



VirusBuster
for Mail Servers
1.2

TABLE OF CONTENTS

VIRUSBUSTER FOR MAIL SERVERS	2
System requirements	4
General information	6
Package naming.....	6
Installation.....	6
Assign to mail sever.....	8
Uninstallation	9
Binary files	9
Statistics screen explanation	10
Registration.....	11
Operation.....	13
First-level modules /filters/ (Level1)	13
Second-level modules /filters/ (Level2).....	13
Database update.....	14
The configuration file	15
Structure of the configuration file	15
Filter definitions.....	16
Action specification.....	18
VBMSRV daemon configuration file (vbmsrv.conf).....	20
Module commands	47
VBMLOG daemon configuration file (vbmlog.conf)	49
Tokens.....	51
END USER AGREEMENT	53
CONTACT	54

VIRUSBUSTER FOR MAIL SERVERS

The VirusBuster for Mail Servers package (hereinafter called VBMSRV protection) provides Sendmail, Qmail, Courier and GroupWise /this one only for Linux/ mail servers with virus and spam protection. The solution integrating to the mail server scans all the mails to be delivered and their attachments and ensures comprehensive protection against viruses, malicious codes and unsolicited mails.

Our product integrates the Commtouch's pre-emptive virus (Zero Hour Virus Protection /ZH filter/ and spam (Extended Spam Protection /ESP filter/) protection module based on the innovative RPD technology, which - as a signature-less solution - provides enhanced detection on the supported platforms.

Important!

The above mentioned ZH and ESP modules are not available in the standard package. Please contact our Sales department at the sales@virusbuster.hu e-mail address for more information.

Main features:

- Filter modules
- * Virus filter: effective virus recognition based on the outstanding virus scan engine
- * File filter: actions could be assigned to specified file formats
- * Spam filter: statistical spam filtering with many evaluation methods
- * Language/script filtering based on the e-mail's content ('language_filter' option of the spam filter)
- * Address filter - It is possible to set Black/White lists
- * Result filter - summarized result, all results of other filters could be used in the result filter
- * RBL filter
- Flexible rule system: ability to use parameters for the actions to be performed
- Heuristic virus analysis to recognize unknown viruses
- Advanced WormBuster function - for blocking I-Worms instantly
- Filtering encrypted archives
- Comprehensive statistical information about mail traffic and events
- Automatic, incremental virus database update
- Log daemon - logging to several output types: file, syslog, standard output

Spam filter:

The anti-virus system's spam filter operates based on statistical scan methods and has numerous leading evaluation techniques to provide effective protection against unsolicited mails.

- Recognizing based on statistical scan methods completed with other evaluation techniques
- Heuristics filtering
- HTML filtering
- UNICODE text handling
- Low false positives and high spam recognition rate
- Filter sensitivity: 3 level of spam filtering
- Frequently updated spam database

Pre-emptive virus and spam protection:

ZH and ESP modules based on the RPD technology don't need a virus or spam database to detect malware delivered by e-mail, but they detect the attacking/spreading wave itself connecting and communicating permanently to a central server. The server analyzes e-mail traffic of the Internet, based on comprehensive information collected from numerous locations of the world. The filter ranks the mails according to the server information so it can reveal the attacks or spam mails some minutes after they have been started and block these e-mails long before the first virus or spam database updates are released which can take several hours sometimes.

- It is effective in the early phase of attacks: protects in a few minutes after the attack has started. Releases of virus or spam database updates for traditional virus/spam scan engines can take several hours.
- Pre-emptive defense: blocks known and unknown malware, spams by detecting attack waves.
- Outstanding detection rate: detects 95 percent of spam mails or e-mails that contains malware in itself, without using any other "traditional" virus or spam protection methods.
- Fully automatic: no maintenance needed

System requirements

Supported operating systems

Linux (i386/amd64)
FreeBSD 5.5, 6.0, 7.0 (i386/amd64)
Solaris 9 (sparc), 10 (i386)

Minimal requirements

Requirements for all the supported platforms:

- 256 MB free memory (512 MB recommended)
- 100 MB free hard disk space
- wget (for update)
- perl5 (for update)

Requirements by platforms:

Linux, FreeBSD:

- Intel Pentium (or compatible) processor at 300 MHz
- Minimal required for Linux: GLIBC 2.2.5, kernel 2.2.1
- Minimal required Linux distributions: SuSE 8.0, RedHat 7.3, Debian 3.0 (woody), Mandrake 9.0, Slackware 8.1

Solaris:

- Ultra Sparc IIe processor at 500 MHz

Minimum required mail server versions

Sendmail: 8.12
Qmail: 1.03
GroupWise: 7

Additional system components needed for the ZH and ESP filters

ZH and ESP filter functions are available on Linux, FreeBSD (all supported versions) and Solaris systems. The following additional system requirements have to be installed on your computer to utilize their features:

Linux:

- Standard C Library 2.3 (glibc 2.3)
- C++ Runtime Library 3.2 (libstdc++.so.5)

FreeBSD:

- Standard C Library (libc.so.4)
- C++ Runtime Library (libstdc++.so.5)

Solaris:

- SMCgcc or SMClibgcc package (download from sunfreeware.com)

These packages copies the needed libs into the /usr/local/lib directory. This path is not registered as the default search path of the dynamic linker (ld.so.1), so this must be registered into the LD_LIBRARY_PATH environmental variable before starting the product.

General information

Package naming

The VirusBuster for Mail Servers package is named according to the following parameters:

```
vbmsrv-<version>-<os>-<architecture>-<minimal libc version>.tgz
```

<version>: The package's version number. For example: 1.0.1

<os>: The package is working on the displayed system. For example: Linux

<architecture>: The processor type. For example: i386

<minimal libc version>: The required minimal version of the libc library. For example: libc6

Installation

You can start the installation by executing the 'vbmsrv-install.pl' program. After executing the following questions should be answered for the successful installation.

Value displayed between square brackets is default answers for the current question, you can simply use the <enter> button to accept it. It is recommended to use these default values.

One of the first steps is to specify the mail server you want to be protected!

```
#Please select the mail server you want to be protected!
```

```
 #(S)endmail, (Q)mail, (C)ourier or (G)roupWise
```

```
#[s]
```

Set the 'run-as-group' option's value. Please see the configuration specification for more information!

```
#In which group do you want to run VirusBuster for Mail Servers?
```

```
#[vbuster]
```

Set the 'run-as-user' option's value. Please see the configuration specification for more information!

```
#With which user permission do you want to run VirusBuster for Mail Servers?
```

```
#[vbuster]
```

You should specify the location of the binary file in the system:

```
#In which directory do you want to install the binary files?
```

```
#[/usr/sbin]
```

Define the path of the library files needed for the program:

```
#In which directory do you want to install the library files?
```

```
#[/usr/lib]
```

Target path of the database files:

```
#In which directory do you want to install the database (virus, spam) files?
```

```
#[/var/lib/vbuster]
```

Specify the location of the text type documentation files:

```
#In which directory do you want to install the documentation files?
```

```
#[/usr/share/doc]
```

Location for the documentation in man page format:

```
#In which directory do you want to install the manual files?
```

```
#[/usr/share/man]
```

The program creates communication and other files needed during its operation:

```
#Which directory do you want to be the run directory?
#[/var/run/vbuster]
```

```
Name the directory of the log file:
Specify the log directory name
[/var/log/vbuster]
```

```
Set the directory storing the initialization scripts:
#What is the directory that contains the init scripts?
#[/etc/init.d]
```

```
Define path for the initialization directories:
#What is the directory that contains the init directories (rc0.d -
#rc6.d)?
#[/etc]
```

If the program detects that a previous version of its configuration file is available in the system, it will offer the following selection:

```
#Found an existing config file (/etc/vbuster/vbmsrv.conf).
#(K)eep the existing file or (C)reate a new one?
#[k]
```

```
Specify name for the log file:
#Specify the log file name
#[vbmsrv.log]
```

```
Mails sent from the IP address you enter will be filtered:
#Which IP address do you want to filter?
#You should use the standard address/length format (example:
#194.222.242.0/24)
#[0.0.0.0/0]
```

```
Please enter your user name:
#Enter your registration user name
#[ ]
```

```
Please enter your registration key:
#Enter your registration key (example: WESAE-WCRVC-CSNEH)
#[ ]
```

If you are about to install the GroupWise protection module, you have to answer additional questions:

```
#Please specify the GroupWise's SMTP Service Queues directory!
```

```
#Please specify the GroupWise's SMTP Queues directory!
```

Consult the Groupwise module settings chapter to get more information about these settings.

The following lines are shown in case of successful installation:

```
Installing files... Done.
Installing config file...
Installing init scripts... Done.
```

Assign to mail sever

To activate anti-virus system you need to perform the following steps beside the configuration settings:

Using Sendmail

The VBMSRV protection must be assigned to the Sendmail so that the mail server and the filter program can communicate to each other. You have to edit the Sendmail's configuration macro file then rebuild it to get the new configuration file.

Please insert one of the following versions into the sendmail.mc file (the name of the Sendmail's macro file may be different on different systems)!

Version A:

This entry consists of 2 lines!

