## NAME

smtpserver − zmailer SMTP server

## SYNOPSIS

**smtpserver**

[**-46aBhigntVvw**] [**-p** *port*] [**-l** *SYSLOG*] [**-l** *logfile*] [**-s**[**ftveRSDh**]] [**-s** `strict`] [**-s** `sloppy`] [**-I** *pidfile*] [**-L** *maxloadaver*] [**-M** *SMTPmaxsize*] [**-P** *postoffice*] [**-R** *router*] [**-C** *cfgfile*] [**-Z** *zenvfile*] [**-T** *'/1.2.3.4/'*]

## DESCRIPTION

This program implements the server side of the `SMTP` protocol as defined in RFC-2821, and knows about the common extensions to the protocol expected by *sendmail* and BSMTP clients.

By default the program will kill the previous *smtpserver*(8zm) daemon, if any, then detach and listen for `SMTP` connections. Incoming messages will be submitted for processing using the *zmailer*(3zm) interface to ZMailer.

Non-trivial address checking is done asynchronously, although this behaviour can be changed by a command line option if you cannot afford to transfer data just to bounce it back.

Complex routing analysis checking is done by executing the *router*(8zm) program in interactive mode, and executing a well-known shell function with well-known parameters for each request.

Traditionally this task has needed starting *router* process (and some other processes) underneath each incoming `SMTP` session, which is not a very lightweight thing.

Current version has prestarted farm of auxiliary server processes that can be used to handle complex things, and they can amortize e.g. *router* startup cost among lots of actual users making the use of routing in reception time very cheap in terms of resources.

Presently system starts a master process that starts a set of auxiliary processes, and keeps internal nameless rendezvous sockets for each of them. Then it creates listening sockets, and begins to accept(2) them (by means of select(2)). The master looks at contactee's IP number, and limits the number of parallel connections from any given IP address to configured value. If these first round limits don't bite, a SMTP talking server is fork(2)ed off, and in addition to the protocol socket, also subserver rendezvous sockets are passed to the protocol talker.

When a protocol talker needs to interact with an auxiliary server, it creates a socketpair(2) and uses nameless rendezvous socket to pass other end of the socketpair to the auxiliary server. The servers reply standard `"#hungry\n"` message to the protocol talker end to inform that they are ready for actual interaction. Exact details of this interface are not documented here, as that is not available to user programs.

The set of auxiliary servers is:

- Interactive *router* multiplexer. (If `"PARAM enable-router"` is activated.)
- Message content-analyzer multiplexer. (If `"PARAM contentfilter"` is defined, and is not pointing to named socket)
- Rate-tracker server. (Always active.)

This server supports following variations of `SMTP` protocol:

- "SMTP" at TCP port 25
- "SUBMIT" at TCP port 587. Serves message submission protocol, which is SMTP with mandatory user authentication. (See RFC 2476)

  By default this mode has exactly same IP-ACL datasets, as the main mode, and isn't really at all different from the primary SMTP service, just running at different port. This mode can also have separate ACL dataset, see "PARAM policydb −submit".
- "SMTPS" at TCP port 465. This is deprecated in favour of using "STARTTLS" command in normal SMTP. IP-ACL dataset behaviour is identical with the SMTP port.

- "LMTP" at TCP port 2525 (or whatever). This is merely a debug tool, not a true LMTP server!

## OPTIONS

-4

Explicitely to use IPv4 type of socket even on machines that is capable to do IPv6 type of sockets.

-6

Explicitely to (try to) use IPv6 type of socket even if the machine does not support it. For a default the server will try to use IPv6, if it has been compiled on an environment where it is present, but will do a fallback to IPv4 in case the runtime system does not have IPv6.

Also by default the system will try to create IPv6 socket at first, and then IPv4 socket.

Some operating systems allow same TCP port to be bound at different protocol sockets, some don't. For systems that don't allow, the IPv6 socket will handle also v4-mapped addresses.

-a

turn on RFC931/RFC1413 indentification protocol, and log the information acquired with it to the submitted file.

-w

turn on usage of "whoson" protocol. This option is available even if actual code is not available.

-g

the gullible option will make the program believe any information it is told (such as origin of a connection) without checking.

-h

check helo-parameter, per default that is **not** checked in any way, with this option, syntax check is done on it.

-i

runs the server interactively, which makes it usable for processing a batched SMTP stream (BSMTP) on stdin. With *-v* option this echoes incoming BSMTP to create more accurate faximille of BITNET BSMTP mailers.

-l *SYSLOG*

A magic value of "SYSLOG" for the *logfile* is interpreted by directing all session log stuff thru syslogd; something which **may** require better syslogd, than your system has by default; see "syslog-ng" at a well indexed free software sites.

-l *logfile*

specifies a logfile and enables recording of incoming SMTP conversations. If you want both file based session log, and syslog() based, issue this option after "-l SYSLOG" one.

-n

indicates the program is being run from *inetd*(8).

-p

specifies the TCP port to listen on instead of the default SMTP port, 25.

-B

flags the email to arrive via BSMTP channel (via BITNET, for example).

-I *pidfile*

specifies an alternate PID file location.

-L *maxloadaver*
: tells the maximum load-average the system is under when we still accept email in.

-M *SMTPmaxsize*
: Defines the asolute maximum size we accept from incoming email. (Default: infinite) (This is local policy issue.)

-P *postoffice*
: specifies an alternate **POSTOFFICE** directory.

-R *router*
: specifies an alternate *router (8zm)* program to use for address verification.

-C *cfgfile*
: specifies nonstandard configuration file location; the default is $ *MAIL-SHARE*/smtpserver.conf

-s strict
: this turns on all kinds of **strict** smtp protocol adherence checks, which in normal life can be relaxed slightly. Great for compliance testing ;)

-s sloppy
: This makes smtpserver even more alike *sendmail*™, with all of its sloppy protocol processing.

  At the same time, you are suggested to define following interactive-router controlling paramers to have some defaults in case normal ones defined in HELO-patterns won't be taken into use due to there not being any HELO/EHLO verb in the beginning of the SMTP session:

```
-s ftveRS
```

`-s [ftvehRSD]`
: specifies the style of address verification to be performed. There are four independent commands that can invoke some kind of address verification, and four independent flags to control whether this should be done. They are:

  f check MAIL FROM addresses

  t check RCPT TO addresses

  v check VRFY command argument

  e check EXPN command argument

  R require addresses to be of syntax: local@remote (strict 2821)

  S
  : allow sloppy input for systems incapable to respect RFC 821 properly; WinCE1.0 (and 2.0) does:
    "MAIL FROM:user@domain"
    :−(

  D Don't discard failed data input, but leave them laying into the input spool. This is debug-use flag, don't use in normal mode!

  h Pass 'HELO'/'EHLO' parameter to the interactive router. There just to complement the setup, not really for active use!

  The flags are concatenated to form the argument to the *-s* option. The default is: **ve**.

-T '[1.2.3.4]'

-T '[ipv6.hhhh:hhhh:hhhh:hhhh:hhhh:hhhh:1.2.3.4]'
: Supply (in interactive mode) test address for policy dataset address testing. The option-set recommended for that case is:

-i -d 1 -T '[1.2.3.4]'

A notable detail is that to see what really is going on in the policy analysis,one must usei the "-d 1" option to turn on the debugging early enough to see its initial verdict at the time the "220.." greeting banner is produced..

Also notable is that **brackets** in the supplied IP address **must** be present, otherwise illegal syntax will be reported. (Using RFC 821 address literal parser here.)

-t

Set when running smtpserver under e.g. inetd, and using service port number 465; a "well-known" deprecated one of SSL/SMTP; (From the era before "STARTTLS" protocol verb.)

