user lookup via authmgr(4gx), bypass user metadata lookup in MySQL, and only perform an LDAP bind. In this sense, authtry is similar to ldapsearch(1), but without performing any search.

`-Z`
Perform LDAP STARTTLS (only meaningful when combined with -L).

`-u` *username/dn*
The Gromox username (usually *user@domain* form), or, in case -H is used, an LDAP Distinguished Name.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

autoconfig(7)

Name

autoconfig — Thunderbird AutoConfig protocol

Description

AutoConfig is a HTTP-based discovery protocol, originally introduced for Mozilla Thunderbird (TB) 3.0 in 2009. It is quite similar to autodiscover(7). This manual page is not normative.

TB tries a bunch of URLs, with HTTP GET:

- `https://autoconfig.example.com/mail/config-v1.1.xml?emailaddress=user@example.com`
- `http://autoconfig.example.com/mail/config-v1.1.xml?emailaddress=user@example.com`
- `https://example.com/.well-known/autoconfig/mail/config-v1.1.xml?emailaddress=user@example.com`
- `http://example.com/.well-known/autoconfig/mail/config-v1.1.xml?emailaddress=user@example.com`
- DNS SRV lookups are not performed.

Note that TLS wildcard certificates do not extend to more than one level; for example, if the e-mail address is `foo@m1.example.com`, and the m1 host presents a server certificates for `CN=example.com`, `subjAltName=*.example.com`, clients that are trying to download `autoconfig.m1.example.com/mail/config-v1.1.xml` may fail TLS verification. The Thunderbird setup wizard (circa version 140) does not show TLS failures, and silently treats it as an unavailable resource.

TB *also* performs AutoDiscover.

config-v1.1.xml

The Thunderbird wiki at https://wiki.mozilla.org/Thunderbird:Autoconfiguration is quite lacking as of 2026-07-01. It has neither formal ABNF syntax, nor does it mention possible values for certain XML attributes. One has to resort to looking at source code, https://github.com/mozilla/releases-comm-central/blob/master/mail/components/accountcreation/modules/readFromXML.sys.mjs#L78.

In Thunderbird 140, incomingServers may be of type "imap", "pop3", "nntp", "exchange". The comm-central master has some more values.

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **autodiscover**(7)

© grommunio GmbH — CC BY-SA 4.0

autodiscover(4gx)

Name

autodiscover — AutoDiscover HTTP Service Protocol handler (responder)

Description

A client would make a HTTP request to the /Autodiscover/Autodiscover.xml endpoint, which is handled by the Gromox oxdisco component.

The Autodiscover response contains the home server name and protocol options (MAPI-RPC-HTTP, MAPIHTTP, IMAP, etc.). A client uses this to set up the mailbox with MAPI services (like `MSEMS` or ``INTERSTOR`\`) within a MAPI profile.

Configuration directives (gromox.cfg)

The following directives are recognized when they appear in /etc/gromox/gromox.cfg.

> `oxdisco_advertise_mh`
> This setting controls whether the AutoDiscover response should include a EXHTTP Protocol section. Possible values: *yes*, *no*, *not_old_mso*, *only_new_mso*. The latter two values can be used to finely control emission in case of clients other than Outlook.
> Default: yes

> `oxdisco_advertise_rpch`
> This setting controls whether the AutoDiscover response should include EXCH/EXPR Protocol sections. Possible values: *yes*, *no*, *only_old_mso*, *not_new_mso*. The latter two values can be used to finely control emission in case of clients other than Outlook.
> Default: yes

> `oxdisco_exonym`
> Globally valid name pointing to a default mailbox server in the Gromox server forest. (Since AutoDiscover nodes are also mailbox nodes, the exonym should simply be: this server's hostname).
> Default: (inherited from http.cfg:host_id, which defaults to the system FQDN)

> `oxdisco_pretty_response`
> A debugging knob to make the Autodiscovery handler emit indented XML responses.
> Default: no

> `oxdisco_request_logging`
> Log AutoDiscover requests. This is independent of http.cfg:http_debug, and setting both would

© grommunio GmbH — CC BY-SA 4.0

log requests twice.
Default: no

`oxdisco_response_logging`
Log AutoDiscover responses. This is independent of http.cfg:http_debug, and setting both would log responses twice.
Default: no

`oxdisco_validate_scndrequest`
When OL opens a non-default store (store of another user) or a public store, it may also make an AutoDiscover inquiry for the extra store. This setting controls whether the AutoDiscover handler should perform a permission check on non-default stores and possibly reject returning connection details. (Inquiry of public stores are always permitted.)
Default: yes

`x500_org_name`
Same as gromox.cfg:x500_org_name.
Default: (unspecified)

Configuration directives (autodiscover.cfg)

The following directives are recognized when they appear in /etc/gromox/autodiscover.cfg. autodiscover.cfg is obsolete in favor of gromox.cfg.

`x500_org_name`
Same as gromox.cfg:x500_org_name.
Default: (unspecified)

Configuration directives (autodiscover.ini)

The following directives are recognized when they appear in /etc/gromox/autodiscover.ini. autodiscover.ini is obsolete in favor of gromox.cfg.

`organization`
Same as gromox.cfg:x500_org_name.

Outlook notes

When Outlook is active, it is possible to Ctrl-MouseBtn3 (right click) on the status tray icon to call up a context menu, from which "Test Email Autoconfiguration..." can be selected to debug AutoDiscover requests and responses from the Windows side.

© grommunio GmbH — CC BY-SA 4.0

Exchange notes

In EXC, MH can be toggled on/off via an EAC Shell command: `Set-OrganizationConfig -MapiHttpEnable \$false` (or $true).

Normative references

- MS-OXDISCO: Autodiscover HTTP Service Protocol
- MS-OXDSCLI: Autodiscover Publishing and Lookup Protocol

See also

gromox(7), **autodiscover**(7)

© grommunio GmbH — CC BY-SA 4.0

autodiscover(7)

Name

autodiscover — AutoDiscover protocols

Description

AutoDiscover is a HTTP-based discovery protocol, originally introduced for Exchange 2007, that helps users configure their email client settings automatically. The MS-OXDSCLI specification speaks about locating such AutoDiscover service, the MS-OXDISCO specification speaks about the XML exchange between such a service and a client. These terms are herein used to avoid ambiguity in the "AD" acronym (ActiveDirectory).

Autodiscover V2 is not an improved version, it is an extra layer that warrants a disgruntled remark about questionable protocol design. Locating the AutoDiscover server still happens via DNS or AD-SCP (Service Connection Points in Active Directory) query. The V2 request contains the user identity and the name of a next-level protocol that the client seeks, e.g. "ActiveSync", "EWS" or "AutoDiscoverV1". The response is now a JSON document and generally contains just one URL, namely for the service sought. Indeed there is no way to obtain MAPI, IMAP or SMTP information in Autodiscover V2.

Note that Autodiscover is not used exclusively by Microsoft Outlook, Autodiscover is the main discovery protocol for any EAS-enabled device and application, such as Apple iOS, Android and other applications.

OXDSCLI summary

The MS-OXDSCLI document is about how clients can collect one or more AutoDiscover URLs/endpoints that, in a second step (OXDISCO), they can make HTTP POST requests to to retrieve server configuration. Behaviors that have been specified in MS-OXDSCLI or have been observed in practice:

- Using an URL from a previous successful run.
- Performing a DNS TXT lookup on the domain name to check for the presence of a Microsoft 365 account identifier and, if found, using the URL <**https://outlook.office365.com/autodiscover/autodiscover.xml**>.
- When joined to an NT Domain/ActiveDirectory, performing an LDAP lookup for service connection points (SCP), i.e. objects matching `(&(objectClass=serviceConnectionPoint)(serviceClassName=ms-Exchange-AutoDiscover-Service))`, and using the URLs obtained from "serviceBindingInformation" attributes.

© grommunio GmbH — CC BY-SA 4.0

- "Root domain" method: Constructing a string based on the user's e-mail address (e.g. u@example.com), <**https://***example.com***/autodiscover/autodiscover.xml**>.
- "AutoDiscover domain" method: Constructing a string based on the user identity, <**https://autodiscover.***example.com***/autodiscover/autodiscover.xml**>.
- Performing a DNS lookup of the name **_autodiscover._tcp.***domain* of record type SRV, and from the response data, constructing the string **https://***rdata*[**:***port*]**/autodiscover/autodiscover.xml**.
- Utilizing an unspecified "local mechanism" method. (might be the same as using lastrun URL?)
- "Redirect" method: Performing an unauthenticated HTTP GET request to <**http://autodiscover.***domain***/autodiscover/autodiscover.xml**, and upon receiving a HTTP 302 response, using the URL in the "Location:" response header for AutoDiscover.

Gromox recommends that administrators employ the AutoDiscover Domain method, i.e. add a **autodiscover.***domain.com* entry to their DNS zones. This is beneficial over the root domain method, because the host serving up e.g. a company's website need not be taught about autodiscover paths. For **autodiscover.***domain.com*, either an AAAA/A pair **or** CNAME resource record can be used. Split-horizon DNS systems **must** publish the autodiscover DNS record(s) in the publicly visible set. The AutoDiscover URL must equally be reachable from any network segment where access is expected from — if need be, set up HTTP reverse proxies.

OXDISCO summary

The MS-OXDISCO document specifies what clients should do once they have an URL collection obtained from the OXDSCLI mechanism(s).

When a user sets up a new email account or changes their existing email client settings, the email client sends an authenticated HTTP POST request to an Autodiscover URL and conveying the user's email address in the request body. The server then responds with an XML document that contains the necessary configuration settings.

The response contains information such as

- the list of supported mail protocols and transports (e.g. MSRPC/RPCH/MAPIHTTP, IMAP, SMTP, etc.)
- the connection parameters for those (e.g. name of the home server, HTTP endpoint URLs)
- for MAPI, any extra mailboxes that should be opened unconditionally (e.g. delegators, public folder)

The email client uses this information to configure the user's email account automatically.

© grommunio GmbH — CC BY-SA 4.0

gromox-dscli notes

The gromox-dscli(8) utility can be used to diagnose problems with AutoDiscover from a command line prompt.

gromox-dscli performs the following probes and in this order: RootDomain, AutoDiscoverDomain, SRV.

Outlook notes

When Outlook is running, there is an Outlook icon in the Windows taskbar's notification area. By pressing Ctrl+RightMouseBtn, a service menu can be brought up, which offers a "Test AutoDiscover" command for diagnosing problems from Windows. Known bugs: The dialog may ignore the contents of the password field and instead use a saved password or SSO, leading to potentially unanticipated authentication successes or failures. If in doubt, use gromox-dscli.

When an AutoDiscover response contains broken information (e.g. unreachable endpoints due to a faulty oxdisco_exonym setting on the server), Outlook may refuse to open mailboxes and the application may exit prematurely. AutoDiscover responses are cached on disk under `%LOCALAPPDATA%/Microsoft/Outlook/* - Autodiscover.xml` and/or `%LOCALAPPDATA%/Microsoft/Outlook/16/AutoD.*.xml` and can be deleted/diagnosed as needed.

Outlook performs the following probes in this primary order: TXT, LDAP-SCP, RootDomain, AutoDiscoverDomain, SRV, Local, Redirect. Secondary order that was observed: TXT, LDAP-SCP, RootDomain, AutoDiscoverDomain, Local, Redirect, SRV.

Individual probes can be disabled via Windows Registry in

- HKEY_CURRENT_USER\Software\Microsoft\Office\16.0\Outlook\AutoDiscover

and/or

- HKEY_CURRENT_USER\Software\Policies\Microsoft\Office\16.0\Outlook\AutoDiscover,

by setting one or more of

- ExcludeLastKnownGoodURL=DWORD:1
- ExcludeExplicitO365Endpoint=DWORD:1
- EnableOffice365ConfigService=DWORD:0
- ExcludeScpLookup=DWORD:1
- ExcludeHttpsRootDomain=DWORD:1
- ExcludeHttpsAutoDiscoverDomain=DWORD:1
- ExcludeSrvRecord=DWORD:1

© grommunio GmbH — CC BY-SA 4.0

- ExcludeHttpRedirect=DWORD:1

as desired. For the Group Policy editor, ADMX template and ADML language packs are available from https://www.microsoft.com/en-us/download/details.aspx?id=49030.

When a DNS zone is M365-enabled, Outlook opens a mini browser window for authenticating with M365. To prevent this, you can set ExcludeExplicitO365Endpoint=1 as described.

Outlook stops probing after the first successful AutoDiscover HTTP POST request. A non-responsive AutoDiscovery server (firewall DROP policy, or TCP RST) is treated the same as a 404 Not Found response.

Known bugs: Outlook ignores the port number in the DNS SRV response. Outlook and/or the Windows HTTP libraries also erroneously show a warning popup whenever the hostname in the SRV result does not match the e-mail domain (even under MS Exchange). Redirection is the key idea of an SRV record and, as far as security considerations go, is no more significant than following a CNAME-typed autodiscover.example.com record.

Outlook re-runs AutoDiscover periodically in the background. This can cause popups such as re-authentication or SRV warnings (particularly after a temporary outage).

Testing scenarios

To force using a particular Autodiscover server in Windows, such as when Gromox is run in a development environment with a fake domain, c:\windows\system32\drivers\etc\hosts can be populated with a static entry for **autodiscover.***example.com* to get that particular scenario working.

See also

gromox(7), **autoconfig**(7), **autodiscover**(4gx)

© grommunio GmbH — CC BY-SA 4.0

ddbg(1gx)

Name

ddbg — MAPI data debugger

Synopsis

/usr/libexec/gromox/ddbg [options...] command [args...]

Description

ddbg can be used to analyze various binary blobs. If no arguments are given, input is read from stdin. If arguments are given, they are treated as immediate values (i.e. content, never the name of a file to read).

Options

`-p`, **--pack**
Employ hex2bin before main action.

Commands

`--bin2hex`
Convert all bytes to hexnibble representation.

`--bin2txt`
Convert all bytes to a textual representation. The environment variable BIN2TXT_MODE can be used to influence the output. Possible values are **cstr** (output as a C string literal without surrounding quotes), **hex** (hex nibbles like bin2hex), **txt** (custom compact encoding).

`-d`, **--decode**
Try all decoders.

`-A`, **--decode-action**
Decode rule action blob.

`-e`, **--decode-entryid**
Decode entryid.

`--decode-guid`
Lookup GUID.

© grommunio GmbH — CC BY-SA 4.0

`--decode-nttime`
Decode an NT timestamp and show the equivalent Unix time and calendar-based date.

`--decode-restrict`
Decode restriction blob (e.g. rule condition).

`--decode-unixtime`
Decode an Unix timestamp and show the equivalent NT time and calendar-based date.

`--htmltortf`
Convert a HTML document to RTF.

`--htmltotext`
Convert a HTML document to plaintext.

`--lzxdec`
Uncompress an lzxpress data stream.

`--lzxenc`
Compress data stream with lzxpress.

`--rtfcp`
Convert RTF to the RTFCP format, particularly the uncompressed "MELA" subformat.

`--rtftohtml`
Convert RTF to HTML.

`--texttohtml`
Convert plaintext to HTML.

`--unrtfcp`
Decompress RTFCP (either "MELA" or "LZFU") to RTF.

Environment variables

GROMOX_HTMLTOPLAIN can be set to "chawan", "pandoc", "w3m", "internal" to pick one particular htmltoplain implementation. If unset or set to "auto", all of these converters are tried until a working one is found.

GROMOX_HTMLTORTF can be set to "pandoc" or "internal" to pick a particular htmltortf implementation. If unset or set to "auto", all of these converters are tried until a working one is found.

GROMOX_RTFTOHTML can be set to "pandoc" or "internal" to pick a particular htmltortf implementation. If unset or set to "auto", all of these converters are tried until a working one is found.

© grommunio GmbH — CC BY-SA 4.0

Examples

- ddbg -p --decode-guid 38a1bb1005e5101aa1bb08002b2a56c2
- ddbg --unrtfcp <body.bin >body.rtf

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

delivery(8gx)

Name

delivery — Backend for local delivery

Synopsis

delivery [**-c** *config*]

Options

> `-c` *config*
> Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/delivery.cfg will be read.

> `-?`
> Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

> `daemons_fd_limit`
> In gromox-delivery, this is treated as an alias for lda_fd_limit.

> `lda_fd_limit`
> Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
> Default: 0

> `outgoing_smtp_url`
> See gromox.cfg(5):outgoing_smtp_url.

Configuration directives (delivery.cfg)

The following directives are recognized when reading from /etc/gromox/delivery.cfg, or when the **-c** option is used to specify a custom file:

© grommunio GmbH — CC BY-SA 4.0

`admin_mailbox`
An e-mail address where to send reports from the "net_failure" code component.
Default: (empty)

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/delivery/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/delivery:/etc/gromox

`data_file_path`
Colon-separated list of directories in which static data files will be searched.
Default: /usr/share/gromox/delivery

`dequeue_max_mem`
Default: 1024M

`dequeue_path`
Default: /var/lib/gromox/queue

`free_context_num`
Default: 512

`host_id`
A unique identifier for this system. It is used for the DSN text of bounce messages. It is used as the value for the EHLO command if and when connecting to an SMTP service (e.g. inbox rules that do forwarding).
Default: (system hostname)

`lda_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`lda_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`running_identity`
An unprivileged user account to switch the process to after startup.
Default: gromox

`work_threads_max`
The number of threads that gromox-delivery may spawn to process incoming messages. This number must be less-or-equal to the exmdb_local(4) exmdb_connection_num directive to avoid rejection of messages in a message storm.
Default: 5

© grommunio GmbH — CC BY-SA 4.0

> `work_threads_min`
> Default: 1

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

delivery-queue(8gx)

Name

delivery-queue — LMTP/SMTP frontend for local delivery

Synopsis

delivery-queue [**-c** *config*]

Description

delivery-queue is an additional mail queue in front of delivery(8gx). It may be removed in a future version.

Options

> `-c` *config*
> Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/smtp.cfg will be read.

> `-?`
> Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

> `daemons_fd_limit`
> In gromox-delivery-queue, this is treated as an alias for lda_fd_limit.

> `lda_fd_limit`
> Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
> Default: 0

> `lda_listen`
> A space-separated list of bind address specifiers to expose standard SMTP/LMTP (unencrypted

© grommunio GmbH — CC BY-SA 4.0

and/or explicit STARTTLS) on.
Default: *[::]:25*

`lda_listen_tls`
A space-separated list of bind address specifiers to expose implicit-TLS SMTP (SMTPS)/LMTP on. (Port 465 is the recommendation from RFC 8314.)
Default: *[::]:465*

`lda_recipient_delimiter`
The set of characters that separate an email address into localpart and extension. This mirrors the "recipient_delimiter" directive from postconf(5).
Default: (empty)

`lda_support_haproxy`
This directive sets the expectation for incoming connections to carry haproxy's "PROXY" protocol extension version 2 (2), or no such header (0). When a (reverse) proxy is placed in front of gromox-delivery-queue, the address that gxdq normally sees is the proxy address (e.g. ::1). A proxy can use this protocol extension to convey the actual client address, and gximap can pick this up for its own reporting, which in turn is useful for e.g. fail2ban setups.
Default: 0

Configuration directives (smtp.cfg)

The following directives are recognized when reading from /etc/gromox/smtp.cfg, or when the **-c** option is used to specify a custom file:

`command_protocol`
Selects what kind of hello command is accepted of clients. Can be *lmtp*, *smtp* or *both*.
Default: both

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/smtp/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/smtp:/etc/gromox

`context_average_mem`
Default: 256K

`context_max_mem`
Default: 2M

`context_num`
Default: 200

`data_file_path`
Colon-separated list of directories which will be scanned when locating data files.

© grommunio GmbH — CC BY-SA 4.0

Default: /usr/share/gromox/smtp

`host_id`
A unique identifier for this system. It is used for greeting lines emitted by delivery-queue on the network.
Default: (system hostname)

`lda_listen_addr`
Deprecated in favor of gromox.cfg:lda_listen / gromox.cfg:lda_listen_tls.

`lda_listen_port`
Deprecated in favor of gromox.cfg:lda_listen.

`lda_listen_tls_port`
Deprecated in favor of gromox.cfg:lda_listen_tls.

`lda_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`lda_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`lda_thread_charge_num`
The maximum number of connections that each thread is allowed to process.
Default: 40

`lda_thread_init_num`
The minimum number of client processing threads to keep around.
Default: 1

`mail_max_length`
Maximum permitted length of a message.
Default: 64M

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

`smtp_certificate_passwd`
The password to unlock TLS certificates.
Default: (unset)

`smtp_certificate_path`
A colon-separated list of TLS certificate files. The complete certificate chain should be present (as

© grommunio GmbH — CC BY-SA 4.0

there is no other config directive to pull CA certs in, and implicit loading from system directories is not guaranteed by Gromox).
Default: (unset)

`smtp_conn_timeout`
If an SMTP connection is inactive for the given period, the connection is terminated.
Default: 3 minutes

`smtp_force_starttls`
This flag controls whether clients must utilize TLS, either by way of implicit TLS (cf. **lda_listen_tls_port**), or through the STARTTLS command.
Default: false

`smtp_private_key_path`
A colon-separated list of TLS certificate private key files.
Default: (unset)

`smtp_support_pipeline`
This flag controls the offering of the PIPELINING extension (RFC 2920) to clients.
Default: true

`smtp_support_starttls`
This flag controls the offering of the STARTTLS extension (RFC 3027) to clients.
Default: false

`tls_min_proto`
The lowest TLS version to offer. Possible values are: **tls1.0**, **tls1.1**, **tls1.2**, and, if supported by the system, **tls1.3**.
Default: tls1.2

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Files

- *data_file_path*/smtp_code.txt: Mapping from internal SMTP error codes to textual descriptions.

See also

gromox(7), **delivery**(8gx), **midb_agent**(4gx)

© grommunio GmbH — CC BY-SA 4.0

dnsbl_filter(4gx)

Name

dnsbl_filter — DNS Blacklist filtering

Description

dnsbl_filter is a module which will query a Domain Name System Realtime Blackhole/Blacklist/Block List to deny access to IP addresses attempting to connect to Gromox services. In particular, (only) the four services use it; gromox-imap(8), gromox-pop3(8), gromox-delivery-queue(8) and gromox-http(8).

Configuration directives

The config file location is /etc/gromox/gromox.cfg; service specific locations are /etc/gromox/http/gromox.cfg, /etc/gromox/imap/gromox.cfg and /etc/gromox/pop3/gromox.cfg.