First:

```
MAIL_FILTER(`vbmsrv', `S=unix:/var/run/vbuster/vbmsrv,F=T, T=S:4m;R:4m')dnl
```

Second:

```
define(`confINPUT_MAIL_FILTERS', `vbmsrv')dnl
```

Version B:

This entry consists of 1 line:

```
INPUT_MAIL_FILTER(`vbmsrv', `S=unix:/var/run/vbuster/vbmsrv,F=T,T=S:4m;R:4m')dnl
```

Please take care of the exact copy!

Using Qmail

1. Rename the original "qmail-queue" to "qmail-queue2" (the "original_qmail_queue" option found in the configuration file must have the same value)
2. Copy the "qmail-queue" found in the package's "qmail" directory to the Qmail's binary directory (default path: /var/qmail/bin)
3. Reset the owner of the "qmail-queue" which had been copied in the previous step to qmailq and its group to qmail with the following commands:
chown qmailq /var/qmail/bin/qmail-queue
chgrp qmail /var/qmail/bin/qmail-queue

Using Courier

1. Rename the original "submit" to "submit2" (the "original_courier_submit" option found in the configuration file must have the same value)
2. Copy the "submit" found in the package's "qcourier" directory to the Courier's binary directory (default path: /usr/lib/courier/courier)

Qmail and Courier interface module (VBRAW client) operation

The Qmail and Courier applications consist of modules connected with each other. VirusBuster integrates its own module between the source module (e.g. SMTP daemon) and the 'submit' (Courier) or the 'qmail-queue' (Qmail) module as a transparent

proxy. Then it will forward the processed/modified mails to the original 'submit' or 'qmail-queue'.

The integrated VBMSRV module will read the access information from the VBMSRV's configuration file ([General]/address option) so as to be able to connect to the main VBMSRV program. It tries to get the path of the configuration file from the VBMSRV_CONFIG environment variable first time, if it fails then from the default path (/etc/vbuster/vbmsrv.conf).

Using GroupWise

If the startup script (vbmsrvctl) detects the GroupWise protection module (vbgwia) in the binary folder of the product (/usr/sbin by default), the module is ready to use, there is no need to do anything else if the module settings are correct in the configuration file.

Uninstallation

Please run the following program file to uninstall the package:
vbmsrv-uninstall.pl

Binary files

The following executable files and their parameters are found in the package. These files are placed in the /usr/sbin directory by default:

vbmsrv [options]

MailScan main daemon program.

Options:

-n, --nodaemon execute in no daemon mode
-v, --version displays the version of vbmsrv and exits
-c, --config=FILE reads configuration from FILE (path needed)
-l --license returns registration data

vbmlog [options]

Log that responsible for controlling log messages.

Options:

-n, --nodaemon execute in no daemon mode
-v, --version displays the version of vbmlog and exits
-c, --config=FILE reads configuration from FILE (path needed)

vbmsrvctl start|stop|restart|cfgreload|dbreload|logrotate|statistic

Control file, you can realize the following functions by using the available parameters:

start: Starts the vbmlog and vbmsrv files.

stop: Stops the vbmlog and vbmsrv files.

restart: Stops and starts the vbmlog and vbmsrv files.

cfgreload: Reloads the [Milter] section's settings of the vbmsrv configuration file and vbmlog's configuration file and applies the new settings.

dbreload: Reloads the virus and spam database.

logrotate: Locks the current log file then opens a new one. This function is useful for archiver programs.

statistic: Displays the statistics and exits.

vbmstat [options]

Statistics screen about anti-virus system's operation.

Options:

-d, --debug RAW output
-v, --version Display the version number.

-q, --quit Run statistics module only once, then quit.
-a ADDRESS, --address=ADDRESS Statserver's address (e.g. ip:host:port or unix:path).
-p N, --processnumber=N **If multi-process operation enabled, specify the process number in the N parameter. In such a case, the module displays global statistics about the programs run in different processes.**

vbasapd

Establishes the connection to the Commtouch's server, provides ZH and ESP functions. Start and stop parameters of the vbmsrvctl file also affect this daemon.

Statistics screen explanation

Date displayed on the right upper corner: system date of the computer running the VBMSRV.

Started at:
Startup date of the VBMSRV.

Uptime is:
Time has elapsed since the last startup.

Program version:
VBMSRV's program version.

Virus database version:
Version of the used virus database.

Virus scan engine version:
Version of the virus scan engine.

Spam database version:
Version of the used spam database.

Spam scan engine version:
Version of the used spam scan engine.

System load average:
System load index number.

Current connections:
Number of connections being currently processed.
(If Sendmail is used: number of clients connected to the server.
If Qmail, Courier or GroupWise is used: number of mails being currently processed.)

Total connections:
If Sendmail is used: total number of connections.
If Qmail, Courier or GroupWise is used: total number of received mails.

Processed mail(s):
If Sendmail is used: total number of processed mails.
If Qmail, Courier or GroupWise is used: the same value as Total connections'.

Blocked mail(s):
Number of refused (dropped or rejected) mails.

Processing error:
Number of errors occurred during the process.

Scanned file for viruses:
Number of attachments scanned by the virus filter.

Virus found:
Number of found viruses.

I-Worms:
Number of found i-worms.

Virus killed:
Number of killed viruses.

Modified attachment(s):
Number of modified attachments.

Deleted attachment(s):
Number of deleted attachments.

Scanned mails for spam:
Number of mails checked by the spam filter.

Spam found:
Number of mails marked as spam by the spam filter.

Additional information displayed when ESP/ZH filter is used

ASAP daemon version:
Version of the vbasapd daemon that requires for the ZH/ESP filters.

ASAP daemon comm. error:
Number of occurred communication errors between the ZH/ESP filter and the vbasapd daemon.

ZH communication error:
Number of occurred communication errors between the Commtouch server and the vbasapd daemon when ZH filter is used.

ESP communication error:
Number of occurred communication errors between the Commtouch server and the vbasapd daemon when ESP filter is used.

ZH virus found:
Number of infected mails found by the ZH filter.

Total virus incident:
Total number of virus incidents found by the ZH filer and the virus filter.

ESP checked mail(s):
Number of mails checked by the ESP filter.

ESP spam found:
Number of mails marked as spam by the ESP filter.

Total spam incident:
Total number of virus incidents found by the ESP filer and the spam filter.

Registration

Standard package

The product can't be used without a valid registration key.

The program warns the user by sending a message into the log filer once a day when the ending of the registration period is coming.

After registration key had expired, the product works as before (without any restriction) until a program update (virus database updating is possible). After program updating, you need a new license (registration key) to use the program.

The registration key must be placed into the anti-virus system's configuration file (serialno option) together with the user name (username option). See the description of the configuration settings for more.

Activate ESP and ZH function

The ESP and ZH modules are not available in the standard package. Please contact our Sales department at the sales@virusbuster.hu e-mail address for more information.

mail-part to the filters modules. The level2 filters also return a string value after processing the MIME-part, you can assign command(s) to the returned value.

Level2 modules (filters):

- virus filter (libflt2_virus.so)
- file filter (libflt2_fileflt.so)

Database update

The product uses incremental virus database update mechanism to keep the virus database up-to-date. The advantage of this method is that the program doesn't need to download the whole virus database file every time (its size is several MBs) but usually only a small additional database package including the virus signatures processed and released recently. Using this mechanism, the download time is decreased to a minimal level so we can release additional virus database packages several times a day to improve the defense. Users can obtain protection against new malware without spending long time and generating considerable network load for the update. The protection is available almost immediately after the signatures of the newly discovered viruses have been processed in our virus lab.

To automate the update processes, use the available scripts attached to the package, they are placed to the /usr/sbin/ directory (vbm_dbupdate.sh and vbm_dbupdate_http.sh).

Execute one of them, it is going to download the virus- and/or spam database, copies it/them into the correct directory and activates it/them. The download and update processes will only be performed, if the database available in the server is newer than one on your computer. Otherwise the database will be left unchanged.

To execute the scripts, you should enter the vbm_dbupdate.sh (through HTTP use the vbm_dbupdate_http.sh) command. It is possible to use parameters, too:

nosdb - the spam database will not be updated
verbose - display progress bar

Example:

```
vbm_dbupdate.sh nosdb verbose
```

The spam database will not be updated, the progress bar will be displayed.

To run these scripts, you need wget program! By the help of cron, you can schedule the script executing to be performed by half an hour. Register into /etc/crontab:

```
0,30 * * * * root /usr/sbin/vbm_dbupdate.sh
```

The database files can be downloaded manually from the following FTP server:

The virus database consists of several files, you need to download all the files from the following folder:

```
update.virusbuster.hu/pub12/vbuster/vdb12/
```

Spam database file, you need to uncompress before use:

```
update.virusbuster.hu/pub12/vbuster/sdb/sdb.tgz
```

The configuration file

Structure of the configuration file

The configuration file stores the program settings in hierarchical structure. The storing mechanism based on the encapsulation concept which means that user has to specify the storing path (section) for each coherent setting group step by step.