-Z *zenvfile*

passes on explicite non-compiled-in-default located ZCONFIG environment file.

-V

prints a version message and exits.

## CONFIGURATION

If the $*MAILSHARE/*smtpserver.conf exists it is read to configure two kinds of things:

PARAM −entries

allow server start-time parametrization of several things, including:

- help-texts
- acceptance/rejection database definitions
- various feature parametrizations

On PARAM lines the system allows $-expansions of ZENV variables. (Special note: '$$' expands as '$', not shell-style process-id number!)

The PARAM lines are managable in groups, which essentially are centered around the service specific "Bind*" keywords so that one can set:

- global defaults
- Binding-specific parameters

A new group (with its own BINDs) begins with: `"PARAM newgroup"`. It inherits settings from "default" group, which are all parameters (except bindings) that are defined before first `"PARAM newgroup"`.

Below individual parameters are labeled to be *(global)* or *(group)* depending on what they are.

It is not absolutely necessary to have anything but default group, but it cleans some service parametrization issues. However, *if there are any grouppings, all of them, default including must have bind-sets!*

The style (*-s*) option

behaviour based on glob patterns matching the **HELO**/**EHLO** name given by a remote client. Lines beginning with a # or whitespace are ignored in the file, and all other lines must consist of two tokens: a shell-style (glob) pattern starting at the beginning of the line, whitespace, and a sequence of style flags. The first matching line is used. As a special case, the flags section may start with a *!* character in which case the remainder of the line is a failure comment message to print at the client. This configuration capability is intended as a way to control misbehaving client software or mailers.

PARAM maxsize

*(global)* This is synonym to start-time *-M* option.

PARAM min-availspace 5000

*(global)* This defines, in **kilobytes**, the minimum available space in **POSTOFFICE** directory after the message has been accepted in.

PARAM max-error-recipients

*(group)* This defines how many recipients can be on a message whose source address is **MAIL FROM:**<>. That is, is an error message. (Sometimes SPAMs are tried to inject in that form...)

PARAM MaxSameIpSource

*(global)* This sets the maximum number of active connections from any given single IP address.

When the limit is reached, system tells the remote end:
**450 Too many simultaneous connections...**
(and then closes the connection.)

When the limit is exceeded by factor of four, the server just closes the connection without telling anything.

**Do note that this works only when the smtpserver is running as its own daemon, not while run from under inetd!**

PARAM MaxParallelConnections

*(global)* This limits how many simultaneous connections the server will accept in total -- e.g. how many childs a master server can have running. Default value: 800.

Exceeding the limit by less than 100 will get a message
**450 Too many simultaneous connections...**
printed to the connection. In every case the connection is closed right after the possible message.

**Do note that this works only when the smtpserver is running as its own daemon, not while run from under inetd!**

PARAM ListenQueueSize

*(group)* This sets the listen queue size parameter for *listen*(2) call at the server.

PARAM TcpRcvBufferSize

*(group)* This sets setsockopt(SO_RCVBUF) value, in case the system default is not suitable.

PARAM TcpXmitBufferSize

*(group)* This sets setsockopt(SO_SNDBUF) value, in case the system default is not suitable.

PARAM RcptLimitCount 10000

*(group)* This sets the maximum number of accepted recipients per one message transaction. Default (and minimum!) value is 100, which is mandated by the RFC 821.

PARAM BindPort 25

PARAM BindAddress [0.0.0.0]

PARAM BindAddress any

PARAM BindAddress [IPv6.0::0]

PARAM BindAddress any6

PARAM BindAddress iface:ifacename

*(group) Deprecated old forms of parameters, use of 'BindSmtp' et.al. (below) instead is preferred.*

Per default the server mode SMTP-server binds to port 25 and any locally accepted address, but occasionally people seem to want to have separate server instances with

different configurations, and for those cases are these parameters.

Multiple instances of *BindAddress* will work, and bind all presently supported ports to all those addresses.

PARAM BindSmtp <address_or_iface> <optional_port>

PARAM BindSmtpS <address_or_iface> <optional_port>

PARAM BindSubmit <address_or_iface> <optional_port>
*(group)* Extended versions of abovementioned *Bindaddress* and *BindPort* parameters. These can be used to bind zero to lots of instances of each protocol at different interfaces/addresses/ports, if so desired.

PARAM DEBUGcmd

PARAM EXPNcmd

PARAM VRFYcmd
*(group)* This trio (DEBUGcmd, EXPNcmd, VRFYcmd) are enablers of like named SMTP verbs which have some uses in the debug mode.

They are normally disabled, but running them enabled does not allow direct attacks with them. (That we know of.)

PARAM enable-router
*(global)* This enables interactive router use where user inputs reach the router. As things turn out, while the canned scripts should be safe against any and all inputs, a careless change in the router scripts may endanger this status.

Per default this is **disabled** to protect your system.

To enable EXPN and VRFY, this must be enabled, but be **very carefull** when you do this. This is also required for interactive router processing of "MAIL FROM" and "RCPT TO" addresses.

PARAM enable-router-maxpar 2
*(global)* Sets the upper limit of how many parallel interactive routers can be activated to handle received routing requests.

Default is 2, accepted value range is 1 thru 20.

PARAM smtp-auth
*(group)* This enables 'SMTP AUTH' facility (AUTH verb, plus optional parameter to MAIL verb). With this the users who are able to 'login' successfully to this host, are then able to relay the email thru the server unlimited.

PARAM auth-failrate 20
*(group)* Limit number of failed SMTP AUTHs per hour to this number, above this, and all AUTHs will always fail...

PARAM no-smtp-auth-on-25
*(group)* Disable SMTP AUTH LOGIN on normal SMTP port. It does still work at SMTPS and SUBMISSION.

PARAM smtp-auth-username-prompt
*(group)* A quoted string whose content is sent to remote end in SMTP AUTH LOGIN exchange - in cases where username is not supplied at the beginning of the exchange...

PARAM smtp-auth-password-prompt
*(group)* A quoted string whose content is sent to remote end in SMTP AUTH LOGIN exchange - possibly shown to user during the protocol exchange, possibly not shown...

PARAM AUTH-LOGIN-also-without-TLS
*(group)* This enables 'SMTP AUTH' facility usage also without running under SSL/TLS security envelope.

PARAM smtp-auth-sasl

*(group)* If the system has been configured with *SASL2* support, use that instead of built-in plain-text authenticator code.

**Presently experimental code!**

PARAM sasl-mechanisms 'mech name list'

*(group)* List only those mechanisms that are wanted to be supported in the running system. Undefined list means: any and all what the SASL-system supports.

Suggested list: `"PLAIN LOGIN"`

Other ways might work, or might not. Experience is partial, and failed in several combinations.

PARAM MSA-mode

*(global) (obsolete)* Enable Message Submission Agent mode, where smtpserver **requires** successful user authentication during SMTP sessions initiated from outside of the trusted networks or the networks with relaying enabled (see `"fulltrustnet"` and `"relaycustnet"` at the sample *proto/db/smtp-policy.src* file).

PARAM SMTP-auth-pipe /path/to/program

*(group)* This is a path to the external authentication program. The authenticator should read a username from command line, and a password from standard input. Exit status 0 means successful authentication.

