GeCAD srl

The Software Company

Configuration file for ravmd

Release date: August 28th, 2002
Product version: 8.4.0
Document revision: 1.41

Copyright © since 2001 GeCAD Software® S.R.L.

GeCAD Software reserves itself the right to revise and modify its products according to its own necessities. This material describes the product, as it was in the moment this material was written and may not correctly describe the latest developments. For this reason, we recommend you to periodically check our website, *http://www.ravantivirus.com*, for the latest versions of product documentations.

GeCAD Software cannot be hold responsible for any special, collateral or accidental damages, related in any way to the use of this document.

GeCAD Software's entire liability, depending on the action, cannot go beyond the price paid for the product described in this material.

GeCAD Software does not guarantee either implicitly or explicitly the suitability of this material for your specific needs. This material is provided on an "as-is" basis.

GeCAD Software trademarks: GeCAD, GeCAD Fast Commander, GFC, RAV Reliable AntiVirus, A.V.A.C., RAlert, RAVUtil, RAVeSpy, R.A.C.E., RAX, WisDOM.

The following are registered trademarks of their respective owners: Times New Roman, Courier, Arial, IBM, OS/2, Intel, Microsoft, MS-DOS, Windows, Windows95, Windows98, WindowsNT, QEMM, F-PROT, TBSCAN, Viruscan, TBAV, DSAV, DrWEB, AVP, MSAV, MS Office, MS Word, MS Access, MS Excel, MS Visual Basic, NetWare.

Terms and conditions of the License Agreement

RAV Reliable AntiVirus is a registered trademark of GeCAD Software S.R.L. (hereafter referred as "GeCAD Software"). All the products from the **RAV AntiVirus** family are offered to our clients under the terms and conditions of the License Agreement accompanying all the products of GeCAD Software.

Before installing or using The Software, please read carefully this License Agreement, because it represents a legal agreement between you and GeCAD Software for the software product you are installing, which includes the software itself and the related documentation. By installing or otherwise using the software, you accept all the terms and conditions of this agreement. If you do not accept the terms of this agreement, you do not have the rights to install or otherwise use The Software.

On the distribution CD-ROM you may find other programs, in addition to the one you have bought. These programs are offered for evaluation only and are the object of separate terms of license. These terms are included in the Evaluation license section of the License Agreement.

CONTENTS

NAME 5

SYNOPSIS 5

Structure 5

What is new in ravmd 8.4.0 6

Definitions 6

SECTION DESCRIPTIONS 8

Regular Expression Declaration section 9

Syntax 9

Action Definitions section 10

Syntax 10

FAQ 1: Scanning existing mails 12

FAQ 2: Separate domain file 12

FAQ 3: Changing the contents of the warn.txt file 12

FAQ 4: Modified message is not appearing in the warn.txt file 13

Warning Mails Message Declarations section 14

Syntax 14

FAQ 5: Warning messages not sent 16

FAQ 6: Omitting the SUBJECT from warning mail 16

FAQ 7: Changing the warning mail message 17

FAQ 8: Stopping update notifications 17

Anti-spam Definitions section 18

How does it work 18

Group Declarations 21

What is a group 21

The [`global`] group 21

Defining other groups 21

THE _include DIRECTIVE 21

How do I configure groups 22

How do I configure separate anti-spam options for my groups 22

FAQ 9: Creating different rules for a domain 22

FAQ 10: Example on how to set the domains and the IP addresses 23

FAQ 11: Global group configuration 23

FAQ 12: Excluding one particular account 23

The Advanced Content Filtering feature 25

What is this 25

How does it work 25

FAQ 13: Rejecting double extension files 25

FAQ 14: Denying certain types of attachments 26

FAQ 15: Example of content filtering for mail subject 26

FAQ 16: Example of content filtering for mail body 27

Explaining the parameters 29

Domain parameters 29

Group members 29

Engine actions 31

Engine parameters **32**
Warning messages **34**
Specifying the sender of the warning mails **40**
Anti-spam parameters **42**
Real-time Blackhole List parameters **44**
White/Black List parameters **46**
Miscellaneous parameters **47**
RAV logging system **51**
Group-specific parameters **53**
Embedded messages **60**
BUGS **61**
Index of Group Parameters **62**

NAME

ravmd.conf - the configuration file for **ravmd** daemon

SYNOPSIS

This is the configuration file for **ravmd** daemon, **ravmd.conf**, and it contains info on the run-time configuration for RAV AntiVirus mail-scanning daemon. After installation, **ravmd.conf** resides in your `/etc/opt/rav/` directory.

This document is part of the *User's Guide for Mail Servers* and should be used only in connection with this User's Guide.

Other documents containing valuable information are also available:

- The *Install*, *UnInstall* and *ReadMe* files included in the `tar` or `tar.gz` files containing the setup programs;
- The configuration manuals for ravmd filter clients included in the *User's Guide for Mail Servers* and in the `tar` or `tar.gz` files containing the setup programs;
- RAV Engine version 8.9 - Release Notes (you can find it in:

  ftp://ftp.ravantivirus.com/pub/rav/engine/common/

Structure

This configuration file consists of:

- Description of four declaration sections (the *Regular Expression Declaration* section, the *Action Definitions* section, the *Warning Mails Message Declarations* section and the *Anti-spam definitions* section),
- Group declarations, and
- Explanation of possible parameters.

The info included in this part of the documentation is for reference purposes. Other valuable sources of information are also available. Information pertaining to the installing and un-installing processes, for instance, is included in the *Install* and *Uninstall* files provided in the *tar.gz* files containing the installation kit for the product you are using. Additional information about **RAV AntiVirus for Mail Servers** (hardware and software requirements, list of installed files, instructions for configuration and updating) is provided in the *ReadMe* file, included in the installation kit.

When presenting the possible parameters in **ravmd**, besides the default values and examples for these parameters *Frequently Asked Questions* are also provided, to help you better understand the functions of **ravmd** and know all the options available in it.

For the users of previous version of **RAV AntiVirus for Mail Servers**, please read the *What is new in ravmd 8.4.0* section below.

What is new in ravmd 8.4.0

The main enhancement brought by **ravmd** version 8.4.0 is the integration of the anti-spam module. The anti-spam module is actually a combination between the new bulk mail module and features already implemented in ravmd, i.e. **Real-time Blackhole List (RBL)** and the **White/Black List (WBL).**

Correctly configured, this combination should be a winner in the fight against unsolicited mails.

Note:	You can find more details on the new features of **ravmd** version 8.4.0 in the *Release Notes for ravmd 8.4.0* and in the *User's Guide for RAV AntiVirus for Mail Servers* (version 1.41 or later of this document). You can find these documents on http://www.ravantivirus.com/.

The new anti-spam module included in **ravmd** version 8.4.0 triggers important changes in this document.

- A new section called *Anti-spam definitions was* added after the *Warning Mails Message declaration* section, explaining how the new anti-spam module works.
- The parameters specific to the anti-spam module are detailed in the *Anti-spam parameters* sub-section under the *Explaining the parameters* section.
- New actions are available for bulk mails: save, embed, add_header, add_subject, deliver, reject, discard.
- New group-specific parameter: **anti-spam_configuration**.

Among other changes to be noted in **ravmd** version 8.4.0:

- the folder structure for **ravmd** has changed. For instance, the **ravmd.conf, actions** files and so on are to be found in the `/etc/opt/rav/` folder, instead of the older `/usr/local/rav8/etc` location. The new anti-spam file is also to be found in the `/etc/opt/rav/` folder.
- the format (font, paragraph) of this document has changed since its previous version.

Definitions

Some concepts frequently used in the programming world might have different understandings for the purpose of this manual. Here you can find these terms and the meanings they are given in this document:

- A **variable** is a place where we can store data;
- A **string** is a sequence of characters ending with a new line or delimited by quotes;
- A **boolean** is one of the keywords: **yes/no**;

- An **enumeration** is a sequence of words separated by commas;
- A **regexp** is a POSIX regular expression.
- A **group** is a category of users (senders or receivers) that have different e-mail addresses or and/or domains, but share the same action parameters for **ravmd**.
- A **macro** is a stored template of instructions to be replaced with actual values by **ravmd**;
- A **commented** line is a line beginning with a hash (#) character. All commented lines are ignored. All the lines containing only white spaces are also ignored;
- A **default** is a predefined value for one parameter. If that **parameter** is missing from **ravmd.conf** then it is considered to have that **default** value.

The values following the '=' sign in *parameters* may be: a **string**, a **boolean**, a **regexp** or an **enumeration**.

Note: The section and parameter names are **not** case sensitive.