> `dnsbl_client`
> This sets the zone suffix to use for queries. If no zone is set, no DNSBL checking takes place.
> Example: xbl.spamhaus.org
> Default: (unset)

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

event_proxy(4gx)

Name

event_proxy — Event sender

Description

event_proxy connects to the event(8gx) daemon and sets itself up so as to be able to send notifications.

event_proxy installs three service functions, "broadcast_event", "broadcast_select" and "broadcast_unselect", though the latter two are just convenience functions for broadcast_event. broadcast_event is for synchronously sending a notification into the event distribution system. Arbitrary notifications and commands can be sent this way. The return value (i.e. in the eventd network protocol) is ignored.

In practice, midb(8gx), imap(8gx) and pop3(8gx) issue FOLDER-TOUCH notifications. Only imap(8gx) issues MESSAGE-FLAG and MESSAGE-EXPUNGE notifications.

See also

event(8gx), **event_stub**(4gx)

© grommunio GmbH — CC BY-SA 4.0

event_stub(4gx)

Name

event_stub — Event receiver

Description

event_stub connects to the event(8gx) daemon and sets itself up to receive notifications asynchronously with the help of an extra thread. (No notifications are sent towards event(8gx) by this component.)

event_stub installs one service function, "install_event_stub", with which imap(8gx) registers a callback function that, in turn, is invoked whenever a notification is received from the event daemon.

In practice, imap handles FOLDER-TOUCH, MESSAGE-FLAG and MESSAGE-EXPUNGE notifications received through this event channel.

See also

event(8gx), **event_proxy**(4gx)

© grommunio GmbH — CC BY-SA 4.0

event(8gx)

Name

event — Folder change notification daemon

Synopsis

event [**-c** *config*]

Description

The event daemon is a software bus, inter-process communication (IPC) mechanism that allows communication between multiple processes running concurrently on multiple machines.

In practice, it is used by midb(8gx), pop3(8gx) and imap(8gx) to notify imap(8gx) instances of changed folder/message states.

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/event.cfg will be read.

`--version`
Output version information and exit.

`-?`
Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives

The usual config file location is /etc/gromox/event.cfg.

`event_hosts_allow`
A space-separated list of individual host addresses that are allowed to converse with the event service. The addresses must conform to gromox(7) § "Host addresses". No networks and no CIDR notations are permitted.
Default: ::1

© grommunio GmbH — CC BY-SA 4.0

`event_listen_ip`
The IPv6 socket address for exposing the event service on. The address must conform to gromox(7) § "Host addresses".
Default: ::1

`event_listen_port`
The TCP port number for exposing the event service on.
Default: 33333

`event_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`event_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`event_threads_num`
The minimum number of client processing threads to keep around.
Default: 50

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

Event protocol

The event service is exposed as a line-based text protocol. Upon connection, the event server gratitiously writes "OK" and will wait for commands. Each connection to the event daemon starts out in Enqueue Mode, and this is the only mode from which commands can be issued.

"FALSE" may be emitted by the server if there is a syntax error.

ID

The command "ID <res_id>" declares the particular connection to be a notification sender. res_id is generally the hostname and the PID. The server always responds with "TRUE". (The connection stays in Enqueue Mode.)

LISTEN

The command "LISTEN <res_id>" declares the particular connection to be a notification receiver. res_id follows the same pattern. The server responds with "TRUE" and the connection state changes to the Dequeue Mode (see below).

© grommunio GmbH — CC BY-SA 4.0

SELECT

The command "SELECT <username> <folder>" subscribes those connections that have registered **as a listener for res_id** to notifications. (This means that a process wishing to use event_stub(4gx) to listen for notifications strictly requires loading event_proxy(4gx) too, and, in essence, use two connections to event(8gx).) The server responds with "FALSE" if no listener exists, or "TRUE" on success.

UNSELECT

The command "UNSELECT <username> <folder>" unsubscribes those connections that had registered as a listener for res_id. The server always responds with "TRUE".

QUIT

Terminate the connection.

PING

Reset inactivity timer on connection.

Partially parsed commands

Any other input is treated as a notification item and is not interpreted by event(8gx) beyond checking the number of fields:

FOLDER-TOUCH

The notification "FOLDER-TOUCH <username> <folder>" informs listeners that the folder metadata has changed and warrants being reloaded. This is also how the arrival of new messages is conveyed.

MESSAGE-FLAG

The notification "MESSAGE-FLAG <username> <folder> <messageid>" informs listeners that the message metadata has changed and warrants being reloaded. (This operation is no longer recognized since Gromox 2.17-26-g10564f3e7.)

MESSAGE-UFLAG

The notification "MESSAGE-UFLAG <username> <folder> <imapuid>" informs listeners that the message metadata has changed and warrants being reloaded.

MESSAGE-EXPUNGE

The notification "MESSAGE-EXPUNGE <username> <folder> <messageid>" informs listeners that the message was deleted.

© grommunio GmbH — CC BY-SA 4.0

Client behavior

Clients in Dequeue Mode will receive notifications. Each notification line received by the client needs to be acknowledged with a "TRUE" response. It is not possible to exit Dequeue Mode; connection termination is the only way out.

Events do not echo for a particular res_id. The event_proxy(4gx) and event_stub(4gx) components use the getpid() function when constructing the res_id for the ID/LISTEN commands. A process like imap(8gx) which uses both components will intentionally not see its own notifications over the gromox-event IPC system this way.

See also

gromox(7), **event_proxy**(4gx), **event_stub**(4gx)

© grommunio GmbH — CC BY-SA 4.0

ews(4gx)

Name

ews — Handler for Exchange Web Services requests

Description

The ews(4gx) component handles requests to the **/EWS/Exchange.asmx** URI path.

Configuration directives (gromox.cfg)

The following directives are recognized when they appear in /etc/gromox/gromox.cfg: outgoing_smtp_url. See the gromox.cfg(5) manpage.

Configuration directives (ews.cfg)

The following directives are recognized when they appear in /etc/gromox/ews.cfg.

> `ews_log_filter`
> Default: !

> `ews_pretty_response`
> Controls whether SOAP/XML responses produced by ews(4gx) are emitted normally ("no") or with extra indent for debugging ("yes").
> Default: no

> `ews_request_logging`
> When set to 1 or higher, requests are logged (subject to restriction by log_filter). When set to 2 or higher, the XML input is dumped as well. The log messages are emitted with "debug" (6) priority, so you may also need to raise http.cfg:http_log_level and/or gromox.cfg:istore_log_level.
> Default: 0

> `ews_response_logging`
> When set to 1 or higher, responses are logged. When set to 2 or higher, the XML output is dumped as well. Like requests, these are logged at "debug" priority level.

Normative references

- OXWAVLS, OXWOOF, OXWSPHOTO

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

exchange_emsmdb(4gx)

Name

exchange_emsmdb(4gx) — Handler for the Wire Format Protocol (Outlook/Exchange RPCs) and Remote Operations Protocol

Description

exchange_emsmdb is a component of http(8gx) which handles (1.) the remote procedure calls for the EMSMDB v1 and AsyncEMSMDB v1 RPC interfaces, and (2.) the Remote Operations encoding protocol that is typically wrapped by EMSMDB's EcDoRpcExt2 call or MAPIHTTP's EXECUTE call.

EMSMDB is a DCE/RPC interface with just a few RPC calls (6 are still used today). ecDoRpcExt2, a call offered by that interface, takes an opaque byte buffer argument not interpreted by DCE/RPC. That byte buffer contains another protocol, "Remote Operation(s) Encoding Protocol" [OXCROPS]. No reason for this wrapping is given in the OXCRPC spec. MAPIHTTP runs OXCROPS directly without the extra DCERPC/EMSMDB framing.

OXCROPS consists of 130 calls that make up the mailbox protocol.

Configuration directives (gromox.cfg)

The following directives are recognized when they appear in /etc/gromox/gromox.cfg.

> `emsmdb_compress_threshold`
> When a ROP response buffer has at least this many bytes, attempt to compress with LZXpress. (Use -1 to disable compression.) This format/implementation is underperforming in modern contexts: the rate is just about 60 MB/s on a 5950X CPU, which is a tenth of what zstd-1.5.7 achieves, and the compression ratio is somewhere between Unix compress(1) and gzip level 1. Default: -1

> `outgoing_smtp_url`
> See gromox.cfg(5):outgoing_smtp_url.

Configuration directives (exchange_emsmdb.cfg)

The following directives are recognized when they appear in /etc/gromox/exchange_emsmdb.cfg.

> `async_threads_num`
> Default: 4

© grommunio GmbH — CC BY-SA 4.0

`ems_max_active_notifh`
Maximum number of concurrently active notify handles.
Default: (unlimited)

`ems_max_active_sessions`
Maximum number of concurrently active EMSMDB sessions. The special value 0 indicates unlimited. The special value 0 indicates unlimited. EMSMDB sessions are not tied to any particular TCP connection; sessions terminate by means of an explicit "ecDoDisconnect" request, or by an inactivity timer (currently 2000 seconds). MFCMAPI sends ecDoDisconnect, but Outlook just breaks off TCP connections, so sessions can pile up.
Default: (unlimited)

`ems_max_active_users`
Maximum number of concurrently active EMSMDB session owners (users). The special value 0 indicates unlimited.
Default: (unlimited)

`ems_max_pending_sesnotif`
Maximum number of pending notifications for an EMSMDB session. [Content tables on search folders can rack up enormous amounts of notifications, so worry not upon seeing W-2305 warnings in the log.]
Default: 64K

`emsmdb_collapse_notif_storm`
Collapse long runs of pending row-added/row-modified table notifications for a session into a single table-changed notification, like is always done for row-deleted runs. This spares the server one synchronous row read per pending notification when the response is serialized, at the price of clients reloading the affected table view. Exchange 2019 delivers individual events; keep this off if exact notification behavior matters more than throughput under bulk changes.
Default: no

`emsmdb_max_cxh_per_user`
The maximum number of EMSMDB sessions (CXH = RPC context handle) for one user. The special value 0 indicates unlimited. EMSMDB sessions are not tied to any particular TCP connection; sessions terminate by means of an explicit "ecDoDisconnect" request, or by an inactivity timer (currently 2000 seconds). Outlook does not send ecDoDisconnect (MFCMAPI does), so sessions can pile up.
Default: 100

`emsmdb_max_obh_per_session`
The maximum number of object handles (e.g. folders/messages/etc.) each ROP logon (contrary to the name, not EMSMDB session) can have at any one time concurrently. Use 0 to indicate unlimited. On average, Outlook creates six ROP logons per mailbox that it opens.
Default: 32768

© grommunio GmbH — CC BY-SA 4.0

`emsmdb_private_folder_softdelete`
Enables soft-delete support for folders in private stores. (This feature is experimental.) Public folders always have this on. (Take note that exmdb_provider.cfg:exmdb_private_folder_softdelete also need to be enabled.)
Default: yes

`emsmdb_rop_chaining`
0: Deactivate ROP chaining. 1: Enable ROP chaining for OL < 15 and OL >= 16.0.10000 (OL2019, OL2021, OLM365). 2: Enabled for all clients.
Default: 1

`mailbox_ping_interval`
Default: 5 minutes

`mail_max_length`
The maximum size for any individual attachment and message. Attempts to store objects larger than this are rejected. The upper limit is 2G, imposed by Exchange protocols.
Default: 64M

`max_ext_rule_length`
Default: 510K

`max_rcpt_num`
The maximum number of recipients that an e-mail is allowed to have.
Default: 256

`rop_debug`
Log every incoming OXCROP call and the return code of the operation in a minimal fashion. Level 1 emits ROPs with a failure return code, level 2 emits all ROPs. Note the daemon log level needs to be "debug" (6), too.
Default: 0

`submit_command`
Default: /usr/bin/php /usr/share/gromox/submit.php

`x500_org_name`
Default: (unspecified)

Files

- *data_file_path*/notify_bounce/: templates for read/nonread notification mails sent to originators

© grommunio GmbH — CC BY-SA 4.0

Outlook notes

Outlook can be started with the **/rpcinfo** command-line parameter to display a status table about the RPC connections it has open. Alternatively, one can Ctrl-MouseBtn3 (right button) on the status tray icon to call up a context menu, from which "Connection status..." can be selected for the same.

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Upon receipt of SIGUSR1, an overview of current EMSMDB sessions to the configured log device (stderr/journal by default). EMSMDB sessions are not linked to any one HTTP connection in particular, and multiple HTTP connections may exercise one session.

Normative references

- DCERPC / C706: Technical Standard DCE 1.1: Remote Procedure Call by The Open Group, 1997
- MS-OXCRPC: Wire Format Protocol. This is the document for the EMSMDB RPC interface.
- MS-OXCROPS: Remote Operations List and Encoding Protocol.

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

exchange_nsp(4gx)

Name

exchange_nsp — Handler for the Name Service Provider Interface Protocol

Description

exchange_nsp is a component for http(8gx) which handles the Name Service Provider Interface Protocol, in essence providing the Address Book for Outlook's EMSMDB32.DLL connector.

Configuration directives

The usual config file location is /etc/gromox/exchange_nsp.cfg.

`cache_interval`
Default: 5 minutes

`hash_table_size`
Default: 3000

`nsp_trace`
Level 1: Log entry into and exit out of NSP functions, with their parameter values to stderr (not http_log_file!). Log data dumps of select calls. Level 2: Dump more data.
Default: 0

`x500_org_name`
Default: (unspecified)

Notes

A number of properties are always synthesized by exchange_nsp and never read from any storage; this includes key properties such as PR_ENTRYID, PR_RECORD_KEY, etc.

For user-attached properties that are read from SQL (cf. table "user_properties"), exchange_nsp (as well as zcore(8gx)'s AB) only handles a subset of property types: PT_BOOLEAN, PT_SHORT, PT_LONG, PT_I8, PT_SYSTIME, PT_BINARY, PT_UNICODE/STRING8 and PT_MV_UNICODE/STRING8.

Signals

Upon receipt of SIGHUP, the address book cache will be dropped; the next regular request for the AB will cause it to be reloaded.

© grommunio GmbH — CC BY-SA 4.0

- MS-OXNSPI: Exchange Server Name Service Provider Interface (NSPI) Protocol

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

exchange_rfr(4gx)

Name

exchange_rfr — Handler for the Address Book Name Service Provider Interface Referral Protocol.

Description

exchange_rfr is a component for http(8gx) which handles the Address Book Name Service Provider Interface Referral Protocol, also referred to as the Directory Service Referral Interface (RFRI).

Configuration directives

This component has no config directives of its own.

Normative references

- MS-OXABREF: Address Book Name Service Provider Interface (NSPI) Referral Protocol

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

exmdb_local(4gx)

Name

exmdb_local — LDA hook that offers a exmdb_provider(4gx) client with a C API

Description

An LDA hook for delivery(8gx) which places mail into a store by connecting to a exmdb_provider(4gx) service.

Configuration directives

The usual config file location is /etc/gromox/exmdb_local.cfg.

> `autoreply_silence_window`
> -> See gromox.cfg(5) manpage instead!

> `cache_scan_interval`
> Interval in which to scan /var/lib/gromox/queue/cache.
> Default: 3min

> `exmdb_connection_num`
> Default: 5

> `lda_mrautoproc`
> Perform meeting request autoprocessing. This feature is currently experimental. Requires lda_twostep_ruleproc to be enabled.
> Default: no

> `lda_twostep_ruleproc`
> If set to *1*, an alternate rule processor codebase will be used which supports cross-store moves, forwarding and OOF condition but (at this time) no delegation or autoreply.
> Default: 0

> `response_audit_capacity`
> Keep track of at most this many {From address, To address} pairs for bounce reports in memory.
> Default: 1000

> `response_interval`
> If a bounce report message for a delivery failure or operational failure is to be generated, that return message will be suppressed if another report for the given {From, To} address pair was

© grommunio GmbH — CC BY-SA 4.0

> produced within the given time period previously.
> Default: 3min

> `retrying_times`
> Default: 30

> `x500_org_name`
> Default: (unspecified)

Notes

Out-of-office replies are only generated when one of the recipient's addresses (primary or alias) appears in the To, Cc or Bcc line of the message, cf. RFC 5230 section 4.5. Mail that reaches a mailbox only by way of e.g. a distribution group thus produces no OOF reply.

Files

- *data_file_path*/local_bounce/: response templates for when mail cannot be delivered

See also

gromox(7), **delivery**(8gx), **exmdb_provider**(4gx)

© grommunio GmbH — CC BY-SA 4.0

exmdb_provider(4gx)

Name

exmdb_provider — Gromox Information Store

Description

exmdb_provider is the mailbox engine. It offers a plethora of stateless low-level functions (124 of them) for operating on mailbox stores. The functionality of the engine is exposed by way of a Gromox-specific network protocol on port 5000.

exmdb_provider is built as a shared library and can run in either the http(8gx) or istore(8gx) processes, depending on the gromox.cfg:istore_standalone config directive.

The shared library contributes an exmdb_client API, which will transparently pick either a local or a remote procedure call depending on whether the mailbox is served by the same process or not.

Configuration directives (gromox.cfg)

The following directives are recognized when they appear in /etc/gromox/gromox.cfg.

`exmdb_force_write_txn`
(Developer option.) Perform all SQLite transactions as write transactions (i.e. with exclusive locking).
Default: no

`exmdb_deep_backtrace`
(Developer option.) Record and report transaction problems with a multi-level backtrace instead of a single-level location indicator.
Default: no

`exmdb_ics_log_file`
Log ICS/synchronization requests (and their results) to this file.
Default: (empty)

`exmdb_optimize_stm`
A debug knob to turn on/off a specific subset of the SQLite prepared statements with extended lifetimes used in the vincinity of property retrieval in conjunction with Content Table querying (e.g. exmdb_server::query_table, or MAPI ropQueryRows), Content Table matching (exmdb_server::match_table, or MAPI ropFindRow), non-instanced full message retrieval (exmdb_server::read_message, often used by our exporters).
Default: on

© grommunio GmbH — CC BY-SA 4.0

`exmdb_parallelize_schemaup`
The maximum number of EXRPC request parsing threads that can concurrently perform schema upgrades in sqlite database files. (Other threads must wait.) This directive is meant to limit the amount of disk I/O.
Default: 4

`exmdb_parallelize_sqliteshut`
On shutdown, close SQLite databases in parallel with as many threads as declared by this directive, capped by the number of processors actually available in the system. This improves the speed of the shutdown procedure. As of sqlite version 3.47, the parallel portion (\alpha in the context of Amdahl's law) is about 0.82.
Default: 4

`outgoing_smtp_url`
See gromox.cfg(5):outgoing_smtp_url.

Configuration directives (exmdb_provider.cfg)

The usual config file location is /etc/gromox/exmdb_provider.cfg.

`cache_interval`
The inactivity timer after which a mailbox's sqlite files are closed.
Default: 1min
Lower limit: 1s

`dbg_synthesize_content`
When this directive is set to 1, missing content files will not be regarded as an error and the respective attachment or property is delivered with a replacement string. If set to 2, a replacement string is always delivered; mode 2 is useful for reducing the amount of data downloaded when debugging ICS.
Default: 0

`enable_dam`
When set to **on**, inbox rule processing is allowed to create Deferred Action Messages (DAM). Furthermore, the "Deferred Actions" folder will have its contents shown. / Conversely, if this directive is **off**, no DAMs will be created, and the DAM folder in inboxes is presented as empty to clients (even if it has content from earlier).
Outlook's DAM handling is poor and if you experience a crash with a primary mailbox that is in non-cached/online mode a few seconds after Outlook has opened it, turn this option off for mitigation.
Default: on

`exmdb_body_autosynthesis`
When a client requests either PR_BODY, PR_HTML or PR_RTF_COMPRESSED, but that property does not exist on a particular message, automatically synthesize the data on-the-fly from another of the

© grommunio GmbH — CC BY-SA 4.0

available formats.
Default: on

`exmdb_file_compression`
Compress content files (bodytexts and attachments). Possible values: **no**, **yes** (zstd-6), **zstd-***level* (level=1..19).
Default: zstd-6

`exmdb_hosts_allow`
A space-separated list of individual host addresses that are allowed to converse with the exmdb service. The addresses must conform to gromox(7) § "Host addresses". No networks and no CIDR notations are permitted. This option deprecates the /etc/gromox/exmdb_acl.txt file used before Gromox 2.8.
Default: ::1

`exmdb_listen_port`
The TCP port number for exposing the timer service on.
Default: 5000

`exmdb_pf_read_per_user`
Keep public folder read states per user (1) or keep one state for all users (0).
Default: 1

`exmdb_pf_read_states`
When set to 0, messages in public stores/folders will always be shown as read and the folder summary will reflect that.
When set to 1, messages will have new/read markings but PR_CONTENT_UNREAD will indicate 0 new messages at all times.
When set to 2, PR_CONTENT_UNREAD indicates the number of new messages for the particular user. (Outlook does not show this number; in Folder Properties, the radiobox is even greyed out.)
Default: 2

`exmdb_private_folder_softdelete`
Enables soft-delete support for folders in private stores. (This feature is experimental.) Public folders always have this on.
Default: yes

`exmdb_schema_upgrades`
This directive controls whether database schemas are automatically upgraded when a mailbox is loaded. During this time, the mailbox is unavailable and operations on it will be delayed. Connection aborts, if any, would be due to timeouts in clients rather than servers. (The procedure takes roughly 36sec per gigabyte of exchange.sqlite3 worth of data, or 36sec per about 110k messages, on a 3700X CPU, single-thread. The file can also temporarily grow to double its size, so ample disk space may be required.)
Default: yes

© grommunio GmbH — CC BY-SA 4.0

`exmdb_eph_prefix`
A path for where variadic data files that are process-temporary can be stored. This may be used to keep the tables.sqlite3 file off an NFS-backed homedir. Required disk space scales linearly with open table handles and linearly with messages in the opened folders, at about 80 bytes per messages. (In other words, don't lump molasses of messages into a single folder in a shared mailbox read by multiple people.) When you create the directory, set its ownership to gromox:gromox and mode to 0770.
Default: (empty)

`exmdb_search_pacing`
When initially populating a search folder (static or dynamic), yield the lock on the sqlite database (file descriptor) after so many messages to give other clients a chance to perform an action.
Default: 250

`exmdb_search_pacing_time`
When initially populating a search folder (static or dynamic), yield the lock on the sqlite database (file descriptor) after this much time has passed to give other clients a chance to perform an action.
Default: 2s

`exmdb_search_nice`
Run the search folder population thread with adjusted niceness, which affects process scheduling. This is not an absolute priority as the nice(1) command would use, but a relative one, as per the nice(2) syscall. The allowed range in Gromox is 0 .. 19; negative values are not supported (and not meaningful, because Gromox will usually be running in an unprivileged setting where it is not possible to raise the priority).
Default: 0

`exmdb_search_yield`
Make the search folder population thread not only give up the lock on the sqlite database temporarily, but also invoke pthread_yield(3) after every work block (cf. exmdb_search_pacing).
Default: no

`exrpc_debug`
Log every incoming exmdb network RPC and the return code of the operation in a minimal fashion to stderr. Level 1 emits RPCs with a failure return code, level 2 emits all RPCs. Note that direct function calls from within the process image are not logged this way, so this will not show exmdb_provider invocations from exchange_emsmdb(4gx). Note the daemon log level needs to be "debug" (6), too.
Default: 0

`listen_ip`
The IPv6 socket address for exposing the exmdb service on. The address must conform to gromox(7) § "Host addresses".
Default: ::1

© grommunio GmbH — CC BY-SA 4.0

`max_ext_rule_number`
Default: 20

`max_router_connections`
As a exmdb server, permit at most this many inbound connections for the purpose of sending notifications on these channels. Note that every incoming TCP connection starts as a data connection and only becomes re-classified as "notification" once the LISTEN_NOTIFICATION RPC has been issued by the client.
Default: unlimited (only limited by ulimits)

`max_rpc_stub_threads`
As a exmdb server, permit at most this many inbound connections for commands.
Default: unlimited (only limited by ulimits)

`max_rule_number`
Default: 1000

`max_store_message_count`
The maximum number of messages any one particular message store is allowed to keep. The technical limit is somewhere around 2^47.
Default: *0* (no limit)

`populating_threads_num`
The number of threads to spawn that will work on asynchronous search folder population.
Default: 4

`rpc_proxy_connection_num`
The maximum number of (idle) connections towards (other) Information Store homeservers that are kept alive for re-use.
Default: 10

`sqlite_busy_timeout`
This sets the maximum time that exmdb_provider/libsqlite will wait in an attempt to start an exclusive write transaction, and if the waittime is exceeded, the surrounding exmdb RPC is aborted with error.
Default: *60s* (max: *1h*)

`sqlite_debug`
If set to 1, every query given to SQLite prepare/execute is logged. If set to 0, only failed queries are logged. (It cannot be made completely silent, since our queries ought to never fail.)
Default: 0

`table_size`
Maximum number of concurrently active mailboxes.
Default: 5000

`x500_org_name`

© grommunio GmbH — CC BY-SA 4.0

Default: (unspecified)

Sharding / Multiple Home Server Cluster

Each exmdb_provider (server) instance evalutes the SQL columns **users.homeserver/servers.hostname** for private stores and **domains.homeserver/servers.hostname** for public stores against the **host_id** directive to determine if it should be served. exmdb_client component, as used by zcore, imapd, etc. and a plethora of command-line utilities, likewise evaluates those SQL columns to determine which server to contact.