**It is relatively easy to make a mistake in external authentication program that follows the specification. Use this option only if you know exactly what you do! BE CAREFULL!**

PARAM use-tcp-wrapper

*(group)* If TCP-WRAPPER is configured in, uncommenting this will activate its use to look service name: smtp-receiver

PARAM No8BITMIME

PARAM NoCHUNKING

PARAM NoDSN

PARAM NoEHLO

PARAM NoENCHANCEDSTATUS(CODES)

PARAM NoETRN

PARAM NoPIPELINING

*(group)* This set contains **disablers** of like named Extended SMTP EHLO responses, plus EHLO verb itself, e.g. using these will turn off given (for example "PIPELINING") response from the EHLO replies, and then a client possibly capable to feed PIPELINING will not do it − unless it breaks rules, and does it even when the server does not report facility being available.

If you want to disable any of these, you better have a good reason for it, as in general they work quite fine.

Of these, 8BITMIME can not in reality be disabled, only its adverticement can be turned off.

PARAM 8BITMIME-OK

PARAM CHUNKING-OK

PARAM DSN-OK

PARAM EHLO-OK

PARAM ENCHANCEDSTATUS(CODES)-OK

PARAM ETRN-OK

PARAM PIPELINING-OK

*(group)* Enablers for previous parameters in case you have chosen to disable something from the default group, and want to enable it now.

PARAM no-multiline-replies

PARAM multiline-replies-ok

*(group)* Turn off ZMailer's default multiline replies; many systems (especially from M$ breed) don't do RFC 821 Appendix E properly... (Enabler available just in case desired group default is disabled.)

PARAM force-rcpt-notify-never

PARAM no-force-rcpt-notify-never

*(group)* Whatever the incoming DNS NOTIFY= value is, we force it always to be NOTIFY=NEVER. This is for those who won't like to let others even to find out that the message made into the system, but still want to support incoming DSN. (That is, not to disable DSN!)

Disabler available in case group default is forced.

PARAM hdr220 a string of stuff

*(global)* This allows full customization of the initial greeting message. For details, see the sample configuration below.

PARAM contact-pointer-message ....

*(group)* This is a final message for many SMTP rejects. By default, it's value is  You may want to put here, e.g., a pointer to a web page with local policy. See example in the sample configuration below.

PARAM help ....

*(global)* This allows adding locally relevant data into the SMTP protocol HELP command response texts. See example in the sample configuration below.

PARAM policydb DBTYPE /path/to/dbfile

*(group)* These defines smtp input policy filtering/analysis database location for all protocols, and (if wanted) separately for submission. See the comments at the sample *proto/db/smtp-policy.src* file.

An error here will be reported with obscureish code; "1" = "DBTYPE parameter unknown/unsupported", "2" = "can't open the database".

This supports also 'sleepyrpc' database type. For more info, see: **FIXME:FIXME:FIXME**

PARAM contentfilter $MAILBIN/smtp-contentfilter

*(global)* An external program for received message content analysis.

Alternate possibility is to have this to point to a named socket, behind which e.g. *zmscanner* (by Eugene Crosser) is accepting connections.

The interface is desribed below at *CONTENTFILTER INTERFACE* section.

PARAM contentfilter-maxpar 2

*(global)* Max parallel contentfilter instances to be running. Default is 2, and internal hard limit is 20.

PARAM debug-contentfilter

*(global)* Sends some debug-data of the content-filter interface into the smtpserver protocol log file.

PARAM perl-hook path...
: Most important email address and content related events have perl hooks that permit easily customizable policy processing.

PARAM tarpit initial exponent toplimit
: *(group)* This defines a pre-reply slow-down factor, and next delay multiplier (All are floating points, not integrs.)

  Default value for the tarpit is: **0.0**, which makes rest irrelevant.

  The "initial" is used as the initial tarpit delay, and "exponent" is multiplier for formula: *next = prev + (prev * exponent)*

  Finally, "toplimit" caps the delay value.

PARAM use-tls

PARAM tls-CAfile $MAILVAR/db/smtpserver-CAcert.pem

PARAM tls-cert-file $MAILVAR/db/smtpserver-cert.pem

PARAM tls-key-file $MAILVAR/db/smtpserver-key.pem

PARAM tls-dcert-file $MAILVAR/db/smtpserver-dcert.pem

PARAM tls-dkey-file $MAILVAR/db/smtpserver-dkey.pem
: *(group)* These are TLSv1 parameters, and all parts of this parameter cluster must be set for the facility to work!

  See doc/guides/openssl, or: http://www.aet.tu-cottbus.de/personen/jaenicke/pfixtls/doc/setup.html (until something ZMailer specific gets written...)

  Also see below section **OPENSSL RELATED PARAMETERS** .

  The "dcert" and "dkey" are for DSA derived private key, and certificates, and they are deprecated.

PARAM tls-dh512 $MAILVAR/db/smtpserver-cert-dh512.pem

PARAM tls-dh1024 $MAILVAR/db/smtpserver-cert-dh1024.pem
: *(group) (deprecated)* Optional file to supply auxiliary DH parameters for the DSA related key exchange, in case the certificate is so old, that it does not contain them. The system has built-in default values for these.

PARAM listen-ssmtp
: *(group) (deprecated)* Listen on port TCP/465, which is deprecated SSL/SMTP listener port.

PARAM outlook-tls-bug
: *(group)* Microsoft does it again... If TLS is set at Outlook, and server port is not 25, it bloody well seems to expect that the server starts in TLS handshake mode.

  This implements a 2 second startup delay in case the port is some other than 25, and if some byte is received from client during that time, and it happens to be 0x80, then this server will initiate TLS negotiation. If nothing happens (well-behaving client), normal SMTP greeting is presented.

PARAM tls-use-scache

PARAM tls-scache-name 'zzzz'

PARAM tls-scache-timeout 3600
: *(group)* Distributed TLS session cache support; incomplete testing.

PARAM tls-loglevel 0

PARAM tls-ccert-vd 0

PARAM tls-ask-cert 0

PARAM tls-require-cert 0

PARAM tls-CApath ... (somewhen: verify client's certificates)

PARAM tls-enforce-tls 1

*(group)* These are some futher thoughs that may materialize some time..

PARAM rcvd-ident

PARAM rcvd-whoson

PARAM rcvd-auth-user

PARAM rcvd-tls-mode

PARAM rcvd-tls-peer

*(group)* This quintet controls what possibly collected data is shown at the published "Received:" header that this system generates.

PARAM smtpserver-cluster node-name-or-address port-number shared-secret

*(global)* In load-balance clusters a network level load-balancer may distribute the incoming SMTP connections to multiple real machines, and by using this parameter (repeatedly) to list those peers, system can track user behaviour consistently regardless of into which node user connection came at any given time.

PARAM etrn-cluster node-name-or-address mq2-username mq2-passwd

*(global)* In load-balance clusters a network level load-balancer may distribute the incoming SMTP connections to multiple real machines, and by using this parameter (repeatedly) to list those peers, system can relay user initiated ETRN requests to all cluster nodes.

*This method requires that the scheduler runs its mailq service in MAILQv2 mode!*

PARAM lmtp-mode

*(global)* When desiring to test LMTP (RFC 2033), this parameter can be turned on, *however ZMailer is no real LMTP server, and this feature exists only for the smtp-client debug purposes!*

PARAM spf-received

*(group)* Create Received-SPF header if SPF check is done and data is available.
See http://spf.pobox.com/ for explanation of SPF.

PARAM spf-threshold keyword

*(group)* Accept incoming messages with level equal or higher than specified threshold. Levels are sorted as follows:

| | |
|---|---|
| fail: | 1 |
| softfail: | 2 |
| none: | 3 |
| neutral: | 4 |
| pass: | 5 |

See http://spf.pobox.com/ for explanation of SPF.