The path (section) must be specified between square brackets in the configuration file:

```
[Milter/Global]
```

Enter comments by using semicolon (;) before the comment text. The characters entered after semicolon will not be interpreted by the parser. You can also use this function to disable a selected option quickly.

```
;command2="copy_mail ('/tmp')"
```

If you have to specify network addresses, enter the following address forms usually in the whole configuration file:

```
unix:/path/to/file
```

```
inet:port@{hostname|ip-address}
```

Example:

```
unix:/var/run/vbuster/vbmsrv
```

```
inet:9009@192.168.2.42
```

```
inet:9427@somebody.com
```

Filter definitions

```
[Milter/FilterRules/Rule/Filter]
; level1 filter settings (1)...
[Milter/FilterRules/Rule/Filter/Action]
; level1 action settings (1)...

[Milter/FilterRules/Rule/Filter]
; level2 general filter settings (2)...
[Milter/FilterRules/Rule/Filter/Level2]
; level2 filter settings (2)...
[Milter/FilterRules/Rule/Filter/Level2/Action]
; level2 action settings (2)...

...
[Milter/FilterRules/Rule/Filter]
; filter settings (n)...
[Milter/FilterRules/Rule/Filter/Action]
; action settings (n)...
```

Filter settings belonging to a rule must be defined after the rule definition line in the [Milter/FilterRules/Rule/Filter] section. Each new filter definition must be placed into a new [Milter/FilterRules/Rule/Filter] section. After the mail has been scanned by the filter, the program returns the filter's result. Based on these values, different actions may be performed on the checked mail. You can specify the required actions and result types in the [Milter/FilterRules/Rule/Filter/Action] section for the level1 filters.

The level2 filters are controlled by a special level1 filter, called: libflt_level2. If you would like to configure a level2 filter, first you have to specify the control filter (libflt_level2) and set its general settings in the [Milter/FilterRules/Rule/Filter] section. After defining general settings, you have to select and set the selected level2 filter in the [Milter/FilterRules/Rule/Filter/Level2] section. The actions (based on the filter result) may be specified in the [Milter/FilterRules/Rule/Filter/Level2/Action] section.

GENERAL FILTER OPTIONS

disable = <0|1>

Disable or enable filter. Possible values: 0 or 1.

1: Filter is disabled.

0: Filter is active, enabled.

filter = <filter type>

Set filter type.

The available values (module level):

libflt_addr - address filter (level1)

libflt_asap - ZH/ESP filter (level1)

libflt_result - result filter (level1)

libflt_bayes - spam filter module (level1)

libflt_rbl - RBL filter

libflt_level2 - level2 module manager (level1)

libflt2_virus - virus filter module (level2)

libflt2_fileflt - file filter module (level2)

In case of level2 filter:

filter2path = <path>

Directory specification of the Level2 filter modules. Location where the level2 modules can be found.

max_mime_depth = <number>

The program is going to scan the embedded e-mail-type mimes down to the specified depth. Setting the 0 (zero) value for the option, none of the embedded e-mail-type mimes will be scanned.

Important!

E-mail-type mimes embedded deeper into the mail than the value of this option will not be scanned, so harmful materials may have found in that deeper levels will get into your system.

Action specification

```
[Milter/FilterRules/Rule/Filter]
filter_option_1
filter_option_2
...
filter_option_n
```

```
[Milter/FilterRules/Rule/Filter/Action] ; action (1)
result=
count=
command=
```

```
[Milter/FilterRules/Rule/Filter/Action] ; action (2)
result=
count=
command=
command2=
command3=
...
```

The filter module returns the result of the mail or attachment scan. The program compares this returned value with the 'result' property of the specified actions and where the values are the same, the action will be performed. You can use regular expression in the 'result' option, in such a case you must insert it between quotes (""). It is also possible to use tokens in the 'command' option.

Note!

Specify level1 actions in the [Milter/FilterRules/Rule/Filter/Action], and level2 actions in the [Milter/FilterRules/Rule/Filter/Level2/Action] section.

Action settings

result = <result value>

You can specify a result value for the filter module. If the filter module returns the same value as you specified, the 'command' options of the section will be performed. Result values may be different by filter modules, these are described in the chapter of the module descriptions. Regular values can be used between quotes ("").

count = <number>

This value specifies that how many times should the commands be performed during the mail processing (0 means unlimited). If 'count' option is not specified to the action, the command will be always performed.

command = <command>

Define actions. The possible commands and their functions is detailed in the "Module commands" chapter. You must insert the commands' value between quotes (""), the parameters between apostrophes (''), separated by comma (,). If you would like to use the apostrophe (') character in the parameters, then use the backslash (\) character right before it.

For example:

```
command="header_modify ( 'Subject', '%subject% \'Scanned e-mail\'' )"
```

Specify several parameters of the same name:

If you would like to specify several parameters of the same name, then you have to number them from the second one (where the number: 2..N).

For example:

```
[Milter/FilterRules/Rule/Filter/Action]
command=
```

command2=
command3=

The daemons of the package have different configuration files which will be detailed below.

VBMSRV daemon configuration file (vbmsrv.conf)

General settings

```
[General]
logaddress=unix:/var/run/vbuster/vbmlog
address=unix:/var/run/vbuster/vbmsrv
socket_permission=0660
run-as-user=user
run-as-group=group
pid_file=/var/run/vbuster/vbmsrv.pid
tempdir=/tmp
process_num=0
module=
```

 The settings:

logaddress=<netcmd address>

You have to specify the communication address of the log component.

Default: logaddress=unix:/var/run/vbuster/vbmlog

address=<socket>

Specify address through which the MTA and the anti-virus application will communicate.

The same setting must be specified in the MTA's configuration Default:

address=unix:/var/run/vbuster/vbmsrv

socket_permission=<octal number>

Set unix socket permission with an octal number. Default: 0660 (in case of Qmail), 0600 (in case of Sendmail).

run-as-user=user

run-as-group=group

VBMLLOG daemon starts with root permission as all the daemon programs usually at the computer startup. However, it is more secure to run with unprivileged user permission. If the VBMLLOG is started with root permission, it is able to change to the user and group specified in these options.

If the VBMLLOG is not started with root permission, it will not be able to change to other user permissions.

pid_file=/var/run/vbuster/vbmsrv.pid

Pid file with path of the anti-virus application.

tempdir=/tmp

Set the temporary directory used by the application.

Default: /tmp

process_num = <processor number>

It is recommended to use this option if the following system components are available: FreeBSD 4.x, multi-processor system, SMP kernel

The kernel is only able to assign the processes to the CPUs existed on FreeBSD 4.x in multi-process environment (the anti-virus protection is thread-based so it will always be run by only one processor). In this option you can specify the number of the anti-virus protection instances to be run, these will be processed by the processors separately. A built-in load-balancing system is responsible for the equal load, it will assign the Sedmail connects to the instances of the anti-virus protection.

Operation:

It creates sockets according to the 'process_num' value:

```
unix:/var/run/vbuster/vbmsrv ->
```

```
unix:/var/run/vbuster/vbmsrv.0
```

```
unix:/var/run/vbuster/vbmsrv.1  
...  
unix:/var/run/vbuster/vbmsrv.n
```

or

```
inet:3333@localhost ->  
inet:3334@localhost  
inet:3335@localhost  
inet:3336@localhost  
inet:3337@localhost
```

module=

Specify the mail server interface module that makes the connection possible between the selected mail server and the anti-virus system.

libvbraw.so - using Qmail, Courier or GroupWise MTA

libvbmilter.so - using Sendmail

Mailserver-module settings

Sendmail module settings

```
[General/Milter]
Milter_timeout = 300
smtpserver =
modify_body=
-----
```

Set the following option if you are using Sendmail:

milter_timeout = <second>

Sets the number of seconds until libmilter is waiting for an MTA connection before timing out a socket.

smtpserver = <server address>

If the processed e-mail is refused by the reject_mail command, you can set an MTA needed to deliver the mail when the send_copy command is used (the set MTA must be different to the one to which the VBMSRV is integrated).

Use the following address form:

```
inet:port@{hostname|ip-address}
```

Example:

```
inet:9442@somebody.com
```

modify_body = <0 or 1>

Certain mail server programs (e.g. Postfix earlier than version 2.3) will not work properly if the mail-body is modified through the milter interface during the virus scan process. Using this option you can enable (1) or disable (0) the modification of mail body if necessary.

Default setting: 1

Qmail module settings

```
[General/Qmail]
original_qmail_queue=/var/qmail/bin/qmail_queue2
accept_mail_retval=0
drop_mail_retval=0
reject_mail_retval=31
-----
```

Set the following options if you are using Qmail:

original_qmail_queue=<path>

Path of the original qmail-queue which will be called by the VBMSRV's own qmail-queue to deliver the mail finally.

Default: /var/qmail/bin/qmail_queue2

accept_mail_retval=<number>

drop_mail_retval=<number>

reject_mail_retval=<number>

Set the return value of the anti-virus system returned to the Qmail module if one of the above incidents (accept mail, drop mail, reject mail) have been detected. Based on these results you can control the Qmail's return value returned to the mailer client. For more information read the Qmail's own manual (man) ('qmail-queue').

Courier module settings

```
[General/Courier]
original_courier_submit=/usr/lib/courier/courier/submit2
```

Set the following options if you are using Qmail:

original_courier_submit=<path>

Path of the original 'submit' which will be called by the VBMSRV's own 'submit' to deliver the mail finally.