The magic value 0 in the homeserver column means any and all server instances consider themselves authoritative. This has serious implications (concurrent writes by multiple hosts) and so, using homeserver=0 for accounts with an actual maildir should be avoided in multiserver setups. (Using 0 for Contact Objects, and using 0 for single-server setups is tolerable.)

For a decidedly single-server cluster, this cicrumstance requires attention too. For example, picture the server.hostname SQL column containing "server01.example.com", which resolves to "2001:db8::1" — this conflicts with the ::1 bind address, making connection attemps fail.

The file /etc/gromox/exmdb_list.txt became obsolete in the development phase that followed the Gromox 3.4 release.

Network protocol

The transmissions on the socket are simple concatenations of protocol data units built using the NDR format. The PDU length is present within the PDU itself near the start.

```
{
  leuint32_t length;
  char pdu[];
}

pdu := {
  uint8_t call_id;
  string directory;
  switch (call_id) {
    ...
  }
}
```

Files

- *data_file_path*/mail_bounce/

© grommunio GmbH — CC BY-SA 4.0

config_file_path and *data_file_path* is determined by the configuration of the program that loaded the exmdb_provider component.

Environment

- ISTORE_JUST_ONE: A development knob that, in conjunction with gromox.cfg:istore_standalone=2/3, starts separate worker(s) only for the one specified user/directory.
- ISTORE_WORKER: A development knob that overrides the executable to use for istore workers. (In conjunction with ISTORE_JUST_ONE, this allows for running one user's mailbox under ASAN/TSAN.)

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-abktconv(8)

Name

gromox-abktconv — Utility for converting between ABKT and JSON

Synopsis

gromox-abktconv {**-b**|**-j**} [**-gw?**] [**-c** *cpid*]

Description

gromox-abktconv can be used to convert between data streams as specified in MS-OXOABKT and a textual representation. It reads and writes to standard input and output, respectively.

Options

`-b`
Produce type-1 ABKT from JSON.

`-c` *cpid*
When converting to ABKT (-b), convert strings to the given codepage and emit them as 8-bit strings.

`-g`
When converting to ABKT (-b), emit extraneous gaps in the data stream to mimic what Exchange would do. Without -g, the ABKT stream will have no unnecessary gaps.

`-j`
Produce JSON from ABKT type-1 or type-2.

`-w`
When converting to ABKT (-b), emit strings in UTF-16 form. This is the default.

`-?`
Display option summary.

Normative references

- MS-OXOABKT: Address Book User Interface Templates Protocol
- oxoabkt.rst: Type-2 ABKT template protocol

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **gromox-abktpull**(8gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-abktpull(8)

Name

gromox-abktpull — Utility to extract ABKT templates from LDIF

Synopsis

ldapsearch *...* | **gromox-abktpull**

Description

gromox-abktpull reads LDIF from standard input and extracts the values from attributes matching /DisplayTable/ — i.e. generally originalDisplayTable, originalDisplayTableMSDOS, addressEntryDisplayTable and addressEntryDisplayTableMSDOS — and saves them in separate files in the current working directory.

Normative references

- RFC 2849: LDAP Data Interchange Format

See also

gromox(7), **gromox-abktconv**(8gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-cleaner.service(8)

Name

gromox-cleaner.service, gromox-cleaner.timer — Soft-deleted message/attachment removal

Description

gromox-cleaner.service is a systemd unit that invokes gromox-mbop(8) to clear out soft-deleted messages from stores, as well as reclaim space by deleting orphaned on-disk attachment files which are no longer referenced by any message. Usually, the service is invoked periodically from gromox-cleaner.timer.

Configuration

/etc/gromox/gromox.cfg is read for the following directives:

> `softpurge_purgetime`
> Controls the lifetime of soft-deleted messages. Messages which have the soft-delete flag (a.k.a. "hidden flag" in Exchange) and which have their PR_LAST_MODIFICATION_TIME older than this interval are hard-deleted.
> Default: 30days

See also

gromox(7), **gromox-mbop**(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-compress(8)

Name

gromox-compress — Utility to recompress Gromox content files

Synopsis

gromox-compress --cid {*directory*| *file*...}

Description

gromox-compress compresses content files (attachments, bodytext) in an existing mailbox after the fact. This utility is useful because the "exmdb_file_compression" config directive only controls compression in the groupware servers for newly created content files.

Options

`--cid`
Treat all arguments given on the command-line as CID directories, and process them appropriately.

`-n`
Dry run. In essence, this only builds the file lists and runs no compressors.

`-z` *level*
Compression level to use. Defaults to 6.

Examples

Compress some:

```
gromox-compress --cid /var/lib/gromox/user/0/1/cid /var/lib/gromox/user/0/2/cid
```

Or string that further to compress cid directories of all mailboxes:

```
find /var/lib/gromox/{user,domain} -type d -name cid -exec gromox-compress --cid {} +
```

Formats

- cid/[0-9]+: content file, with proptag-dependent header and trailer

© grommunio GmbH — CC BY-SA 4.0

- cid/[0-9]+.v1z: content file, with proptag-dependent header and trailer, compressed
- cid/[0-9]+.zst: content file, headerless, compressed

See also

gromox(7), **exmdb_provider**(4gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-dbop(8)

Name

gromox-dbop — User database maintenance utility

Synopsis

gromox-dbop [-CU] [**--create-old**] [**-c** *mysql_adaptor.cfg*]

Options

`-C`
Create the initial set of tables for the user information database (in MySQL).

`-U`
Upgrade the schema of the user information database. The database (MYSQL_DBNAME) should exist and be blank.

`--create-old`
Create blank initial database using version 0. (This can be used for testing the upgrade procedure.)

`-c` *mysql_adaptor.cfg*
Path to a configuration file that defines MYSQL_HOST, MYSQL_USERNAME, etc. If omitted, gromox-dbop will read /etc/gromox/http.cfg to locate mysql_adaptor.cfg to locate the mysql parameter.

See also

gromox(7), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-dscli(8)

Name

gromox-dscli — AutoDiscover command line utility

Synopsis

PASS=*secret* **gromox-dscli** [**--eas**] [**-H** *https://host/path*] [**-h** *hostonly*] [**-e** *emailaddr*] [**-v**]

Description

This utility facilitates sending/receiving AutoDiscover request/responses to/from a server for testing.

Options

`-H` *https://host/Autodiscover/Autodiscover.xml*
The full URL for making the request. Useful when the request URI is non-standard. Use of -H disables DNS SRV lookup.
Default: https://localhost/Autodiscover/Autodiscover.xml

`-e` *user@domain.example*
Username to send along in the request.
Default: (none)

`-h` *host*
The hostname to use for making the request. From this, the full URL is constructed as "https://*host*/Autodiscover/Autodiscover.xml". Use of -h disables DNS SRV lookup.
Default: derived from -e argument, or *localhost* (when -x is used)

`-u` *username*
Use a distinct username for authentication.
Default: inherited from -e argument

`-v`
Be verbose. Log messages are emitted to stderr, the HTTP/XML request is emitted to stderr, and the HTTP/XML response (if any) is emitted to standard output. (These choices allow the output XML to be fed to e.g. xmllint for pretty-printing).

`-x` */o=1234578/ou=Exchange Administrative Group (FYDIBOHF23SPDLT)/cn=Recipients/cn=username*

© grommunio GmbH — CC BY-SA 4.0

> LegacyDN field to send along in the request.
> Default: (none)

> `--eas`
> Use the request schema from [MS-ASCMD].

> `--ac`
> Perform Mail Autoconf request instead of AutoDiscover. MA is used by e.g. (Mozilla) Thunderbird, (GNOME) Evolution, (KDE) KMail, (KDE) Kontact, K9 Mail, to name a few.

Examples

Discover your store:

```
PASS=letmein gromox-dscli -e user@domain.example
```

Test public store discovery:

```
PASS=letmein gromox-dscli -e public.folder.root@domain.example -u
user@domain.example
```

Normative references

- MS-OXDISCO: Autodiscover HTTP Service Protocol
- MS-OXDSCLI: Autodiscover Publishing and Lookup Protocol

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

gromox-e2ghelper(8)

Name

gromox-e2ghelper — Helper program for exchange2grommunio.ps1

Synopsis

gromox-e2ghelper [**-u** *username*] [**--** *pff-args*...]

Options

> `-u` *emailaddr*
> Username of store to import to.

All non-options arguments are passed through to gromox-pff2mt(8).

Description

e2ghelper is an implementation detail in the exchange2grommunio.ps1 script and not meant to be called directly.

© grommunio GmbH — CC BY-SA 4.0

gromox-edb2mt(8)

Name

gromox-edb2mt — Utility for analysis of Exchange .edb files

Synopsis

gromox-edb2mt -l *mdb01.edb*

gromox-edb2mt [**-pt**] *mdb01.edb* **-x** *mbid*

Description

gromox-edb2mt reads one mailbox from an Exchange .edb file, and dumps the structure to stderr with -t/-p.

edb2mt is not currently suitable for import of mailbox data, see also https://github.com/libyal/libesedb/issues/68.

Options

`-l`
Show the mailbox table from the .edb file.

`-p`
Show properties in detail (enhances **-t**).

`-t`
Show a diagnostic tree view of the source data as it is being read.

`-x` *mbid*
Extract the given mailbox from the .edb file.

`--loglevel` *n* Maximum verbosity of general logging (not connected to **-p**, **-t** or **-v**). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

Obtaining edb files

On the Exchange server, stop the service known as "Microsoft Exchange Information Store" (as reported in services.msc) or "Microsoft.Exchange.Store.exe" (as reported in Task Manager). This will release file locks on edb files so that they can be copied elsewhere.

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **gromox-import**(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-eml2mbox(8)

Name

gromox-eml2mbox — Utility for converting RFC5322 Internet Mail messages into a RFC4155 mbox-format mailbox

Synopsis

gromox-eml2mbox [*file*...] **>***out.mbox*

Description

gromox-eml2mbox reads one or more RFC5322-formated e-mail messages and re-exports them as a Unix mbox to stdout. The purpose is to make messages openable with command-line MUAs such as Alpine https://alpineapp.email/.

Examples

When **-** is given as an argument, standard input is expected to contain a **list of filenames** (i.e. indirection). This can help when *.eml leads to an expansion the system cannot handle:

gromox-eml2mbox *.eml >all.mbox*

May result in a "Argument list too long" error in sh. Remedy for that:

find . -maxdepth 1 -type f -name "*.eml*" | *gromox-eml2mbox* - >all.mbox

To convert a single message coming from elsewhere via pipe:

gromox-exm2mbox <1.eml >1.mbox

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

gromox-eml2mt(8)

Name

gromox-eml2mt — Utility for analysis/importing various formats

Synopsis

gromox-eml2mt [**-Ppt**] *file.eml*[...]

gromox-ical2mt [**-Ppt**] *file.ics*[...]

gromox-mbox2mt [**-Ppt**] *file.mbox*[...]

gromox-tnef2mt [**-Ppt**] *file.tnef*[...]

gromox-vcf2mt [**-Ppt**] *file.vcf*[...]

Description

gromox-eml2mt reads one or more RFC5322-formatted e-mail messages and re-exports the data in a Gromox-specific mailbox transfer format to stdout for consumption by pipe by the gromox-import(8) program. Each file must contain at most one RFC5322 message.

When called as gromox-mbox2mt, the input is treated as RFC4155-formatted Unix mailbox.

When called as gromox-ical2mt, the input is treated as RFC5545/5546-formatted calendaring and scheduling objects (.ics).

When called as gromox-vcf2mt, the input is treated as RFC4770/6530-formatted vCard objects (.vcf).

When called as gromox-tnef2mt, the input is treated as a MS-OXTNEF object.

eml2mt will resolve email addresses to Gromox objects already, so the emitted data stream should be consumed by a gromox-import invocation on the *same* Gromox cluster.

All objects in the output stream are unanchored so that the -B option of gromox-import can be used to select placement. (Prior to Gromox 2.46, some modes like --ical and --vcard emitted an anchor.)

Options

> `--decap`=*n*
> Once a MAPI message object is constructed, select attachment number *n*'s embedded message as the "top-level" message and discard the rest of the outer message. *n* is 1-based.

© grommunio GmbH — CC BY-SA 4.0

`--ical`
Treat all file arguments as iCalendar input. This is the default if the program was invoked as gromox-ical2mt.

`--loglevel` *n* Maximum verbosity of general logging (not connected to **-p**, **-t** or **-v**). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`--mail`
Treat all file arguments as Internet Mail input. This is the default if the program was invoked as gromox-eml2mt.

`--mbox`
Treat all file arguments as Unix mboxes. This is the default if the program was invoked as gromox-mbox2mt.

`--oneoff`
Do not resolve email addresses to EX addresses, but to ONEOFF instead. This way, streams created by eml2mt can be imported into *other* Gromox clusters that do not have the same user set.

`--tnef`
Treat all file arguments as Transport Neutral Encapsulation Format objects. This is the default if the program was invoked as gromox-tnef2mt.

`--vcard`
Treat all file arguments as vCard input. This is the default if the program was invoked as gromox-vcf2mt.

`-P`
Enable super-pedantic mode when parsing VCARDs and reject everything that is not recognized. (Not recommended)

`-p`
Show properties in detail (enhances **-t**). Using -p twice prints properties with human-readable mnemonic names. The BIN2TXT_MODE environment variable can be set to control how binary properties are presented; possible values are "co" (C string representation with octal escapes), "cstr" (C string representation, any form) and "hex" (two-hex-nibble representation), "txt" (compact representation for light-binary text-heavy content).

`-t`
Show a diagnostic tree view of the source data as it is being read.

Examples

Import of an RFC5322 message (sometimes with .eml file extension) to drafts:

gromox-eml2mt msg.eml | gromox-import -u recipient@domain.example -B drafts

© grommunio GmbH — CC BY-SA 4.0

Import of calendar objects:

gromox-ical2mt meeting.ics | gromox-import -u recipient@domain.example

See also

gromox(7), **gromox-export**(8), **gromox-import**(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-exm2eml(8)

This is an alias. See **gromox-export(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-exm2ical(8)

This is an alias. See **gromox-export(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-exm2mt(8)

This is an alias. See **gromox-export(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-exm2tnef(8)

This is an alias. See **gromox-export(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-exm2vcf(8)

This is an alias. See **gromox-export(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-export(8)

Name

gromox-export — Utility to export Gromox messages to various formats

Synopsis

gromox-exm2eml -u *user@domain.example* [*folder_id***:**]*message_id*

gromox-exm2ical -u *user@domain.example* [*folder_id***:**]*message_id*

gromox-exm2vcf -u *user@domain.example* [*folder_id***:**]*message_id*

gromox-exm2mt -u *user@domain.example* [**-ars**] {*folder_spec*|**f***folder_id*|[*folder_id***:**]*message_id*}... >file.mt

gromox-exm2tnef -u *user@domain.example* [*folder_id***:**]*message_id* >file.tnef

Description

gromox-export reads folders or messages from an exmdb information store and exports it in various formats. depending on which name it was invoked with. ("exm" is short for exmdb and refers to the network protocol used to talk to the message store.) Multiple messages may only be exported to GXMT. ACLs are not extracted yet. Export to PST is not possible: the external library libpff has only implemented reading.

Folders can be selected by symbolic name or by folder path (see gromox-mbop(8) section "Folder specification" for that). To select a folder by numeric ID, use the 'f' prefix, e.g. **f13** or **f0xd**. To select a message by numeric ID, use the ID **262145** or **0x10001**. The syntax **13:262145** reads a message using the Instance API (i.e. using a copy-on-write object) and validates that the message is indeed in that folder; this mode is only relevant for developers really.

An alternate way to get an EML representation is using grommunio-web's "Export as > EML file(s)" function from the context menu of a mail item.

Options

`--ical`
Selects iCalendar (RFC 5545/5546) as the output format. This is the default if the program was invoked as gromox-exm2ical.

© grommunio GmbH — CC BY-SA 4.0

`--mail`
Selects Internet Mail (RFC 5322) as the output format. This is the default if the program was invoked as gromox-exm2eml.

`--mt`
Selects Gromox Mailbox Transfer (GXMT) as the output format. This is the default if the program was invoked as gromox-export or gromox-exm2mt.

`--tnef`
Selects Transport Neutral Encapsulation Format (TNEF) as the output format. This is the default if the program was invoked as gromox-exm2tnef.

`--vcard`
Selects vCard (RFC 6350) as the output format. This is the default if the program was invoked as gromox-exm2vcf.

`-a`
Include Folder Associated Information (FAI), a.k.a. MAPI_ASSOCIATED messages in the export as well. (PST export from Outlook would omit them; hence this is also a separate option for gromox-export.)

`-p`
Show properties in detail (enhances **-t**).

`-r`
Process folders recursively. (Only in conjunction with GXMT export.)

`-s`
Mark objects for splicing when later imported. For example, objects exported from, and originally located in Sent Items are, upon import, placed in the target mailbox's Sent Items.

`-t`
As the source message is read, print a diagnostic tree view of the MAPI properties to stderr.

`-u` [*user*]**@***domin.example*
Source mail store from which to load the message from. For the public folder of a domain, leave out the local part, i.e. use **@***domain.example***.**

`--loglevel` *n*
Maximum verbosity of general logging (not connected to **-p, -t or -v). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.**
Default: 4 *(notice)*

Examples

- Export entire mailbox: gromox-exm2mt -u user@example.com -ar / >dump.mt

© grommunio GmbH — CC BY-SA 4.0

- Export most of the mailbox (as Outlook would): gromox-exm2mt -u u@e.xz -r IPM_SUBTREE >dump.mt
  Note: Favorites, Shortcuts, Quick Steps, Search Folders, some view settings, grommunio-sync states are all stored outside of IPM_SUBTREE. In Exchange, softdeleted items are stored outside of IPM_SUBTREE as regular messages; in Gromox 3.6, they are stored as invisible items with the actual folder and never part of an export (this might change in the future, though).
- Export two folders without subordinates: gromox-exm2mt -u u@e.xz IPM_SUBTREE/Foo IPM_SUBTREE/Bar >dump.mt
- Export a single message as RFC5322: gromox-exm2eml -u u@e.xz 0x10001 >10001.eml
- Export a MAPI contact as vcard: gromox-exm2vcf -u u@e.xz 0x30001 >30001.vcf

See also

> `gromox(7), gromox-eml2mt(8)`

© grommunio GmbH — CC BY-SA 4.0

gromox-ical2mt(8)

This is an alias. See **gromox-import(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-import(8)

Name

gromox-import — Utility for importing various mail items into Gromox

Synopsis

gromox-import [...] [**-Dcpt**] [**-u** [*localpart*]**@***domain.example*] < input.mt

Description

gromox-import reads a Gromox-specific mailbox transfer format data stream (GXMT) from standard input as generated by gromox-pff2mt(8) or other similarly named utilities. When the username option (-u) is provided, folders and messages included in the stream are written to the specified Gromox mail store, otherwise the utility will just parse the stream (useful with -t,-p).

There is no cross-check for already-imported messages at this time. Messages from the input stream will always generate new messages in the target mailbox.

Note that messages are always owned by the store they are in. Note that, when importing to a public store, you may need to set some additional permissions on that public store's folders so that a particular user is able to modify messages after they have been imported. (All import/export utilites use superuser access and need no permissions.)