PARAM spf-localpolicy local_policy

*(group)* Specify local policy for SPF check.
See http://libspf2.org/docs/api.html#SPF_compile_local_policy for local policy explanation.
Note: if your local policy contains space(s) quote it using single (') or double ( ") quote pair.

PARAM spf-whitelist-use-default true |false

*(group)* Use (true) or not use (false) default globally maintained whitelist of known trusted email forwarders.

PARAM report-auth-file $ {MAILSHARE}/scheduler.auth

*(group)* If exists, is used to authenticate a set of interactive report queries that can be done thru the *smtpserver* about its internal state.

This can share the authenticator file with the *scheduler*.

This one expects to find token: `"SMTPIP"` from the file that for all intents and purposes is same what scheduler uses.

Here is a possible configuration file:

```
#
# smtpserver.conf - autogenerated edition
#
#### global parameters ####
#
#PARAM maxsize              10000000    # Same as -M -option
#PARAM min-availspace           5000    # Minimum free in POSTOFFICE after
#                                       # message has arrived; in KILOBYTES.
#PARAM MaxSameIpSource            10    # Max simultaneous connections
#                                       # from any IP source address
#PARAM MaxParallelConnections    800    # Max simultaneous connections
#                                       # in total to the server
#PARAM max-unknown-commands       10    # Max unknown cmds before we hung up
#                                       # because we thing the client is
#                                       # broken/spammer/something
#
PARAM  enable-router    # This is a security decission for you.
#       # This is needed for EXPN/VRFY and interactive
#       # processing of MAIL FROM and RCPT TO addresses.
#       # However it also may allow external user entrance
#       # to ZMailer router shell environment with suitably
#       # pervert input, if quotation rules are broken in
#       # the scripts.  Not having this enabled does mean
#       # that input can freely feed whatever locally non-
#       # existent addresses that are then bouncing via
#       # routing, or via delivery attempts.
#       # Spamming being what it is, any bounce message _you_
#       # don't need to send is all the better...
PARAM  enable-router-maxpar 2
#       # Resource control: Start at most 2 interactive
#       # routers, if such are needed. The value range
#       # is 1 thru 20 ...
#
# HDR220 metatags:
#  %%  =  '%' character
#  %H  =  myhostname
#  %I  =  '+IDENT' if 'identflg' is set
#  %i -- remote IP address in text literal form
#  %V  =  VersionNumb
#  %T  =  curtime string
#  %X  =  xlatelang parameter
#
```

```
#PARAM hdr220 %H ZMailer ESMTP-server %V running at Yoyodyne Propulsion Inc
#PARAM hdr220 %H ESMTP (NO UCE)(NO UBE) our local time is now %T
#
# Note above the "ESMTP" words are present because *some* MTA systems won't
# do EHLO greeting, unless they see "ESMTP" - against RFC 1869 part 4.
# "EHLO is to be done blindly, server responses are not to be studied for
#  any possible 'ESMTP' keyword!"
#
#
#PARAM help ============================================================
#PARAM help  This mail-server is at Yoyodyne Propulsion Inc.
#PARAM help  Our telephone number is: +1-234-567-8900, and
#PARAM help  telefax number is: +1-234-567-8999
#PARAM help  Our business-hours are Mon-Fri: 0800-1700 (Timezone: -0700)
#PARAM help
#PARAM help  Questions regarding our email service should be sent via
#PARAM help  email to address  <postmaster@OURDOMAIN>
#PARAM help  Reports about abuse are to be sent to: <abuse@OURDOMAIN>
#PARAM help ============================================================

# A load-balanced server cluster may want to communicate
# the ETRN request to cluster components, here is how:
# See also:  doc/guides/etrn-cluster
#
#PARAM etrn-cluster localhost              mq-etrn-user mq-etrn-pw
#PARAM etrn-cluster node-2-name-or-address mq-etrn-user mq-etrn-pw
#PARAM etrn-cluster node-3-name-or-address mq-etrn-user mq-etrn-pw
#...
#PARAM etrn-cluster node-40-name-or-address mq-etrn-user mq-etrn-pw

#PARAM lmtp-mode        # When wanting to emulate LMTP server,
#                       # In reality this is only for debugging
#                       # the smtp TA's LMTP mode.

# Uncomment following for not to strip incoming addresses of format:
# <@aa,@bb:cc@dd> into non-source-routed base form: <cc@dd>
#
#PARAM  allowsourceroute
#      # DON'T ENABLE UNLESS YOU USE ROUTER BASED
#      # POLICY ANALYSIS!  (Which doesn't exist anymore!)


##### GROUP PARAMETERS -- DEFAULT SETTINGS #####

#PARAM max-error-recipients        3    # More than this is probably SPAM!
#
#PARAM TcpRcvBufferSize        32000    # Should not need to set!
#PARAM TcpXmitBufferSize       32000    # Should not need to set!
#
#PARAM ListenQueueSize         20000    # listen(2) parameter
#
#PARAM RcptLimitCount          10000    # Max number of recipients for one
#                                       # MAIL FROM session. Minimum: 100
```

```
#
# SPF support options (see http://spf.pobox.com/)
# Zmailer needs to be compiled with libspf_alt
# (see http://www.midwestcs.com/spf/libspf-alt/)
#
#PARAM spf-localpolicy  "ip4:1.2.3.4/24 ?exists:%{ir}.trusted-forwarders.domain.com"
#                       # SPF localpolicy (see libspf2 documentation)
#PARAM spf-whitelist-use-default  false
#                       # use (true) or not use (false) default whitelist
#                       # (see libspf2 documentation)
#
#
#PARAM spf-received     # Create Received-SPF header if SPF check is
#                       # done and data is available.
#
#PARAM spf-threshold  softfail    # worst acceptable SPF check result
#                       # 'fail' - always accept
#                       # 'softfail' - accept if 'softfail',
#                       #   'unpublished', 'neutral', 'pass'
#                       # 'none' - treat 'softfail' as 'fail'
#                       #   accept all better results
#                       # 'neutral' - reject unpublished
#                       # 'pass' - accept only 'pass' and
#                       #   DNS errors.
#      See http://spf.pobox.com/ for more explanation of SPF.
#
# Enablers of some commands:
#PARAM  DEBUGcmd
#PARAM  EXPNcmd
#PARAM  VRFYcmd
#
# Matching disablers (original default state)
#PARAM  no-DEBUGcmd
#PARAM  no-EXPNcmd
#PARAM  no-VRFYcmd
#
#PARAM  smtp-auth-sasl
#       # Authentication with SASL[2] mechanisms
#       # in the system.  Enabling this takes
#       # precedence over smtp-auth below!
#
#PARAM  sasl-mechanisms  "LIST OF SASL MECHANISMS"
#       # A quoted space delimittered list of SASL-
#       # mechanisms we want to support.
#       # Possibly just: "plain" !
#
#PARAM  smtp-auth
#       # Enable, if you want to allow SMTP to autenticate
#       # with the default code against system  /etc/passwd
#       # (or whatever source  getpwnam() uses for it..)
#       # This is intended to be used WITH the TLS network
#       # encryption!  This supports just plaintext logins.
#
#PARAM  auth-failrate 20
```

```
#       # Limit number of failed SMTP AUTHs per hour per IP
#       # to this number, above this, and all AUTHs will
#       # always fail... (Anti-oracle machinery.)
#
#PARAM  no-smtp-auth-on-port-25
#       # No "AUTH LOGIN"/"AUTH=LOGIN" on port 25, which
#       # appears to be troublesome to many users.
#       # This allows authentication in SUBMISSION, though.
#
#PARAM smtp-auth-username-prompt "Username:"
#PARAM smtp-auth-password-prompt "Password:"