Default: /usr/lib/courier/courier/submit2

GroupWise module settings

[General/Groupwise]

daemon_pid_file=/var/run/vbuster/vbgwia.pid

service_queue_dir=

queue_dir=

check_incoming=yes

check_outgoing=yes

checking_period=10

Set the following options if you are using GroupWise:

daemon_pid_file=<path>

Path of the vbgwia pid file.

Default: /var/run/vbuster/vbgwia.pid

service_queue_dir=<path>

Set the SMTP Service Queue of the GroupWise Internet Agent to this option.

Find the requested path in the ConsoleOne program:

1. In the selected domain, click with the right mouse button on the GWIA you want to protect then select the 'Properties' menu item.
2. Select the 'Server Directories' panel.
3. Click the 'Advanced' button.

Enter the path found in the 'SMTP Service Queues Directory' setting to the 'service_queue_dir' option.

queue_dir=<path>

Enter the Groupwise Internet Agent's directory here. You can also find this path in the ConsoleOne program:

Do the 1st and 2nd steps mentioned above, find the 'SMTP Queues Directory' option on the 'Server Directories' panel and set it's value to the 'queue_dir' options.

check_incoming=<yes/no>

Yes: Incoming mails will be scanned (this is the default setting).

check_outgoing=<yes/no>

Yes: Outgoing mails will be scanned (this is the default setting).

checking_period=<period>

Set a time period (second) after the product scan the specified folders.

Default: 10

Log settings

```
[Logging]
logscreen=0
```

The setting:

logscreen=<0|1>

Log to screen.

1: The log messages came from the filter modules will be displayed on the screen.
This function could be used in non-daemon mode.

0: Inactive.

General settings of the virus scan engine

```
[Engine]
max_decompress_size=0
max_decompress_ratio=0
max_decompress_depth=5
vdb_file=/var/lib/vbuster/vdb9.xml
-----
```

Specify the general setting of the scan engine in the [Engine] section.

max_decompress_size=0

If this file size limit is exceeded while uncompress an archive, the program stops the uncompression and scanning of the file and returns the 'archive_exploit' result. (Option's value is in MByte).

Specifying the 0 value means using the virus scan engine's default value for this option.

max_decompress_ratio=0

If the size of the decompressed file is 50 times (or more) greater than the compressed file's, the program will return the 'archive_exploit' result.

Specifying the 0 value means using the virus scan engine's default value for this option.

Other explanation (option's value in percent): $1/n*100$, where n is the value.

For example the value is 50. $1/50*100 = 2\%$ so if the compression ratio is better than 2% the program will return the 'archive_exploit' result.

max_decompress_depth=5

The program will scan the multi-level archives down to the specified depth. If the program finds more depth levels, it will return the 'archive_depth_limit' result and files that are deeper than the specified level will not be scanned.

vdb_file = <file name with path>

Location of the XMS descriptor file for the virus database.

General settings of the spam scan engine

[Bayes]

```
bayes_sdb=/var/lib/vbuster/vbuster.sdb
```

Specify the general setting of the spam scan engine in the [Bayes] section.

bayes_sdb = <File name with path>

The spam database file's name and location in the system.

ZH/ESP general settings (Asap daemon)

```
[Asap]
asapd_path      = /usr/sbin/vbasapd
proxy           =
proxy-auth      =
proxy-domain    =
max-connection  = 128
pid-file        = /var/tmp/asapd.pid
cache-size      = 4000
do-detect       = yes
listen          = 127.0.0.1:9999
protocol-version =
```

The following options are available to set the ZH/ESP filter below the [Asap] label:

asapd_path = /user/sbin/vbasapd

vbasapd binary path. The Watchdog controls this daemon so it needs to know where the daemon can be found.

proxy = <host:port>

Set proxy server, if the sever running vbasapd is not able to connect directly to the Commtouch's server. You can also set user name and password as follows:

```
[username[:password]@]host[:port]
```

Default value: none

proxy-auth

Proxy authentication mode: Basic, NTLM or NoAuth

Default value: NoAuth

proxy-domain

Domain name used by the proxy server.

Default value: none

max-connection = 128

Maximum number of queries.

Default value: 128

pid-file = /var/run/vbuster/vbasapd.pid

vbasapd pid file with path.

Default value: /var/run/vbuster/vbasapd.pid

cache-size = 10000

Maximum number of queries stored in the vbasapd daemon's cache.

Default value: 10000

do-detect = yes

Whether the Commtouch server updates its database with unknown mails or not?

If this function active, those mails that are unknown for the server will be followed by the system. If the unknown mail type is reported from many different locations, it may be marked as bulk mail in the database.

listen = 127.0.0.1:9999

The vbasapd daemon's IP address. VBMSRV uses this IP to communicate with the daemon.

Default value: 127.0.0.1:9999

protocol-version = 3

Set the ZH filter version you would like to use in the product.

This option has to be introduced because of the expansion of the ZH filter-levels.

Available values for this option: 3 or 2

The only difference between the two versions is the number of the filter levels. The version 3 filter allows more detailed level settings than the version 2 in which the previous levels are available.

Default setting: 2

See the ZH module setting part of this document for more information about the filter-level versions.

Global settings

```
[Milter]
[Milter/Global]
username=user_name
serialno=xxxxxx-xxxxxx-xxxxxx
filters=/usr/lib/vbmsrv/
acceptnomatch=1
cfg-watch-timer=120
stataddr=unix:/var/run/vbuster/vbmstat
max-connections=100
-----
```

You can find the MAILFILTER daemon settings in the [Milter] section. Inside this section the general settings are in the [Milter/Global] section.

username = <user name>

Specifying the user name based on your license.

serialno = <registration key>

Specifying the registration key in the following form: XXXXX-XXXXX-XXXXX

Note!

You are not allowed to use the program without (valid) registration data!

filters = <path>

Level1 filter modules' directory specification. The location where the level1 modules could be found.

acceptnomatch = <number>

How the program handles the mails which not matching the rules?

0: Refuses them.

1: Accepts them, but it does not perform filtering them, mails will be forwarded without checking.

cfg-watch-timer = <second>

The cfg-watch-timer field sets the intervals at which the program should check if the configuration file has been modified. If so the file will be reloaded.

stataddr=unix:/var/run/vbuster/vbmstat

The statistical server communicates through the specified address.

Default: stataddr=unix:/var/run/vbuster/vbmstat

max-connections=100

Client connection limit. Maximum number of the clients that will be allowed to connect to the anti-virus system. If this limit is reached 4xx error message (temporary unavailable) will be returned to the MTA in case of every further attempt.

Rule definition

```
[Milter/Filterrules]
[Milter/Filterrules/Rule]
sourcemask=194.222.242.0/24
-----
```

Specify a rule, the filter modules defined for this rule will be applied to the mails matching this rule. Define the rule in the [Milter/Filterrules/Rule] section inside the [Milter/Filterrules] section.

sourcemask = <domain>

Filter modules defined after the sourcemask option will be applied to the mails sent from the specified domain. These filter modules belong to this rule. If you insert a new sourcemask option (with the required section specifications) the filter modules defined after the new sourcemask option will be applied the mails matching the new rule (sourcemask).

Global module settings (level1)

```
[Milter/Filterrules/Rule/Filter]
```

```
disable=0  
filter=libflt_global
```

```
[Milter/Filterrules/Rule/Filter/Action]
```

```
result=true  
command="add_header('X-VBMSRV', 'Scanned by VBMSRV')"
```

In this module you can set actions which will be performed on each mail processed. Set the global filter module for the 'filter' option (libflt_global).

Actions

This module returns 'true' value without exceptions. Commands could be used in the 'command' option are detailed in the "Module commands" chapter.

In this instance, the selected 'command' will mark the processed mail. The 'add_header' option adds a new field and its content to the mail header.

Virus filter module settings (level2)

```
; ---- level2 filter module initialization
[Milter/Filterrules/Rule/Filter]
disable=0
filter=libflt_level2
filter2path= /usr/lib/vbuster/filters/
; ---- end of level2 filter module initialization

[Milter/Filterrules/Rule/Filter/Level2]
disable=0
filter2=libflt2_virus

filemask=%DEFAULTMASKLIST%, *.jpg
exclude-filemask=

search_method=strict
heuristic_level=normal
macro_delete=no
containers=yes

[Milter/Filterrules/Rule/Filter/Level2/Action]
result="infe.*"
command="modify_header ('Subject','%virusname%')"
command2= \
"replace('*.txt','iso-8859-2',
'*****
** Attachment %filename% was infected with %virusname% virus,
** attachment part was removed.
*****')"
```

```
[Milter/Filterrules/Rule/Filter/Level2/Action]
result=cleaned
command="continue"

[Milter/Filterrules/Rule/Filter/Level2/Action]
result=i-worm
command="drop_mail"
-----
```

The above configuration part is a possible example of the virus filter module setting. Because the virus filter module is level2 module, first the level2 manager module (libflt_level2) and the location of the level2 modules ('filter2path') must be defined. This is the initialization method of the level2 modules.