Options

`-B` *name*
Place unanchored messages (loose objects that do not request placement in any particular folder) in the folder specified by -B. The accepted names are documented in the gromox-mbop(8) manpage under section "Folder specification". Unanchored messages can be produced by e.g. gromox-eml2mt, gromox-tnef2mt, gromox-pff2mt --only-obj, gromox-kdb2mt --only-obj.
Default: draft

`-D`
Use delivery mode, i.e. treat messages as if they were sent through the mail system rather than being imported. As such, messages are subject to inbox rules, and folders in the MT stream will be ignored altogether, which also implies that -B is ignored.

`-c`
Continuous operation mode. Errors during the import of a message are reported, but the utility will continue with importing more messages. The default is to exit after reporting a delivery error.

© grommunio GmbH — CC BY-SA 4.0

`-p`
Show properties in detail (enhances **-t**).

`-t`
Print folder/message summary as these items are processed.

`-u` [*user*]**@***domain.example*
Target mail store which to import mails to. For the public folder of a domain, leave out the local part, i.e. use **@***domain.example***.**

`-v`
Verbose mode: report individual FID/MIDs for newly created objects.

`-x`
When importing an MT stream that does not request message splicing, the utility will raise an error if a to-be-created folder already exists. This behavior can be turned off with -x. This option can be thought of what mkdir's -p option would do.

`--loglevel` *n* **Maximum verbosity of general logging (not connected** to **-p, -t or -v). 1=crit, 2=error, 3=warn, 4=notice,** 5=info, 6=debug.
Default: 4 *(notice)*

`--skip-notif`
Skip emitting MAPI notifications (when -D is used). This is for development only.

`--skip-rules`
Skip executing rules (when -D is used).

Exit status notes

All input streams must have at least a valid magic signature (first eight bytes) — an input stream of length zero is treated as invalid rather than as a valid stream without any commands.

Examples

PFF import to a private mailbox:

gromox-pff2mt sample.pst | gromox-import -u user@domain.example

PFF import to a public folder:

gromox-pff2mt sample.pst | gromox-import -u @domain.example

See also

`gromox(7), gromox-pff2mt(8)`

© grommunio GmbH — CC BY-SA 4.0

gromox-kdb2mt(8)

Name

gromox-kdb2mt — Utility for analysis/importing of Zarafa/Kopano SQL-stored mailboxes

Synopsis

gromox-kdb2mt [**-pstv**] [**--sql-host** *hostname*] [**--sql-user** *identity*] [**--sql-port** *number*] [**--sql-db** *name*] **--src-attach** *dir* {**--mbox-guid** *mboxguid*|**--mbox-name** *username*|**--mbox-mro** *username*} [...]

Description

gromox-kdb2mt reads one store from a Zarafa/Kopano SQL database and, for attachments, the associated filesystem. The data is then re-exported in a Gromox-specific mailbox transfer format to stdout, intended for consumption by pipe by the gromox-import(8) program. Optionally, kdb2mt can print a summary of the hierarchy during extraction.

The SQL server that carries the Zarafa/Kopano database for the home server of the user must be active. Databases with a schema version **n61** or newer (ZCP versions 7.0.3 and onwards, and all KC versions) are supported. The kopano-server(8) process need not be running. Its LDAP need not be available either.

ACLs can be extracted, but, owing to the independence of the Kopano LDAP data model, require pre- or post-processing (see below).

The directory of Kopano attachments of the home server should be made available in the mount namespace wherever gromox-kdb2mt runs.

Properties of the *store object itself* are only shown (if and when -p is used), but never dumped to the MT data stream. Properties of the store *root* folder object are(!) transferred.

Options

`-p`
Show properties in detail (enhances **-t**).

`-s`
Map the source mailbox folder hierarchy (and its subobjects) to the target mailbox's hierarchy and splice objects accordingly. Only use -s when both the source side and the target side are private stores. See the section "Splice mode" below for more information. When --with-hidden is not

© grommunio GmbH — CC BY-SA 4.0

selected, -s will imply --without-hidden by default to avoid polluting e.g. "QuickStep settings" (which may already have settings).

`-t`
Show a diagnostic tree view of the source data as it is being read.

`-v`
Print message count progress while processing larger folders. This option has no effect if (the even more verbose) **-t** option was used.

`--loglevel` *n* Maximum verbosity of general logging (not connected
to **-p**, **-t** or **-v**). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`--user-map` *file*
Use the given file to perform ACL mapping and ZARAFA Address Type rewriting. See sections "ACL Extraction" and "ZARAFA Address Type" below for details. The file format is described in kdb-uidextract(8).

`--sql-host` *hostname*
Hostname for the source SQL connection.
Default: (MySQL default; *localhost*)

`--sql-port` *number*
Port for the source SQL connection.
Default: (MySQL default; automatic)

`--sql-user` *identity*
Username for the source SQL connection.
Default: root

`--sql-db` *dbname*
Database name.
Default: kopano

`--src-attach` *directory*
Required specification to the /var/lib/kopano/attachments directory mounted somewhere locally. (To skip over file-based attachments, use the empty value, i.e. **--src-attach ""**.)

`--mbox-guid` *guid*
Selects the mailbox with the particular GUID for extraction. (This may be used to read orphaned stores.)

`--mbox-user` *username*
Scan the user map (cf. --user-map) for a mailbox which was used by the given username, and use it for extraction. (To get a listing of all stores, use **--mbox-user ""**.)

© grommunio GmbH — CC BY-SA 4.0

`--mbox-mro` *username*
Scan the source database for a mailbox which appears to have last been used by *username*, and use it for extraction. There are a number of **caveats** related to this lookup; see the section "Store lookup by name" further below. (To get a listing of all stores, use **--mbox-mro ""**.)

`--l1` *x*, **--l2** *y*
If you are using "attachment_storage=files_v1-*x*-*y*" in /etc/kopano/server.cfg, call kdb2mt with the L1 and L2 options.

`--only-obj` *hid*
Extract just the object with the given hierarchy id. This option may be specified multiple times to selectively extract more objects. In the output stream, objects so extracted will be declared as unanchored so they can be imported to a folder of choice later (cf. gromox-import's -B option).

`--acl`={*auto*|*no*|*noextract*|*extract*|*convert*}
Handling for ACLs on MAPI objects: ignore, extract to synthetic address, or convert to an e-mail address. *auto* plus the presence of --user-map leads to *convert*. *auto* plus the absence of --user-map leads to *noextract*. See the section "ACL Extraction" below for some more details.

`--with-hidden`, **--without-hidden**
This option controls the import of folders that have PR_ATTR_HIDDEN=1.

Splice mode

Normally, kdb2mt will have a folder mapping table that specifies one entry, which is "(Source root)" to "(Target root)\Top of Information Store\Import of GUID @date". Any objects within an entry's source folder (including more folders) are copied. This default entry makes sure absolutely everything is imported, without loss.

Using the -s option, this behavior will change. The default mapping is replaced by one that will intermix imported folders with an existing hierarchy. Specifically, special folders such as the root folder item, Top of Information Store (a.k.a. TOIS or IPM_SUBTREE), Inbox, etc. are mapped. This only works when both the source and target are private stores!

Special folders are identified by metadata, not by name. This way, kdb2mt can support localized folder names and correctly map, for example, a German "Gesendete Elemente" to a French "Éléments envoyés" (Sent Items). Regular folders will be processed normally (by name), e.g. "(Source root)\Top of Information Store\Invoices" will be imported at "(Target root)\Top of Information Store\Invoices".

The -s option is most useful when importing one's own store from one system to another that's new and blank. If importing someone *else's* store into yours, leaving out -s is normally the desired behavior, since you may not want want to mix your (existing) with their mails.

© grommunio GmbH — CC BY-SA 4.0

Environment variables

`SQLPASS`
Password for the source SQL connection.

Examples

Common scenario (Separate hosts)

When Gromox and Kopano run on different hosts, and you wish to have the Gromox host to initiate all necessary connections.

Step 1. Establish an sshfs mount. This is used to get at the attachments directory of Kopano Core. Command:

```
sshfs root@kp:/var/lib/kopano/attachments /mnt
```

For this to work, root logins need to be possible in some form (password or pubkey-based authentication).

Step 2. Establish an SSH tunnel. This is used to get at the MariaDB/MySQL database, assuming that this database is not already accepting connections on port 3306. Command:

```
ssh -L 12345:localhost:3306 root@kp
```

This way, the database can be accessed as 127.0.0.1:12345 later.

Step 3. Locate the MariaDB connection parameters that you want to use. You can use the MariaDB "root" user (if available), or reuse the credentials from /etc/kopano/server.cfg (often a "kopano" user).

Step 4. Run the conversion. The use of "127.0.0.1" is necessary to bypass the special meaning of "localhost" (which implies the use of an AF_LOCAL socket, e.g. /run/mysql/mysql.sock). Command:

```
SQLPASS=kopanosqlpass gromox-kdb2mt --sql-host 127.0.0.1 --sql-port 12345 --sql-user kopano --src-attach /mnt --mbox-mro jdoe | gromox-import -u user@domain.example
```

Done! The speed of the operation depends on the capabilities of the network and the source database (latency more so than throughput).

Other options

If the Gromox host is not allowed to connect to the Kopano host for reasons of networking and/or firewall setups, there are plenty of other ways to carry over the data. Administrators are asked to use their experience to mix and match the plethora of utilities available at their disposal. Possible operations include mysqldump(1), sftp(1), rsync(1), tar(1) and curl(1).

© grommunio GmbH — CC BY-SA 4.0

Store lookup using Kopano tools

If kdb2mt's built-in heuristic --mbox-mro resolution mechanism is not adequate enough, you can use utilities from the Kopano installation, provided that is still active.

- `kopano-storeadm -M` is the gold standard. This dumps the entire store list, in JSON representation. The GUIDs can then be used together with --mbox-guid.
- The global "SYSTEM" user object in Kopano also happens to have a private store, titled "Inbox - SYSTEM". This store however is practically empty and it is unlikely it will ever need extraction. Alternatively, its GUID can also be shown with `kopano-admin --details SYSTEM`.
- The global public store in Kopano, if it exists, is owned by the "Everyone" *group object*. In kopano-storeadm output, it can be found by looking for the display name "Public Folders". There is no way to see the GUID via kopano-admin.
- Just for completeness: There is no per-company SYSTEM user (and hence no store). If anything, companies re-use the global SYSTEM user as a member.
- The per-company public folder, if it exists, is owned by the respective *company object*. In kopano-storeadm output, it can be found by looking for the display name "Public Folders - MyCompany". Alternatively, the GUID can also be shown with `kopano-admin --type company --details MyCompany`.

Store lookup by name

Generally, Kopano SQL databases do not store usernames. Store ownership is recorded with a Kopano-level numeric ID, which itself is mapped to a site-specific attribute of an authentication service, e.g. the uidNumber field of an LDAP. Only the authentication service would know the username, and kdb2mt does not rely on the existence of such authentication provider.

Every store has a metadata field for the **most recent owner** (MRO). This field was intended for orphaned stores and has informational value only. The MRO field is not always updated by Kopano services, which can lead to --mbox-mro not necessarily finding an expected match. In particular, kopano-server misses doing the MRO update on store detach, and on changes to the username in LDAP.

Furthermore, because it is possible to detach/orphan and create a new store for a user (and repeatedly so), the MRO field value is **not unique** across the set of all stores.

Furthermore, the MRO field is missing the domain/company part of the username. Company public stores (in hosted setups) use the company name as MRO. This all contributes to --mbox-mro possibly matching multiple stores.

When more than one store matches in any way, kdb2mt will print the result set with GUIDs and exit, at which point you need to use --mbox-guid instead.

© grommunio GmbH — CC BY-SA 4.0

ACL Extraction

Because kdb2mt works completely LDAP-less, it knows nothing about users except for their numeric user object ID on the homeserver and a reference to an LDAP object (the so-called "Extern id", e.g. objectUUID/uidNumber). The user object ID is local to a kopano-server instance. The composition of the object ID and server instance GUID forms a unique token. ACEs are carried over such that that permissions for user with a given *objid* are transformed to the synthetic identity *objid*@*serverguid*.kopano.invalid.

> sqlite3 /var/lib/gromox/user/1/1/exmdb/exchange.sqlite3 sqlite> select *from permissions;* member_id folder_id username permission --------- --------- ---------- ---------- ---------- 1 15 default 2048 2 24 default 2048 3 2090545 256@aa8e2b20b2054ca98987ea1053c3bb16.kopano.invalid 1177

kdb2mt can be instructed to map these to a new email address using the --user-map command-line option. That file can be generated by executing kdb-uidextract(8) or the kdb-uidextract-limited(8) helper programs on the original, live Kopano system. (With some effort, the file can also be manually constructed. See that manpage for format details.)

ZARAFA Address Type

MAPI as a system supports referencing message participants with arbitrary address types, including, but not limited to, SMTP, EX (Exchange 4.0 Directory Service identifier), FAX numbers, etc.

Zarafa/Kopano systems define a "ZARAFA" address type, and the identifiers contain the username. How exactly it is formatted is system-dependent, cf. /etc/kopano/server.cfg:loginname_format.

See also

gromox(7), **gromox-import**(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-mailq(8)

Name

gromox-mailq — Local delivery agent queue lister

Synopsis

```
gromox-mailq
```

Description

gromox-mailq looks in the queue directory for delivery-queue(8gx) and prints the summaries for all queued mail.

See also

gromox(7), **delivery-queue**(8gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-mbck(8)

Name

gromox-mbck — Mailbox check and repair utility

Synopsis

gromox-mbck [**-p**] x.sqlite

Description

mbck can be used to check one or more mailboxes for problems, and optionally repairing them.

mbck directly operates on the filesystem, which is not ideal, but it is believed it is "mostly fine":

It is technically safe to run gromox-mbck while gromox-http has a mailbox open, provided Gromox is version >= 2.30. HOWEVER, gromox-http (still as of Gromox 2.36) does not anticipate databases being write-locked by another process for undue amounts of time (gromox.cfg:sqlite_busy_timeout), and signals an operational error to the caller. For example, mail cannot be delivered to the mailbox while mbck is running in repair/write mode.

Options

`-p`
Perform repairs / write operations. (Default: just readonly checks)

`-?`
Display option summary.

© grommunio GmbH — CC BY-SA 4.0

gromox-mbop(8)

Name

gromox-mbop — Mailbox operations utility

Synopsis

gromox-mbop [**-d** *mbox*|**-u** [*recipient*]**@domain.example**] *command* [command-args...]

Summary

gromox-mbop can be used to perform various administrative tasks on mailboxes. Technically, mbop contacts the respective home server(s) and instructs the server to perform the requested action. Unlike tools such as mbck, mkprivate/mkpublic/mkmidb, mbop itself does not do any direct filesystem access on the mailbox.

Global options

`-c`
Continuous operation mode. If a command in a series (e.g. with foreach.*) fails, do not stop.

`-d` */var/lib/gromox/user/1/2*
Lookup the mailbox parameters from the associated filesystem location.

`-u` [*user*]**@example.com**
Lookup the mailbox parameters by the associated username. (To access a public store of a domain, leave out the local part, i.e. use **@***example.com*.)

`-v`
Verbose mode.

Commands

- **(** command1 c1args **) (** command2 c2args **)**: command chaining
- cgkreset: reset synchronization state (PR_CHANGE_KEY, PR_PREDECESSOR_LIST)
- clear-photo: delete user picture
- clear-profile: delete user's PHP-MAPI profile
- clear-rwz: delete IPM.RuleOrganizer FAI messages from the inbox

© grommunio GmbH — CC BY-SA 4.0

- delmsg: issue "delete_message" RPCs for a mailbox
- echo-maildir: return maildir (for use with foreach.here.*)
- echo-username: return username (for use with foreach.*)
- emptyfld: remove objects from folders
- exaddrxlat: replace all "EX" type addresss in the mailbox's contents
- foreach.*: iterate over security objects
- freeze: halt all operations on a mailbox
- get-freebusy: test FB schedule lookups
- get-photo: retrieve user image from store and print to stdout
- get-websettings, get-websettings-persistent, get-websettings-recipients: retrieve settings for grommunio-web
- movemsg: move or copy a message between folders
- ping: cause a mailbox's sqlite files to be opened
- purge-datafiles: remove orphaned attachments/content files from disk
- purge-softdelete: remove soft-deleted items from a folder
- recalc-sizes: recalculate store size
- set-locale: reset UI language and special folders' names
- set-photo: read user image from stdin and save to store
- set-websettings, set-websettings-persistent, set-websettings-recipients: read new grommunio-web settings from stdin and save to store
- sync-midb: trigger a midb synchronization run
- thaw: unfreeze a mailbox
- unload: issue the "unload_store" RPC for a mailbox
- vacuum: issue the "vacuum" RPC for a mailbox
- zaddrxlat: replace all "ZARAFA" type addresses in the mailbox's contents

Further documentation

- SQLite recovery: https://docs.grommunio.com/kb/sqlite.html

© grommunio GmbH — CC BY-SA 4.0

Command chaining

(*subcommand1 sub1args…* **)**…

It is possible to run multiple mbop commands in sequence for a user. For the option parser to recognize when a command ends and the next one starts, each subcommand invocation shall be wrapped in **(** and **)**. This becomes even more handy in conjunction with the foreach.* pseudocommand.

Subcommands reading data from standard input (e.g. set-photo) cannot be realiably used with chaining, because stdin would be fully consumed the first time around and (…) does not cache the input for any subcommands.

Examples

Run two commands for a user: gromox-mbop -u a@b.de (purge-softdelete -r /) (purge-datafiles)

cgkreset

cgkreset resets Change Numbers on all folder and message objects, PR_CHANGE_KEY and PR_PREDECESSOR_LIST values. The use cases for cgkreset are:

- when the mailbox has CN corruption and Incremental Change Synchronization (by e.g. Outlook or grommunio-sync) is hampered (e.g. message flags/color updates not transferred)
- when the mailbox has CN corruption and gromox-http/emsmdb has thrown the error "INSERT INTO messages ... UNIQUE constraint failed: messages.change_number"

After execution, .ost files referencing the reset mailbox should be deleted.

clear-photo

The clear-photo command will delete the user picture. Note that, when there is no mailbox-level profile picture set, Gromox server processes may serve an image from another source, e.g. LDAP.

clear-profile

Similar to MSMAPI, PHP-MAPI keeps a MAPI profile which contains a store list and also the settings for grommunio-web. The clear-profile command will delete the copy of this data stored in exmdb. Note that zcore(8) may still hold a copy of the MAPI profile in memory and could write that back to exmdb, nullifying the effect of the clear-profile command. Also, if the store list is absent, a new one will implicitly be created when PHP-MAPI/zcore is used.

© grommunio GmbH — CC BY-SA 4.0

clear-rwz

Deletes IPM.RuleOrganizer FAI messages from the inbox.

delmsg

Synopsis

delmsg -f *folder_spec msgid*[...]

Description

This command hard-deletes messages from a store, including issuing proper PR_CHANGE_KEY metadata updates for the sake of Cached Mode clients.

The message IDs taken as arguments on the command-line should be of the GC-value form, i.e. as they appear in the the SQLite database. (For details about GCV, see glossary.rst in the source distribution.)

Subcommand options

> `-f` *folder_spec*
> The folder from which to delete the messages. See section "Folder specification" below for syntax details of *folder_spec*. (If a msgid is specified which is not located in the particular folder, that message will not be deleted.)

> `--soft`
> Perform a soft deletion.

emptyfld

Synopsis

emptyfld [**-MRa**] [**-t** *age*] [**--soft**] *folder_spec*...

Description

This command deletes objects from one or more folders. emptyfld is normally a one-shot server-side operation. The use of -R,-t is not covered by the existing network protocols, which means that, if either of these options is used, the mbop client program performs the desired recursion and/or timestamp matching locally. This incurs multiple round trips to the server and so takes a bit more time than a "trivial" emptyfld call.

Just to spell it out again explicitly, emptyfld can be in one of three modes:

- server-assisted operations:

© grommunio GmbH — CC BY-SA 4.0

    - clear contents and/or FAI, no time conditions, no recursion
    - clear contents and/or FAI, no time conditions, nuke subfolders (recursion barred)
- client-side traversal:
    - clear contents and/or FAI, with or without evaluating timestamps, with or without recursion into subfolders, with or without subfolder deletion if empty

Subcommand options

> `-M`
> Exempt normal messages from deletion.

> `-R`
> Recurse into subfolders.

> `-a`
> Select associated messages (FAI) for deletion.

> `-t` *relative-age*
> Limit deletion to messages which have a last modification timestamp older than *relative-age*. See gromox(7), section "Time span syntax" for details. There is currently no option for specifying absolute time.

> `--delempty`
> If, after message deletion, any subfolder is empty, delete it.

> `--nuke-folders`
> Unconditionally delete subfolders outright. For obvious reasons, deleting subfolders disables recursion via -R (because when they are deleted, there is nothing left to recurse into).

> `--soft`
> Switch from hard deletion to soft deletion.

Soft deletion notes

Soft deletion sets the soft-delete flag (also called "hidden" in Exchange) on messages and/or folders. Soft-deleted objects can be restored/unhidden by the user. Users are technically empowered to perform hard deletions as well, but most mail clients do not offer a user control (e.g. checkbox widget) for it, requiring the use of diagnostic utilities like MFCMAPI or gromox-mbop instead.

When a folder's soft-delete flag changes, the messages and subfolders within are left untouched; their soft-delete flag does not change. In fact, this behaves exactly like setting a directory in the file system to hidden.

© grommunio GmbH — CC BY-SA 4.0

Examples

- Clear one folder's contents like Outlook/grommunio-web: gromox-mbop -u a@b.de emptyfld --soft DRAFTS
- Outlook/grommunio-web behave differently when clearing trash! The equivalent mbop command is: gromox-mbop -u a@b.de emptyfld --soft --nuke-folders DELETED
- Deletion of objects in trash only if untouched for a while: gromox-mbop -u abc@example.com emptyfld -Rt 1week --soft DELETED

exaddrxlat

Synopsis

exaddrxlat -m *file* [**-r**] *folder_spec*[...]

Description

Scans messages in the given folders and replaces PR_ADDRTYPE="EX" and related properties by SMTP addresses.

MAPI as a system supports referencing message participants with arbitrary address types, including, but not limited to, SMTP, EX (Exchange/ActiveDirectory DN), FAX numbers, etc.

Contemporary Exchange systems (>= 2010) and/or Outlook versions should be adding secondary metadata when receiving/sending messages such that EX-typed participants become a EX+SMTP hybrid. Really old messages may lack the properties, and GUIs may fail to fill the To: line when replying to old messages. exaddrxlat will replace EX and EX+SMTP hybrids by pure SMTP ones, removing most historic Exchange Directory and ActiveDirectory information.