#       # Define text prompts for builtin hard-coded
#       # AUTH LOGIN processing.  Most clients don't
#       # show any of these to the users  :-(
#
#PARAM  SMTP-auth-pipe /path/to/program
#       # External authentication program. The
#       # authenticator should read a username from
#       # command line and a password from standard input.
#       # Exit status 0 means successful authentication.
#       # Works only without SASL
#
#PARAM  AUTH-LOGIN-also-without-TLS
#       # Enable, if the "AUTH LOGIN" must be allowed to
#       # be used without running under SSL/TLS encryption
#       # envelope.   Plain-text authentications should
#       # not be used, but you are probably using them
#       # at POP3 and IMAP services, anyway...
#
#PARAM  use-tcp-wrapper
#       # If TCP-WRAPPER is configured in, uncommenting this
#       # will activate its use to look service name: smtp-receiver
#
# Disablers of some facility adverticements/supports:
#
#PARAM  NoEHLO
#PARAM  NoPIPELINING
#PARAM  No8BITMIME
#PARAM  NoCHUNKING
#PARAM  NoDSN
#PARAM  NoETRN
#
# Matching re-enabler, if default-group got disabled:
#PARAM EHLO-OK
#PARAM PIPELINING-OK
#PARAM 8BITMIME-OK
#PARAM CHUNKING-OK
#PARAM DSN-OK
#PARAM ETRN-OK

PARAM   no-multiline-replies # except to EHLO (Bloody M$ RFC821/AppE violators)
#PARAM  multiline-replies-ok # except to EHLO (Bloody M$ RFC821/AppE violators)

#PARAM   force-rcpt-notify-never
```

```
#     # Some want to hide the delivery knowledge.
#PARAM   no-force-rcpt-notify-never
#     # invert previous, default
#
#
# Final message for SMTP rejects.  You can put a pointer to your policy
# web page or such things.
#PARAM contact-pointer-message "Ask HELP for our contact information."
#
#
# The policy database:  (NOTE: See  'makedb'  for its default suffixes!)
#
PARAM  policydb        $DBTYPE  $MAILVAR/db/smtp-policy
#
# External program for received message content analysis:
#PARAM  contentfilter   $MAILBIN/smtp-contentfilter
#PARAM  contentfilter-maxpar 2  # Max parallel content filters; 1 .. 20
#PARAM  debug-contentfilter # Debug the content-filter interface protocol

#PARAM perl-hook  $MAILBIN/smtpserver-perl-hook.pl  # (still experimental)
#
#
#PARAM  tarpit 0 0 0   # No "tarpit" for 4XX/5XX reply codes (default)
#PARAM  tarpit 20 0.1 300  # Initial delay: 20 secs, next = prev + (prev * 0.1)

#
# TLSv1/SSLv[23] parameters; all must be used for the system to work!
#
# See  doc/guides/openssl,  or:
# http://www.aet.tu-cottbus.de/personen/jaenicke/pfixtls/doc/setup.html
#
#PARAM use-tls     # enable to serve STARTTLS
#PARAM no-use-tls  #(default)
#
#PARAM outlook-tls-bug
#     # Fix problems when with Outlook is talking to SUBMIT port
#
#PARAM  tls-CAfile      $MAILVAR/db/smtpserver-CAcert.pem
#PARAM  tls-cert-file   $MAILVAR/db/smtpserver-cert.pem
#PARAM  tls-key-file    $MAILVAR/db/smtpserver-key.pem
#
#  # If system default SSL-session-cache is to be used ?
#PARAM  tls-use-scache
#PARAM  tls-scache-name "zzz"
#PARAM  tls-scache-timeout 3600 # (cache timeout in seconds)
#
#  # Then some futher thoughs that may materialize some time..
#PARAM tls-loglevel     0
#PARAM tls-ccert-vd     0
#PARAM tls-ask-cert     0
#PARAM tls-require-cert 0
##PARAM tls-CApath ... (somewhen: ways to verify client's certificates)
##PARAM tls-enforce-tls 1
```

```
# Elements to be added into "Received:" header's initial comment part:
#
#PARAM rcvd-ident       # The ident lookup result (or even admitting it)
#PARAM rcvd-whoson      # Likewise for "whoson"
#PARAM rcvd-auth-user   # Authenticated Username
#PARAM rcvd-tls-mode    # Cipher, or not
#PARAM rcvd-tls-peer    # Client Certificate reference
#  (plus their   no-rcvd-*  versions)


#
#PARAM report-auth-file ${MAILSHARE}/scheduler.auth
#   Defines where ACL/AUTH data for "Z-REPORT" SMTP-verb
#   is stored.
#

#
#   ### deprecated old forms of binds ###
#PARAM BindPort                   25    # Binding port
#PARAM BindAddress          [0.0.0.0]    # Binding address - for multihomers..
#PARAM BindAddress               any    # Same as [0.0.0.0]
#PARAM BindAddress              any6    # ditto for IPv6
#PARAM BindAddress       [IPv6.0::0]    # and here is for IPv6 - NO SPACES!
#PARAM BindAddress      iface:eth0:2    # Addresses of that interface
#          ###### Number of address binders is not limited!
#
#   ### preferred new forms of binds, addresses as above ###
#PARAM BindSmtp    <addr_or_if>  <optional_port>
#PARAM BindSmtpS   <addr_or_if>  <optional_port>
#PARAM BindSubmit  <addr_or_if>  <optional_port>
#          ###### Number of protocol address binders is not limited!

#PARAM newgroup
#   Begins a new group definition, and is intended to be
#   used along with service specific Binds:
#
# < at first default settings common to everything       >
# < including TLS setups, however without authentication >
# PARAM BindSmtp    any 25
#
# PARAM newgroup  #  <initiate new group>
#         # We have external hardware SSL accelerator
#         # we implement backend service for SMTPS
#         # but STARTTLS is now forbidden.
# PARAM no-use-tls
# PARAM BindSmtp    any 2501   #
# PARAM smtp-auth
#   <and other things needed>
#
# PARAM newgroup  #  <initiate new group>
#         # Define SMTPS service
# PARAM use-tls
# PARAM BindSmtpS    any 465   #
# PARAM smtp-auth
```

```
#   <and other things needed>
#
# PARAM newgroup  #  <initiate new group>
#         # Define SUBMIT service
# PARAM use-tls
# PARAM BindSubmit    any 587   #
# PARAM smtp-auth
#   <and other things needed>
#

#
#
# HELO/EHLO-pattern     style-flags / !reject_message
#               [max loadavg]
#
#  Note about the style-flags: 've' set needs enable-router!
#   'ft' mean nothing anymore, and 'FT' disable use of enabled
#    inline router address analysis (if 'enable-router' is set.)
#    The system will not complain about lack of it (since 2.99.56),
#    but without that enable, those four flags have no effect.
#
#  List of supported style flags:
#    F     DISABLE processing of MAIL FROM thru the interactive router
#    T     DISABLE processing of RCPT TO thru the interactive router
#    v     Process VRFY thru the interactive router
#    e     Process EXPN thru the interactive router
#    R     Require always fully-qualified addresses for MAIL FROM,
#          and for RCPT TO; that is, have @ and domain.
#          ( <postmaster> is the only exception! )
#    S     Allow utter sloppyness in input syntax; mainly extra
#          white-spaces in places where they don't belong are
#          tolerated.  Also lack of HELO/EHLO greeting is allowed!
#    h     Process HELO parameter thru the interactive router
#          Use of this IS NOT RECOMMENED, but is here for the
#          completeness sake...
#    D     Don't Discard; protocol timeouts leave behind files in
#          the $POSTOFFICE/public/ directory with suffixes:
#               .SMTP-TIMEOUT   .DATA-EOF    .BDAT-EOF
#          depending on where the abort happened..
#
#localhost           * ftveR
#some.host.domain    * !NO EMAIL ACCEPTED FROM YOUR MACHINE

# If the host presents itself as:  HELO [1.2.3.4], be lenient to it..
# The syntax below is due to these patterns being SH-GLOB style patterns
# where the brackets are special characters.

#\[*\]                * ve

# Per default demand strict syntactic adherence, including fully
# qualified addresses for  MAIL FROM, and RCPT TO.  To be lenient
# on that detail, remove the "R" from "veR" string below:

*                   * veR
```

## CONTENTFILTER INTERFACE

The *contentfilter* interface has been modified a few times, see your **README.UPGRADING** file for details matching your running setup.