SECTION DESCRIPTIONS

Four different sections are included in the **ravmd.conf** file: the *Regular Expression Declarations* section, the *Action Definitions* section, the *Warning Mails Message Declarations* section and the *Anti-spam Configuration* section. These sections are presented below in the following format:

- Short description;
- Keyword representing the beginning of the section;
- Syntax;
- Example.

Some *Frequently Asked Questions* (FAQs) are also provided in connection with some relevant aspects of these sections. The *Frequently Asked Questions* are meant to help you with the practical aspects of using **RAV AntiVirus for Mail Servers**.

The *Frequently Asked Questions* were selected from the questions asked by users of **RAV AntiVirus for Mail Servers** on the mailing lists available for our products.

You can subscribe to these discussion lists by visiting our website http://www.ravantivirus.com/ (*RAV Discussion Lists* section) or by sending an empty e-mail message to: listname-subscribe@lists.ravantivirus.com. For more *Frequently Asked Questions*, please visit our website at http://www.ravantivirus.com/ or consult our *Knowledge Base* available at the following address: http://www.ravantivirus.com/kb.

Regular Expression Declaration section

In the *Regular Expression Declaration* section you can define all the regular expressions you will use for the content filtering feature. This section can appear anywhere in the configuration file, as long as it is placed before the group definitions.

This section begins with the keyword:

```
_define_regular_expressions
```

Syntax

variable = string

or

variable = regexp

Example 1:

```
for_subject_filter = I love you
```

This regular expression defines a filter for all the e-mails including the string “I love you” in the **Subject**.

Example 2:

```
file_regexp = .*\.exe
```

This regular expression defines a filter for all mail messages having **.exe** attachments.

Action Definitions section

In the *Action Definitions* section you can define the actions you want to be used by **ravmd**. The *Action Definitions* section can appear anywhere in the configuration file as long as it is placed before group definitions.

This section starts with the keyword: `_define_actions.`

Syntax

variable = enumeration

Depending on the scanning status of the mail message (infected, suspicious, subject/attachment/content filter match), the variable will be associated with some different actions (*clean, move, copy, delete, rename, ignore, reject, discard*). The enumeration contains one or multiple actions separated by a comma. The actions from this enumeration are executed by **ravmd** in the order you specify.

Depending on the scanning status, the following valid actions are supported:

- for infected files: clean, move, copy, delete, rename, ignore, reject, discard.
- *for suspicious files*: move, copy, delete, rename, ignore, reject, discard.
- *for e-mail files matching the subject filter*: copy, ignore, reject, discard.
- *for files matching the attachment filter*: move, copy, delete, rename, ignore, reject, discard.
- *for e-mail files matching the content filter*: move, copy, delete, ignore, reject, discard.
- *for mails tagged as spam:* save, embed, add_header, add_subject, deliver, reject, discard.

Note 1:	If the last action you define for infected mails is not one of the following: **Discard, Reject** or **Ignore**, the **Reject** action is automatically applied.
Note 2:	If the last action you define for bulk mails is not one of the following: **Deliver, Reject** or **Discard**, the **Deliver** action is automatically applied.

For more information, please refer to the `actions` file included in your setup program.

The following table includes a description of all the actions available in **ravmd** (first column), a description of these actions (second column) and a listing of circumstances when each specific action is available (third column), depending on the scanning status:

Action	Description	Scanning status
Clean	Asks RAV AntiVirus to clean the infected file.	Infect
Move	Asks RAV AntiVirus to move the file to quarantine (equivalent to `Copy` + `Delete` actions).	Infect, suspicious, attach match, content match
Copy	RAV AntiVirus is copying the file to quarantine.	Infect, suspicious, subject match, attach match, content match
Delete	RAV AntiVirus is deleting the file and replacing it with a new file automatically generated. The file's name is `warn.txt` and it contains the following text: "*RAV AntiVirus has deleted this file because it contained dangerous code.*" This text is customisable (please see FAQ no 1 for details). Note that **ravmd** doesn't change the mail file size because of some protocols (like IMAP) may request the mail size first and then the mail body. So, the `warn.txt` file will be filled with spaces to fit the original file length.	Infect, suspicious, attach match, content match
Rename	The file will be renamed using the `rename_ext` extension specified in the configuration file.	Infect, suspicious, attach match
Ignore	The file is ignored, no action is taken and the e-mail is delivered.	Infect, suspicious, subject match, attach match, content match
Reject	The e-mail is rejected; it will not be delivered to any of its recipients, but will be bounced back to the sender.	Infect, suspicious, subject filter, attach match, content match, bulk mail
Discard	The e-mail is discarded; it will not be delivered to any of its recipients and will not be bounced back to the sender.	Infect, suspicious, subject filter, attach match, content match, bulk mail
Deliver	The original mail is delivered to its recipients even after being tagged as Spam.	Bulk mail
Save	The bulk mail is saved in the quarantine directory.	Bulk mail
Embed	Creates an embedded mail containing a custom message and having attached the bulk mail tagged as Spam.	Bulk mail
add_header	Add a custom extra header to the bulk mail tagged as Spam.	Bulk mail
add_subject	Affix a custom string in the Subject field of a bulk mail tagged as Spam.	Bulk mail

Table 1: Actions available in **ravmd.**

The following *Frequently Asked Questions* will eventually help you better understand the characteristics of the actions available in **ravmd**.

FAQ 1: Scanning existing mails

Question: I just installed **RAV AntiVirus for Mail Servers**. How can I scan e-mail messages existing prior to this installation?

Answer: If you wish to scan e-mail messages existing prior to the installation of **ravmd**, do the following:

```
/etc/opt/rav/bin/ravav -AM --smart --report=/tmp/ravreport.txt
[path to mail accounts]
```

This setting results in scanning a mailbox (with *ignore* as default action) and delivering a report (`ravreport.txt`) in the `tmp` directory.

FAQ 2: Separate domain file

Question: Is it possible to put all scannable domains in a separate hash or normal file?

Answer: Create the `/etc/opt/rav/domains.list` file and/or operate the following changes:

In the **ravmd.conf** file, replace the “domains to scan (separate them with ',')” section with:

```
domain1.com, domain2.com, domain3.net, domain4.com,
domain5.org
```

Then use:

```
domain = _include /etc/opt/rav/domains.list
```

FAQ 3: Changing the contents of the warn.txt file

Question: Recipients who get a virus-infected email have their attachments replaced with a `warn.txt` file. How can I change the contents of the `warn.txt` file?

Answer: Please edit the `/etc/opt/rav/languages/English` file and customize the value of:

```
warn_txt_msg_english = "Your message".
```

Then restart **ravmd** with:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

Please note that `warn.txt` has exactly the size of the deleted attachment and in some cases, because of the small size of the attachment, you will not be able to view your complete customized message.

FAQ 4: Modified message is not appearing in the warn.txt file

Question: I have modified the `warn.txt` message in my configuration but the modified message is not appearing in the `warn.txt` file. Why is that happening?

Answer: Either you did not correctly change the `warn.txt` message or the infected attachment file is to small to allow the entire `warn.txt` message to be displayed (see the answer to FAQ 3 above). The steps to be followed are:

- In `/etc/opt/rav/languages/english` use:

```
warn_txt_msg_english = "Your message here"

-in /etc/opt/rav/languages/english.equiv

warn_txt_msg = warn_txt_msg_english
```

- Restart **ravmd** with:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

Warning Mails Message Declarations section

In the *Warning Mails Message Declarations* section you can define the subjects and the messages for the warning mails that will be sent to those interested. The *Warning Mails Message Declarations* section can be declared anywhere in the configuration file as long as it is placed before the group definitions.

This section starts with the keyword:

```
_define_strings
```

Syntax

variable = string

The string defining one warning mail should give the user some basic information such as: what file has caused the warning, who sent that file and whom was it intended for, what is the warning about, what action was taken by **ravmd** and so on. One example of good warning message would be the following:

> "*That* file coming from *that* sender and addressed to *this* user is infected with *this* virus. The action taken by **ravmd** was *this*."

Of course, not all the warning mails are about virus-infected files. There are some other situations when you might want to be alerted by **ravmd**. For instance, you might want **ravmd** to alert you when an e-mail containing one specific type of attachment has arrived on your mail server or when one of your users is trying to transmit a confidential document. Of course, in these cases the warning mail should change accordingly to that specific situation. However, in all the cases some information is *always* included (the name of the file causing the alert, the sender, the action, etc.).

This information always included in a warning mail is based on *macros*. One macro is (for the purpose of this document) a stored template of instructions containing one piece on info from the string representing the warning mail. `FILE_NAME`, for instance, is an example of macro containing the name of the file causing the alert you're receiving. **ravmd** will replace this macro with the actual name of the trouble-making file.