Options

> `-m` *file*
> User map to utilize for translation. See kdb-uidextract(8) for a format description.

> `-r`
> Process folders recursively.

foreach.*

Synopsis

foreach.*filter*[**.***filter*]* [**-j** *jobs*] *command* [command-args...]

© grommunio GmbH — CC BY-SA 4.0

Description

foreach.* is a pseudoaction for running another subcommands that gromox-mbop offers (e.g. ping, unload, purge-softdelete, etc.) for a number of users. Subcommands reading data from standard input (e.g. set-photo) cannot be realiably used with foreach, because stdin would be fully consumed the first time around and foreach does not cache the input for any subcommands.

Filters

- secobj: limit to objects that can be used in ACLs
- user: regular users
- dl: distribution lists (groups)
- sharedmb: shared mailboxes
- room: room objects
- equipment: equipment objects
- contact: GAB contact objects
- active: active entities
- susp: entities marked as "suspended"
- deleted: entities marked as "deleted"
- mb: entity has a mailbox directory defined
- here: entity has current host as homeserver (compares `hostname --fqdn` where mbop is run with the SQL.servers.hostname column)

There is no "all" filter. Security objects and Contacts are so vastly different that it just does not make sense to operate on them in the same run.

Options

> `-j` *jobs*
> Maximum parallel execution factor. (Experimental.) 0 means autosizing. Only ping/vacuum/unload support this, and the option is otherwise ignored. Use external tools like parallel(1) or make(1) for guaranteed parallelization.
> Default: 1

Examples

- Hard-delete all objects which are currently softdeleted: gromox-mbop foreach.mb.here purge-softdelete -r /

© grommunio GmbH — CC BY-SA 4.0

get-freebusy

Synopsis

get-freebusy [**-a** *start_time*] [**-b** *end_time*] [**-x** *username*]

Description

Runs the get_freebusy routine on the mailbox specified by the global -d/-u option(s) [or the mailbox currently in scope when using foreach.*], and asks for free/busy status within the given time period.

Options

`-a` {*yyyy-mm-dd***T***hh:mm:ss*[**Z**|**+***hhmm*|**-***hhmm*]|*unixtime*}
Limit returned events to occuring on or after this timestamp. Can either be a Unixtime or a ISO 8601 timestamp.

`-b` {*yyyy-mm-dd***T***hh:mm:ss*[**+***hhmm*|**-***hhmm*]|*unixtime*}
Limit returned events to ocurring before this timestamp.

`-x` *username*
Evaluate access control lists as this identity. If the -x option is omitted, ACL checks are omitted.

freeze

Synopsis

freeze [**--no-wait**]

Description

Tell the Information Store to halt all new operations on the mailbox. Outstanding operations that are currently in processing are allowed to complete. New requests for the mailbox will be rejected. Special requests like (mbop's) *unload* or *thaw* are nevertheless allowed in frozen state, for obvious reasons.

Options

`--no-wait`
Do not wait for outstanding operations to complete.

© grommunio GmbH — CC BY-SA 4.0

get-photo

Synopsis

get-photo >*somefile*

Description

Reads the user photo from the store and dumps it to stdout. If stdout is a terminal, no output is shown, in which case, if stderr is (also) a terminal, a summary will be shown there.

get-websettings

Synopsis

get-websettings >*file.json*
get-websettings-persistent >*file.json*
get-websettings-recipients >*autocomplete.json*

Description

Reads various grommunio-web settings from the store and dumps it to stdout.

movemsg

Synopsis

movemsg -f *folder_spec* **-t** *folder_spec* [**--copy**] *msgid*[...]

Description

Moves, or with **--copy**, copies one or more messages from one folder to another within the same store.

The message IDs taken as arguments on the command-line should be of the GC-value form, i.e. as they appear in the SQLite database. (For details about GCV, see glossary.rst in the source distribution.)

Subcommand options

> `-f` *folder_spec*
> The source folder. See section "Folder specification" below for syntax details of *folder_spec*.

> `-t` *folder_spec*
> The destination folder.

© grommunio GmbH — CC BY-SA 4.0

> `--copy`
> Copy the messages instead of moving them.

ping

Causes the respective mailbox to be opened by the server. (Any request to the information storage server causes the respective mailbox to be opened; and ping is technically just a no-op request type.)

sync-midb

Synopsis

sync-midb [**-f** *folder_spec*]

Description

Sends a request to midb for opening the mailbox and updating the midb-specific folder indices, as well as potentially building RFC5322 representations for newly-appeared messages. (Once the mailbox is open in midb, it uses asynchronous notifications to stay up to date.)

Options

> `-f` *folder_spec*
> Forcibly rerun the sync routine for a single specific folder. See section "Folder specification". In addition, the special keyword "all" is recognized.

purge-datafiles

The "purge-datafiles" RPC makes exmdb_provider remove attachment and content files from disk that are no longer referenced by any message.

purge-softdelete

Synopsis

purge-softdelete [**-r**] [**-t** *timespec*] *folder_spec*...

Description

This command hard-deletes all messages from a folder which are marked as soft-deleted. (The entire mailbox can be processed by specifying the root folder plus the -r option.)

© grommunio GmbH — CC BY-SA 4.0

Subcommand options

`-r`
Recurse into subfolders.

`-t` *timespec*
Specifies the minimum time to the last modification that soft-deleted messages must have before they are hard-deleted. See gromox(7), section "Duration specification" for timespec's syntax. Default: *0* (immediate deletion)

Examples

- To process an entire mailbox and wipe everything older than a few days: gromox-mbop -u abc@example.com purge-softdelete -r / -t 10d

recalc-sizes

Recalculates the store size.

set-locale

Synopsis

set-locale [**-Tv**] -l *id*

Description

First, the set-locale operation changes the "preferred language" setting for the user account. This affects the display of user interfaces like grommunio-web, and also affects the folder language selection when a mailbox is truncated/re-created with gromox-mkprivate(8).

Second, provided Gromox has default folder name translations for the desired locale, set-locale also resets the display names of the mailbox's built-in folders.

Options

`-T`
Run a trivial performance test against exmdb by repeatedly setting the folder names.

`-l` *d*
A locale identifier in the form of *language*_**[***territory***]**, where language is a ISO 639-1 code and territory is a ISO 3166-1 Alpha 2 code, e.g. ja_JP, pt_BR, pt_PT. This is like the well-known XPG/POSIX locale identifier syntax https://www.gnu.org/software/libc/manual/html_node/Locale-Names.html, but no Codeset and no Modifier should be used in Gromox.

`-v`
Verbose mode. (Same as global -v.)

© grommunio GmbH — CC BY-SA 4.0

Examples

- gromox-mbop -u abc@example.com set-locale -l ja_JP

set-photo

Synopsis

set-photo <*somefile*

Description

Reads a new user photo from standard input and writes it to the store.

set-websettings

Synopsis

set-websettings <*file.json*
set-websettings-persistent <*file.json*
set-websettings-recipients <*autocomplete.json*

Description

Reads new grommunio-web settings from standard input and writes it to the store.

unload

Normally, exmdb_provider(4gx) keeps stores open for up to exmdb_provider.cfg:cache_interval. The "unload_store" RPC to exmdb_provider(4gx) causes the sqlite database (in /var/lib/gromox/.../exmdb/exchange.sqlite3) to be closed. Any subsequent RPC may reopen it, though. The unload RPC is useful after a mailbox was deleted and/or reinitialized with grommunio-admin-api or tools like gromox-mkprivate(8)/gromox-mkpublic(8). [zcore also has store state in memory. This would also need to be purged — but there is no RPC for such action at this time.] unload will fail to succeed if there is still a client connected to the mailbox via a notification channel.

vacuum

Issue the SQLite ".vacuum" command on the user's exchange.sqlite3 file in an attempt to reclaim unused disk space and shrink it. This operation can potentially run for quite some time, during which the mailbox is inaccessible.

© grommunio GmbH — CC BY-SA 4.0

zaddrxlat

Synopsis

zaddrxlat -m *file* **[-r]** *folder_spec***[...]**

Description

Scans messages in the given folders and replaces PR_ADDRTYPE="ZARAFA" and related properties by normal SMTP addresses. See gromox-kdb2mt(8) section "ZARAFA Address Type" for some details. The use case for zaddrxlat is for fixing up addresses after the fact, i.e. when the administrator forgot to employ a user map at the time of running kdb2mt. zaddrxlat does not change ACLs.

Options

> `-m` *file*
> User map to utilize for translation. See kdb-uidextract(8) for a format description.

> `-r`
> Process folders recursively.

Example

zaddrxlat -m u.map -r "IPM_SUBTREE/Import of blah @2025-12-26"

Folder specification

folder_spec **must conform to one of three forms. Either:**

- a numeric identifer (e.g. 13, 0xd)
- a folder path starting with a slash, optionally followed by a slash-separated sequence of subordinate folder names
- a folder path starting with a fixed symbolic name, optionally followed by a slash-separated sequence of subordinate folder names

The backslash may be used as a hierarchy separator instead; in any case, the chosen separator must be used consistently in the entire path.

The recognized strings are: CALENDAR, COMMON_VIEWS, CONFLICTS, CONTACTS, DEFERRED_ACTION, DELETED (TRASH, WASTEBASKET), DRAFT, FINDER, INBOX, IPM_SUBTREE, JOURNAL, JUNK, LOCAL_FAILURES, NOTES, OUTBOX, SENT, SERVER_FAILURES, SHORTCUTS, SYNC_ISSUES, TASKS, VIEWS.

The purpose of these names is for referencing a built-in folder irrespective of its assigned name, which is dependent upon translation settings. The symbolic names can be used with private stores

© grommunio GmbH — CC BY-SA 4.0

only; there are no names defined for public folder contents at this time. There is also no parsing support for slashes in folder names. The slash character is always treated as a hierarchy separator.

Examples

- Using the MAPI root: /Top of Information Store/Sent Items/2022
- Using a symbolic name: IPM_SUBTREE/Sent Items/2022
- Using a symbolic name: SENT/2022
- Referencing a folder with a slash can be done by using backslash as the hierarchy separator: SENT\Winter break 2022/2023

The MAPI root is not visible in most clients. MUAs like Outlook and grommunio-web show hierarchy starting at IPM_SUBTREE only.

See also

> `gromox(7)`

© grommunio GmbH — CC BY-SA 4.0

gromox-mbox2mt(8)

This is an alias. See **gromox-eml2mt(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-mbsize(8)

Name

gromox-mbsize — Mailbox size analysis

Synopsis

gromox-mbsize *directory*

Description

Shows a detailed view of how a mailbox size translates to on-disk usage. Explanation of the columns/rows follows. The reported numbers may slightly deviate from what du(1) would output, as mbsize does not count e.g. the config/ directory, tmp/ directory, sqlite auxiliary files, any other unreferenced attachments (cf. gromox-mbop(8) for the purge-datafiles command), and other stuff left there by outside actions.

Apparent size: This is the exact size of the object, or simply the sum of sizes of objects.

On FS: This is the space that is used on the filesystem, and is subject to fs block sizes. Details about this behavior may be found on https://en.wikipedia.org/wiki/Block_(data_storage). As a result, the on-disk size may be larger than the apparent size.

RFC5322/Mbox: For the sake of IMAP, RFC5322 copies of messages and some metadata is retained.

RFC5322 Received: Applies to messages received via delivery(8gx).

RFC5322 Sent: Applies to any other message.

Body analysis: A set of 4 MAPI properties that usually get stored as files on disk rather than inside the sqlite database: PR_BODY, PR_HTML, PR_RTF_COMPRESSED and PR_TRANSPORT_MESSAGE_HEADERS. In gromox-mbsize, these are considered "body".

Attachment analysis: What it says. Not all MIME parts are or stay an attachment; for example, calendar items/meeting requests are usually converted to MAPI objects.

Missing items/Apparent: The number of MAPI properties/attachments which have a dangling reference into the filesystem.

Missing items/FS: The on-disk number of files that seem to be absent. This number can be lower than Apparent due to internal data deduplication that is transparent to MAPI/exmdb clients.

© grommunio GmbH — CC BY-SA 4.0

Informational content: The logical amount of data that is represented by those four MAPI properties / by MAPI attachments.

After deduplication: The logical amount of unique bodies/attachments.

Dedup ratio/gains: For the "body" group, there are usually little gains to be observed in practice; bodies are just very unique. Messages like "test" are ironically the ones that benefit. Attachments dedup a little better, owing to many people sending/receiving redundant information, such as company logos.

After compression: Besides deduplication, Gromox can also compress before data goes to disk. This is also the final form and so there is an apparent and an on-disk value. The on-disk value may be higher due to aforementioned filesystem block sizing.

File compress ratio/gains: The earnings going from Dedup to Compressed.

IFC compress ratio/gains: The earnings going from Informational Content to Compressed.

MAPI reported sizes: The PR_MESSAGE_SIZE property of a store, folder, message, and the PR_ATTACH_SIZE property of an attachment, all give a close approximation to the amount of data needed to transfer the object(s) over a MAPI connection.

NTS deviation: how much the Network Transfer Size is off from the on-disk size.

Provisioning factor: The ratio between on-disk usage and the logical mailbox size reported inside MUAs.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

gromox-mkmidb(8)

Name

gromox-mkmidb — Tool for creating a blank message index database

Synopsis

gromox-mkmidb [**-Uv**] [**-c** *config*] [**-fP**] *username*

Description

mkmidb is used to generate the midb.sqlite3 file for a private or public store, which is used by midb(8gx) to keep additional IMAP/POP3-specific state about the mailbox and its contents. mkmidb directly operates on the filesystem, which has implications. First, the program ought to be executed on the correct homeserver in case a multi-server Gromox installation is used. Second, there is no coordination with midb(8gx), and midb might continue using file descriptors opened earlier, working with the old copy of the mailbox, or get confused because e.g. a WAL file no longer matches the sqlite db. Ideally, you would use something like `gromox-mbop -u ... midb-unload` to make midb close the mailbox before resetting it.

Options

`-U`
Rather than creating the SQLite file, upgrade it. (The -f option has no effect.) mkmidb does not coordinate with midb(8gx) and data corruption is possible if both try to edit the file at the same time. You should let midb(8gx) do upgrades instead, via the "midb_schema_upgrades" directive, or at the very least, stop midb when using mkmidb -U.

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/mysql_adaptor.cfg will be read for MySQL connection parameters if that file exists.

`-f`
Force overwrite when an existing SQLite database is detected for the user. By default, mkmidb will not touch midb.sqlite3.

`-v`
Turn on verbose mode for -U.

`--create-old`
Create blank initial database using version 0. (This can be used for testing the upgrade

© grommunio GmbH — CC BY-SA 4.0

> procedure.)

> `--integrity`
> Perform SQLite integrity check, either standalone or, if -U is also given, as part of an upgrade.

> `-?`
> Display option summary.

Files

- */var/lib/gromox/.../*exmdb/midb.sqlite3: Message index database for IMAP.

See also

gromox(7), **gromox-mkprivate**(8), **midb**(8gx), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-mkprivate(8)

Name

gromox-mkprivate — Tool for creating a blank private store

Synopsis

gromox-mkprivate [**-Uv**] [**-T** *template_dir*] [**-c** *config*] [**-f**] *username*

Description

mkprivate is used to generate the exchange.sqlite3 file for a private store. mkprivate directly operates on the filesystem, which has implications. First, the program ought to be executed on the correct homeserver in case a multi-server Gromox installation is used. Second, there is no coordination with exmdb_provider(4gx), and exmdb_provider might continue using file descriptors opened earlier, working with the old copy of the mailbox, or get confused because e.g. a WAL file no longer matches the sqlite db. Ideally, you would use something like `gromox-mbop -u ... ( freeze ) ( unload )` to make exmdb_provider close the mailbox before resetting it. Even then, services that build on top of exmdb, such as exchange_emsmdb(4gx) or zcore(8gx), may also get confused when the mailbox (now as an abstract object rather than a concrete file descriptor) or any key characteristics like PR_STORE_RECORD_KEY.

Options

> `-T` *path*
> This option can be used to override the built-in data path for folder name translations and prepoulated named properties.
> Default: /usr/share/gromox

> `-U`
> Rather than creating the SQLite file, upgrade it. (The -f option has no effect.) mkprivate does not coordinate with exmdb_provider(4gx) and data corruption is possible if both try to edit the file at the same time. You should let exmdb_provider(4gx) do upgrades instead, via the "exmdb_schema_upgrades" directive, or at the very least, stop http(8gx)/exmdb_provider when using mkprivate -U.

> `-c` *config*
> Read configuration directives from the given file. If this option is not specified, /etc/gromox/mysql_adaptor.cfg will be read for MySQL connection parameters if that file exists.

© grommunio GmbH — CC BY-SA 4.0

`-f`
Force overwrite when an existing SQLite database is detected for the user. By default, mkprivate will not touch exchange.sqlite3.

`-v`
Turn on verbose mode for -U.

`--create-old`
Create blank initial database using version 0. (This can be used for testing the upgrade procedure.)

`--integrity`
Perform SQLite integrity check, either standalone or, if -U is also given, as part of an upgrade.

`-?`
Display option summary.

Files

- *data_file_path*/folder_names.txt: Translations for essential folders in a message store.
- *data_file_path*/propnames.txt: Initial set of named properties to add to the new private store.
- */var/lib/gromox/...*/exmdb/exchange.sqlite3: MAPI object database.

See also

gromox(7), **gromox-mkmidb**(8), **gromox-mkpublic**(8), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-mkpublic(8)

Name

gromox-mkpublic — Tool for creating a blank public store

Synopsis

gromox-mkpublic [**-Uv**] [**-T** *template_dir*] [**-c** *config*] [**-f**] *domainname*

Description

mkpublic is used to generate the exchange.sqlite3 file for a public store. mkpublic directly operates on the filesystem, and has the same considerations as mentioned in gromox-mkprivate(8).

Options

`-T` *path*
This option can be used to override the built-in data path for folder name translations and prepopulated named properties.
Default: /usr/share/gromox

`-U`
Rather than creating the SQLite file, upgrade it. (The -f option has no effect.) mkpublic does not coordinate with exmdb_provider(4gx) and data corruption is possible if both try to edit the file at the same time. You should let exmdb_provider(4gx) do upgrades instead, via the "exmdb_schema_upgrades" directive, or at the very least, stop http(8gx)/exmdb_provider when using mkpublic -U.

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/mysql_adaptor.cfg will be read for MySQL connection parameters if that file exists.

`-f`
Force overwrite when an existing SQLite database is detected for the user. By default, mkpublic will not touch exchange.sqlite3.

`-v`
Turn on verbose mode for -U.

`--create-old`
Create blank initial database using version 0. (This can be used for testing the upgrade

© grommunio GmbH — CC BY-SA 4.0

procedure.)

`--integrity`
Perform SQLite integrity check, either standalone or, if -U is also given, as part of an upgrade.

`-?`
Display option summary.

Files

- *data_file_path*/propnames.txt: Initial set of named properties to add to the new private store.
- */var/lib/gromox/.../*exmdb/exchange.sqlite3: MAPI object database.

See also

gromox(7), **gromox-mkprivate**(8), **mysql_adaptor**(4gx)

© grommunio GmbH — CC BY-SA 4.0

gromox-mt2exm(8)

This is an alias. See **gromox-import(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-oxm2mt(8)

Name

gromox-oxm2mt — Utility for analysis of Outlook .msg files

Synopsis

gromox-oxm2mt [**-pt**] *input.msg*

Description

gromox-oxm2mt reads an Outlook .msg file, and re-exports the data in a Gromox-specific mailbox transfer format to stdout for consumption by pipe by the gromox-import(8) program. Optionally, oxm2mt can print a tree summary of the message.

Options

`--decap`=*n*
Select attachment number *n*'s embedded message as the "top-level" message and discard the rest of the outer message. *n* is 1-based.

`-p`
Show properties in detail (enhances **-t**).

`-t`
Show a diagnostic tree view of the source data as it is being read.

`--loglevel` *n* Maximum verbosity of general logging (not connected to **-p**, **-t** or **-v**). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

Examples

Import of a message to drafts:

gromox-oxm2mt saved.msg | gromox-import -u user@domain.example -B drafts

Normative references

- MS-CFB: Compound File Binary Format (CFBF)

© grommunio GmbH — CC BY-SA 4.0

- MS-OXMSG: Outlook Item (.msg) File Format

Outlook message files use the "Object Linking and Embedding (OLE) / Component Object Model (COM) structured storage compound file implementation binary file format", or just CFB/CFBF for short. It resembles something of a FAT filesystem. The file(1) utility identifies those as "CDFV2 Microsoft Outlook Message" (Compound Document Format). A proposed MIME type is "application/vnd.ms-outlook".

oxm2mt uses the libolecf C library to read the CDF structure of .msg files per [MS-CFB], and then applies own code to make sense of the files as per [MS-OXMSG].

See also

gromox(7), **gromox-import**(8), **olecfexport**(1)

© grommunio GmbH — CC BY-SA 4.0

gromox-pff2mt(8)

Name

gromox-pff2mt — Utility for analysis/import of PFF/PST/OST files

Synopsis

gromox-pff2mt [**-pst**] [...] *input.pst*

Description

gromox-pff2mt reads a file that conforms to the Personal Folder File (PFF) and the Offline Folder File (OFF) format and re-exports the data in a Gromox-specific mailbox transfer format to stdout for consumption by pipe by the gromox-import(8) program. Optionally, pff2mt can print a tree summary of the PFF. The PFF format is used in several file types:

- PAB (Personal Address Book)
- PST (Personal Storage Table)
- OST (Offline Storage Table)

By default, pff2mt creates a new folder "Import of <xyz.pst> on <date>" within IPM_SUBTREE ("Top of Information Store") and places all PFF objects into that new subfolder.

Note that folders of non-default mailboxes are stored in a flat fashion in OST. A secretary.ost file with hierarchy like "(Boss)\Inbox\2021\Invoices", "(Boss)\Inbox\2022\Invoices" is flattened to "(Boss)\Inbox", "(Boss)\2021", "(Boss)\Invoices", "Boss\2022" and "(Boss)\Invoices". The presence of **two** folders with the same name **at the same level** means gromox-import errors out when it tries to create a folder whose name already exists. This limits the usefulness of importing OST files.

Options

`-p`
Show properties in detail (enhances **-t**).