You have to specify and set the requested level2 module in the [Milter/Filterrules/Rule/Filter/Level2] section.

filter2 = <level2 filter module type>

Set the requested filter module, in the present case this is the virus filter module (libflt2_virus).

filemask =

Please specify attachment names and mask that you would like to be scanned by the filter module. These must be separated by commas (,). If you specify the star (*) character, then all the files will be scanned. Use the following mask to cover attachments without any extensions: "*."

The %DEFAULTMASKLIST% token is also available to use, it represents the default mask list of the engine (if the virus filter is active, this mask list is displayed in the log file (in case of INFO or greater log level)).

Remark!

In case of appearance of a new, critical threat, extra file masks may be added to the existed list automatically, supplied by the virus database. This file mask will also be scanned by default while the threat spells real danger.

exclude-filemask=

Attachment matching one of the filemasks specified in this option will not be virus scanned, even if its extension is registered in the 'filemask' option. Specify this option in the same way as 'filemask'.

search_method = <fast/strict/full>

Specify the search method. The virus scanning engine is able to scan for and detect viruses according to the set methods/levels. It is possible to choose the needed scanning method in the components in the software. The following levels are available:

fast:

Only scans those parts of the file, which are most likely to contain a virus and does not detect viruses, which can only be detected by using a major amount of system resources (e.g. Excel FORMULA viruses).

extensive:

Optimized scanning method, which detects all viruses registered in the virus database and scans those parts of the file, which are most likely to contain a virus.

full:

Detects all viruses registered in the virus database and scans the whole file, even those parts, where viruses are not likely to be found.

heuristic_level = <off/normal/high>

During the heuristic analysis, the software tries to detect codes and programs, which have virus-like characteristics but are not registered in the virus database. If such a suspicious file is found, the user is notified. The following levels of heuristic analysis are available:

off:

No heuristic analysis.

normal:

The depth of the analysis is limited, the possibility of false positives is low, but the chance of detecting unknown viruses is not too high.

high:

The chance of detecting unknown viruses is higher, but there is a higher possibility of false positives.

macro_delete = <yes/no>

yes: all the macros will be deleted.

no: inactive.

containers = <yes/no>

Scanning in compressed files.

yes: scanning in container files (archives, compressed files). The anti-virus system recognizes the compressed, archived files automatically.

Returned results, actions

Available return values of the virus filter module:

none: there was no virus found

i-worm: infected file, I-Worm type incident

cleaned: infected file, virus successfully killed

infected: infected file, fail to kill virus

encrypted_archive: compressed file protected by password

archive_exploit: too big archive (exploit)

archive_depth_limit: if the limit of the 'max_decompress_depth' option exceeded

error: error occurred during processing

These values are available to use as the value of the 'result' option. Commands could be used in the 'command' option are detailed in the "Module commands" chapter.

In the example

1.
result="(infected|i-worm)"
The result value is specified as a regular expression. If the filter module returns a string either 'infected' or 'i-worm' then the specified action will be performed. In this example the command is header modification: the program inserts the name of the virus into the subject field.
Because this result has a secondary command (command2) so it also will be performed: The infected attachment will be replaced to the warning text file.
2.
result=cleaned
If the attachment was infected, but the virus was killed successfully, then the command to be performed is the 'continue', the mail will be forwarded to the MTA to deliver.
3.
result=i-worm
If the virus filter recognizes the mail as Internet worm, then the drop_mail action will be allied to the mail, the mail will not be delivered.

File filter module settings (level2)

```
[Milter/Filterrules/Rule/Filter/Level2]
disable=0
filter2=libflt2_fileflt

filemask=*.pif, *.scr, *.vbs
use-regex=no

[Milter/Filterrules/Rule/Filter/Level2/Action]
result=true
command="delete"
-----
```

Initialization of level2 modules has been described in the virus filter section. Initialization must be done only once so if it has been initialized in the virus filter (or rather in the first module specification in the configuration file) you don't need to do it again.

filter2 = <level2 filter module>

Set the requested filter module, in the present case this is the file filter module (libflt2_fileflt).

filemask =

You can specify as filemask as you wish by using the * and ? characters. If one of the filemask matches the attachment files, the specified command will be applied to the file. The file mask values must be separated by commas (,).

Returned results, actions

Return values of the file filter module:

true: the name of the file (attachment) matched one of the values of the 'filemask' option
false: the name of the attachment) didn't match the values of the 'filemask' option
error: error occurred during processing

In the example

result=true

If one of the values of the 'filemask' option matches the name of the file, the program will delete the attachment according to the command option's value.

Spam filter module settings (level1)

```
[Milter/Filterrules/Rule/Filter]
disable=0
filter=libflt_bayes

;language/script filter activation
language_filter=yes

;spam filter level setting
filter_level=high

;actions for each spam level
[Milter/Filterrules/Rule/Filter/Action]
result=low_level_spam
command="drop_mail"

[Milter/Filterrules/Rule/Filter/Action]
result=normal_level_spam
command="reject_mail('550','Recognized as SPAM')"
command2="add_rcpt_to ('admin@domain.com')"

[Milter/Filterrules/Rule/Filter/Action]
result=high_level_spam
command="modify_header ('Subject','***SPAM*** %subject%')"

;common spam filter action for spams
;[Milter/Filterrules/Rule/Filter/Action]
;result=true
;command="modify_header ('Subject','***SPAM*** %subject%')"

;for language/script filter
[Milter/Filterrules/Rule/Filter/Action]
result=language_chinese
command="drop_mail"
```

The spam filter returns the 'true' or 'false' result to indicate the mail is spam ('true') or not ('false'). In case of spam it also returns the level of the spam. Set your security spam level in the 'filter_level' option and...

- ...use the 'true' result in the rule if you don't want to set different actions for the spam according to its spam level (common action).
- ...use the name of the security levels in the rules to assign different actions for the spam according to its spam level.

Language/script filtering (language_filter):

The language filter function provides great ability to filter e-mails according to the language and script-type of their text parts. The language/script-type database that needs for the recognition is built in the vbuster.sdb (spam database) file so you need to have the spam database file downloaded and available to activate the language filter. The 'language' means the natural language of the mail-text, for example: English, Hungarian, Russian, Chinese, etc. The 'script-type' means the character-set used in the mail, for example: Latin, Cyrillic, Greek letters, Far-Eastern letters, etc. The language filter works based on heuristics detection so its result will be the most likely script-type/language used in the mail.

Options:

filter = <levell filter module>

Set the requested filter module, in the present case this is the spam filter module (libflt_bayes).

filter_level = <low/normal/high>

Filter level setting. The filter marks the mail as spam which is found on the specified spam level or below.

low:

Insignificant false positives, the spam detection rate is normal. This means that the spam filter only marks that mails which are real spam by the spam database, normal mails are not affected (low false positives).

normal:

The false positive index increases a bit compared to 'low' level. This level provides effective spam recognition. This is the optimal level.

high:

The number of false positives increases but the filter filters out almost all the spam mails on this level. This setting is recommended if mails marked as spam can be reviewed because of the relatively high number of false positives.

If the mail is marked as spam on the selected level, the specified action will be performed. Different levels should have another actions. The following actions are recommended for the levels:

low: drop_mail

normal: reject_mail, add_rcpt_to

high: modify_header

language_filter = <yes/no>

Enable/disable language/script filter.

Returned results, actions

Return values of spam filtering:

true: the spam filter marked the mail as spam based on the specified setting

false: the mail is not spam according to the spam filter

If the result is 'true', the spam levels also be returned (explanation read above):

low_level_spam

normal_level_spam

high_level_spam

Return values of language filtering:

If the language/script filter is enabled, the program also returns the language/script type of the language filter.

When the program is started, it copies the available values to the log file if the language/script filter is active (language_filter=yes). The language/script database is improved continuously, that's why you can see the supported languages and scripts in the log file (in case of INFO or greater log level).

The supported language/script values can be set to the 'result' option.

Example:

result=language_english

or

```
result=script_latin
```

It is possible to invert the result of the comparison if you use a '!' (exclamation) mark at the beginning of the value. In such a case, the action(s) will be applied all the mails having different language/script to the specified one(s).

Example:

```
result=!language_hungarian|language_english
```

In such a case, the action(s) will not be applied to the mails using hungarian or english but to all the others.

Other possible result:

```
error: error occurred during processing
```

In the example

Assign different actions for spam mails:

result=low_level_spam

The mail is surely spam, the program simply does not forward the mail.

result=normal_level_spam

The mails is spam most likely, so the program rejects the mail 'command1' and sends a copy to the address specified in the 'command2' command.

result=high_level_spam

Because of the increase number of false positives, the program only modifies the subject field of the mail and forwards the mail back to the MTA.

Common action for spams:

result=true

If the mail is marked as spam on the high level, the program modifies the subject field of the mail and forwards it back to the MTA.

Language/script filtering:

result=language_chinese

All the mails written in Chinese language will be dropped (command="drop_mail").

Address filter /White/Black list/ (level1)

```
[Milter/Filterrules/Rule/Filter]
```

```
disable=0  
filter=libflt_addr
```

```
[Milter/Filterrules/Rule/Filter/Address]  
sender=1  
entry=*@domain.com  
external_file=/etc/vbuster/wlistaddr.txt
```

```
[Milter/Filterrules/Rule/Filter/Action]  
result=all_rcptto_listed  
command="accept_mail"
```

This module filters the sender or recipient(s) of the mail based on the specified address(es).