The string that the warning message will be created from can contain the following macros:

`FILE_NAME`	The full path to the scanned file and its name.
`ATTACH_NAME`	The name of the scanned attachment.
`VIRUS_NAME`	The name of the virus discovered by **ravmd**.
`FROM_USER`	The e-mail address of the sender.

ON_HOST	The host **ravmd** is running on.
TO_USER(S)	The e-mail addresses of the receivers.
SUBJECT	The mail's subject.
QUARANTINE_NAME	The name of the file saved to RAV Quarantine using the *Move* or *Copy* action.
SAVED_FILE_NAME	The name of the mail file saved to quarantine when save_infected=yes and/or save_suspicious=yes.
HEADER_RECEIVED	This macro is replaced with all the **Received:** lines from the received e-mail's header (if this information exists). This helps you finding the infected machine more easily.
HEADER	This macro is replaced with the entire received e-mail's header.

These macros are combined in one warning string that might look like this:

```
"The file ATTACH_NAME attached to mail (with subject:SUBJECT)
sent by FROM_USER to TO_USER(S)is infected with virus:
VIRUS_NAME".
```

Note: The macros **FROM_USER** and **VIRUS_NAME** can be used in the notification e-mail subject string too.

In this case, one attachment file is virus-infected. Info is provided about the attachment's name, the subject of the e-mail containing the infected attachment (this is an optional info), the sender's name, the addressees' names and the virus name.

To eliminate any mystification, here are some lines about the differences between the most confusing two pairs of macros:

1. QUARANTINE_NAME *vs.* SAVED_FILE_NAME

   QUARANTINE_NAME represents the name of the file saved to Quarantine as a result of using the *Move* or *Copy* action. The file is saved by **ravmd** in the Quarantine directory with the extension .qto. The files are encrypted, but starting with version 8.3.3 **ravav** can decrypt them.

   SAVED_FILE_NAME represents the name of the file saved in the Quarantine folder as a result of using the parameters save_infected=yes and save_suspicious=yes. The file is saved by **ravmd** in the Quarantine directory in the **UNIXTIME-RAV{PID_OF_RAV_FILTER}** format, where RAV_FILTER is ravexim, ravpostfix, etc. For sendmail, the file is saved in the **UNIXTIME-df{MESSAGE-ID}** format, and for sendmail-milter the format is **UNIXTIME-RAV{MESSAGE-ID}**.

2. `HEADER_RECEIVED` *vs.* `HEADER`

   The `HEADER` macro is replaced with the entire e-mail header, while `HEADER_RECEIVED` is replaced only by the lines beginning with "**Received:**". Therefore, the lines *Message-Id*, *X-Sender* and *X-Mailer*, for instance, are only included in the HEADER macro is used, but not if the `HEADER_RECEIVED` macro is used.

The following *Frequently Asked Questions* will hopefully help you better understanding this very interesting feature of **ravmd**.

FAQ 5: Warning messages not sent

Question: Why don't I get warning messages when viruses are found?

Answer: You probably did not configure correctly the **ravms** mail account. Here are the excerpts from the documentation that should help you:

"Default values:

```
ravms_name = ravms

on_host = the official name returned by the gethostbyname()
function

smtp_server = same as host (IP address)

smtp_port = 25

ravms_full_name = displayed.name (RFC822)
```

Default values are provided and they will probably work. Define these fields only if warning mails are sent when a virus is found or if you want to use a different account instead of **ravms**. Specify **smtp_server** IP address only if that machine is behind a firewall and **ravmd** can't get its host name".

FAQ 6: Omitting the SUBJECT from warning mail

Question: Is it possible to define that `SUBJECT` should be omitted from the warning for specific viruses? Some of the viruses disclose potentially sensitive information from the victim's hard drive, and puts it in the subject of the email.

Answer: In `/etc/opt/rav/languages/english`, edit the line:

```
infected_m_english = "The file ATTACH_NAME attached to mail
(with subject:SUBJECT) sent by FROM_USER to TO_USER(S)is
infected with virus: VIRUS_NAME."
```

and remove `"with subject: SUBJECT"`.

The subject will not be displayed anymore in your warning mails. Please note that this solution will be applied to all e-mails, not just for specific viruses.

FAQ 7: Changing the warning mail message

Question: When a mail with `.exe`, `.com` or other attachments of this type is sent, a warning message is also sent, which is OK, but it says the file is infected with the virus: `UNAUTHORIZED_MAIL_ATTACHMENT`, which seems to scare the average user. Can the message be modified?

Answer: Yes, the message can be modified:

- In `/etc/opt/rav/languages/your_language` define a variable, for example:

  ```
  infected_attach = "The file ATTACH_NAME attached to mail
  (with subject:SUBJECT) sent by FROM_USER to TO_USER(S) has
  an unauthorized file extension"
  ```

- Then in the `/etc/opt/rav/languages/your_language.equiv` file, use (in the `_attachment_filter_warning_messages` section):

  ```
  infected_msg = infected_attach
  ```

  instead of

  ```
  infected_msg = infected_m_english
  ```

- Restart **ravmd** with:

  ```
  kill -HUP `cat /var/opt/rav/run/ravmd.pid`
  ```

  and you're done.

FAQ 8: Stopping update notifications

Question: How do I stop update notifications?

Answer: To stop the update notifications, in `/etc/opt/rav/bin/ravupdate.sh` change this:

```
#Specify when should the administrator be notified by
the update process

#VERBOSE="silent"

VERBOSE="noisy"

#VERBOSE="errors"
```

to:

```
#Specify when should the administrator be notified by
the update process

#VERBOSE="silent"

#VERBOSE="noisy"

VERBOSE="errors"
```

Anti-spam Definitions section

The **Anti-spam** module is available in **RAV AntiVirus for Mail Servers** starting with version 8.4.0. This feature is designed to protect the users of unsolicited mail messages.

The parameters pertaining to the anti-spam module of **RAV AntiVirus for Mail Servers** are described in the *Anti-spam parameters* sub-section of this document.

A default configuration is included in the `/etc/opt/rav/antispam` file.

The Anti-spam functionality is based on the new bulk mail module, working in close co-operation with older features like **Real-time Blackhole List** (RBL) or **White-Black List** (WBL), available in **ravmd** since version 8.3.3.

How does it work

When a mail message reaches a RAV-protected mail server, it is first confronted with the **White/Black List (WBL)**. This is a static configurable list that any system administrator can use for specifying the mail addresses from which he wants to automatically *accept* messages (**Static White List**) or the mail addresses from which he wants to automatically *reject* messages (**Static Black List**).

If the mail just received by the RAV-protected mail server comes from an address found in the **Static White List**, then the search in the **RBL** is not executed anymore and **ravmd** jumps directly to the scanning process for the respective mail. Also, the anti-spam module is not used for the mail coming from addresses found in the **Static White List**. If the mail just received by the RAV-protected mail server comes from an address in the **Static Black List**, the mail is automatically rejected.

If the address is not found in the **White/Black List (WBL)**, the mail is confronted against the **Real-time Blackhole List (RBL)**. This is a dynamic list (`rbl_site`) with sites containing listings of known spammers. If any of the IP addresses from the mail's header is listed on one of the websites defined in the `rbl_site`, the mail is automatically rejected. If no IP address from the mail's header is found in the `rbl_site` list or the **use_rbl** parameter is set to **No**, **ravmd** jumps to the scanning process for the respective mail. The system administrator can of course customize **ravmd** not to use **WBL** or **RBL** feature. In this case, **ravmd** scans directly the received mail, using the options defined for the corresponding group.

Assuming the mail is passing OK through the virus-scanning process, the anti-spam search is executed. **ravmd** is looking for patterns known to be specific to spammers on the **Header** and **Body** levels of the mail message. Depending on its specifics, the mail message is classified in one of the following *accuracy levels*: **Low**, **Medium**, **High** and **Very High**.

A „low” accuracy level means that **ravmd** reports the suspected mail message as spam even if only few spam patterns exist. This means that you might get some „false alarms”, but no spam messages will pass by **ravmd**.

A „medium” accuracy level means that **ravmd** reports the suspected mail message as spam if several spam patterns exist. This means that the number of „false alarms” is lower than in case of LOW ACCURACY SPAM, but several spam messages might pass by **ravmd**.

A „high” accuracy level means that **ravmd** reports the suspected mail message as spam if more spam patterns exist. This means that the number of „false alarms” is low, but some spam messages might pass by **ravmd**.

A “very high” accuracy level means that **ravmd** reports the suspected mail message as spam only if lots of spam patterns exist. Therefore, it is unlikely for **ravmd** to report false alarms, but more spam messages will probably pass by **ravmd**.