`-s`
Splice objects from the PFF into existing folders. Specifically, the PFF root is mapped to the store root, and special folders (Sent Items, Deleted Items, etc.) are mapped to special folders in the store if the PFF has the necessary information. PST files have no such mapping hint for Inbox; only OST do. When --with-hidden is not selected, -s will imply --without-hidden by default to avoid polluting e.g. "QuickStep settings" (which may already have settings).

© grommunio GmbH — CC BY-SA 4.0

`-t`
Show a diagnostic tree view of the source data as it is being read.

`--loglevel` *n* Maximum verbosity of general logging (not connected to **-p**, **-t** or **-v**). 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`--with-hidden`, **--without-hidden**
This option controls the import of folders that have PR_ATTR_HIDDEN=1.
Default: if -s is present, import without hidden folders

`--only-obj` *nid*
Extract just the object with the given PFF node id. This option may be specified multiple times. The objects will be unanchored; see gromox-import(8) for where messages are placed.

Examples

Import of a PFF-compatible file into a Gromox mailbox via exmdb transport:

gromox-pff2mt sample.pst | gromox-import -u user@domain.example

gromox-import has more target options; see its manpage.

Bugs

Embedded messages are treated as subitems by libpff. Luckily, the only consequence is that the tree view (-t) shows duplicate NID visits.

> _ [id=21cee4h ntyp=unknown-4h type=appointment nset=1 nent=161] _ [id=8005h ntyp=atx type=attachment nset=1 nent=19] _ [attachment type=i embedded_msg] _ [id=21cf04h ntyp=unknown-4h type=appointment nset=1 _ [id=21cf04h ntyp=unknown-4h type=appointment nset=1 nent=50] _ [id=8025h ntyp=atx type=attachment nset=1 nent=19] _ [attachment type=i embedded_msg] _ [id=21cf24h ntyp=unknown-4h type=appointment nset=1 _ [id=21cf24h ntyp=unknown-4h type=appointment nset=1 nent=49]

See also

gromox(7), **gromox-import**(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-selinux(5)

NAME

gromox_selinux - Security Enhanced Linux Policy for the gromox processes

DESCRIPTION

Security-Enhanced Linux secures the gromox processes via flexible mandatory access control.

The gromox processes execute with the gromox_t SELinux type. You can check if you have these processes running by executing the **ps** command with the **-Z** qualifier.

For example:

```
ps -eZ | grep gromox_t
```

ENTRYPOINTS

The gromox_t SELinux type can be entered via the **gromox_exec_t** file type.

The default entrypoint paths for the gromox_t domain are the following:

/usr/libexec/gromox(/.*)?

PROCESS TYPES

SELinux defines process types (domains) for each process running on the system

You can see the context of a process using the **-Z** option to **psP**

Policy governs the access confined processes have to files. SELinux gromox policy is very flexible allowing users to setup their gromox processes in as secure a method as possible.

The following process types are defined for gromox:

```
gromox_t
```

Note: **semanage permissive -a gromox_t** can be used to make the process type gromox_t permissive. SELinux does not deny access to permissive process types, but the AVC (SELinux denials) messages are still generated.

© grommunio GmbH — CC BY-SA 4.0

BOOLEANS

SELinux policy is customizable based on least access required. gromox policy is extremely flexible and has several booleans that allow you to manipulate the policy and run gromox with the tightest access possible.

If you want to dontaudit all daemons scheduling requests (setsched, sys_nice), you must turn on the daemons_dontaudit_scheduling boolean. Enabled by default.

```
setsebool -P daemons_dontaudit_scheduling 1
```

If you want to allow all domains to execute in fips_mode, you must turn on the fips_mode boolean. Enabled by default.

```
setsebool -P fips_mode 1
```

MANAGED FILES

The SELinux process type gromox_t can manage files labeled with the following file types. The paths listed are the default paths for these file types. Note the processes UID still need to have DAC permissions.

> `cluster_conf_t`

/etc/cluster(/.*)?

> `cluster_var_lib_t`

/var/lib/pcsd(/.*)?
/var/lib/cluster(/.*)?
/var/lib/openais(/.*)?
/var/lib/pengine(/.*)?
/var/lib/corosync(/.*)?
/usr/lib/heartbeat(/.*)?
/var/lib/heartbeat(/.*)?
/var/lib/pacemaker(/.*)?

> `cluster_var_run_t`

/var/run/crm(/.*)?
/var/run/cman_.*
/var/run/rsctmp(/.*)?
/var/run/aisexec.*

© grommunio GmbH — CC BY-SA 4.0

/var/run/heartbeat(/.*)?
/var/run/pcsd-ruby.socket
/var/run/corosync-qnetd(/.*)?
/var/run/corosync-qdevice(/.*)?
/var/run/corosync.pid
/var/run/cpglockd.pid
/var/run/rgmanager.pid
/var/run/cluster/rgmanager.sk

> `gromox_log_t`

/var/log/gromox(/.*)?

> `gromox_var_lib_t`

/var/lib/gromox(/.*)?

> `gromox_var_run_t`

/var/run/gromox(/.*)?

> `root_t`

/sysroot/ostree/deploy/.*-atomic/deploy(/.*)?
/
/initrd

FILE CONTEXTS

SELinux requires files to have an extended attribute to define the file type.

You can see the context of a file using the **-Z option to lsP**

Policy governs the access confined processes have to these files. SELinux gromox policy is very flexible allowing users to setup their gromox processes in as secure a method as possible.

> `STANDARD FILE CONTEXT`

SELinux defines the file context types for the gromox, if you wanted to store files with these types in a different paths, you need to execute the semanage command to specify alternate labeling and then use restorecon to put the labels on disk.

semanage fcontext -a -t gromox_var_run_t '/srv/mygromox_content(/.*)?'
`restorecon -R -v /srv/mygromox_content`

Note: SELinux often uses regular expressions to specify labels that match multiple files.

The following file types are defined for gromox:

© grommunio GmbH — CC BY-SA 4.0

```
gromox_exec_t
```

- Set files with the gromox_exec_t type, if you want to transition an executable to the gromox_t domain.

```
gromox_log_t
```

- Set files with the gromox_log_t type, if you want to treat the data as gromox log data, usually stored under the /var/log directory.

```
gromox_var_lib_t
```

- Set files with the gromox_var_lib_t type, if you want to store the gromox files under the /var/lib directory.

```
gromox_var_run_t
```

- Set files with the gromox_var_run_t type, if you want to store the gromox files under the /run or /var/run directory.

Note: File context can be temporarily modified with the chcon command. If you want to permanently change the file context you need to use the **semanage fcontext** command. This will modify the SELinux labeling database. You will need to use **restorecon** to apply the labels.

COMMANDS

semanage fcontext can also be used to manipulate default file context mappings.

semanage permissive can also be used to manipulate whether or not a process type is permissive.

semanage module can also be used to enable/disable/install/remove policy modules.

semanage boolean can also be used to manipulate the booleans

system-config-selinux is a GUI tool available to customize SELinux policy settings.

AUTHOR

This manual page was auto-generated using **sepolicy manpage .**

© grommunio GmbH — CC BY-SA 4.0

SEE ALSO

selinux(8), gromox(8), semanage(8), restorecon(8), chcon(1), sepolicy(8), setsebool(8)

© grommunio GmbH — CC BY-SA 4.0

gromox-snapshot(8)

Name

gromox-snapshot — Helper to create snapshots of mailboxes

Synopsis

```
/usr/libexec/gromox/gromox-snapshot
```

Description

gromox-snapshot calls btrfs(8) or cp(1)/--reflink to create snapshots of the current state of mailboxes as needed. gromox-snapshot is meant to be periodically invoked by a systemd timer (or, failing that, a classic cron job). The default interval of gromox-snapshot.timer is hourly.

The program generates snapshots "for this day" / "for this week" / etc., rather than snapshots which are at least one day / one week / etc. apart. When the snapshotter is run periodically, this evens out and is not a concern at all.

Options

The program has no command-line options.

Configuration file

/etc/gromox/snapshot.cfg will be read on startup. It is a trivial key=value file, with one variable per line.

retention_days=*n*
Make daily snapshots and keep them for *n* days. Use *0* to deactivate daily snapshotting.
Default: 7

retention_hours=*n*
Make hourly snapshots and keep them for *n* hours. Use *0* to deactivate hourly snapshotting.
Default: 0

retention_months=*n*
Make monthly snapshots and keep them for *n* months (more precisely, *n**31 days). Use *0* to deactivate monthly snapshotting.
Default: 0

© grommunio GmbH — CC BY-SA 4.0

retention_weeks=*n*
Make weekly snapshots and keep them for *n* weeks. Use *0* to deactivate weekly snapshotting.
Default: 4

> `subvolume_root`
> When btrfs is used, this directive specifies the root directory of the nearest btrfs subvolume. (As of Linux 6.2.1, it is not possible to snapshot arbitrary directories; it has to be the root of a subvolume. Confer with `btrfs sub list` to see subvolumes.) Otherwise, this must point to the Gromox data directory.
> Default: /var/lib/gromox

> `snapshot_archive`
> The directory where snapshots (these are subdirectories) will be placed. This directory needs to be on the same device as the snapshot source (subvolume_root, see above).
> Default: /var/lib/gromox-snapshots

Each time "category" (days, weeks, etc.) is independently evaluated and snapshots are always made from the subvolume_root, never from another snapshot. A weekly snapshot generated on the first day of the week is not strictly equal to the daily snapshot generated for the same day, as there is technically a very small time window between individual btrfs commands.

Errors

- Invalid cross-device link: snapshot_archive was not on the same device as subvolume_root.
- Operation not supported: the filesystem employed on snapshot_archive does not support reflinks

Known issues

When a reflink-based snapshot is deleted (which happens via /bin/rm), rm may modify the directory's timestamp. If that rm call is interrupted without completing, the next run of gromox-snapshot will erroneously consider the directory as "too new" and not resume the deletion until it has sufficiently aged again.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

gromox-tnef2mt(8)

This is an alias. See **gromox-eml2mt(8)**.

© grommunio GmbH — CC BY-SA 4.0

gromox-vcf2mt(8)

This is an alias. See **gromox-import(8)**.

© grommunio GmbH — CC BY-SA 4.0

http(8gx)

Name

http — Protocol handler for HTTP and RPCH

Synopsis

http [**-c** *config*]

Description

http(8gx) is a trivial HTTP server. It understands the special HTTP methods as used by RPC-over-HTTP protocol as used by Outlook, it can serve files verbatim, or forward requests to a FastCGI server such as php-fpm(8).

Generally, http(8gx) also executed the Information Store, exmdb_provider(4gx), which is a loadable module.

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/http.cfg will be read.

`-?`
Display option summary.

URI processing order

- Requests are passed to the mod_rewrite(4gx) module (built-in) to have their URI potentially rewritten.
- If a HTTP request is using the methods RPC_IN_DATA or RPC_OUT_DATA, the data stream is handed off to the exchange_emsmdb(4gx) component.
- Otherwise, HTTP processing modules (HPM) are invoked. Processing ends when one module signals that the request was handled. The order depends on the HPM list (which is fixed): ews, mh_emsmdb, mh_nsp, oxdisco, oab.
- Otherwise, the mod_fastcgi(4gx) module (built-in) is invoked. Processing ends if the module handled the request.

© grommunio GmbH — CC BY-SA 4.0

- Otherwise, the mod_cache(4gx) module (built-in) is invoked. Processing ends if the module handled the request.
- Otherwise, the request is rejected.

RPC-over-HTTP

RPC-over-HTTP utilizes two special HTTP methods, RPC_IN_DATA and RPC_OUT_DATA. These requests can, similarly to HTTP CONNECT, be very long-lived. The RPC data stream is handled by the included exchange_emsmdb(4gx) component.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`daemons_fd_limit`
In gromox-http, this is treated as an alias for http_fd_limit.

`http_basic_auth_cred_caching`
Perform credential caching for HTTP Basic.
Default: 1minute

`http_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

`http_listen`
A space-separated list of bind address specifiers to expose HTTP on.
Default: *[::]:80*

`http_listen_tls`
A space-separated list of bind address specifiers to expose implicit-TLS (HTTPS) on.
Default: *[::]:443*

`http_remote_host_hdr`
The name of the HTTP request header which contains the actual client IPv6/IPv4 address. When a (reverse) proxy is placed in front of gromox-http, the address gxhttp normally sees is the proxy address (e.g. ::1). If the proxy sets a custom header to convey the actual client address, Gromox can pick this up for its own reporting, which in turn is useful for e.g. fail2ban setups.
Default: (empty)

© grommunio GmbH — CC BY-SA 4.0

`istore_standalone`
A bitmask controlling how the Information Store code should run as separate processes in separate address spaces. See istore(8gx) for details.
Default: 0

Configuration directives (http.cfg)

The following directives are recognized when reading from /etc/gromox/http.cfg, or when the **-c** option is used to specify a custom file:

`block_interval_auths`
The amount of time a user is blocked from connecting to the service after too many failed logins.
Default: 1 minute

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/http/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/http:/etc/gromox

`context_average_mem`
Default: 256K

`context_num`
Default: 400

`data_file_path`
Colon-separated list of directories which will be scanned when locating data files.
Default: /usr/share/gromox/http

`fastcgi_exec_timeout`
Maximum execution time for CGI scripts.
Default: 10 minutes

`gss_program`
The helper program to use for authenticating HTTP requests when Negotiate-SPNEGO headers are presented. The value is rudimentarily tokenized at whitespaces, and no special characters may be used. If necessary, write your own wrapper. The special value "internal-gss" uses libgssapi directly.
Negotiate was meant to carry GSS-API auth data (appearing as "Authorization: Negotiate YII..." in HTTP protocol dumps). NTLM can be wrapped in SPNEGO (also shows up as "YII"), but a handful of clients may also send raw NTLM tokens (appearing as "Authorization: Negotiate TlRMTVNT..."). Whether raw NTLM tokens are accepted by internal-gss depends on your GSS library and, more broadly, your Kerberos setup. Otherwise, you may need to use a helper program like the one from Squid. internal-gss also does not offer a way to specify a separate keytab or replay cache parameters, so use Squid's helper if you need such parameters.

© grommunio GmbH — CC BY-SA 4.0

Default: internal-gss
Example: */usr/lib/squid/negotiate_wrapper_auth --ntlm /usr/bin/ntlm_auth* --helper-protocol=squid-2.5-ntlmssp --kerberos /usr/lib/squid/negotiate_kerberos_auth -s GSS_C_NO_NAME

`host_id`
A unique identifier for this system. It is used for the Server HTTP responses header, for components like exmdb_provider(4gx), which makes use of it for SMTP HELO lines, for DSN report texts, for MIDB database/EML cache. The identifier should only use characters allowed for hostnames.
Default: (system hostname)

`http_auth_basic`
Enable HTTP Basic authentication.
Default: yes

`http_auth_spnego`
Enable HTTP Negotiate authentication.
Default: no

`http_auth_times`
The number of login tries a user is allowed before the account is blocked.
Default: 10

`http_certificate_passwd`
The password to unlock TLS certificates.
Default: (unset)

`http_certificate_path`
A colon-separated list of TLS certificate files. The complete certificate chain should be present (as there is no other config directive to pull CA certs in, and implicit loading from system directories is not guaranteed by Gromox).
Default: (unset)

`http_conn_timeout`
If a HTTP connection is inactive for the given period, the connection is terminated.
Default: 3 minutes

`http_debug`
If set to **1**, prints all incoming and outgoing HTTP traffic to stderr (not http_log_file!).
Default: 0

`http_enforce_auth`
Enforce authentication at all times. This is a debugging knob.
Default: no

`http_krb_service_principal`
Kerberos service principal to use when gss_program=internal-gss. The form is often something

© grommunio GmbH — CC BY-SA 4.0

like *HTTP/fqdn@REALM*, but may vary. When using an external GSS authentication helper, http_krb_service_principal has no effect, and any principal you want to use needs to be passed via the **gss_program** directive in some way. be
Default: (empty)

`http_listen_addr`
Deprecated in favor of gromox.cfg:http_listen / gromox.cfg:http_listen_tls.

`http_listen_port`
Deprecated in favor of gromox.cfg:http_listen.

`http_listen_tls_port`
Deprecated in favor of gromox.cfg:http_listen_tls.

`http_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`http_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`http_private_key_path`
A colon-separated list of TLS certificate private key files.
Default: (unset)

`http_rqbody_flush_size`
If the HTTP request to a CGI endpoint has a HTTP body larger than the limit given here, the data is buffered in a file rather than kept in memory. If the request uses Chunked Transfer Encoding, a file is used unconditionally.
Default: 512K

`http_rqbody_max_size`
If the Content-Length of a HTTP request to a CGI endpoint is larger than this value, the request is rejected.
Default: 50M

`http_support_tls`
This flag controls whether (or not) the server offers TLS at all. The default is false because you need a certificate for this first.
Default: false

`http_thread_charge_num`
Connection load factor (oversubscription ratio) for a processing thread.
Default: 20

© grommunio GmbH — CC BY-SA 4.0

`http_thread_init_num`
The initial and also minimum number of client processing threads to keep around. This is similar to php-fpm's start_servers/min_spare_servere. (The maximum number of threads, i.e. what would be max_spare_servers, is determined by: context_num divided by http_thread_charge_num)
Default: 5

`msrpc_debug`
Log every completed RPC call and the return code of the operation in a minimal fashion to stderr. Level 1 emits RPCs with a failure return code, level 2 emits all RPCs. Note the daemon log level needs to be "debug" (6), too.
Default: 0

`request_max_mem`
The maximum hint size for fragmented RPC PDU requests that will be allowed (C706 §12.6.3.7, MS-RPCE v33 §2.2.2.6).
Default: 4M

`tls_min_proto`
The lowest TLS version to offer. Possible values are: **tls1.0**, **tls1.1**, **tls1.2**, and, if supported by the system, **tls1.3**.
Default: tls1.2

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

`user_default_lang`
Default: en

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Upon receipt of SIGUSR1, a dump of the currently active HTTP connections will be printed to stderr.

Normative references

- MS-RPCE: Remote Procedure Call Protocol Extensions
- DCERPC / C706: Technical Standard DCE 1.1: Remote Procedure Call by The Open Group, 1997

See also

gromox(7), **mod_cache**(4gx), **mod_fastcgi**(4gx), **mod_rewrite**(4gx)

© grommunio GmbH — CC BY-SA 4.0

imap(8gx)

Name

imap — Gromox IMAP server

Authentication

The IMAP server supports impersonation. The username given to the IMAP login normally specifies both the mailbox and the user performing the access. To use a different identity for authentication, prefix the mailbox name by the user identity and separate it with an exclamation mark, e.g. "myaccount@domain.example!sharedmbox@domain.example". Accessing a store in such manner is only possible when the authenticating user has store ownership over the mailbox.

(The exclamation mark was once used for path routing in Usenet, which might explain how the exclamation mark never became a character that is widely used for email addresses, despite being a valid character for an RFC 5322 addr-spec. That drove the choice to use the character for denoting impersonation; it could be regarded as a form of routing. ("Through myaccount@, access sharedmbox@"))

Synopsis

imap [**-c** *config*]

Options

> `-c` *config*
> Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/imap.cfg will be read.

> `--version`
> Output version information and exit.

> `-?`
> Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

© grommunio GmbH — CC BY-SA 4.0

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`daemons_fd_limit`
In gromox-imap, this is treated as an alias for imap_fd_limit.

`imap_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

`imap_accept_haproxy`
This directive sets the expectation for incoming connections to carry haproxy's "PROXY" protocol extension version 2 (2), or no such header (0). When a (reverse) proxy is placed in front of gromox-imap, the address that gximap normally sees is the proxy address (e.g. ::1). A proxy can use this protocol extension to convey the actual client address, and gximap can pick this up for its own reporting, which in turn is useful for e.g. fail2ban setups.
Default: 0

`imap_listen`
A space-separated list of bind address specifiers to expose standard IMAP (unencrypted and/or explicit STARTTLS) on.
Default: *[::]:143*

`imap_listen_tls`
A space-separated list of bind address specifiers to expose implicit-TLS IMAP (IMAPS) on.
Default: *[::]:993*

Configuration directives (imap.cfg)

The following directives are recognized when reading from /etc/gromox/imap.cfg, or when the **-c** option is used to specify a custom file:

`block_interval_auths`
The amount of time a user is blocked from connecting to the service after too many failed logins.
Default: 1 minute

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/imap/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/imap:/etc/gromox

`context_average_mem`
Default: 128K

© grommunio GmbH — CC BY-SA 4.0

`context_average_mitem`
The expected average upper bound of number of mails for a folder. Together with context_num, this directive controls the size of the memory pool for listings.
Default: 64K

`context_num`
Maximum number of concurrently active sessions.
Default: 200

`data_file_path`
Colon-separated list of directories in which static data files will be searched.
Default: /usr/share/gromox/imap

`default_lang`
Default: en

`enable_rfc2971_commands`
RFC 2971 specifies the "ID" command with which a client can inquire the program name and version of the server. This is disabled by default, as it can facilitate potential attackers' information gathering.
Default: no

`host_id`
A unique identifier for this system. It is used in the IMAP protocol greeting lines (positive as well as negative). It is furthermore used as a unique identifier among the set of all midb(8gx) clients to construct filenames for the MIDB database/EML cache. The identifier should only use characters allowed for hostnames.
Default: (system hostname)

`imap_auth_times`
The number of login tries a user is allowed before the account is blocked.
Default: 10

`imap_autologout_time`
If an authenticated IMAP connection is idle for the given period, the connection is terminated. RFC 2060 §5.4 recommends 30 minutes minimum. (Connections that have not authenticated are subject to the regular imap_conn_timeout.)
Default: 30 minutes

`imap_certificate_passwd`
The password to unlock TLS certificates.
Default: (unset)

`imap_certificate_path`
A colon-separated list of TLS certificate files. The complete certificate chain should be present (as there is no other config directive to pull CA certs in, and implicit loading from system directories is

© grommunio GmbH — CC BY-SA 4.0

not guaranteed by Gromox).
Default: (unset)

`imap_cmd_debug`
Log every incoming IMAP command and the return code of the operation in a minimal fashion to stderr. Level 1 emits commands that have failed execution, level 2 emits all commands. (The response text is **not** sent to the log, because of size. Deep analysis can be done with socat/telnet/tcpdump; shallow analysis for end-users is possible with the protocol-compliant error-reporting MUA "Alpine" https://alpineapp.email/.)
Default: 0

`imap_conn_timeout`
If an IMAP connection stalls (writing responses to client) for the given period, the connection is terminated. If unauthenticated IMAP connections do not have any activity (requests from clients) for the given period, the connection is terminated.
Default: 3 minutes

`imap_expunge_on_delete`
When a STORE command sets the \Deleted flag, expunge the affected messages immediately instead of waiting for an EXPUNGE or CLOSE command. Some clients mark messages as deleted without ever expunging them. For example, Thunderbird does this when moving messages to a different account, which leaves such messages visible to non-IMAP clients indefinitely.
Default: false

`imap_force_tls`
This flag controls whether clients must utilize TLS, either by way of implicit TLS, or through the STARTTLS command.
Default: false

`imap_listen_addr`
Deprecated in favor of gromox.cfg:imap_listen / gromox.cfg:imap_listen_tls.

