NAME

mailbox − zmailer local delivery transport agent

SYNOPSIS

mailbox

[−**8abCDHPrRSUVX**] [−**F** `edquot`] [−**c** *channel*] [−**h** *"localpart"*] [−**l** *logfile*] [−**d** *dirpath*]

DESCRIPTION

mailbox is a **ZMailer** transport agent which is usually only run by the *scheduler*(8zm) program to deliver mail to local user mailbox files. The *mailbox* program must be run with root privileges and invoked with the same current directory as the *scheduler*, namely *POSTOFFICE*/**transport**.

Recipient addresses are processed as follows:

- Strip doublequotes around the address, if any.
- Strip prefixing backslashes, if any.
- If the address starts with a '|', the rest of the recipient address string is interpreted as a shell command to be run.
- If the address starts with a '/', the recipient address is a filename to append the message to.
- Otherwise the recipient address must be a local user id.
- If user is not found, and the first character of the address is a capital letter, the entire address is folded to lowercase and the user lookup is retried.

If delivering to a user mailbox (**MAILBOX**/*userid*) which doesn't exist, *mailbox* will try to create it. If the **MAILBOX** directory is mounted from a remote system this will succeed if the directory is group-writable.

Some sanity checks are done on deliveries to files and mailboxes:

- The file being delivered to must have 1 link only, and must be either /dev/null or a regular file.
- The file lock must be held. (See below for a section about locks.)

There is a further sanity check on mailbox deliveries, namely if the mailbox is not empty the *mailbox* program will enforce 2 newlines as a separator before the message to be delivered. This guarantees that User Agents, like *Mail*(1), can find the about-to-be delivered message even if the current contents of the mailbox is corrupt.

When delivering to a process (by starting a Bourne shell to execute a specified command line), the environment is set up to contain several variables which are listed below at the "Subprogram Environment Variables" section. The **SIGINT** and **SIGHUP** signals are ignored, but **SIGTERM** is treated normally. If the process dumps core, it will be retried later. Sub-process exit codes are interpreted according to <sysexits.h> codes, and of those EX_NOPERM, EX_UNAVAILABLE, EX_NOHOST, EX_NOUSER, and EX_DATAERR are treated as permanent errors, all others are treated as temporary failures.

The actual data delivered to a file, mailbox, or process, is identical. It consists of the concationation of a UUCP style separator line, the message header specified in the message control file, and the message body from the original message file. The separator line starts with `"From "` and is followed by the sender address and a timestamp.

After all deliveries and just before exiting, the *mailbox* process will poke *comsat*(8C) in case recipients have turned on *biff*(1). The program may be compiled to look in the rwho files on the system for recipient names logged onto neighbouring hosts, in which case the *comsat* on the remote host will be poked. Even if this compile-time option is enabled, this will only be done for users that have a **.rbiff** file in their home directory. (Unless an '-DRBIFF _ALWAYS' compile option is used.)

OPTIONS

−8 enables MIME-QP-decoder to decode incoming MIME-email with Quoted-Printable encoded characters.

−a the access time on mailbox files is, by default, preserved across delivery, so that programs such as *login*(1) can determine if new mail has arrived. This option disables the above action.

−b disables biff notification.

−c *channel*
specifies which channel name should be keyed on. The default is **local**.

−C Canonify username by using internally version of username received in *pw_name* field of the *getpwnam()* call result.

−d *"dirpath"*
This sets the directory prefix where individual mailbox files reside at. In lacking of this, ZENV-variable MAILBOX value is used, and lacking it, following set is used:

/var/mail
/usr/mail
/var/spool/mail
/usr/spool/mail

Of those the one which yields first a directory is chosen.

The *"dirpath"* can also be a %-char containing format string:

%%	the '%' alone
%a	address as is
%u	userid
%U	long user name (userid if not supported)
%D	full domain name
%x	next character derived from PJW hash of userid
%X	next character derived from crc32 hash of userid
%h	userid's home directory
%n	(unimplemented, but reserved)
%N	(unimplemented, but reserved)

Some examples:

/var/mail/%u	standard mail directory
/var/mail/%x/%x/%u	hashed directory
%h/Mail/INBOX	mailbox in user's home
%h/mbox	mailbox in user's home for UW-IMAP..
/var/virt/%D/mail/%X/%X/%u	hashed spool with virtual domain

If parametrization, or default pickup fails, this program yields a "*TEMPFAIL*" status, and syslog's ALERT level messages.