The *contentfilter* program is started without parameters running userid of *daemon* in directory $POSTOFFICE.

The protocol in between the *smtpserver* (8zm), and the content-policy program is a clone from the scheduler to transport-agent one. Namely:

0) server: spawn a sub-process for the policy program

1) policy: `"#hungry\n"`

2) server: `"jobfilepath\n"`

3) policy: `"RESULT DATA\n"`

The loop repeats from 1, and terminates at 2, when the content-filter program reads an EOF.

The *smtpserver* does expect that the *contentfilter* program behaves according to this protocol, and e.g. a system where the filter program runs once, and exists, has in the past proven incompatible with the system.

General rule:

```
-1 550 5.7.1 negatives are condemned into rejection
 0 250 2.6.0 zero is ok! gladly accepted
 1 550 5.7.1 positives are sent into the freezer
```

The program *may* choose to order rejection, and report acceptance, or which way ever:

```
-1
-1 250 2.7.1 Glad to see some spam, immediately destroyed :)
 0
 0 250 2.6.0 Message OK!
 1
 1 550 5.7.1 That is spam, rejected!
```

If the message has no text, some defaults are supplied. If the message text starts with numbers, it is presumed that it contains both the SMTP reply code, and ENHANCEDSTATUSCODE before the text. (If no ENHANCEDSTATUSCODE part is present, then some possibly senseless default is supplied.)

Interface message text lines beginning with anything except signed integer are logged, and the communication channel from the smtpserver to the contentfilter program is closed. Interface continues to scan things reported by the contentfilter program, and if no properly formatted line appears, default is to send the message into the freezer with code `"1"`.

The message text **may** contain CR characters, in which case the code in smtp-server will produce multi-line replies to the message originator.

Keep always in mind, that SMTP protocol (and thus the message texts here) are presumed to be plain 7-bit US-ASCII! However: You may (most of the time) get away with 8-bit chars in the texts...

## PERL-HOOK INTERFACE

System is able to use perl embedding (when compiled with `--with-embed-perl and that can be used to make quite versatile policy processing.`

An example of this is `"zpostgrey"` which does call *Postgrey* server to do *Greyfiltering.*

The interface consists of following callable functions:

```
 ...
package ZSMTP::hook;
 ...
sub set_ipaddress(@) {
}
sub helo(@) {
}
sub set_user(@) {
}
sub rset(@) {
}
sub mailfrom(@) {
}
sub rcptto(@) {
}
sub data {
}
```

Details of the API are not fixed as of this writing. Some details are being altered to better support paradigm of "report as late as possible". (To let e.g. postmaster be posted to, while policy codes otherwise reject postings.)

## TCP-WRAPPER AND SMTPSERVER

If the ZMailer system is configured with *tcp-wrapper* code, and "PARAM use-tcp-wrapper" is active in configuration, then service-id `"smtp-receiver"` is looked for all those addresses that are allowed to feed SMTP email in.

Usually this mode of operation is not used, and files *hosts.allow*, and *hosts.deny* contain following kind of entries:

*/etc/hosts.allow*

```
mailq : ALL@1.2.3.0
smtp-receiver: ALL@ALL
```

*/etc/hosts.deny*

```
ALL : ALL@ALL
```

Alternatively, all the functions which tcp-wrapper could supply are also available thru the policy database machinery.

*(Do note that scheduler(8zm) has also tcp-wrapper support, which becomes active simultaneously with smtpserver's tcp-wrapper code!)*

## SASL[2] AUTHENTICATION MECHANISMS

The *smtpserver* does contain **experimental** code supporting authentication interaction using SASL mechanisms as they are implemented in *CMU Cyrus-SASL-2* library.

This will also necessitate adding SASL-2 library related configuration telling what backside systems are to be used. Possible configuration file is:

```
# ---- /usr/lib/sasl2/smtpserver.conf ----
pwcheck_method: saslauthd
```

File paths, etc. for this are highly dependent of said library setup.

## TLS MODE

**TO BE WRITTEN;** Some further notes on how to setup the TLS encryption on the *smtpserver*(8zm). For the time being, see notes at "doc/guides/openssl" as well as what is written above at the "PARAM use-tls", or there abouts.

## SMTP-AUTH – AUTHENTICATE (RELAY) USER

The idea with "smtp-auth" is to authenticate the user who wants extra privileges from the SMTP service, namely if a user wants to send email to an address considered non-local at the system, this user needs special privilege.

In usual cases the privilege is granted based on IP address of the user (See "smtp-policy.relay*" below in SMTP-Policy Configuration).

The privilege can also be granted by doing a "login" procedure where the user gives some personal identifier, and related secret.

Becase the normal authenticator is a plain-text password, this operation should be done under the security envelope of the SSL. (Or equivalent under IPSEC encryption, although ZMailer's smtpserver does not know how to detect the session being IPSEC protected!)

You will likely need:

- "PARAM smtp-auth"
- The TLS related notes above.

Possibly also:

- "PARAM AUTH-LOGIN-also-without-TLS "
- "PARAM SMTP-auth-pipe /path/to/program "
- And possibly also PAM-support for SMTP-Auth, see below.

## PAM-SUPPORT FOR SMTP-AUTH

If the system has <security/pam_appl.h> file, and admin has chosen to compile system using PAM support. For *Linux* the following file would also be needed:

------- /etc/pam.d/smtpauth-login -----------

```
#%PAM-1.0
auth     required  pam_pwdb.so shadow
auth     required  pam_nologin.so
account  required  pam_pwdb.so
```

---

An alternate configuration for Linux is:

------- /etc/pam.d/smtpauth-login -----------

```
#%PAM-1.0
auth       required     pam_stack.so service=system-auth
account    required     pam_stack.so service=system-auth
```

---

For *Solaris* the setup needs editing */etc/pam.conf* file:

------- /etc/pam.conf ---------

```
 ....
smtpauth-login  auth required   pam_unix_auth.so.1
smtpauth-login  auth required   pam_authtok_get.so.1
 ....
```

---

## SMTP-POLICY CONFIGURATION

This subsystem of *smtpserver*(8zm) does control acceptability of recipient envelope addresses per several criteria:

- "contactee IP address"
- SMTP "MAIL FROM:<..>" address

- SMTP "RCPT TO:<..>" addresses
- "various control files"

Specifically the $*MAILBIN/policy-builder.sh* script uses following files:

| | |
|---|---|
| $MAILVAR/db/smtp-policy.src | The boilerplate |
| $MAILVAR/db/localnames | ('= _local_names') |
| $MAILVAR/db/smtp-policy.relay.manual | ('= _full_rights') |
| $MAILVAR/db/smtp-policy.relay | ('= _full_rights') |
| $MAILVAR/db/smtp-policy.mx.manual | ('= _relaytarget') |
| $MAILVAR/db/smtp-policy.mx | ('= _relaytarget') |
| $MAILVAR/db/smtp-policy.spam.manual | ('= _bulk_mail') |
| $MAILVAR/db/smtp-policy.spam | ('= _bulk_mail') |

**You should review the** *smtp-policy.src* **file for things that apply to you, and then execute** *policy-builder.sh* **script.**

You may roll your own *policy-builder.sh* script, *but you must be extremely carefull* to supply sufficient set of parameters and defaults on generated final *smtp-policy.dat* file, that will then be compiled into the binary policy database to be used by the *smtpserver*(8zm) program *If you don't, the policy database will most likely fail to restrict access, and you end up having an open relay!*

Basically these various source files (when existing) are used to combine knowledge of valid users around us. Some datasets have *two* input source files, *smtp-policy.NN* and *smtp-policy.NN.manual*, the ".manual" is intended to be overrider of of possibly autogenerated data at the "plain" version of files.

localnames
: Who we are -- ok for receiving; does not grant outgoing relay capability.

smtp-policy.relay

smtp-policy.relay.manual
: Who can use us as outbound relay.