Functioning:

- if all the recipients or the sender of the mail (according to the setting) are/is included in the address list then the action specified will be applied on the mail (in case of mailfrom_listed or all_rcptto_listed results)
- if there is at least one of all the recipients or the sender of the mail (according to the setting) who are/is not included in the address list and the mail would be blocked for this, the mail will be delivered without modification for those recipient(s) who are included in the address list. In this case even those actions will not be applied which would not modify the mail. (eg. copy mail)
- if there is at least one of all the recipients who is not included in the address list but the mail would not be blocked for this, the mail will be delivered to all the recipients with possible modifications which were set in the 'command' options.

sender=[0|1]

- 0: the module will filter the recipient addresses
- 1: it will filter the sender address

entry=[address(es)]

Enter address(es) to be filtered. Use comma (,) character to enumerate a number of addresses. You can use the * joker character in the localpar of the addresses. Example: *@domain.com, aaa@bbb.ccc

external_file=[file name with path]

Addresses to be filtered could be stored in an external file, too. You can specify the filename with its path in this option. The addresses will be read by lines from the file. If a semicolon (;) is placed at the beginning of the line, that line will be considered as comment.

Example for external file content:

```
*@domain.com  
a@b.com  
;d@e.com
```

Returned results, actions

Available return values of the address filter module:

mailfrom_listed: the sender is included in the list

all_rcptto_listed: all the recipients/sender are included in the list

rcptto_listed: there is at least one recipient/sender who is not included in the list

These values are available to use as the value of the 'result' option. Commands could be used in the 'command' option are detailed in the "Module commands" chapter.

In the example

According to the result of the filter (resultt=all_rcptto_listed) all the recipients are included in the address list so the mail will be accepted (command="accept_mail").

Result filter module settings (level1)

```
[Milter/Filterrules/Rule/Filter]
disable=0
filter=libflt_result

[Milter/Filterrules/Rule/Filter/Action]
result=bayes_true.*l2virus_infected
command="drop_mail"

[Milter/Filterrules/Rule/Filter/Action]
result=addr_mailfrom_listed.*bayes_true
command="add_rcpt_to ('admin@domain.com')"
```

All results of other filters (virus, spam, ...) specified in the configuration file before the Result filter are available in this special filter. You can connect filters by assigning actions based on their result combinations specified in the Result filter.

So the Result filter provides in a string (result string) all the filters' results which have been performed before the Result filter. With the help of regular expressions you can compare various conditions with the result string and assign actions to the mail if there is a correspondence.

Actions as reject_mail, drop_mail or accept_mail specified before the Result filter can block the activation of the Result filter because these ones break the mail process so the Result filter can not be activated. Keep this in mind when composing the configuration file and the actions.

Other possibility is not to assign actions to the filters specified before the Result filter but set them in the Result filter getting their results from the result string.

Because the results can be the same even if they are resulted by two different filters (e.g. true), these values must be distinguished from each other. Use the following prefixes at the beginning of the result separated by an underline:

```
address filter: addr
spam filter: bayes
file filter: l2fileflt
virus filter: l2virus
global filter: global
rbl filter: rbl
zh/esp filter: asap
```

In the example

1.
Reject infected and simultaneously spam mails:
virus filter result: infected
spam filter result: true

Use this mask if the spam filter is placed before the virus filter:
result=bayes_true.*l2virus_infected
command="drop_mail"

2.
Forward spam mails to the administrator that come from a specified sender:
spam filter result: true
address filter result: mailfrom_listed

Use this mask if the address filter is placed before the spam filter:

```
result=addr_mailfrom_listed.*bayes_true  
command="add_rcpt_to ('admin@domain.com')"
```

Important!

For correct operation, place the filters' results into the result mask in the same order as the filters are specified in the configuration file. Also keep the order of values inside a filter: first use the incident level (e.g. bayes_high_level_spam) then the incident flag (bayes_true).

RBL filter settings (level1)

```
[Milter/Filterrules/Rule/Filter]
disable=0
filter=libflt_rbl

[Milter/Filterrules/Rule/Filter/DNSRBL]
host=relays.ordb.org
[Milter/Filterrules/Rule/Filter/DNSRBL]
host=sbl.spamhaus.org

[Milter/Filterrules/Rule/Filter/Action]
result=true
command="drop_mail"
```

In the RBL filter, you can specify web sites providing realtime database of IP addresses of verified spam sources, supplied as a free service to help email administrators better manage incoming email streams. Before receiving a mail, the program checks if the sender's IP address can be found in the specified database(s). If so, the selected command(s) will be performed.

Filter settings:

disable = <number>

Enable/disable the filter. Values: 0 or 1
1: The filter is disabled
0: The filter is enabled

filter = <filter type>

Specifying filter type. Set the libflt_rbl (level1) to the RBL filter.

Set RBL server in the 'host' option of the [Milter/Filterrules/Rule/Filter/DNSRBL] section (you are allowed to set it several times).

Returned results, actions

Available return values of RBL filter module:

```
true: IP address of the sender is found on the list(s)
false: IP address of the sender is NOT found on the list(s)
```

These values are available to use as the value of the 'result' option. Commands could be used in the 'command' option are detailed in the "Module commands" chapter.

In the example

```
result=true
```

If the RBL servers' IP list includes the client's IP (true), the program execute the drop_mail command ('command').

ZH/ESP filter settings (level1)

```
[Milter/Filterrules/Rule/Filter]
```

```
disable=0
```

```
filter=libflt_asap
```

```
ip=127.0.0.1
```

```
port=9999
```

```
timeout=5000
```

```
virus_level=high
```

```
;spam_level=confirmed
```

```
retry=5
```

```
ip_information=ip-ignore-list
```

```
ip_ignore_list=127.0.0.1/8, 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
```

```
received-num=3
```

```
[Milter/Filterrules/Rule/Filter/Action]
```

```
result=virus_true
```

```
command="drop_mail"
```

```
command2="add_rcpt_to ('vod@vlab.virusbuster.hu')"
```

Filter settings:

disable = <0|1>

Disable or enable filter. Possible values: 0 or 1.

1: Filter is disabled its settings will not be performed.

0: Filter is active.

filter = <filter type>

Sets libflt_asap filter type.

ip = <ip address>

IP address of the computer that executes vbasapd. (default: 127.0.0.1)

port = nnn

If vbasapd can not be found on the standard port, it sets the used one for the VBMSRV. (default: 9999)

timeout = nnn

VBMSRV is waiting for the answer of vbasapd until specified time interval expires (msec). (default: 1000)

virus_level =

Sets virus sensitivity level. If the filter returns the selected level (or above) the mail will be considered as infected mail (virus_true).

The filter levels that can be set in this option depend on the value of the 'protocol-version' option (ZH filter-version). (We had to introduce the filter-version number because of the filter level expansion and the need of compatibility to the previous versions.)

Available filter levels when setting version 3 ZH filter:

- VIRUS: Virus threat has been detected in the message.
- HIGH: High likelihood of the message presenting a virus threat.
- MEDIUM: Probable threat of virus in the message has been detected.
- UNKNOWN: Threat for virus could not be determined at this time.
- NONE: Confirmed that message does not contain a virus.

Available filter-levels when setting version 2 ZH filter:

- HIGH: High likelihood of the message presenting a virus threat or virus threat has been detected in the message.

- MEDIUM: Probable threat of virus in the message has been detected.
- UNKNOWN: Threat for virus could not be determined at this time.
- NONE: Confirmed that message does not contain a virus.

We recommend you to use version 3 ZH filter so that you can set more detailed threat-levels for the filter

spam_level =

If the filter returns the selected level (or above) the mail will be considered as spam (spam_true).

The meanings of the levels are (from the strongest to the weakest):

- CONFIRMED: The mail is a confirmed spam.
- BULK: The mail is not appearing in the database as spam, but the database server has this sample from more locations already. (Attention! The mail is not a confirmed spam! It's typical also in the case of bulk news letters.)
- SUSPECT: spam suspected mail.
- UNKNOWN: there are no information to classify the mail
- NONE: the mail is surely not a spam

retry = <N>

Number of reconnection attempts in case of communication error.

Default: 3

ip_information=real-ip/ip-ignore-list/received-header

Source of the IP address forwarded to the Commtouch server. The sent IP address also affects the result of ZH/ESP filter.

real-ip

It will use the IP address of the mailer client connected to the MTA.

ip-ignore-list

It will check the IP addresses put into the received field from the latest entry. If the IP address found in the field matches one of the 'ip_ignore_list' values, it will be ignored, the following will be checked and the first allowed will be used. If it could not find allowed IP address, the 'real-ip' will be used.

received-header

The IP address will be determined based on the 'received-num' option. If it could not find valid IP address, the 'real-ip' will be used.

ip_ignore_list=127.0.0.1/8, 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16

List of IPs, IP masks that must be ignored. Use comma (,) to separate values.

received-num=3

Checks the IP addresses put into the received field from the latest entry. The set value is an ordinal number, which IP address will be considered. This option is useful in case of sequenced mail servers to determine the mail's original IP address.