−D[D..]
For a user with name as: *abcdef*, one *-D* will place the mailbox file into directory *£MAILBOX/a/abcdef*. With *-DD* the mailbox file will be placed into directory: *£MAILBOX/a/b/abcdef*. The limit on number of 'D's and resulting subdirs is 10.

If there are less chars in user name than given hash level says, hashing stops at the end of the name.

−F `edquot`
"Fatalify."

Parameter-full option that can turn into **fatal** things that previously were mere *TEMPFAIL*s.

This makes "quota exceeded" condition instantly fatal.

−h *"localpart"*
: specifies which of the possible multiple recipients is to be picked this time. Default is "none", which selects all local channel recipients, however when the routing is done with scripts storing some tokens (other than "−") into the "host"-part, it is possible to process "host-wise", i.e. so that each **user** has his/her own lock-state, and not just everybody hang on the same lock(s)..

−H
: Keep headers in 8-bit characters, not converting them to "MIME-2".

−l *logfile*
: specifies a logfile. Each entry is a line containing message id, pre-existing mailbox size in bytes, number of bytes appended, and the file name or command line delivered to.

−M
: enables the creation of MMDF-style mail-folder in the incoming mail folder. The default is "classic" UNIX-style folder.

−P[P..]
: This uses much of similar method as *-D[D..]* option, but directory names are derived from much more smoothly distributing hash function over user names, namely: *pjwhash32*().

  The hash is split modulo 26 into a reversing buffer, and then output encoded as uppercase characters. 'A' for 0, 'Z' for 25. E.g. for *-PPP* that would be analogous to base-10 numeric printout of: 654321 -> "3/2/1/"

  The result of these *-P[P..]* derived directory paths is something like: *£MAILBOX/X/username* or *£MAILBOX/Y/X/username*

  Note1: The Base-26 output consumes 4.7 bits of the hash at the time, which means that a 32 bit hash exhausts all of its bits in 7 levels.

  Note2: Depth of hash tree should be determined by individual filesystem capabilities. For example Solaris 8 UFS can handle up to 254 things on one directory level in fastest possible manner, anything over it, and things get more and more sluggish.

−r
: disables remote biff notification (if supported).

−S
: This option enables "Return-Receipt-To:" message header recognition and processing along with sending receipt to given address. *(Newer sendmails don't anymore support this facility per default..)*

−V
: prints a version message and exits.

−X[X..]
: This is similar to *-P[P..]* option, but used hash function is *crc32*(). Resulting distribution is slightly different, and in fact quite smooth.

INTERFACE

This program reads in processable file names relative to the current working directory of the scheduler (namely: *£POSTIOFFICE/transport/*). Optionally on the same line the scheduler may tell which host is to be looked for from the recipients of the message.

relative-spool-path [<TAB> *hostname*]

This program produces diagnostic output on the standard output. Normal diagnostic output is of the form:

id/offset<TAB>*notify-data*<TAB>*status message*

where *id* is the inode number of the message file, *offset* is a byte offset within its control file where the address being reported on is kept, *status* is one of **ok**, **error**, or **deferred**, and the *message* is descriptive text associated with the report. The text is terminated by a linefeed. Any other format (as might be produced by subprocesses) is passed to standard output for logging in the **scheduler** log.

The exit status is a code from file <**sysexits.h**>.

LOCKS

Locking scheme used at the system is configurable at the runtime, and has separate parameters for mailboxes, and files. The data is configurable with zenv variable **MBOXLOCKS** at which following characters have meanins:

':' Separates mailbox locks, and file-locks at the string. The left side has mailbox locks, and the right side has locks for other regular files. (Files with explicit paths defined.)

'.' For mailboxes only: Does "dotlock" (userid.lock), or (at Sun Solaris) maillock() mechanism.

'F' If the system has *flock()* system call, uses it to lock the entire file. *(Ignored at systems without flock()!)*

'L' If the system has *lockf()* system call, uses it to lock the entire file. *(Ignored at systems without lockf()!)*

Locks are acquired in the same order as the key characters are listed.

Default for the lockf() capable systems is:

MBOXLOCKS=".L:L"

You can choose insane combinations of lock mechanisms, which at some systems cause locks to fail always, like at Linux-2.0 series where program must not use both lockf() and flock() locks.

It is extremely important, that selected locking methods are same throughout the system at all programs trying to acquire locks on mail spools.

SECURITY

Like all parts of the **ZMailer**, the *mailbox*(8zm) chooses to err into overtly cautious side. In case of pipes being run under the *mailbox*(8zm), the program in pipe is started thru */bin/sh* with severely sanitized environment variables, and with only file descriptors STDIN, STDOUT, and STDERR. Programs are refused from running, if address analysis has found suspicuous data; external messages can't directly run programs, nor those addresses that have had a security breach detected during *.forward*-, or other aliasing analysis. (Same applies also with writing into explicitly named files.)