Below you can find a scheme detailing the flow described above.

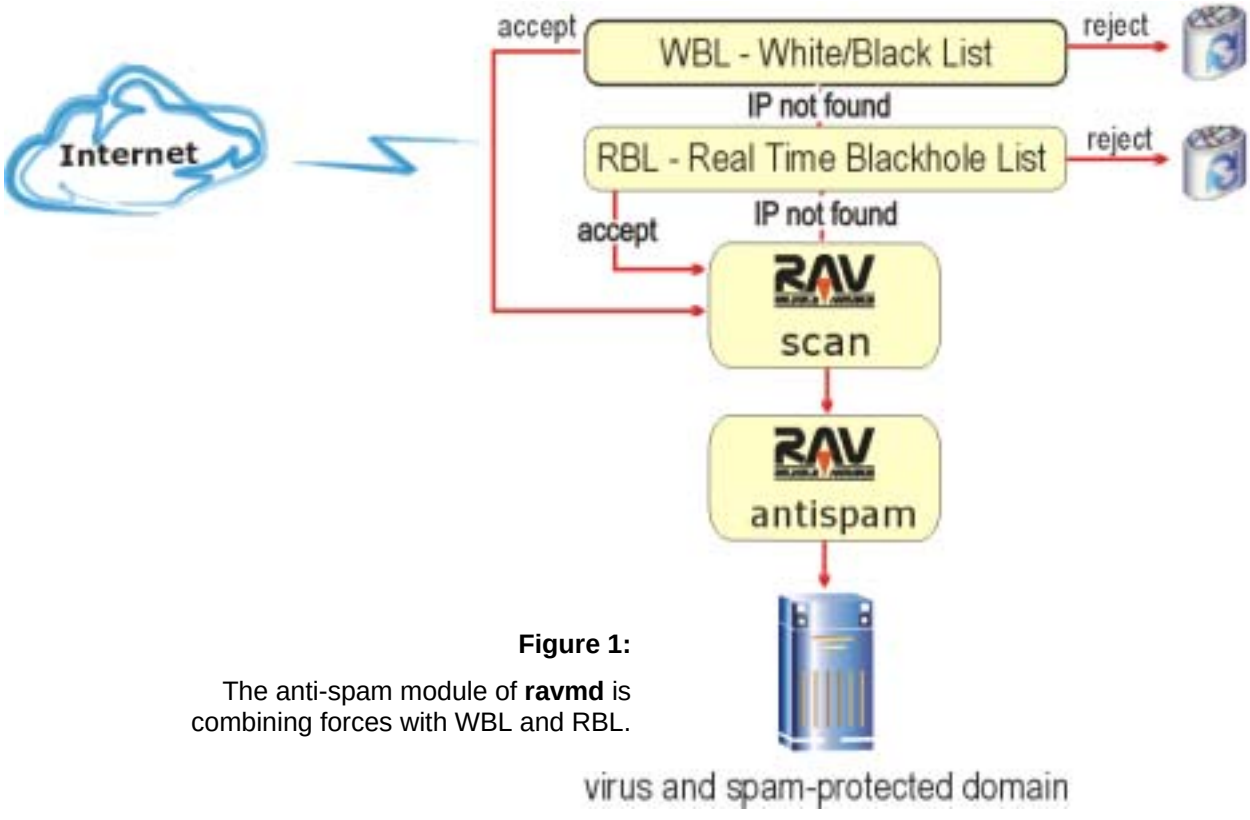

Figure 1:

The anti-spam module of **ravmd** is combining forces with WBL and RBL.

For each of the four accuracy levels specified above you can configure different strings and separate actions to be used/taken by **ravmd**. The actions specific for the Anti-spam module of **ravmd** are:

- **save** (the mail is saved in the **Quarantine** folder, for further analysis, before any other action is executed on the respective mail)**,**

- **embed** (the original mail is sent as embedded mail)**,**
- **add_header** (a header is added to the original mail)**,**
- **add_subject** (a user-defined variable is affixed in the **Subject** field)**,**
- **discard/reject/deliver** (the mail is discarded/rejected/delivered).

Here is the checklist to be followed when using the Anti-spam module of **ravmd**:

- define in **ravmd.conf** (after the **Regular Expression Declarations**, **Action Definitions** and **Warning messages** sections and before the **Group configuration** section) the name of the section, flanked by two "@" symbols (i.e. `@bulk_detection_low@`);
- define accuracy level. This should be one of the following four keywords: **accuracy_low, accuracy_medium, accuracy_high, accuracy_very_high**;
- define the strings that will be used by the **extra_header**, **extra_subject** and **embedded_msg** parameters;
- define the actions to be executed for each of the four bulk detection levels;
- define the path for the Quarantine folder.

Important:	If no action is defined for one specific accuracy level, **ravmd** will automatically assume the actions defined for the level having the immediate lower priority.

Here is one example:

```
@bulk_high_precision@

accuracy_high

extra_header = bulk_header_high_english

extra_subject = bulk_subject_high_english

embedded_msg = bulk_embedded_high_english

actions = bulk_actions_high

quarantine = /var/opt/rav/bulk
```

Please refer to the corresponding parameters in the *Explaining the parameters* section of this document for more details. The `bulk_header_high_english`, `bulk_subject_high_english` and `bulk_embedded_high_english` (and the other similar parameters) are to be defined in the selected language files (`/etc/opt/rav/languages/eng`, for instance) and the `bulk_actions_high` parameter - in the `_define_actions` section.

Group Declarations

What is a group

A group is a category of users (senders or receivers) that have different e-mail addresses or and/or domains, but share the same action parameters for **ravmd**. Using the group feature of **ravmd**, you can use the same characteristics (for the scanning engine or the update process, for instance) and the same actions (filters, warning mails and so on) for users having different e-mail addresses and mail domains. This group structure is very useful in case you use **ravmd** in a network with hundreds of mailboxes.

In **ravmd** you have one default group, called [`global`], containing at the beginning all the users and mail domains protected by **RAV AntiVirus for Mail Servers**. When you define a new group, its members are taken away from the [`global`] group and included in the new group, for which you have to define new group-specific options.

The [`global`] group

This is the default group, which contains all the users and mail domains that are not defined in the other groups. The [`global`] group MUST NOT contain any member declaration.

Defining other groups

A new group definition begins with the group name written between "`[]`". The group definition must be followed by the member declarations (it is mandatory that the members are declared before any other parameters) and the group options.

THE _include DIRECTIVE

In order to keep the configuration file more readable you can use the **_include** directive to insert other files in the main one. This can be very useful if you have a large number of groups. Defining them on a single configuration file will make the future maintenance difficult. Instead of using a single large configuration file, you can split it in more small files.

If you want to add a new group called [`mygroup`] all you have to do is append a line to the main configuration file:

```
_include /etc/opt/rav/groups/mygroup_file
```

and define that group in the `mygroup_file`:

```
[mygroup]

sender = user_1@domain.com
```

The following *Frequently Asked Questions* are providing the answers for some of the confusions of the existing users of **RAV AntiVirus for Mail Servers** in aspects pertaining to group configuration.

How do I configure groups

The configuration file must include the [global] group, containing the default options for all mail scanning processes. Besides the [global] group, you can customize **ravmd** by creating additional groups, with different configurations.

If no value is specified for one parameter in the configuration file for one group, **ravmd** will use the *default value* for that paramater. If no *default value* is defined, **ravmd** will use the value specified in the [global] group for that parameter, with the following exceptions:

filter_subject, filter_attachment, filter_content, warn_sender, warn_receivers, warn_admin, do_not_warn, do_not_show, admin_addr, do_not_scan, advertising_msg, wbl_reject, wbl_accept, use_rbl, embed_clean_mail, embed_cleaned_mail, embed_unclean_mail, use_embedded_msg, use_embedded_warning, antispam_configuration.

How do I configure separate anti-spam options for my groups

If you intend to use the anti-spam feature of **ravmd**, you should specify what *anti-spam configuration* you want to use for the targeted groups. You do that using the antispam_configuration parameter, described in the *Explaining parameters* section, under *Group-specific parameters* sub-section.

FAQ 9: Creating different rules for a domain

Question: How do I create different rules for a domain?

Answer: You can define different rules for a domain by creating a group in which you specify from_host and/or to_host. Then specify the actions that **ravmd** can perform for the newly created domain. Here are the steps you should follow for creating different rules for a domain:

- In **ravmd.conf**, in the section _define_actions use:

```
act_for_my_group = clean, delete, ..
```

- At the end of ravmd.conf define the group:

```
[my_domain]

_include /etc/opt/rav/groups/my_domain
```

- Create the file /etc/opt/rav/groups/my_domain and edit it:

```
#from_host=a.com

to_host=a.com

infected_actions=act_for_my_group
```

FAQ 10: Example on how to set the domains and the IP addresses

Question: I need an example on how to set the domains and the IP addresses.