Returned results, actions

Available return values of the additional spam filter module:

```
virus_true: virus found*
virus_false: virus not found*
spam_true: spam found*
spam_false: spam not found*
(*)based on the settings
```

The filter also returns the level which the mail was found on:

virus_<level>: virus filter (ZH) result, returned by the Commtouch server. The value of the <level> can be: VIRUS/HIGH/MEDIUM/UNKNOWN/NONE

Example: virus_high

spam_<level>: spam filter (ESP) result, returned by the Commtouch server. The value of the <level> can be: CONFIRMED/BULK/SUSPECT/UNKNOWN/NONE

Example: spam_confirmed

These values are available to use as the value of the 'result' option.

In the example

result=virus_true

If the ZH filter found the mail infected (virus_true), the program performs the drop_mail command ('command' option), then forwards the current mail to the specified e-mail address based on the 'command2' command.

Module commands

Commands belong to level1 modules

continue

The mail processing may continue.

accept_mail

The mail will not be scanned, it will be accepted without checking.

reject_mail

The processing (filtering) may not continue. The mail will be rejected without scanning and the error codes and messages will be returned to the mailing client.

Parameters: error code, error message (only 5xx type error codes accepted)

Example: `command="reject_mail('550','Mail recognized as SPAM')"`

Important!

Using Q-mail mailing system, the given parameters will not be returned because the Q-mail will overwrite them with its own error code and message!

drop_mail

The mail processing may not continue, the mail will be accepted but will not be forwarded.

copy_mail

If one of the modules break the mail processing then the program will copy the whole mail named as 'mailXXXXXX' where the XXXXXX is a random generated number. Parameter: target directory.

Example: `command="copy_mail ('/tmp')"`

add_rcpt_to

A copy of the mail will also be sent to the recipient specified in the parameter.

Parameter: e-mail address

Example: `command="add_rcpt_to ('admin@domain.com')"`

send_copy

The e-mail will be forced to forward to the address specified in this command, even if the mail is refused with the 'reject_mail' or 'drop_mail' command. If the returned value is 'continue' this command will work the same as 'add_rcpt_to'.

Parameters: mail state, e-mail address

The mail state parameter is not considered yet but you need to specify.

Example: `command="send_copy ('original', 'user@domain.com')"`

override_rcpt_to

The original and the added (send_copy, add_rcpt_to) recipients of the mail will be overwritten by the address specified in this command if the mail should be delivered.

Parameter: e-mail address

Example: `command="override_rcpt_to ('user@domain.com')"`

Important!

Use multiple add_rcpt_to and send_copy commands to set more than one recipients. It is pointless to use the override_rcpt_to command repeatedly because the mail will only be forwarded to the last-set one.

modify_header

Modify the mail's header. If the specified field could not be found in the header, then it will be inserted. Parameters: field name, value.

Example: `command="modify_header ('Subject','%virusname%')"`

add_header

Insert the specified field to the mail's header. Parameters: field name, value.

Example: `command="add_header('X-VBMSRV', 'Scanned!')"`

execute_command

Execute external command. Parameters: name of the program to be executed (with path), possible command line switches. The anti-virus systems' tokens can be used in the command line.

Example: `execute_command('touch /tmp/command-executed')`

Commands belong to level2 modules

All the commands belonging to the level1 modules are available completed with the following:

delete

Delete attachment.

replace

Replace attachment to text file. Parameters: file extension, char set, text.

Example: `command="replace('*.txt','iso-8859-2','*The %filename% attachment is infected by the %virusname% virus*')`

modify

Modification is allowed. (For example the virus filter module is killed the virus)

copy

Makes a copy of the original file. The file will be named as 'mailXXXXXX' where the XXXXXX is a random generated number. Parameter: target directory.

Example: `copy('/var/lib/quar')`

VBMLLOG daemon configuration file (vbmlog.conf)

The LOG component is responsible for storing and handling the log messages came from other modules of anti-virus system. The modules could send messages to the LOG daemon with the help of netcmd.

General settings

```
[General]
netcmdaddr=unix:/var/run/vbuster/vbmlog
pid_file=/var/run/vbuster/vbmlog.pid
run-as-user=user
run-as-group=group
```

 The settings:

netcmdaddr = <netcmd address>

You have to specify the communication address of VBMLLOG component.
 Default: netcmdaddr=unix:/var/run/vbuster/vbmlog

pid_file= <pid file>

Pid file with path of vbmlog.

run-as-user=user

run-as-group=group

VBMLLOG daemon starts with root permission as all the daemon programs usually at the computer startup. However, it is more secure to run with unprivileged user permission. If the VBMLLOG is started with root permission, it is able to change to the user and group specified in these options.

If the VBMLLOG is not started with root permission, it will not be able to change to other user permissions.

Log output settings

```
[OutputSetting]
[OutputSetting/Output]
type=file
filename=/var/log/vbuster/vbmsrv.log ;if type=file
perpid=0 ;if type=file
facility= ;if type=syslog
ident= ;if type=syslog
format="[%d/%m/%Y %H:%M:%S %z] $k $l $A $C PID:$P TID:$T \"$M\""
```

 There can be only one [OutputSetting] section specified in the vbmlog.conf file. The output sub-sections could be defined inside this section.

You can set the log's general setting in the [OutputSetting/Output] section. The number of [OutputSetting/Output] sections is not restricted. Each of these sections have different rules. Rules determine which messages occurred in the system should be logged. The VBMLLOG daemon processes the [OutputSetting/Output] sections and if the new log message's type matches one of the rule, it registers the log according to the settings of the specific output section.

type = <file|syslog|stdout>

Specify the type of the log file of the specific log section.

file: the log messages will be registered into a simple text file

syslog: the entries will be registered into syslog

stdout: the log will be written to the standard output

filename = <log file name>

If you would like the program to make log file (type=file), you can set the name of the file. Default: /var/log/vbuster/vbuster.log

perpid = <0|1>

This function available if type=file is set:

1: in case of making log file the program will insert the PID into the name of the log file

0: inactive

facility =

This function available if type=syslog is set:

In this option you can define which type of log messages will be considered. A type belongs to all the entries, it makes the search easier in the log file. You can specify several types separated by comma, or the ALL keyword.

For example: MAIL, USER

The following types are available:

KERN, USER, MAIL, DAEMON, AUTH, SYSLOG, LPR, NEWS, UUPC, CRON, AUTHPRIV, FTP

ident = <identifier>

This function available if type=syslog is set:

Identifier which will be placed before the log record. Default: vbuster

format =

Specification of the log file's structure. The following tokens are available:

Building the date:

%d - day

%m - month

%Y - year

%H - hour

%M - minute

%S - second

%z - time zone

Other:

\$a/\$A - computer name (hostname)

\$c/\$C - component name (which created the record)

\$k/\$K - facility (\$k returns counter, \$K returns name)

\$l/\$L - log priority (Level) (\$l returns counter, \$L returns name)

\$m/\$M - log message

\$n/\$N - new line character

\$p/\$P - PID

\$t/\$T - TID

\$\$ - insert \$ character

Default: [%d/%m/%Y %H:%M:%S %z] \$k \$l \$A \$C PID:\$P TID:\$T \"\$M\"

Output rules

```
[OutputSetting/Output/RuleSetting]
```

```
[OutputSetting/Output/RuleSetting/Rule]
```

```
components=ALL
```

```
priority=DEBUG3
```

```
facility=ALL
```

The rules should be inserted in a new section inside the [OutputSetting/Output] section. There is a main rule section [OutputSetting/Output/RuleSetting] and inside this section you can create several rules in the [OutputSetting/Output/RuleSetting/Rule] section.

components = <component names separated by comma or ALL>

This section will log those messages which created by the specified component(s). You can enumerate different components, these must be separated by comma. The ALL keyword means all the components.

Available component(s): vbmsrv

priority = <keywords or 0..10>

Log level. Only those messages will be registered which have the equal or lower level to the specified level.

Available values:

EMERG	0	system is unusable
ALERT	1	action must be taken immediately
CRITICAL	2	critical conditions
ERROR	3	error conditions
WARNING	4	warning conditions
NOTICE	5	normal, but significant, condition
INFO	6	informational message
DEBUG0	7	debug-level message
DEBUG1	8	debug-level message
DEBUG2	9	debug-level message
DEBUG3	10	debug-level message

facility = <0..4 or keywords or ALL>

In this option you can define which type of log messages will be considered. A type belongs to all the entries, it makes the search easier in the log file. You can specify several types separated by comma, or the ALL keyword.

For example: VIRUS, SPAM

The following types are available:

ALL	0	All kind of message types
SYSTEM	1	System log message
VIRUS	2	Virus found log message
SPAM	3	Spam found log message
DEBUG	4	Debug log message

In the example

According the rule the program will register the logs messages come from one of the components and the log type is not important either. The DEBUG3 or higher level log messages will be registered.

Example 2:

```
[OutputSetting/Output/RuleSetting/Rule]
components=vbmsrv
priority=INFO
facility=SPAM VIRUS
```

In this case only those log messages will be logged which come from the vbmsrv component in case of spam- or virus found (facility=SPAM VIRUS) being on INFO or higher log level (priority=INFO).