The pipe subprogram is run with user-id it gets thru the address privilege analysis during message routing, and it gets the group-id thru lookup of: getpwuid(uid). That is, if you have multiple usernames with same uid, there are no guarantees as to which of them is used for the gid entry.

The pipe subprogram is started **without** use of */bin/sh* command line interpreter (i.e. "system()" call), when the command line begins with slash, and does not contain characters: '$' and ' >'. If any of those rules is not fulfilled, the subprogram is started with "/bin/sh -c "$cmdlinestr"" call. This allows running pipes with carefully formed parameters, when the *mailbox* program is running inside shell-less chroot environment.

SUBPROGRAM ENVIRONMENT VARIABLES

The *mailbox* sets following environment variables for the subprograms it runs in the pipes:

HOME
The homedirectory path is taken from abovementioned *getpwuid*() lookup.

USER Likewise the textual username.

SENDER
is the incoming "MAIL FROM:<..>" address without brackets. For an incoming error message, value "<>" is used.

ORCPT
: when present, is the XTEXT encoded ORCPT value received at the message injection into this system. See RFC 1891 for details.

INRCPT
: A **ZMailer** specific thing which is supposed to carry the RCPT TO address that was given at the incoming SMTP session, independent of ORCPT data.

NOTIFY
: Possible (usually) externally received DSN NOTIFY parameter data.

BY
: Possible externally received DELIVERBY parameter data.

INFROM
: A **ZMailer** specific thing which is supposed to carry the MAIL FROM address that was given at the incoming SMTP session.

EZMLM
: A **ZMailer** specific thing which is actually present only at *listexpand*-utility expanded email list.

ENVID
: when present, is the XTEXT encoded ENVID value received at the message injection into this system. See RFC 1891 for details.

ZCONFIG
: is the location of the ZMailer ZENV file.

MSGSPOOLID
: Is the message spool-id in the ZMailer; subprograms may use this info in co-operation with ZMailer to e.g. *syslog*(3) what they have done to the arrived message.

MESSAGEID
: Is the RFC 822 "Message-ID:" header data as possibly copied into the control file; another item to support *syslog*(3) at programs.

MAILBIN
: is the value from ZENV.

MAILSHARE
: is the value from ZENV.

PATH
: is the value from ZENV, or "/usr/bin:/bin:/usr/ucb" in case no ZENV value is available.

SHELL
: is constant value: "/bin/sh".

IFS
: is constant value: " \t\n".

TZ
: is value from scheduler's environment variables via normal environment inheritance rules. Supposedly that of *systemwide* time-zone setting. Available to subprogram only if set when the *mailbox* was started.

ENVIRONMENT VARIABLES

ZCONFIG
: This environment variable is expected to be inherited from the *scheduler*(8zm), and it tells where scheduler's idea of *ZENV*-variables are located at.

Z−ENVIRONMENT VARIABLES

Following *ZENV-variables* are used by the **mailbox** program:

DEFCHARSET
: Supplies value for default charset, if builtin ISO-8859-1 is not appropriate, and 8-bit chars in headers need to be converted into proper "MIME-2" format.

MAILBOX
A directory path at which mailboxes reside. See above for option "−d".

MBOXLOCKS
This variable is used to define locking schemes used for mailbox spool files, and separately for other regular files. See the "locks" section above.

PATH This is passed onwards to subprograms.

ZCONFIG
This is passed onwards to subprograms.

MAILBIN
This is passed onwards to subprograms.

MAILSHARE
This is passed onwards to subprograms, and also on occasion used by the **mailbox** to find "$MAILSHARE/forms/return-receipt " form file.

FILES

/opt/mail/zmailer.conf	(ZCONFIG)
/var/spool/postoffice	(POSTOFFICE)
/var/spool/mail	(MAILBOX)

SEE ALSO

scheduler(8zm), *comsat*(8C), *biff*(1), *flock*(2), *Mail*(1), *mboxpath*(1zm), *zmailer.conf*(5zm).

RFC 822/2822	The basic Internet email format specification
RFC 1123	Various 822 parameter clarifications
RFC 1341/1521/2045	MIME specification (body, formats)
RFC 1342/1522/2047	"MIME-2" specification (headers)

AUTHOR

This program authored and copyright by:
Rayan Zachariassen <no address>
Extensive modifications by:
Matti Aarnio <mea@nic.funet.fi>