Answer: In `/etc/opt/rav/ravmd.conf`, in the [`global`] section use:

```
domain = localhost, your.host.com
```

In order to receive the warning mails, in `/etc/opt/rav/groups/global` use:

```
ravms_name = ravms

on_host = your.host.com

smtp_server = ip.address.of.host

ravms_full_name = displayed.name (RFC822)
```

FAQ 11: Global group configuration

Question: How is the [`global`] group configured?

Answer: Please define in `/etc/opt/rav/groups/global` the following options:

```
ravms_name=ravms

on_host=your.host.com

smtp_server=127.0.0.1 (IP address)

smtp_port=25

ravms_full_name = displayed.name (RFC822)
```

Then start **ravmd** with `/etc/init.d/ravmail start`.

Note:	In `/etc/host` you should have defined at least: `127.0.0.1 localhost.localdomain localhost`

FAQ 12: Excluding one particular account

Question: I need to exclude one account in particular as I receive a daily email of over 60 MB in size and I don't want RAV to try to process that file. What do I get it done?

Answer: You can create a group with that sender and choose not to scan that mail. Please make the following changes:

In `/etc/opt/rav/ravmd.conf`, add at the end of the file:

```
[group1]

_include /etc/opt/rav/groups/group1
```

Then create the file `/etc/opt/rav/groups/group1` and edit it:

```
sender=sender@host.com

do_not_scan=yes
```

Then restart **ravmd** with:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

The Advanced Content Filtering feature

What is this

The Advanced Content Filtering is a very powerful feature of **ravmd** that can help you configure the product to answer your specific needs. Using this feature, you can define filters on different levels (subject, body, attachment contents or attachment file name). You can afterwards define specific actions for the mail messages matching the rules you specify. For instance, you can instruct **ravmd** to deny incoming mail messages containing one specific string in the **Subject** field (i.e. “I love you”), deny outgoing mail messages containg user-specified strings (i.e. “confidential”, “balance sheet”) and deny incoming/outgoing mail containing attachment with user-specified file types/names.

How does it work

The *Content Filtering* module of **ravmd** is using:

- **POSIX regular expressions** for searches executed on the *Subject* and *Attachment file names* levels;
- **User-defined strings** for searches executed on the *Body* level (this starting with version 8.4.0 of **ravmd**; previous version of **ravmd** have used POSIX regular expressions on the Body level).

The rules you define are processed in the following order:

- Subject,
- File names,
- Body.

If you define more rules for one single component, the rules will be processed in the order you are defining them.

Here are some *Frequently Asked Questions* users of **RAV AntiVirus for Mail Servers** have asked about the *Advanced Content Filtering* feature of **ravmd**. You can even find an example of configuring a content filter for mail messages containing one specific string.

FAQ 13: Rejecting double extension files

Question: Has anyone setup the configuration file to reject any attachment with a double extension?

Answer: Please use in **ravmd.conf**:

```
var_regexp = .*/..*/..*

var_action = reject
```

In [global]:

```
filter_attachment var_regexp var_action
```

Please note that this restrictive action will also filter the tar.gz files, for instance.

FAQ 14: Denying certain types of attachments

Question: How can I deny certain types of attachments?

Answer: The content filtering feature of **RAV Antivirus for Mail Servers** can help you if you want to deny a certain type of attachment from entering your company via email. The content filtering module of RAV AntiVirus uses POSIX regular expressions to find a pattern in *subject* and *attached file names* and user-defined strings to find a pattern in *message body* (and attachment contents). The rules are processed in the following order: **subject, attachments, body**. If you define more rules for the same component, they will be processed in the same order they are specified.

Here is an example for how to deny **.exe** attachment files:

In the define_regular_expressions section from /etc/opt/rav/etc/ravmd.conf:

```
define file_regexp = .*\.exe
```

In the _define_actions paragraph from /etc/opt/rav/ravmd.conf define:

```
file_action = delete, reject
```

In the /etc/opt/rav/groups/global file define (wherever inside the file):

```
filter_attachment file_regexp file_action
```

To add other types of attachments separate them with a | symbol on the regexp definition.

Example:

```
file_regexp = .*\.((vbs)|(vbe)|(js)|(exe)|(com)|(pif)
|(lnk)|(scr)|(bat)|(shs)|(sh))
```

As is the case for each change in the RAV configuration files (**ravmd.conf, global, english, english.equiv**), you must restart **ravmd**:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

FAQ 15: Example of content filtering for mail subject

Question: Can I have an example of content filtering for mail messages containing one defined expression in the **Subject** field?

Answer: Yes, you can. Here you can find how to create a content filtering rule for mail messages containing the expression "xxx" in the **Subject** field.

Please note that this feature is not available in **RAV AntiVirus for Mail Servers** running on Sendmail if the mail server is not Milter-enabled.

The following options are available for the mail messages matching the content filter: **move, copy, delete, ignore, reject, discard**.

For this scenario, the e-mail will be allowed to pass **RAV AntiVirus for Mail Servers**. The sender and the receiver will not receive warnings, but a notification will be sent to the administrator.

- In the `_define_regular_expressions` section from **ravmd.conf**, define the expression `xxx`:

```
subjxxx_regexp = xxx
```

- Define the action for `xxx` in the `_define_actions` section from **ravmd.conf**:

```
define xxxsubj_action = ignore
```

  This will allow the `xxx` mail to pass the RAV content filter.

- To add other words, separate them with a | symbol on the `regexp` definition.

  Example:

```
subjxxx_regexp = (word1)|(word2)|(word3)
```

- Edit the file `/etc/opt/rav/groups/global`.
- Specify the administrator's e-mail address for alerting him that an e-mail matching an expression from content filter has been entering in the company:

```
admin_addr=administrator@ravantivirus.com
```

- Activate the content filter by adding the following line:

```
filter_subject subjxxx_regexp xxxsubj_act
```

- As is the case for each change in the RAV configuration files (**ravmd.conf, global, english, english.equiv**), you must restart **ravmd**:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

FAQ 16: Example of content filtering for mail body

Question: Can I have an example of content filtering for mail messages containing one defined expression in the mail bodies?

Answer: Yes, you can. Here you can find how to create a content filtering rule for mail messages containing the following expressions in their bodies: “confidential”, “salaries” and “balance sheet”.

For this scenario, the e-mail will not be allowed to pass **RAV AntiVirus for**

Mail Servers. The sender and the receiver will not receive warnings, but a notification will be sent to the administrator.

- In the `_define_regular_expressions` section from **ravmd.conf**, define the string `confidential`:

```
bodyconfidential_string = confidential
```

- To add other words, separate them with a | symbol on the `string` definition.

Example:

```
bodyconfidential_string = confidential|salaries|balance
sheet
```

- Define the action for `confidential` in the `_define_actions` section from **ravmd.conf**:

```
define bodyconfidential_action = reject
```

This will not allow the mail messages containg to pass the RAV content filter.

- Edit the file `/etc/opt/rav/groups/global`.
- Specify the administrator's e-mail address for alerting him that an e-mail matching an expression from content filter has been entering in the company:

```
admin_addr=administrator@ravantivirus.com
```

- Activate the content filter by adding the following line:

```
filter_subject bodyconfidential_string define
bodyconfidential_action
```

- As is the case for each change in the RAV configuration files (**ravmd.conf, global, english, english.equiv**), you must restart **ravmd**:

```
kill -HUP `cat /var/opt/rav/run/ravmd.pid`
```

Explaining the parameters

The parameters included in this configuration file for **ravmd** (**ravmd.conf**) have different functions. Some are used for specifying the domain parameters, other for specifying the group members, the actions for the scanning engine, the warning messages and the sender of warning messages and so one. All the parameters have been grouped below depending on their function.

Some of these parameters are group-specific, meaning that they are different for each defined group. Other parameters are common for different groups and/or are inherited from the [`global`] group. This classification is also important for understanding the way **ravmd** is working.

Domain parameters

domain = enumeration

Description: This parameter specifies the domains scanned by RAV AntiVirus.

Note:	During the evaluation period of 30 days you can set only two domains. Mails for/from other domains are not scanned. They are normally delivered.

This parameter is a global one. It is sufficient to define it in the [`global`] group.

Important:	You **must** specify at least one domain name, or else **ravmd** will not start. Starting with the 8.3.3 version of **ravmd**, the following default groups are provided: `machine.name` and `domain.name`.