  Use here

  [ip.number]/maskwidth

  for listing those senders (networks) we absolutely trust. Additionally you may give (at the same line) some attributes as parameters for this key entry:

```
fulltrustnet +
trustrecipient +
maxinsize 30000000
filtering +/-
ratelimitmsgs 300
ratelimitmsgs 300,6000
```

  The 1st pair will accept any source address, and any recipient addresses that are fed to the server.

  The 2nd will verify the source IP-address, but after that it will accept any recipient addresses.

  The 3rd pair sets EHLO-reported "SIZE nn" parameter to be limited to 30 million bytes. (Or to global definition, if that is in range of 1 thru 'nn'.)

  The 4th pair sets explicite content-filtering activation/disabling ('+'/'-') for IP address masks given in these files. See the comments in the "smtp-policy.src" about this! It might make sense to place this 4th pair into boilerplate "_full_rights" definition. This functions without regard to "fulltrustnet +" overriding all other tests e.g. those of SMTP protocol MAIL FROM and RCPT TO addresses.

The 5th pair sets limit on how many messages are accepted in an 1.0-1.15 hour period. Excess messages (MAIL FROMs actually) are treated with cold shoulder ( "450" code). If the numeric parameter value is NEGATIVE, then the rejection is instant and permanent with "550" code. It might make sense to place this 5th pair into boilerplate "_full_rights" definition.

The 6th pair with comma-separated integer parameter sets limit on how many messages, and how many recipients in total are accepted in an 1.0-1.15 hour period. Excess messages, or recipients, are treated with a cold shoulder ( "450" code). If the numeric parameter value is NEGATIVE, then the rejection is instant and permanent with "550" code. It might make sense to place this 6th pair into boilerplate "_full_rights" definition.

You may also enter domains which are looked up for the hostname of reversed IP address, but it is not very wise; IP-reversal is not trustworthy data. It may also cause double-entry/level descendance problems when two domain-suffixes have same ending suffix (or are the same).. (Name/keyspace problem)

We can set the internal "always_accept" flag at the source IP test, and never after.

smtp-policy.mx

smtp-policy.mx.manual
Domains, who really are our MX clients.

Use this when you really know them, and don't want just to trust that if recipient has MX to you, it would be ok... You can substitute this knowledge with a fuzzy feeling by using 'acceptifmx +' attribute at the generic boilerplate.

List here *domain names.*

You **can** also list here all **postmaster** addresses you accept email routed to:

postmaster@local.domain postmaster@client.domain

these are magic addresses that email is accepted to, even when everything else is blocked. (well, not exactly, if MAIL is rejected, or connection at all...)

smtp-policy.spam

smtp-policy.spam.manual
Those users & domains that are absolutely no-no for the senders, or the recipients, no matter what earlier analysis has shown. (Except for those that we absolutely trust..)

Short usage instructions:

- Fill in/modify related files
- Execute *MAILBIN/policy-builder.sh* script

## SMTP-POLICY TESTING

You can run the smtpserver in a mode where you can claim to be from any address in the outside world you wish:

$MAILBIN/smtpserver -i -d 1 -T '[1.2.3.4]'

The mode must be interactive (-i), and supplying debug mode (-d 1) to it is good help.

Actual claimed connection source address is to be given inside square brackets as a SMTP IP address literal.

Now you can try things like:

```
220 ...
EHLO foo
...
```

```
MAIL FROM:<>
...
RCPT TO:<address@local.domain>
...
RCPT TO:<address@elsewere.domain>
...
```

(Substitute some real domains into those RCPT TO lines -- "local.domain" is a hint about what to pick for it..)

Depending what kind of address you have supplied to the -T parameter, they get either accepted, or rejected.

## SMTP-POLICY RBL-TYPE BLOCKING LISTS

**There is a generic problem in these RBL things: They tend to grow stale, go to defunct, and sometimes become poisoned. You have been warned!**

Per default the system **does not** use RBL-type blocking lists. There are two ways how to take them into use:

1. You can start rejecting at the connection setup and then at MAIL FROM (and RCPT TO).

   However many (especially M$ environment) SMTP clients won't react on that properly, and will just keep repeating the delivery attempts.

2. You can delay the rejections until RCPT addresses are given.

## SMTP-POLICY; IMMEDIATE REJECTION BY RBL

Like mentioned above, this method has a problem with many clients who don't believe that HELO can give 500-series response.

Method is as follows:

Pick your choice of databases to the second variant " `_rbl0`" label by joining your selection from various things exemplified here below by using " `:`" character as glue in between:

"`+`" alias "`rbl.maps.vix.com`"
"`relays.mail-abuse.org`"
"`dul.maps.vix.com`"
"`relays.orbs.org`"

An example for the resulting attribute pair: (RBL+DUL+RSS)

```
#| Second RBL variant: Early block with RBL+DUL+RSS
_rbl0   test-dns-rbl  dul.maps.vix.com:relays.mail-abuse.org
_rbl1   # Nothing
```

## SMTP-POLICY; DELAYED REJECTION BY RBL

Delay the rejection report to "`RCPT TO`" verbs by using the "Third RBL variant":

```
#| Third RBL variant: Late block with DUL+RSS
_rbl0   rcpt-dns-rbl      dul.maps.vix.com:relays.mail-abuse.org
_rbl1   test-rcpt-dns-rbl +
```

The sample boilerplace will use these as defaults unless you choose to explicitely have " `test-rcpt-dns-rbl -`" at some of the recipient domains you list at *smtp-policy.mx* file:

```
#sample.domain.with.rbl
sample.domain.no.rbl    test-rcpt-dns-rbl -
```

## OPENSSL RELATED PARAMETERS

Because the TLS related code is fairly straight copy from *Postfix* specific one, the document is fairly direct copy, too..