`imap_listen_port`
Deprecated in favor of gromox.cfg:imap_listen.

`imap_listen_tls_port`
Deprecated in favor of gromox.cfg:imap_listen_tls.

`imap_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`imap_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

© grommunio GmbH — CC BY-SA 4.0

`imap_private_key_path`
A colon-separated list of TLS certificate private key files.
Default: (unset)

`imap_rfc9051`
Enable RFC 9051 (IMAP 4.2) related logic and protocol elements.
Default: yes

`imap_support_tls`
This flag controls the offering of TLS modes. This affects both the implicit TLS port as well as the advertisement of the STARTTLS extension and availability of the STARTTLS command (RFC 2595) to clients.
Default: false

`imap_thread_charge_num`
Connection load factor (oversubscription ratio) for a processing thread.
Default: 40

`imap_thread_init_num`
The initial and also minimum number of client processing threads to keep around. This is similar to php-fpm's start_servers/min_spare_servere. (The maximum number of threads, i.e. what would be max_spare_servers, is determined by: context_num divided by imap_thread_charge_num)
Default: 5

`running_identity`
An unprivileged user account to switch the process to after startup.
Default: gromox

`tls_min_proto`
The lowest TLS version to offer. Possible values are: **tls1.0**, **tls1.1**, **tls1.2**, and, if supported by the system, **tls1.3**.
Default: tls1.2

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Files

- *data_file_path*/folder_lang.txt: Translations for IMAP folder names.
- *data_file_path*/imap_code.txt: Mapping from internal IMAP error codes to textual descriptions.

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **midb_agent**(4gx)

© grommunio GmbH — CC BY-SA 4.0

istore(8gx)

Name

istore — Gromox Information Store launcher

Synopsis

/usr/libexec/gromox/istore [**-c** *config*]

Description

The Gromox Information Store, exmdb_provider(4gx), is built as a shared object, and can be either be loaded by gromox-http(8) or run in a separate process which is gromox-istore. The benefit of a separate process is improved debuggability.

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg will be read if it exists.

`-x` *userdir*
Used internally by istore to re-launch itself as a single-store worker process. This option is not meant to be used by administrators.

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/istore/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.) Default: /etc/gromox/istore:/etc/gromox

`daemons_fd_limit`
In gromox-istore, this is treated as an alias for istore_fd_limit.

`data_file_path`
Colon-separated list of directories which will be scanned when locating data files.

© grommunio GmbH — CC BY-SA 4.0

Default: /usr/share/gromox/istore:/usr/share/gromox

`host_id`
A unique identifier for this system. It is used for the Server HTTP responses header, for components like exmdb_provider(4gx), which makes use of it for SMTP HELO lines, for DSN report texts, for MIDB database/EML cache. The identifier should only use characters allowed for hostnames.
Default: (system hostname)

`istore_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

`istore_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`istore_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`istore_standalone`
A bitmask controlling how the Information Store code should split itself up to run as separate processes in separate address spaces.
Flag 0x1: Separation of the Information Store from the HTTP server. If set, the Information Store is to be launched from /usr/libexec/gromox/istore (e.g. the gromox-istore.service systemd unit) and not from /usr/libexec/gromox/http (gromox-http.service).
Flag 0x2: Separation of individual stores. If set, the Information Store director, i.e. the code that has the port 5000 listening socket open, will spawn worker processes instead of threads.
Default: 0

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7), **exmdb_provider**(4gx)

© grommunio GmbH — CC BY-SA 4.0

kdb-uidextract(8)

Name

kdb-uidextract — Helper for creating a gromox-kdb2mt ACL map

Synopsis

```
python /usr/libexec/gromox/kdb-uidextract
```

Description

kdb-uidextract is a Python script utilizing python-kopano bindings to read user object descriptions off a Kopano installation and produce a user listing suitable for consumption by the gromox-kdb2mt --user-map option.

This script is meant to be executed on a live Kopano system and does not rely on Gromox components at all.

kdb-uidextract first queries the server on the current machine for all participating Kopano servers in the cluster. This requires that all Kopano servers accept TLS connections (/etc/kopano/server.cfg:server_ssl_port, server_ssl_key_file, sslkeys_path) and have authentication keys set up for the SYSTEM account (in the directory specified by sslkeys_path).

The resulting map for kdb2mt is printed to stdout.

Options

This program offers no command-line options.

Files

By way of the *kopano* Python module, /etc/kopano/admin.cfg is sourced for TLS certificate parameters. Confer with the kopano-admin.cfg(5) manpage.

User map format

The output of kdb-uidextract is a JSON file containing an array of user objects. If running uidextract is not possible, or not applicable (e.g. in case of Exchange user translation), the JSON file can also be constructed by other means, including manual input. Each user object is a dictionary with zero or more attributes; these can be:

© grommunio GmbH — CC BY-SA 4.0

- "na": original username associated with the Kopano account (this can have many forms, including, but not limited to, "user", "user@domain", "domain_user", "domain\user"; see "loginname_format" line of /etc/kopano/server.cfg)
- "dn": X.500 Legacy DN associated with an Exchange account
- "sv": server GUID, represented as 16 ASCII characters, case-insensitive
- "st": store GUID, represented as 16 ASCII characters, case-insensitive
- "id": per-database(!) numeric user ID
- "em": original e-mail address associated with the Kopano account
- "to": target e-mail address in the Gromox system

null values and empty strings are allowed. Take note that in multi-server Kopano installations, every LDAP user will appear in **all** the kopano-server databases, and with generally **different** user IDs.

For sender/recipient address substitution (done by kdb2mt, or mbop-zaddrxlat): The "to" attribute is needed. One (or both) of "na" and "em" is also needed.

For sender/recipient address substitution (done by mbop-exaddrxlat), the "to" and "dn" attributes are needed.

For ACL substitution (done by kdb2mt), the "id", "sv" and "to" attributes are needed.

For kdb2mt --mbox-user lookup, the "sv", "id" and "st" attributes are needed.

Example Kopano user map

```
[

 {"em": "boss@example.domain", "na": "boss", "sv":
  "0123456789abcdef0123456789abcdef", "st": "0123456789abcdef012345678
  "to": "boss@domain.example", "id": 3},

 {"em": "boss@domain.example", "na": "boss", "sv":
  "123456789abcdef0123456789abcdef0", "st": "0123456789abcdef012345678
  "to": "boss@domain.example", "id": 91}

]
```

Example Exchange user map

```
[ {"dn": "/o=foobar/ou=Gobbledygook/cn=00000000-boss", "to":
"boss@domain.example"} ]
```

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **gromox-kdb2mt**(8), **kdb-uidextract-limited**(8)

© grommunio GmbH — CC BY-SA 4.0

kdb-uidextract-limited(8)

Name

kdb-uidextract-limited — Helper for creating a gromox-kdb2mt ACL map

Synopsis

```
perl /usr/libexec/gromox/kdb-uidextract-limited
```

Description

kdb-uidextract-limited is a Perl script utilizing just the Kopano command line interface of the kopano-admin(8) and kopano-stats(8) programs to read user object descriptions off a Kopano installation and produce an ACL map for use with the gromox-kdb2mt --user-map option.

This script is meant to be executed on a Kopano system and does not rely on Gromox at all.

Unlike kdb-uidextract, kdb-uidextract-limited also does not rely on the presence of python-kopano bindings (e.g. for when installing those is considered infeasible). However, kdb-uidextract-limited can only evaluate the host it is executed on and has no automatic multi-server capability. This is because kopano-stats(8) never loads a TLS certificate needed to make the -h option work. To produce a complete map, you need to manually execute kdb-uidextract-limited on all Kopano server instances in a cluster and then join the results in accordance with the JSON syntax, e.g. by use of jq(1).

kopano-server need not be set up for TLS.

The resulting map for kdb2mt is printed to stdout.

Options

This program offers no command-line options.

See also

gromox(7), **gromox-kdb2mt**(8), **kdb-uidextract**(8)

© grommunio GmbH — CC BY-SA 4.0

ldap_adaptor(4gx)

Name

ldap_adaptor — LDAP connector for authentication

Description

ldap_adaptor is a component for Gromox that facilitates the use of an LDAP server for authentication purposes. Since the authoritative user database is in MySQL, LDAP is only used to perform authentication (Bind operations), and metadata searches that lead up to such Binds, i.e. looking for the LDAP object that has a particular "mail" attribute.

Gromox versions 1.33 and onwards have support for using different LDAP servers per organization. The SQL database for Gromox users (and domains and organizations) has room to store properties on the individual organization objects. Use Grommunio AAPI/AWEB to create and/or change organizations and their properties. These properties from SQL can selectively override the six config directives ldap_host, ldap_start_tls, ldap_bind_user, ldap_bind_pass, ldap_search_base and ldap_mail_attr.

Configuration directives

The configuration file, /etc/gromox/ldap_adaptor.cfg, serves not only the ldap_adaptor component, but is also read by the Grommunio Admin API.

Gromox directives

`auth_connections`
The maximum size of the connection pool for authentication requests. This parameter has fixed value (same as data_connections) and is currently not settable!
(Authentication operations incur an implicit logout of whatever identity was used before, which could incur extra latency if authentication operations and metadata lookups were to be done on the same connection, which is why ldap_adaptor has two separate connection pools.)
Default: (same as data_connections)

`data_connections`
The number of LDAP connections that will be kept active to the LDAP server for the purpose of metadata searches.
Default: 4

`ldap_edirectory_workarounds`
Attempt to deal with wire protocol violations brought about by Novell/NetIQ eDirectory server

© grommunio GmbH — CC BY-SA 4.0

implementations.
Default: false

`ldap_bind_user`
An LDAP binddn to use for metadata searches. You can only use Simple Authentication at this time. If an organization object defines LDAP credentials of its own, those will be used in preference to ldap_bind_user.
Default: (unset)

`ldap_bind_pass`
Password for Simple Authentication of ldap_bind_user.
Default: (unset)

`ldap_host`
Whitespace-separated set of LDAP URIs in the form of *ldap[si]://[name[:port]]* for the default LDAP tree. The openldap2 ldap.conf(5) manpage does not specify trailing slashes or DN bases like RFC 2255 does, and because of this, you should not use them. openldap2 utilities accept-ignore such part of the URI, while other implementations like python-ldap3 fail to connect. Per-organization LDAP credentials override ldap_host as necessary.
Default: (libldap default, see ldap.conf(5))

`ldap_mail_attr`
The name of the LDAP attribute which holds the primary e-mail address of the user. Pick **mail** (OpenLDAP as well as Active Directory schemes).
Default: (empty)

`ldap_search_base`
Default: (libldap default)

`ldap_start_tls`
Use the STARTTLS mechanism on LDAP connections. Prefer using Explicit TLS (ldaps:// in the URI field) in favor of ldap:// with STARTTLS; see Internet blog posts "STARTTLS considered harmful" for details.
Default: off
Take note that libldap may reject self-signed certificates from the LDAP server. This may be worked around with the "TLS_REQCERT allow" directive in ldap.conf. See the ldap.conf(5) manpage for details. However, by its description, TLS_REQCERT will also make encryption optional, which means becoming the victim of a downgrade attack is a possibility.

Grommunio Admin API directives

`ldap_host`

`ldap_bind_user`

`ldap_bind_pass`

`ldap_search_base`

© grommunio GmbH — CC BY-SA 4.0

`ldap_start_tls`

`ldap_mail_attr`
(These six as above)

`ldap_disabled`
If true, Grommunio Admin API will not make use of LDAP (which generally just means synchronization). This directive has no effect on Gromox; users which have been synchronized previously and which exist in MySQL keep their validity as far as Gromox is concerned.

`ldap_object_id`
The name of the LDAP attribute which holds a unique, unchanging object identifier for synchronization purposes. Pick **entryUUID** for OpenLDAP, **objectGUID** for Active Directory.
Default: (empty)

`ldap_user_filter`
An LDAP search filter that specifies which users should be synchronized. Recommendations are **(objectClass=posixAccount)** for OpenLDAP/RFC2307bis, **(objectClass=user)** for Active Directory.
Default: (empty)

`ldap_user_displayname`
The name of the LDAP attribute which holds the value for PR_DISPLAY_NAME. Pick **displayName** (OpenLDAP as well as Active Directory schemes).
Default: (empty)

`ldap_user_search_attrs`
The name(s) of LDAP attributes which the Admin API will compare when using AAPI's search function. To specify multiple attributes, repeat this directive, and specify one attribute per line, i.e. put *ldap_user_search_attrs=mail* and *ldap_user_search_attrs=cn*, etc. in the config file.
Default: (empty set)

`ldap_user_templates`
The name(s) of Admin API templates to use. Multi-value directive like search_attrs. Pick **ldap_user_templates=common** and **ldap_user_templates=OpenLDAP** for OpenLDAP, or **ldap_user_templates=common** and **ldap_user_templates=ActiveDirectory** for Active Directory.
Default: (empty set)

`ldap_user_aliases`
The name of the LDAP attribute which contains secondary e-mail addresses. Pick **mailAlternativeAddress** (OpenLDAP) or **proxyAddresses** (Active Directory). The **smtp:** prefix in proxyAddresses is automatically trimmed when read.
Default: (empty)

© grommunio GmbH — CC BY-SA 4.0

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7), **authmgr**(4gx)

© grommunio GmbH — CC BY-SA 4.0

mapi(4gx)

Name

mapi.so — PHP module providing MAPI functions

Description

mapi.so is a PHP module that makes available a number of functions to PHP for connecting to Gromox services. In particular, the PHP module will regularly invoke RPCs to zcore(8gx).

Configuration

The PHP ini fragment, mapi.ini, may look like this:

```
extension=mapi.so
[mapi]
zcore_socket=/run/gromox/zcore.sock
```

See also

gromox(7), **zcore**(8gx)

© grommunio GmbH — CC BY-SA 4.0

mh_emsmdb(4gx)

Name

mh_emsmdb — Handler for MAPIHTTP-wrapped EMSMDB requests

Description

mh_emsmdb is a component for http(8gx) which reacts to URIs starting in **/mapi/emsmdb/**. The ROP stream inside the HTTP request bodies is handed off to exchange_emsmdb(4gx) for subsequent processing.

Configuration directives

This component has no config directives of its own.

Normative references

- MS-OXCMAPIHTTP: Messaging Application Programming Interface (MAPI) Extensions for HTTP

See also

gromox(7), **http**(8gx), **exchange_emsmdb**(4gx)

© grommunio GmbH — CC BY-SA 4.0

mh_nsp(4gx)

Name

mh_nsp — Handler for MAPIHTTP-wrapped NSPI requests

Description

mh_nsp is a component for http(8gx) which reacts to URIs starting with **/mapi/nspi/**. The ROP stream inside the HTTP request bodies is handed off to exchange_nsp(4gx) for subsequent processing.

Configuration directives

This component has no config directives of its own.

Normative references

- MS-OXCMAPIHTTP: Messaging Application Programming Interface (MAPI) Extensions for HTTP

See also

gromox(7), **http**(8gx), **exchange_nsp**(4gx)

© grommunio GmbH — CC BY-SA 4.0

midb_agent(4gx)

Name

midb_agent — Client for midb(8gx)

Configuration directives

The usual config file location is /etc/gromox/midb_agent.cfg.

`connection_num`
The number of connections to keep open towards every midb target.
Default: 5

`context_average_mem`
Enables and sets the size of a memory pool (in object count; the actual size is 256 bytes * context_num * context_average_mem). imap(8gx) and pop3(8gx) require this to be non-zero for full functionality.
Default: 1024

`midb_agent_command_buffer_size`
Certain midb commands can produce large results (such as P-SRHU with "ALL"). To avoid unbounded memory allocation, the result set is limited in size. If midb responds with a larger result, the midb connection is terminated and midb_agent returns an error to e.g. IMAP/POP. The default of 256K is good for a result set of around 24000 to 32000 messages.
Default: 256K

Multiserver map

The SQL column **users.homedir** specifies a home directory location in an abstract namespace. This abstract namespace is shared between all Gromox programs, and can be used to divide users into custom subsets and steer connections to different servers.

midb_list.txt specifies how to map from this namespace to midb servers. Each line in this file consists of 3 columns separated by whitespace:

- Initial prefix to match a user's exmdb home directory on. The pattern should almost always end in a '/' character, otherwise a prefix of "/home" is able to match a userdir of "/home2/username" as well, which may be undesired.
- The IPv6 address of the midb server to connect to for all requests involving this prefix. The address must conform to gromox(7) § "Host addresses".

© grommunio GmbH — CC BY-SA 4.0

- The port number.

In the absence of midb_list.txt, an implicit default entry is used:

```
/ ::1 5555
```

Files

- *config_file_path*/*instance*.cfg: configuration file for the instance of midb_agent. (Instance is usually **midb_agent**, as there is not much point in loading midb_agent twice.)
- *config_file_path*/midb_list.txt: midb multiserver map

config_file_path is determined by the configuration of the program that loaded the midb_agent component.

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

midb(8gx)

Name

midb — Message Index database daemon

Synopsis

midb [**-c** *config*]

Description

midb is used by imap(8gx) and pop3(8gx) to generate and keep track of folder and message summaries. In particular, it enumerates messages to give them their IMAP UID and have at all times a suitable UIDNEXT value for folders ready. midb also caches the Message-Id, modification date, message flags, subject and sender to facilitate IMAP listings.

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/midb.cfg will be read.

`--version`
Output version information and exit.

`-?`
Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`daemons_fd_limit`
In gromox-midb, this is treated as an alias for midb_fd_limit.

`midb_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the

© grommunio GmbH — CC BY-SA 4.0

system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

`midb_listen`
A space-separated list of bind address specifiers to expose the MIDB service on.
Default: *[::1]:5555*

Configuration directives (midb.cfg)

The following directives are recognized when reading from /etc/gromox/midb.cfg, or when the **-c** option is used to specify a custom file:

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/midb/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/midb:/etc/gromox

`data_file_path`
Colon-separated list of directories which will be scanned when locating data files.
Default: /usr/share/gromox/midb

`midb_cache_interval`
The time after the last use of a particular midb.sqlite3 that the sqlite gets unloaded.
Default: 30minutes

`midb_cmd_debug`
Log every incoming MIDB command and the return code of the operation in a minimal fashion to stderr (not midb_log_file!). Level 1 emits commands with a failure return code, level 2 emits all commands. Logs are written to stderr only.
Default: 0

`midb_hosts_allow`
A space-separated list of individual host addresses that are allowed to converse with the midb service. The addresses must conform to gromox(7) § "Host addresses". No networks and no CIDR notations are permitted. This option deprecates the /etc/gromox/midb_acl.txt file used before Gromox 2.8.
Default: ::1

`midb_listen_ip`
Deprecated in favor of gromox.cfg:midb_listen.

`midb_listen_port`
Deprecated in favor of gromox.cfg:midb_listen.

`midb_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or

© grommunio GmbH — CC BY-SA 4.0

"*syslog*" are recognized.
Default: - (auto)

`midb_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`midb_reload_interval`
The time after a midb.sqlite3 was first loaded that it will be unloaded.
Default: 60min

`midb_schema_upgrades`
This directive controls whether database schemas are automatically upgraded when a message index database (midb.sqlite) is loaded. During this time, that DB is unavailable and operations on it will be delayed. Connection aborts, if any, would be due to timeouts in components other than midb. (The file can temporarily grow to double its size, so ample disk space may be required.)
Default: yes

`midb_table_size`
Default: 5000

`midb_threads_num`
The exact number of client processing threads to keep around. This also controls the maximum number of concurrent connections by external clients towards midb.
Default: 100

`rpc_proxy_connection_num`
The maximum number of (idle) connections towards Information Store homeservers that are kept alive for rapid re-use.
Default: 10

`sqlite_debug`
If set to 1, every query given to SQLite prepare/execute is logged. If set to 0, only failed queries are logged. (It cannot be made completely silent, since our queries ought to never fail.)
Default: 0

`x500_org_name`
Default: (unspecified)

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

mod_cache(4gx)

Name

mod_cache — HTTP handler for serving objects from a local filesystem

Description

mod_cache serves local files when certain URIs are requested. Note that mod_fastcgi(4gx) has a table of its own and higher precedence.

Configuration directives

This component shares **http.cfg**. See http(8gx).

URI map

The filemap that specifies which URIs to handle is **cache.txt**, which is searched for in *config_file_path*. The usual location is /etc/gromox/cache.txt.

Each line in this file consists of 3 columns separated by whitespace:

- Domain or wildcard to match the HTTP Host: header with.
- URI path (prefix) to match
- Target file/directory within the filesystem

If the file has no lines, no documents will be served this way. If the file is absent however, a set of default entries will be used.

Default entries

```
<em> /web /usr/share/grommunio-web</em>
<em> /EWS/Messages.xsd /usr/share/gromox/Messages.xsd</em>
<em> /EWS/Services.wsdl /usr/share/gromox/Services.wsdl</em>
<em> /EWS/Types.xsd /usr/share/gromox/Types.xsd</em>
```

Files

- *config_file_path*/cache.txt: URI map specifying which paths are served

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

mod_fastcgi(4gx)

Name

mod_fastcgi — http(8gx) handler for proxying requests to FastCGI servers

Description

mod_fastcgi can forward HTTP requests to one or more FastCGI servers when certain URIs are requested. Take special note of the file suffix column documented below; in particular, to override an entire directory, you may also need to edit the table for mod_cache(4gx).

Configuration directives

This component shares **http.cfg**. See http(8gx).

URI map

The filemap that specifies which URIs to handle is **fastcgi.txt**, which is searched for in *config_file_path*. The usual location is /etc/gromox/fastcgi.txt.