Example:

```
domain = mail.domain.com, domain.com

domain = second.domain.net
```

Important:	In the acception of **ravmd** and according to the **License Agreement**, a 'domain' is defined as "*the string which follows the '@' symbol in an e-mail address*".

Group members

sender = enumeration

Description: This parameter is used to specify the e-mail addresses of the senders who will be members in the actual group.

Example:

```
sender = user1@domain1.com, user2@domain1.net

sender = user3@domain2.org
```

receiver = enumeration

Description: This parameter is used to specify the e-mail addresses of the receivers who will be members in the actual group.

Example:

```
receiver = user4@domain4.com, user5@domain4.org

receiver = user5@domain2.org
```

from_host = enumeration

Description: This parameter is used to specify the hosts that are members in the actual group.

Example:

```
from_host = mail.domain1.com, domain1.net

from_host = domain2.org
```

to_host = enumeration

Description: This parameter is used to specify the receiving host names members in the actual group.

Example:

```
to_host = domain3.com, domain3.net

to_host = domain3.com
```

Engine actions

The following parameters are used to define the actions for the scanning engine: **infected_actions, suspicious_actions.** The `infected_actions` parameter helps you define the actions to be performed when an infected object is found, and the `suspicious_actions` parameter - the actions to be performed when a suspicious object is found.

infected_actions = variable

Description: This parameter is used to specify a variable name defined in the `define_actions` section, which contains the actions to be performed when an infected object is found. If this parameter is not defined then the *reject* action is used for the infected e-mails.

Example:

```
infected_actions = act_for_infected_files
```

Where:

```
act_for_infected_files = clean, move, delete, reject, discard
```

In this example, when an infected file is detected, **ravmd** tries first to *clean* that file. If the cleaning action is completed successfully, **ravmd** moves to the next file. If the cleaning action fails, **ravmd** tries to *move* (*copy* to quarantine and *delete* that file from e-mail) the file. If the moving action is completed successfully, **ravmd** moves to the next file. If the moving action fails, **ravmd** tries to *delete* the file, and so on. The *ignore*, *reject* and *discard* actions **always** return success.

Note: RAV AntiVirus appends a *reject* action to the actions enumeration by default.

suspicious_actions = variable

Description: This parameter is used to specify a variable name defined in the `_define_actions` section, which contains the actions to be performed when a suspicious object is found. If this parameter is not defined then the *reject* action is used for the suspicious e-mails.

Example:

```
suspicious_actions = act_for_suspicious_files
```

Where:

```
act_for_suspicious_files = move, rename, delete, ignore
```

As in the previous example, when **ravmd** detects a suspicious file, it tries

successively to *move*, *rename* and *delete* that file. If the move action fails, **ravmd** tries to rename the file (see bellow the description for ***rename_ext***). If the *rename* action fails, **ravmd** tries to *delete* the file. If one of the previous actions is successfully completed, **ravmd** moves to the next file. Eventually, if all of the actions (*move, rename, delete*) fail, **ravmd** will ignore that file.

Engine parameters

use_heuristics = boolean

Description: This parameter is used to control the heuristic methods for detecting new viruses.

Default value: **Yes**.

Example:

```
use_heuristics = no
```

use_cf_inside_embedded_objects = boolean

Description: Use this parameter to specify if **ravmd** will use the content filtering feature for embedded objects (i.e. files included in archives, scripts, OLE objects).

Default value: **Yes**.

Example:

```
use_cf_inside_embedded_objects = no
```

scan_packed_executables = boolean

Description: This parameter is used to control the scanning process for packed executables (i.e. `vvpack`, `ucexe`, `pepack`, etc.)

Default value: **Yes**.

Example:

```
scan_packed_executables = no
```

scan_archives = boolean

Description: This parameter is used to control scanning in archive files, like **ZIP**, **ARJ**, **RAR**, **LHA**, **LHZ**, **ACE**, **CAB**, etc. If the boolean value is set to **Yes**, **ravmd** will scan inside archives. If the boolean value is set to **No**, **ravmd** will not scan inside archives.

Default value: **Yes**.

Example:

```
scan_archives = yes
```

rename_ext = string

Description: This parameter is used to specify the *extension* used to *rename* the infected/suspicious files. This function is necessary for preventing inexperienced users from accessing by accident infected/suspicious files moved to Quarantine.

Default extension is: _**??**

Example:

```
rename_ext = _??
```

smart_scan = boolean

Description: This parameter is used to specify the scanning modes for **ravmd**. The available options are:

- scan all files,
- let RAV decide what files to scan.

Note:	If **smart_scan** *is not defined* then **ravmd** will scan all files. By default, the smart scanning is enabled.

Example:

```
smart_scan = yes
```

Warning messages

The following parameters help you define the messages used to create warning notifications in different circumstances (virus found, subject filter match, attachment filter match, content filter match).

If some warning messages are not specified for the [`global`] group, then **ravmd** uses a default string (`<<not defined>>`). For all the other groups, the warning messages are inherited from the [`global`] group.

_virus_warning_messages

Description: This parameter is used to specify the messages used to create the warning notifications when a virus is found in the e-mail.

_subject_filter_warning_messages

Description: This parameter is used to specify the messages used to create the warning notifications when the content filtering is enabled and the e-mail subject matches one of the defined rules.

_attachment_filter_warning_messages

Description: This parameter is used to specify the messages used to create the warning notifications when the content filtering is enabled and an attached file name matches one of the defined rules.

_content_filter_warning_messages

Description: This parameter is used to specify the messages used to create the warning notifications when the content filtering is enabled and the e-mail body or one of the attachments contains a string matching one of the defined rules.

warning_mail_subj = variable

Description: This parameter is used to specify the subject used in the notification e-mail.

Example:

```
warning_mail_subj = wm_sbj
```

Note:	In the **Warning Mails Message Declarations** section you must specify: `wm_sbj = "RAV AntiVirus scan results."`

infected_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when an *infected* file is detected.

Example:

```
infected_msg = wm_inf_msg
```

Where:

```
wm_inf_msg = "The file ATTACH_NAME attached to mail (with
subject:SUBJECT) sent by FROM_USER to TO_USER(S) is infected
with virus: VIRUS_NAME."
```

suspicious_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when a *suspicious* file is detected.

Example:

```
suspicious_msg = wm_sus_msg
```

Where:

```
wm_sus_msg = "The file ATTACH_NAME attached to mail (with
subject:SUBJECT) sent by FROM_USER to TO_USER(S) contains
suspicious code."
```

ignored_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when an *infected/suspicious* mail file is ignored.

Example:

```
ignored_msg = wm_ign_msg
```

Where:

```
wm_ign_msg = "All defined actions have failed. Do not use this
file."
```

rejected_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by

ravmd when an *infected/suspicious* mail file is rejected.

Example:

```
rejected_msg = wm_rej_msg
```

Where:

```
wm_rej_msg = "The e-mail was rejected because it contains
dangerous code."
```

discarded_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when an *infected/suspicious* mail file is discarded. **Discard** is a new action available starting with **ravmd** version 8.3.3.

The value for **variable** must be declared in the language file and the `language.equiv` file must be included in the group file. So, in the `/etc/opt/rav/languages/english.equiv` file, define:

```
_virus_warning_messages

discarded_msg=discarded_m_english

.......

_subject_filter_warning_messages

discarded_msg=discarded_m_english

.......

_content_filter_warning_messages

discarded_msg=discarded_m_english

.......

_attachment_filter_warning_messages

discarded_msg=discarded_m_english
```

Example:

```
discarded_msg = discarded_m_english
```

Where:

```
discarded_m_english = "This e-mail was discarded. Please
contact your system administrator."
```

cleaned_msg = variable

Description: This parameter is used to specify the *body* of the info e-mail sent by **ravmd** when a file is *cleaned*.

Example:

```
cleaned_msg = clean_ok
```

Where:

```
clean_ok = "The file was successfully cleaned by RAV
AntiVirus."
```

moved_msg = variable

Description: This parameter is used to specify the *body* of the info e-mail sent by **ravmd** when a file is *moved*.

Example:

```
moved_msg = move_ok
```

Where:

```
move_ok = "The file was successfully moved to quarantine with
name: QUARANTINE_NAME."
```

copied_msg = variable

Description: This parameter is used to specify the *body* of the info e-mail sent by **ravmd** when a file is *copied*.

Example:

```
copied_msg = copy_ok
```

Where:

```
copy_ok = "The file was successfully copied to quarantine
with name: QUARANTINE_NAME."
```

deleted_msg = variable

Description: This parameter is used to specify the *body* of the info e-mail sent by **ravmd** when a file is *deleted*.