Tokens

Tokens available in the system:

%productversion%	program's version number
%from%	from field of the mail
%to%	to field of the mail
%mailid%	value of the mail's 'message-id' field if it exists.
%vdbversion%	virus database version

%sdbversion% spam database version
%subject% content of 'subject' field
%virusname% name of found virus
%filename% current attachment's name
%sender% e-mail address of the sender
%realip% address of the e-mail client which connected to the MTA
%recipient% e-mail address of the recipient
%mailfilename% file name and path of the copy of the original e-mail created by the antivirus system
%languagefilter% contains the result of the language filter (script/language)

ZH/ESP filter tokens:

%zhfilter% return value of ZH filter.
%espfilter% return value of ESP filter.
%asaprefid% reference string resulted by ZH/ESP filter

END USER AGREEMENT

THIS SOFTWARE END USER LICENSE AGREEMENT ("EULA") IS A LEGAL AGREEMENT BETWEEN YOU AND VirusBuster Ltd. READ IT CAREFULLY BEFORE COMPLETING THE INSTALLATION PROCESS AND USING THE SOFTWARE. IT PROVIDES A LICENSE TO USE THE SOFTWARE AND CONTAINS WARRANTY INFORMATION AND LIABILITY DISCLAIMERS. BY INSTALLING AND USING THE SOFTWARE, YOU ARE CONFIRMING YOUR ACCEPTANCE OF THE SOFTWARE AND AGREEING TO BECOME BOUND BY THE TERMS OF THIS AGREEMENT. IF YOU DO NOT AGREE TO BE BOUND BY THESE TERMS THEN DO NOT INSTALL THE SOFTWARE.

IMPORTANT NOTICE TO USERS: THE SOFTWARE IS NOT FAULT-TOLERANT AND IS NOT DESIGNED OR INTENDED FOR USE IN ANY HAZARDOUS ENVIRONMENT REQUIRING FAIL-SAFE PERFORMANCE OR OPERATION. THIS SOFTWARE IS NOT FOR USE IN THE OPERATION OF AIRCRAFT NAVIGATION, NUCLEAR FACILITIES, OR COMMUNICATION SYSTEMS, WEAPONS SYSTEMS, DIRECT OR INDIRECT LIFE-SUPPORT SYSTEMS, AIR TRAFFIC CONTROL, OR ANY APPLICATION OR INSTALLATION WHERE FAILURE COULD RESULT IN DEATH, SEVERE PHYSICAL INJURY OR PROPERTY DAMAGE.

1. Definitions

- (a) "Educational Version" means a version of the Software, so identified, for use by students and faculty of educational institutions only. "Home version" means a version of the Software, so identified, for use by individuals on a single computer at home only. Educational and Home Versions may not be used for, or distributed to any party for, any commercial purpose.*
- (b) Henceforward VirusBuster Ltd. means VirusBuster Ltd. and (where interpretable) its suppliers and licensors, if any.*
- (c) "Not For Resale (NFR) Version" means a version of the Software, so identified, to be used to review and evaluate the Software, only.*
- (d) "Software" means the VirusBuster Ltd. (R) VirusBuster(TM) software program supplied by VirusBuster Ltd. herewith, which may also include documentation, associated media, printed materials, and online and electronic documentation.*

2. License

This EULA allows you to:

- (a) Install and use the Software on a single computer; OR install and store the Software on a storage device, such as a network server, used only to run or install the Software on your other computers over an internal network, provided you have a license for each separate computer on which the Software is installed or run from the storage device. A license for the Software may not be shared or used concurrently on different computers.*
- (b) Educational and Home Version Only. If you have purchased a license for the Educational and/or the Home Version of the Software, then you may install or store the Software on a storage device, such as a network server, used only to run or install the Software on your other computers over an internal network for use by a total number of concurrent users not to exceed the number of user licenses you have been granted; provided, you agree to implement reasonable controls to ensure that your use of the Software does not exceed the number of licenses you have been granted. You agree that VirusBuster Ltd. may audit your use of the Software for compliance with the EULA at any time, upon reasonable notice.*
- (c) Make one copy of the Software in machine-readable form solely for backup purposes. You must reproduce on any such copy all copyright notices and any other proprietary legends on the original copy of the Software.*

3. License Restrictions

- (a) Other than as set forth in Section 2, you may not make or distribute copies of the Software, or electronically transfer the Software from one computer to another or over a network.*
- (b) You may not decompile, reverse engineer, disassemble, or otherwise reduce the Software to a human-perceivable form.*
- (c) You may not sell, rent, lease, transfer or sublicense the Software.*
- (d) You may not modify the Software or create derivative works based upon the Software.*
- (e) You may not use the Software in automatic, semi-automatic or manual tools designed to create virus signatures, virus detection routines, any other data or code for detecting malicious code or data.*
- (f) In the event that you fail to comply with this EULA, VirusBuster Ltd. may terminate the license and you must destroy all copies of the Software.*

4. Upgrades

If this copy of the Software is an upgrade from an earlier version of the Software, it is provided to you on a license exchange basis. You agree by your installation and use of this copy of the Software to voluntarily terminate your earlier EULA and that you will not continue to use the earlier version of the Software or transfer it to another person or entity.

5. Ownership

The foregoing license gives you limited rights to use the Software. VirusBuster Ltd. and its suppliers retain all right, title and interest, including all copyrights, in and to the Software and all copies thereof. All rights not specifically granted in this EULA, including International Copyrights, are reserved by VirusBuster Ltd. and its suppliers.

6. LIMITED WARRANTY AND DISCLAIMER

- (a) LIMITED WARRANTY. VirusBuster Ltd. warrants that, for a period of ninety (90) days from the date of delivery (as evidenced by a copy of your receipt) that the physical media on which the Software is furnished will be free from defects in*

materials and workmanship under normal use.

(b) NO OTHER WARRANTY. EXCEPT AS SET FORTH IN THE FOREGOING LIMITED WARRANTY, VirusBuster Ltd. AND ITS SUPPLIERS DISCLAIM ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, OR OTHERWISE INCLUDING THE WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. ALSO, THERE IS NO WARRANTY OF NONINFRINGEMENT, TITLE OR QUIET ENJOYMENT. IF APPLICABLE LAW IMPLIES ANY WARRANTIES WITH RESPECT TO THE SOFTWARE, ALL SUCH WARRANTIES ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF DELIVERY. No verbal or written information or advice given by VirusBuster Ltd. its dealers, distributors, agents or employees shall create a warranty or in any way increase the scope of this warranty.

7. Exclusive Remedy

Your exclusive remedy under Section 6 is to return the Software to the place you acquired it, with a copy of your receipt and a description of the problem. VirusBuster Ltd. will use reasonable commercial efforts to supply you with a replacement copy of the Software that substantially conforms to the documentation, provide a replacement for defective media. VirusBuster Ltd. shall have no responsibility if the Software has been altered in any way, if the media has been damaged by accident, abuse or misapplication, or if the failure arises out of use of the Software with other than a recommended hardware configuration.

8. LIMITATION OF LIABILITY.

NEITHER VirusBuster Ltd. NOR ITS SUPPLIERS SHALL BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES (INCLUDING DAMAGES FOR LOSS OF BUSINESS, LOSS OF PROFITS, BUSINESS INTERRUPTION OR THE LIKE), ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE OR THIS EULA BASED ON ANY THEORY OF LIABILITY INCLUDING BREACH OF CONTRACT, BREACH OF WARRANTY, TORT (INCLUDING NEGLIGENCE), PRODUCT LIABILITY OR OTHERWISE, EVEN IF VirusBuster Ltd. OR ITS REPRESENTATIVES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES AND EVEN IF A REMEDY SET FORTH HEREIN IS FOUND TO HAVE FAILED OF ITS ESSENTIAL PURPOSE.

9. Basis of Bargain

The Limited Warranty, Exclusive Remedies and Limited Liability set forth above are fundamental elements of the basis of the agreement between VirusBuster Ltd. and you. VirusBuster Ltd. would not be able to provide the Software on an economic basis without such limitations.

10. Consumer End Users Only

The limitations or exclusions of warranties and liability contained in this EULA do not affect or prejudice the statutory rights of a consumer, i.e., a person acquiring goods otherwise than in the course of a business.

11. General Provisions

The internal laws of Hungary shall govern this EULA. This EULA contains the complete agreement between the parties with respect to the subject matter hereof, and supersedes all prior or contemporaneous agreements or understandings, whether oral or written. All questions concerning this EULA shall be directed to VirusBuster Ltd.

VirusBuster and VirusBuster logo are trademarks or registered trademarks of VirusBuster Ltd. in Hungary and/or other countries. Other marks are the properties of their respective owners.

CONTACT

This manual provides comprehensive information on operational of our virus protection product. If you have any additional questions about it or would like to share your experience or proposals with us do not hesitate to contact us! Turn to us with confidence, your demands and remarks will be respected.

Address VirusBuster Ltd.
Budapest 1518,
Pf. 54.
Hungary

Phone (+36) 1 382-7000
Fax (+36) 1 382-7007
Web <http://www.virusbuster.hu>
Support <https://support.virusbuster.hu>
E-mail sales@virusbuster.hu
support@virusbuster.hu