Each line in this file consists of 7 columns separated by whitespace:

- Domain or asterisk-based wildcard ("*", "*.example.com") to match the HTTP Host: request header with.
- URI path (prefix) to match
- Mapped path that will be passed to the FastCGI executor. Note that the FastCGI process may additionally have a document root setting that could map the path one more time.
- File suffix (without dot) to match on, e.g. *php*.
- A file to use as the default file for a directory (similar to Apache httpd's DirectoryIndex).
- A set of headers, separated by the pipe symbol |, which should be forwarded to the CGI handler. In absence of any desired extra headers, a single pipe can be used to fill the column.
- An AF_LOCAL socket path to make the FastCGI request to.

If the file has no lines, no documents will be served this way. If the file is absent however, a set of default entries will be used.

© grommunio GmbH — CC BY-SA 4.0

Default entries

```
<em> /sync /usr/share/grommunio-sync php index.php | /run/php-fpm/php-grommunio-s
<em> /web /usr/share/grommunio-web php index.php | /run/php-fpm/php-grommunio-web
<em> /dav /usr/share/grommunio-dav php index.php | /run/php-fpm/php-grommunio-dav
```

NOTE: The path designated for FPM sockets varies between distributions, and because some distributions *also* have security policies that involve path matching (e.g. via AppArmor), we are unable to choose a path that works out-of-the-box on all distro-provided platform configurations without fail. /run/php-fpm is used as the default in line with the Grommunio Appliance.

Files

- *config_file_path*/fastcgi.txt: Map specifying with paths mod_fastcgi should react to, and which FastCGI backend to route requests to

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

mod_rewrite(4gx)

Name

mod_rewrite — http(8gx) handler for altering HTTP request URIs before processing

Description

mod_rewrite can alter request URIs. It runs before any of the other built-in or HTTP processing modules.

Configuration directives

This component shares **http.cfg**. See http(8gx).

Rewrite map

The filemap that specifies which URIs to handle is **rewrite.txt**, which is searched for in *config_file_path*. The usual location is /etc/gromox/rewrite.txt.

Each line in this file consists of 3 columns separated by whitespace:

- A POSIX Basic Regular Expression (cf. regcomp(3)) for matching the original URI. For safety, this should always be anchored with the beginning-of-line metacharacter (^; circumflex).
- The fixed sequence "=>".
- Replacement string. Captures can be spliced using **\1**, **\2**, etc. The sequence **\0** splices the entire string (equivalent of Perl's **$&**). If a particular pattern has successfully matched, no other rewrite rules are processed.

If the file has no lines, no paths will be rewritten. If the file is absent however, a set of default entries will be used.

Default rules

```
^/Microsoft-Server-ActiveSync(/\|$) => /sync/index.php
```

Examples

© grommunio GmbH — CC BY-SA 4.0

```
^/([a-z+])/([a-z]+).txt$ => /\2/\1.txt
```

Files

- *config_file_path*/rewrite.txt: Map specifying how to rewrite URIs

See also

gromox(7), **http**(8gx)

© grommunio GmbH — CC BY-SA 4.0

mysql_adaptor(4gx)

Name

mysql_adaptor — MySQL/MariaDB connector for user metadata and authentication

Description

mysql_adaptor is a component for integrating user accounts from a MySQL/MariaDB database.

Configuration directives

The usual config file location is /etc/gromox/mysql_adaptor.cfg.

`connection_num`
Number of SQL connections to keep active. Note that the SQL server may have limits in place, such as "max_connections" and "wait_timeout" (cf. `SHOW GLOBAL VARIABLES LIKE "wait_timeout"`). Automatic reconnection happens by Gromox when a query is about to be executed.
Default: 8

`enable_firsttimepw`
This flag determines whether non-LDAP users with no recorded password (empty users.password SQL column) will have the account's password set to whatever credential was passed along in the first authentication request.
Default: no

`mysql_dbname`
Default: email

`mysql_host`
The hostname/IP address for contacting the SQL server.
Default: localhost

`mysql_password`
Default: (unset)

`mysql_pool_idle_timeout`
Pooled SQL connections that have not been used for this long are closed when convenient, the exact time of which is unspecified, and connections are re-established on-demand. This keeps long-lived but mostly-idle processes, e.g. per-store istore workers when istore_standalone is enabled, from each holding a connection indefinitely and collectively running the SQL server into its max_connections limit. A value of 0 disables the reaper.
Default: 1min

© grommunio GmbH — CC BY-SA 4.0

> `mysql_port`
> The TCP port number for contacting the SQL server.
> Default: 3306

> `mysql_rdwr_timeout`
> This sets all three of MYSQL_OPT_READ_TIMEOUT, MYSQL_OPT_WRITE_TIMEOUT and MYSQL_OPT_CONNECT_TIMEOUT option values on the MySQL connection.
> Default: *0* (unlimited) for read/write, *20* for connect

> `mysql_tls_cert`
> The path name of an optional client public key certificate file for authentication.
> Default: (empty)

> `mysql_tls_key`
> The path name of an optional client private key file for authentication.
> Default: (empty)

> `mysql_username`
> Default: root

> `scan_interval`
> Default: 1 minute

> `schema_upgrade`
> This controls what to do when a database schema update is available. Because the mysql_adaptor component is loaded by many programs across potentially multiple machines, the basic default is to do nothing.

host:*xyz*
Perform automatic schema upgrades if the program which has loaded mysql_adaptor is http(8gx) and only if http.cfg's **host_id** value matches *xyz*.

(any other value)
No automatic schema upgrades are performed in the process which has loaded the component. The component may be unable to process or produce certain data, e.g. if it relies on a table that has not yet been created. Generally, no restart is needed when that upgrade is finally performed, because SQL queries just start working. However, there may be some caches, e.g. in the zcore(8gx) address book that could remain empty until the refresh timer expires or a manual reload is triggered.

Default: host:(system_hostname)

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **authmgr**(4gx)

© grommunio GmbH — CC BY-SA 4.0

oab(4gx)

Name

oab — Handler for the Offline Address Book

Description

oab is a component for http(8gx) which handles URIs with the /OAB/ prefix.

Configuration directives (exchange_nsp.cfg)

If it exists, /etc/gromox/exchange_nsp.cfg will be read.

> `cache_interval`
> Address book cache invalidation time.
> Default: 5 minutes

© grommunio GmbH — CC BY-SA 4.0

pam_gromox(4gx)

Name

pam_gromox — a PAM plugin to authenticate with Gromox

Description

This module feeds authentication requests to Gromox's authmgr(4gx) module, and thus mysql_adaptor(4gx) and/or ldap_adaptor(4gx). It does not rely on the availability of any Gromox service; just MySQL/LDAP is enough. pam_gromox is meant to be used in conjunction with non-Gromox processes that an administrator may wish to integrate with, such as an SMTP daemon.

Incantation in /etc/pam.d/smtp

Gromox accounts are not mapped from or to any Unix accounts, so the pam_unix.so module that is present in the default /etc/pam.d/smtp module list within Linux distributions is not suitable and can be wholly replaced. In otherwords, /etc/pam.d/smtp need just contain:

```
auth required pam_gromox.so service=smtp
account required pam_permit.so
```

(pam_gromox does not provide a usable "account" handler, therefore "account required pam_gromox.so" would do nothing. The PAM framework always starts out with an initial deny policy, so at least one module needs to be called to make the PAM request succeed. For this reason, if there are no other "account" modules listed, pam_permit.so should be used.)

PAM module arguments

service=*s*
Check for a specific privilege bit on the user account. Possible values for *s* are: **exch**, **smtp**, **imap**, **pop3**, **chat**, **video**, **files**, **archive**.
Default: *smtp*

Configuration directives

The usual config file location is /etc/gromox/pam.cfg.

© grommunio GmbH — CC BY-SA 4.0

`config_file_path`
Colon-separated list of directories in which further configuration files.
Default: /etc/gromox/pam:/etc/gromox

`pam_prompt`
If pam_gromox detects the absence of a password but presence of a PAM conversation function, it will attempt to retrieve the password that way, and in doing so, will show this label just ahead of the nonechoing password prompt.
Default: Password:

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

php-mapi(4)

This is an alias. See **mapi(4gx)**.

© grommunio GmbH — CC BY-SA 4.0

pop3(8gx)

Name

pop3 — Gromox POP3 server

Authentication

The POP3 server supports impersonation; see imap(8gx) for details.

Synopsis

pop3 [**-c** *config*]

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/pop3.cfg will be read.

`--version`
Output version information and exit.

`-?`
Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`daemons_fd_limit`
In gromox-pop3, this is treated as an alias for pop3_fd_limit.

`pop3_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

© grommunio GmbH — CC BY-SA 4.0

`pop3_accept_haproxy`
This directive sets the expectation for incoming connections to carry haproxy's "PROXY" protocol extension version 2 (2), or no such header (0). When a (reverse) proxy is placed in front of gromox-pop3, the address that gxpop3 normally sees is the proxy address (e.g. ::1). A proxy can use this protocol extension to convey the actual client address, and gxpop3 can pick this up for its own reporting, which in turn is useful for e.g. fail2ban setups.
Default: 0

`pop3_listen`
A space-separated list of bind address specifiers to expose standard POP3 (unencrypted and/or explicit STARTTLS) on.
Default: *[::]:110*

`pop3_listen_tls`
A space-separated list of bind address specifiers to expose implicit-TLS POP3 (POP3S) on.
Default: *[::]:995*

Configuration directives (pop3.cfg)

The following directives are recognized when reading from /etc/gromox/pop3.cfg, or when the **-c** option is used to specify a custom file:

`block_interval_auths`
The amount of time a user is blocked from connecting to the service after too many failed logins.
Default: 1 minute

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/pop3/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/pop3:/etc/gromox

`context_average_mem`
Default: 256K

`context_average_units`
Lower clamp is 256.
Default: 1024

`context_max_mem`
Network buffer per client.
Default: 2M

`context_num`
Default: 200

© grommunio GmbH — CC BY-SA 4.0

`data_file_path`
Colon-separated list of directories in which static data files will be searched.
Default: /usr/share/gromox/pop3

`enable_capa_implementation`
When enabled, the server will include an "IMPLEMENTATION" line in the CAPA response (RFC 2449 §6.9). This is disabled by default, as it can facilitate potential attackers' information gathering.
Default: no

`host_id`
A unique identifier for this system. It is used in the POP3 protocol greeting lines (positive as well as negative). The identifier should only use characters allowed for hostnames.
Default: (system hostname)

`pop3_auth_times`
The number of login tries a user is allowed before the account is blocked.
Default: 3

`pop3_certificate_passwd`
The password to unlock TLS certificates.
Default: (unset)

`pop3_certificate_path`
A colon-separated list of TLS certificate files. The complete certificate chain should be present (as there is no other config directive to pull CA certs in, and implicit loading from system directories is not guaranteed by Gromox).
Default: (unset)

`pop3_cmd_debug`
Log every incoming POP3 command and the return code of the operation in a minimal fashion to stderr (not pop3_log_file!). Level 1 emits commands that have failed execution, level 2 emits all commands.
Default: 0

`pop3_conn_timeout`
If a POP3 connection is inactive for the given period, the connection is terminated.
Default: 3 minutes

`pop3_force_tls`
This flag controls whether clients must utilize TLS, either by way of implicit TLS, or through the STLS command.
Default: false

`pop3_listen_addr`
Deprecated in favor of gromox.cfg:pop3_listen / gromox.cfg:pop3_listen_tls.

© grommunio GmbH — CC BY-SA 4.0

`pop3_listen_port`
Deprecated in favor of gromox.cfg:pop3_listen.

`pop3_listen_tls_port`
Deprecated in favor of gromox.cfg:pop3_listen_tls.

`pop3_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`pop3_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`pop3_private_key_path`
A colon-separated list of TLS certificate private key files.
Default: (unset)

`pop3_support_tls`
This flag controls the offering of TLS modes. This affects both the implicit TLS port as well as the advertisement of the STARTTLS extension and availability of the STLS command (RFC 2595) to clients.
Default: false

`pop3_thread_charge_num`
Connection load factor (oversubscription ratio) for a processing thread.
Default: 40

`pop3_thread_init_num`
The initial and also minimum number of client processing threads to keep around. This is similar to php-fpm's start_servers/min_spare_servere. (The maximum number of threads, i.e. what would be max_spare_servers, is determined by: context_num divided by imap_thread_charge_num)
Default: 5

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

`tls_min_proto`
The lowest TLS version to offer. Possible values are: **tls1.0**, **tls1.1**, **tls1.2**, and, if supported by the system, **tls1.3**.
Default: tls1.2

© grommunio GmbH — CC BY-SA 4.0

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Files

- *data_file_path*/pop3_code.txt: Mapping from internal POP3 error codes to textual descriptions.

See also

gromox(7), **midb_agent**(4gx)

© grommunio GmbH — CC BY-SA 4.0

timer_agent(4gx)

Name

timer_agent — Client for timer(8gx)

Description

timer_agent connects to a remote timer(8gx) daemon and locally installs two service functions, "add_timer" and "cancel_timer", which can be used to set up and rescind, respectively, jobs for later execution.

In practice, this is used by exchange_emsmdb(4gx) and zcore(8gx) to implement delayed sending of messages.

Configuration directives

The usual config file location is /etc/gromox/timer_agent.cfg.

`connection_num`
Number of connections to keep active.
Default: 8

`timer_host`
The hostname/IP address for contacting the timer daemon.
Default: ::1

`timer_port`
The TCP port number for contacting the timer daemon.
Default: 6666

See also

gromox(7), **timer**(8gx)

© grommunio GmbH — CC BY-SA 4.0

timer(8gx)

Name

timer — deferred command executor

Synopsis

timer [**-c** *config*]

Description

The timer daemon can be used to schedule commands to be executed once, at a particular time in the future. It is similar to the at(1) command and its associated daemon, atd.

timer(8gx) generally receives commands from timer_agent(4gx).

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/timer.cfg will be read.

`--version`
Output version information and exit.

`-?`
Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Files

- /var/lib/gromox/timer.txt: This file is used to save the state of timer(8gx) and persist them across restarts.

Configuration directives

The usual config file location is /etc/gromox/timer.cfg.

© grommunio GmbH — CC BY-SA 4.0

`running_identity`
An unprivileged user account to switch the process to after startup. To inhibit the switch, assign the empty value.
Default: gromox

`timer_hosts_allow`
A space-separated list of individual IPv6 or v4-mapped IPv6 host addresses that are allowed to converse with the timer service. No networks and no CIDR notations are permitted.
Default: ::1

`timer_listen_ip`
An IPv6 address (or v4-mapped address) for exposing the timer service on.
Default: ::1

`timer_listen_port`
The TCP port number for exposing the timer service on.
Default: 6666

`timer_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`timer_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`timer_state_path`
Default: /var/lib/gromox/timer.txt

`timer_threads_num`
The minimum number of client processing threads to keep around.
Default: 50

Timer protocol

The timer service is exposed as a line-based text protocol. Upon connection, the event server gratitiously writes "OK", following which the server will wait for timer commands, and execute them synchronously.

The command "ADD <seconds> <command>" installs a new timer for the given command to be executed in that many seconds from now. The server will respond with "FALSE 2", "FALSE 3", or respond with the timer ID as "TRUE <id>".

The command "CANCEL <id>" revokes the timer with the chosen ID.

© grommunio GmbH — CC BY-SA 4.0

See also

gromox(7), **timer_agent**(4gx)

© grommunio GmbH — CC BY-SA 4.0

user_filter(4gx)

Name

user_filter — User logon limiter

Description

This component implements two concepts (each independently configurable):

- A mechanism for banning user identities for a set time window. When a user repeatedly fails to successfully authenticate, the http(8gx), imap(8gx), pop3(8gx) daemons can add the user to this list and set a time during which all authentication requests for the user are rejected. This is a bit like fail2ban, but operates on usernames rather than hosts/IP addresses.
- A mechanism for rate-limiting authentication attempts. Whenever a user tries to authenticate, the daemons convey the occurrence to the user_filter component, and the component ensures that only a given amount of attempts can be made per time quantum, per user. This is a bit like iptables -m (hash)limit.

Configuration directives (gromox.cfg)

`userfilter_icase`
Treat usernames as case-insensitive within the user_filter component.
Default: true

`userfilter_maxbans`
Controls how much memory the banlist mechanism of user_filter is allowed to use at most, by limiting the number of unique usernames recorded. The list replacement policy is none (so, slightly different from MRU). The value 0 therefore deactivates user_filter's banlist mechanism.
Default: 1000

`userfilter_maxusers`
Controls how much memory the rate-limiting mechanism of user_filter is allowed to use at most, by limiting the number of unique usernames. The list replacement policy is none. The value 0 therefore deactivates user_filter's rate-limiting mechanism.
Default: 0

`userfilter_rl_maxtries`
Rate-limit all authentication calls to rl_maxtries per rl_window. Note that there can be **a lot** of requests, particularly over MAPI/HTTP since every single HTTP request counts as one attempt. (Opening a message with MFCMAPI already incurs 4 HTTP requests. The Windows EMSMDB

© grommunio GmbH — CC BY-SA 4.0

> connector is anything but efficient.)
> Default: 10

> `userfilter_rl_window`
> Rate-limit all authentication attempts to rl_maxtries per rl_window.
> Default: 1minute

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0

zcore(8gx)

Name

zcore — Bridge for PHP-MAPI requests

Synopsis

zcore [**-c** *config*]

Description

zcore is a bridge process (proxy) between PHP-MAPI and the Information Store (see manpages mapi(4gx) and exmdb_provider(4gx), respectively). It listens on /run/gromox/zcore.sock (hardcoded) for zcore RPCs, a Gromox-specific protocol and issues exmdb RPCs to exmdb_provider(4gx). As exmdb_provider connections have no state to speak of, zcore is the process that defines the logins sessions. zcore needs to run on the same server as the program that uses the PHP-MAPI functions.

Options

`-c` *config*
Read configuration directives from the given file. If this option is not specified, /etc/gromox/gromox.cfg and /etc/gromox/zcore.cfg will be read.

`--version`
Output version information and exit.

`-?`
Display option summary.

All time-based command-line options and configuration file directives are subject to the syntax described in gromox(7), section "Duration specifications".

Configuration directives (gromox.cfg)

The following directives are recognized when reading from /etc/gromox/gromox.cfg, or when the **-c** option is used to specify a custom file:

`daemons_fd_limit`
In gromox-zcore, this is treated as an alias for zcore_fd_limit.

© grommunio GmbH — CC BY-SA 4.0

`outgoing_smtp_url`
See gromox.cfg(5):outgoing_smtp_url.

`zcore_fd_limit`
Request that the file descriptor table be at least this large. The magic value 0 indicates that the system default hard limit (rlim_max, cf. setrlimit(2)) should be used.
Default: 0

Configuration directives (zcore.cfg)

The following directives are recognized when reading from /etc/gromox/zcore.cfg, or when the **-c** option is used to specify a custom file:

`address_cache_interval`
Default: 5 minutes

`address_table_size`
Default: 3000

`config_file_path`
Colon-separated list of directories which will be scanned when locating further configuration files, especially those used by subcomponent instances. (For example, mysql_adaptor(4gx) would be directed to look at /etc/gromox/zcore/mysql_adaptor.cfg before /etc/gromox/mysql_adaptor.cfg.)
Default: /etc/gromox/zcore:/etc/gromox

`data_file_path`
Colon-separated list of directories which will be scanned when locating data files.
Default: /usr/share/gromox/zcore

`host_id`
A unique identifier for this system. It is used for the HELO line of outgoing SMTP connections, and as an unused identifier within muidStoreWrap entryids. The identifier should only use characters allowed for hostnames.
Default: (system hostname)

`mailbox_ping_interval`
Default: 5 minutes

`mail_max_length`
Default: 64M

`max_ext_rule_length`
Default: 510K

`max_rcpt_num`
The maximum number of recipients that an e-mail is allowed to have.
Default: 256

© grommunio GmbH — CC BY-SA 4.0

`rpc_proxy_connection_num`
The maximum number of (idle) connections towards Information Store homeservers that are kept alive for rapid re-use.
Default: 10

`submit_command`
Default: /usr/bin/php /usr/share/gromox/submit.php

`user_cache_interval`
Sets the time how long the MAPI profile is cached before it is written to disk.
Default: 1 hour

`user_table_size`
Default: 5000

`x500_org_name`
Default: (unspecified)

`zcore_listen`
The named path for the AF_LOCAL socket that zcore will listen on.
Default: /run/gromox/zcore.sock

`zcore_log_file`
Target for log messages here. Special values: "-" (stderr/syslog depending on parent PID) or "*syslog*" are recognized.
Default: - (auto)

`zcore_log_level`
Maximum verbosity of logging. 1=crit, 2=error, 3=warn, 4=notice, 5=info, 6=debug.
Default: *4* (notice)

`zcore_max_obh_per_session`
The maximum number of object handles each session can have at any one time (e.g. folders/messages/etc. open simultaneously). Use 0 to indicate unlimited. There is one session for each time a mailbox is opened.
Default: 500

`zcore_threads_num`
The minimum number of client processing threads to keep around.
Default: 10

`zrpc_debug`
Log every incoming zcore RPC and the return code of the operation in a minimal fashion to stdout. Level 1 emits RPCs with a failure return code, level 2 emits all RPCs. Note the daemon log level needs to be "debug" (6), too.
Default: 0

© grommunio GmbH — CC BY-SA 4.0

Network protocol

The transmissions on the zcore socket are simple concatenations of protocol data units built using the NDR format. The PDU length is present within the PDU itself near the start.

```
{
  leuint32_t length;
  char pdu[];
}

pdu := {
  uint8_t call_id;
  string directory;
  switch (call_id) {
    ...
  }
}
```

Store lookup

zcore determines the homeserver and store path for a user from the user database, which is provided by mysql_adaptor(4gx).

Each line in this file consists of 4 columns separated by whitespace:

- A portion of the store path to match on
- The type of store ("private" or "public")
- The IPv6 socket address of the server running exmdb_provider(4gx). The address must conform to gromox(7) § "Host addresses".
- The TCP port number of the server

Signals

Upon receipt of SIGHUP, configuration files are re-read, but only a few select directives can be changed this way, as many parts do not implement reload.

Upon receipt of SIGHUP, the address book cache will be dropped; the next regular request for the AB will cause it to be reloaded.

Files

- *data_file_path*/folder_names.txt: Translations for essential folders in a message store.

© grommunio GmbH — CC BY-SA 4.0

- *data_file_path*/notify_bounce/: templates for read/nonread notification mails sent to originators

Notes

Behavior for the address book generally mirrors exchange_nsp(4gx), so see that manpage for additional notes.

See also

gromox(7)

© grommunio GmbH — CC BY-SA 4.0