Example:

```
deleted_msg = delete_ok
```

Where:

```
delete_ok = "The file was successfully deleted by RAV
AntiVirus."
```

renamed_msg = variable

Description: This parameter is used to specify the *body* of the info e-mail sent by **ravmd** when a file is *renamed*.

Example:

```
renamed_msg = rename_ok
```

Where:

```
rename_ok = "The file was successfully renamed by RAV
AntiVirus."
```

saved_inf_msg = variable
saved_sus_msg = variable

Description: This parameter is used to specify the string used in the notification e-mail when the infected/suspicious file is saved to quarantine.

Example:

```
saved_inf_msg = save_ok

saved_sus_msg = save_ok
```

Where:

```
save_ok = "The e-mail file SAVED_FILE_NAME was saved to
quarantine."
```

cannot_clean_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when a file *cannot be cleaned*.

Example:

```
cannot_clean_msg = not_cleaned
```

Where:

```
not_cleaned = "Cannot clean this file."
```

cannot_move_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when a file *cannot be moved*.

Example:

```
cannot_move_msg = not_moved
```

Where:

```
not_moved = "Cannot move this file."
```

cannot_copy_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when a file *cannot be copied*.

Example:

```
cannot_copy_msg = not_copied
```

Where:

```
not_copied = "Cannot copy this file."
```

cannot_delete_msg = variable

Description: This parameter is used to specify *body* of the warning e-mail sent by **ravmd** when a file *cannot be deleted*.

Example:

```
cannot_delete_msg = not_deleted
```

Where:

```
not_deleted = "Cannot delete this file."
```

cannot_rename_msg = variable

Description: This parameter is used to specify the *body* of the warning e-mail sent by **ravmd** when a file *cannot be renamed*.

Example:

```
cannot_rename_msg = not_renamed
```

Where:

```
not_renamed = "Cannot rename this file."
```

cannot_save_inf_msg = variable
cannot_save_sus_msg = variable

Description: This parameter is used to specify the *string* used in the notification e-mail when the infected/suspicious file cannot be saved to quarantine.

Example:

```
cannot_save_inf_msg = not_saved

cannot_save_sus_msg = not_saved
```

Where:

```
not_saved= "The infected e-mail file cannot be saved to
quarantine."
```

Specifying the sender of the warning mails

ravms_name = string
on_host = string
smtp_server = string
smtp_port = number
ravms_full_name = string

Description: Using these parameters you can define the e-mail address for the sender of the warning mails. *Default values* are provided and they will probably work. Define these fields only if warning mails are sent when a virus is found or if you want to use a different account instead of **ravms**. Specify the **smtp_server** IP address only if that machine is behind a firewall and **ravmd** can't get its host name. If you are using Postfix as MTA then you can set **ravmd** to use a specified port when sending warning e-mails. Setting **smtp_port** on 10026 (in our configuration example) will make Postfix to send those e-mails without being scanned.

`ravms_full_name` is a parameter included in **ravmd** starting with version 8.3.3. This parameter is used for compiling the sender's e-mail address according to RFC822, the standard for the format of ARPA Internet text messages.

Default values:

ravms_name = ravms
on_host = official host name
smtp_server = official host IP address
smtp_port = 25
ravms_full_name = RAV Antivirus

Example:

```
ravms_name = ravms

on host = ravantivirus.com

smtp_server = 127.0.0.1

smtp_port = 25

ravms_full_name = RAV AntiVirus Scanner
```

The e-mail address displayed in the warning mail will be: "RAV Antivirus Scanner" <ravms@ravantivirus.com>.

no_subject = string

Description: This parameter is used to specify the string replacing the `SUBJECT` macro in the warning e-mail if **ravmd** does not find a valid subject in the infected email.

Default value: **--no subject found--.**

Example:

```
no_subject = "original e-mail didn't contain any
subject field"
```

mailer_daemon = string

Description: This parameter is used to specify the name that will replace the `FROM_USER` macro when the e-mail sender is **<>**.

Default value: **--unknown--**.

Example:

```
mailer_daemon= "MAILER-DAEMON"
```

Anti-spam parameters

The parameters pertaining to the anti-spam module of **RAV AntiVirus for Mail Servers** are described below.

The default configuration is included in the `antispam` file that you can find in the `/etc/opt/rav` directory.

accuracy_low
accuracy_medium
accuracy_high
accuracy_very_high

Description: Keywords designating the accuracy level.

quarantine = string

Description: String specified in the `_define_strings` section defining the location where the messages meeting certain spam patterns will be saved for the groups where the **save** action is configured.

Default value: **`/var/opt/rav/bulk.`**

extra_header = string

Description: String specified in the `_define_strings` section, defining the extra-header that will be added to the message tagged as spam.

The *default values* depend on the corresponding bulk detection accuracy level.

Example:

```
extra_header = bulk_header_high_english
```

extra_subject = string

Description: String specified in the `_define_strings` section, defining the extra-subject that will be added in the **Subject** field of the messages meeting certain spam patterns.

The *default values* depend on the corresponding bulk detection accuracy level.

Example:

```
extra_subject= bulk_subject_high_english
```

embedded_msg = string

Description: String specified in the `_define_strings` section, defining the message that will be used for the embedded mail body.

The *default values* depend on the corresponding bulk detection accuracy level.

Example:

```
embedded_msg= bulk_embedded_high_english
```

actions = variable

Description: String defined in `_define_actions` section specifying the actions to be taken by **ravmd** for the corresponding bulk detection level.

The *default values* depend on the corresponding bulk detection accuracy level.

```
bulk_actions_low = add_subject, add_header, deliver

bulk_actions_medium = add_subject, add_header, deliver

bulk_actions_high = embed, add_subject, add_header, deliver

bulk_actions_very_high = save, discard
```

Example:

```
actions = bulk_actions_high
```

Real-time Blackhole List parameters

Real-time Blackhole List (RBL) is a feature available in **RAV AntiVirus for Mail Servers** starting with version 8.3.3. This functionality is implemented in `librbl.so` and it consists in defining a dynamic list (`rbl_site`) with sites containing listings of known spammers.

Note 1:	The `rbl_site` list has to be configured and updated by your system administrators.

When this feature is enabled, **RAV AntiVirus for Mail Servers** checks if any of the IP addresses from the mail's header is listed on one of the websites defined in the `rbl_site`. If it does, the mail is automatically rejected.

No warning mail is sent; no file is saved in the RAV Quarantine folder. The `librbl.so` library is loaded only if at least one site is included in `rbl_site`.

The **rbl** options are used for the [`global`] group. You cannot set different values for different groups. If you do not want the mails for one of your groups to be checked against the **Real-time Blackhole List**, you should use the following parameter in the configuration for that group:

```
use_rbl = no
```

Below you can find the parameters associated with this feature.

use_rbl = boolean

Description: Use this parameter to specify if the Real-time Blackhole List (RBL) feature is used or not for one specific group.

Accepted values: **Yes** or **No**.

Default value: **No**.

rbl_site = enumeration

Description: List containing the sites where **ravmd** will be looking for IP addresses for known spammers to be checked against the IP addresses from the e-mail header.

rbl_cache_file = string

Description: Variable containing the path where the **ravmd** cache is saved when the program is stopped. When **ravmd** is restarted, the cache is re-loaded from this path.

Default value: **`/var/opt/rav/rbl.cache`.**

rbl_cache_size= number

Description: Library caching the latest IP addresses **rbl** has been looked for. The parameter following the '=' sign specifies how many IP addresses are included in this cache.

Default value: **10007.**

rbl_timeout = number

Description: Number specifying how many seconds **ravmd** will be waiting for one site to answer to its DNS request.

Accepted values: **minim=2**, **maxim=30**.

Default value: **5.**

rbl_retry = number

Description: Number specifying how many times the DNS request is sent to the same server in case of timeout.

Accepted values: **minim=1, maxim=5**.

Default value: **4.**

White/Black List parameters

wbl_accept IP, IP/MASK, IP/netmask, mail@address, mail.domain
wbl_reject IP, IP/MASK, IP/netmask, mail@address, mail.domain

Description: These parameters are used for specifying the parameters for the **Static White/Black List**, a feature included in **ravmd** starting with version 8.3.3.

Using the **White/Black List** feature you can define IP addresses and mail addresses/domains to be included in the **Static White List** (e-mail messages to be received) or in the **Static Black List** (e-mail messages to be rejected). The keyword `wbl_accept` anticipates the IP addresses and mail addresses/domains added to the **Static White List**. The keyword `wbl_reject` anticipates the IP addresses and mail addresses/domains added to the **Static Black List**.