*To use TLS we do need a certificate and a private key. Both must be in be encrypted, that does mean: it must be accessable without password. Both parts (certificate and private key) may be in the same file.*

PARAM tls-cert-file /etc/postfix/server.pem
PARAM tls-key-file /etc/postfix/server.pem

*The certificate was issued by a certification authority (CA), of which the CA-cert must be available. This file may also contain the the CA certificates of other trusted CAs. You must use this file for the list of trusted CAs if you want to use chroot-mode.*

PARAM tls-CAfile /etc/postfix/CAcert.pem

*To verify the peer certificate, we need to know the certificates of certification authorities. These certificates in The same CAs are offered to clients for client verification. Don't forget to create the necessary $OPENSSL_HOME/bin/c_rehash /etc/postfix/certs. A typical place for the CA-certs may also be $OPENSSL_HOME/certs, so there is no default and you explicitly have to set the value here!*

PARAM tls-CApath /etc/postfix/certs

*To get additional information during the TLS setup and negotiations you can increase the loglevel from 0..4:*

0: No output about the TLS subsystem

1: Printout startup and certificate information

2: 1 + Printout of levels during negotiation

3: 2 + Hex and ASCII dump of negotiation process

4: 3 + Hex and ASCII dump of complete transmission after STARTTLS

*Use loglevel 3 only in case of problems. Use of loglevel 4 is strongly discouraged.*

PARAM tls-loglevel 0

*By default TLS is disabled, so no difference to plain ZMailer is visible. Explicitely switch it on here:*

PARAM use-tls

*If the operating system isn't equipped with /dev/*random devices, OpenSSL's RAND_bytes*(3) *function will use compilation time defaults to locate EGD compatible entropy source. See dokumentation at: RAND_egd*(3) *man-page. If you have something similar in your system, but in non-default location (see the man-page above), you can set its location with this:*

PARAM tls-random-source /var/run/prngd-socket

## IMPLEMENTED SMTP VERBS AND EXTENSIONS

### TO BE COMPLETED

HELO

EHLO
See RFC 2821.

MAIL
See RFC 2821.

RCPT
See RFC 2821.

DATA
: See RFC 2821.

BDAT
: See RFC 3030.

QUIT
: See RFC 2821.

EXPN
: See RFC 2821.

VRFY
: See RFC 2821.

HELP
: See RFC 2821.

RSET
: See RFC 2821.

ETRN
: See RFC 1985.

AUTH
: Mainly "AUTH LOGIN", see RFC 2554+Netscape... (= plaintext login)

STARTTLS
: If enabled and configured, starts TLS enryption. See RFC 3207.

Z-REPORT
: Special interactive authenticated and ACL controlled command for querying *smtpserver*'s internal resource trackers for given IP address (or some other things.)

  If access-control (can use/share `scheduler.auth` file mechanisms with "SMTPIP" token) allows access to this command, the report will be something like:

```
Z-REPORT <username> <password> IP 11.22.33.44
200-Slot-ages:  117 630 1146 1658 2175 2691 3203 3720
200 Slot-IPv4-data: 0 9 2 3 9 0 3 7 MAILs: 99 SLOTAGE: 90 Limit: 99 Excesses: 0 0 0 I
```

  where the first reply line ("Slot-ages:") is telling counting bin ages (in seconds) in same order as the counts are given in the second line ( "Slot-IPv4-data:")

  Additional items in the second line are:

  - "MAILs:" the count of "MAIL FROM" lines in total during the lifetime of this counting entry. *NOTE: The lifetime may be a lot longer, than just the 8 counting bins!* The counting entry disappears completely only when all 8 bins are zero!
  - "SLOTAGE:" the *age* of this tracking entry in seconds.
  - "Limit:" last "ratelimitmsgs" limit parameter value used when the policy processing did query the rate data.

  The **DUMP** variant is primarily for debug purposes:

```
Z-REPORT <username> <password> DUMP
200-DUMP BEGINS
200- ... dump lines ...
200 DUMP ENDS
```

  If access-control fails, command issuer receives "550 .. unknown command " reply.

Z-IDENT
: Tool to see, what "ident" query results for the received SMTP session.

Z-DEBUG
**If enabled!**

Turns on "debug" flag within the session, and produces lots of murky output into the datastream. May, or may not break SMTP protocol. May, or may not work in tandem with TLS-cipher mode. For manual debuggers ONLY.

Security note: Will not let users to enter anything that they would not otherwise be able to enter, although will allow users to see things that normally are not shown.

TURN
RFC 821, not supported due to security problems in it.

RFC 1985: ETRN supercedes this.

ONEX, SEND, SOML, TICK, EMAL, ESND, ESOM, ESAM, EVFY, VERB
Various proprietary *sendmail* things, and or other semi-well-known extensions.

## ENVIRONMENT VARIABLES USED BY THE SMTPSERVER

The *zmailer*(3zm) library's *mail_open()* et.al. functions use several environment variables. See that man-page.

## Z-ENV VARIABLES USED BY THE SMTPSERVER

A collection of system configuration things

ALLOWSOURCEROUTE
Alternate way to tell the system the same thing as config does:

```
PARAM allowsourceroute
```

**Preferrably this is not to be used! Security dangers aplenty!**

BINDADDR
If exists (with valid content), **BINDADDR** specifies to which local interface to bind *smtpserver*, *smtp* transport agent, and *scheduler*. Possible specification formats are:

[0.0.0.0]
[IPv6.0::0]
iface:eth0:1

The *iface:* syntax is not really recommended.

INPUTDIRS
See *zmailer*(3zm).

INPUTNOTIFY
See *zmailer*(3zm).

MAILBIN
Used to find related *router*(8zm) program for interactive routing things.

PATH Passed on authentication subprocess program, see: *PARAM SMTP-auth-pipe ...*

POSTOFFICE
See also *zmailer*(3zm).

ROUTEDIRHASH
See *zmailer*(3zm).

ROUTERNOTIFY
See *zmailer*(3zm).

SYSLOGFLG

ZCONFIG
Passed on authentication subprocess program, see: *PARAM SMTP-auth-pipe ...*

## FILES

*/opt/mail/zmailer.conf*
system global parameters

*/var/spool/postoffice/.pid.smtpserver*
$POSTOFFICE/.pid.smtpserver

*/opt/mail/smtpserver.conf*
$MAILSHARE/smtpserver.conf

*/etc/pam.d/smtpauth-login*
if PAM mechanism is present and plain-password authentication is wanted

*/etc/hosts.allow*, */etc/hosts.deny*
Files for the *tcp-wrapper*, if compiled and configured into use.

${MAILSHARE}/scheduler.auth
Possible ACL control file for the Z-REPORT SMTP verb.

## SEE ALSO

*router*(8zm), *zmailer.conf*(5zm).

| | |
|---|---|
| RFC 821/2821 | The basic SMTP specification |
| RFC 1123 | Various 821 parameter clarifications |

Several extended SMTP facilities are implemented:

| | |
|---|---|
| RFC 1341/1521/2045 | MIME specification (body, formats) |
| RFC 1342/1522/2047 | MIME specification (headers) |
| RFC 1425/1651/1869 | ESMTP EHLO framework |
| RFC 1426/1652 | ESMTP 8BITMIME |
| RFC 1427/1653/1870 | ESMTP SIZE |
| RFC 1428 | Basic MIME conversion rules |
| RFC 1830/3030 | ESMTP CHUNKING |
| RFC 1854/2197/2920 | ESMTP PIPELINING |
| RFC 1891/3461 | ESMTP DSN |
| RFC 1893/3463 | Enhanced Mail System Status Codes |
| RFC 1985 | ESMTP ETRN |
| RFC 2033 | LMTP server mode (for testing) |
| RFC 2034 | ESMTP ENHANCEDSTATUSCODES |
| RFC 2476 | SUBMIT protocol (port 587) |
| RFC 2487/3207 | ESMTP STARTTLS |
| RFC 2222 | SASL mechanism base definition |
| RFC 2554+M$ Exchange | ESMTP AUTH LOGIN |
| RFC 2554+NetScape | ESMTP AUTH=LOGIN |
| RFC 2852 | ESMTP DELIVERBY (incomplete implementation) |

## AUTHOR

This program authored and copyright by:
Rayan Zachariassen <no address>
Extended SMTP, policy facilities, etc. by
Matti Aarnio <mea@nic.funet.fi>