The rules are applied in the order you define them. The first rule to match will give the result of the **wbl** search: accept or reject. The **wbl** options are not inherited from the [`global`] group. The **wbl** search is done before the **rbl** search. If there is a **reject** rule match for the current mail, the message is automatically rejected. If there is a `wbl_accept` rule match for the current mail, no **rbl** search is executed. If no rule is found in the **wbl** for the current mail, the **rbl** search is executed (unless you specified `use_rbl=no` for the corresponding group).

No warning mail is sent; no file is saved to the RAV Quarantine folder. The `libwbl.so` library is loaded only if at least one **wbl** rule is defined for one group.

Example:

1. If you want to accept mail messages only from the IP address 193.230.245.100 and to reject all mail messages from the entire class 193.230.245.0/24 (193.230.245.0/255.255.255.0), use:

```
wbl_accept 193.230.245.100

wbl_reject 193.230.245.100/24
```

2. If you want to accept mail messages only from the `user@domain.com` mail address and to reject all mail messages from the other mail addresses from the `domain.com`, use:

```
wbl_accept user@domain.com

wbl_reject domain
```

Miscellaneous parameters

These parameters are inherited by additional groups from the [global] group.

warn_header_msg = variable

Description: This parameter is used to specify the text used as a header in the notification e-mail.

Example:

```
warn_header_msg = notification_header
```

Where:

```
notification_header = "This message is automatically generated
by RAV AntiVirus."
```

warn_footer_msg = variable

Description: This parameter is used to specify the text used as footer in the notification e-mail.

Example:

```
warn_footer_msg = notification_footer
```

Where:

```
notification_footer = "The e-mail scanned was received from:
HEADER_RECEIVED."
```

warn_txt_msg = variable

Description: This parameter is used to specify the text appended after the default one in the `warn.txt` file.

Example:

```
warn_txt_msg = append_to_warn_txt
```

Where:

```
append_to_warn_txt = "Please contact your system administrator
for more information."
```

For more information about the `warn.txt`, please read the FAQ 4.

charset = string

Description: This parameter (available beginning with the 8.3.3 version of **ravmd**) is used to specify the value for the **charset** field used in the MIME header of the virus notification mails. If the warning mails contain strings with a character encoding system different from ASCII you should specify the respective encoding using the **charset** parameter.

Default value: **US-ASCII**.

Example:

```
charset = iso-2022-jp
```

custom_msg = number

The notification e-mails are created using the strings defined by the user in the `_define_strings` section and RAV-related information (always added during the evaluation period). A notification message looks as follows:

RAV AntiVirus for OSTYPE version: x.x.x (snapshot-yyyymmdd)
Copyright (c) 1996-2001 GeCAD The Software Company. All rights reserved.
X more days to evaluate. (or: Registered version for N domain(s).)
Running on host: HOSTNAME

The file `ATTACHED_NAME` attached to mail (with subject:`SUBJECT`) sent by `FROM_USER` to `TO_USER(S)` is infected with virus: `VIRUS_NAME`. The file was successfully deleted by RAV AntiVirus.

Scan engine 8.7 () for i386.
Last update: Thu, 27 Jun 2002 15:44:53 +0300
Scanning for 68249 malwares (viruses, trojans and worms).

To get a free 30-days evaluation version of RAV AntiVirus v8 (yet fully functional) please visit:
http://www.ravantivirus.com

The macros are replaced with their corresponding values. In the registered version of **RAV AntiVirus for Mail Servers**, all the RAV-related information can be omitted, except for the first header line.

In the registered version the warning e-mails can be customized.

Default value: **255** (use all RAV information).

Accepted values:

= 0 – No information.

+ 1 – Add “Registered version ...”

+ 2 – Add “Running on host ...”

+ 4 – Add “Scan engine ...”

+ 8 – Add "Last update ..."

+16 – Add "Scanning for ..."

+32 – Add "To get a free 30-days ..."

+64 – Add "Copyright ..."

+128 – Add "RAV AntiVirus for ..."

Example:

custom_msg = 136

```
RAV AntiVirus for OSTYPE version: x.x.x (snapshot-
yyyymmdd)

The file ATTACHED_NAME attached to mail (with subject:
SUBJECT) sent by FROM_USER to TO_USER(S) is infected
with virus: VIRUS_NAME.

The file was successfully deleted by RAV AntiVirus.

Last update: Thu, 27 Jun 2002 15:44:53 +0300
```

max_processes = number

Description: This parameter is used to specify the maximum number of **ravmd** scanning processes running at the same time.

Accepted values: between **1** and **128**.

Default value: **24**.

Example:

```
max_processes = 80
```

timeout_per_file = number
timeout_per_mega = number

Description: These parameters are used to specify the maximum time in seconds that a scanning process can spend on a file. The total timeout is computed using the following formula:

timeout_per_file + timeout_per_mega * filesize/1Mb

Accepted values: **10-600** (for **timeout_per_file**), respectively **5-600** (for **timeout_per_mega**).

Default values: **300** for `timeout_per_file` **60** for `timeout_per_mega`.

Example:

```
timeout_per_file = 120

timeout_per_mega = 25
```

save_infected = boolean
save_suspicious = boolean

Description: This parameter is used to specify if infected/suspicious e-mail files are saved to the local disk before executing any action. You can set these options to **Yes** or **No**.

Default value: **Yes**.

Note:	The infected/suspicious messages will be placed in the quarantine regardless of the defined `infected_actions` and `suspicious_actions`.

Example:

```
save_infected = no

save_suspicious = yes
```

quarantine = string

Description: String used to specify the directory where the infected/suspicious e-mails are saved.

Default value: **`/var/spool/rav/quarantine.`**

Example:

```
quarantine = /tmp/rav/quarantine
```

RAV logging system

When **ravmd** is launched, it logs some information in the system mail info file (using `syslog`) then it switches to the internal log. By default the log files are created in: `/var/spool/rav/log/`.

It is possible to use a different log file for every group, with different options for it. For this you have to declare one of the log options in that group. If a group doesn't contain any log options then the log data for the [`global`] group will be used.

Here are the parameters used for RAV logging system:

log_file_name = string

Description: This parameter is used to specify the value of string must be the full path and the name of the log file. Any other values should determine **ravmd** to not start.

Default value: **`log_file_name = /var/spool/rav/log/group_name`** (for the `group_name` group).

Example:

```
log_file_name = /var/spool/rav/log/global

log_file_name = /var/spool/rav/log/my_group
```

log_max_length = {number}(Kb|Mb)

Description: This parameter is used to specify the maximum log file size.

Default value: **500Kb**.

Accepted values: **10-1000Kb** and **1-5Mb**.

log_rotate_after = {number}(m|h|d)

Description: This parameter is used to specify the period of time elapsing before creating a new log file.

Default value: **6h** (hours).

Accepted values: **10-60m** (minutes, with 23m rounded to 20m, 25m rounded to 30m), **1-24h** (hours) or and **1-30d** (days).

log_delete_after = {number}(h|d|m)

Description: Parameter used to specify the period elapsing before deleting log files older than the specified period.

Default value: **7d** (days).

Accepted values: **1-24h** (hours), **1-30d** (days) and **1-12m** (months).

log_use_zip = boolean

Description: This parameter is used to specify if the log files should be archived (using the zlib library) or not.

Default value: **Yes**.

Accepted values: **Yes** or **No.**

log_level = number

Description: Number controlling the logging level used by **ravmd**.

Default value: **2047** (use all RAV information).

Accepted values:

0 – No log information.

+ 1 – Add error messages (i.e. “can't fork”, “error reading from socket”, etc.).

+ 2 – Add the name of the e-mail file.

+ 4 – Add mime part scanned.

+ 8 – Add final scan result.

+ 16 – Add actions taken during scanning.

+ 32 – Add the e-mail addresses of the sender and the first receiver.

+ 64 – Add the group name matched.

+128 – Add information generated by the external triggered update.

+256 – Add LICENSE LIMIT warnings.

+ 512 – Add **WBL** logs.

+ 1024 – Add **RBL** logs.

Group-specific parameters

The following parameters must have **different** values for every defined group. If these parameters are not defined for each group, their values are **NOT** copied from the [`global`] group. The *default values* are specified for each of these parameters in the following section.

filter_subject variable_1 variable_2

Description: This parameter is used to filter the e-mail subject.

variable_1 is a regular expression defined in the *Regular Expression Declarations Section*.

variable_2 is a variable defined in the *Action Declarations Section*. If this parameter is not defined then the subject filter is disabled.

Example:

```
filter_subject subj_regexp subj_actions
```

Where:

```
subj_regexp = I love you

subj_actions = reject
```

Using this rule, e-mails having the "I love you" string in the subject field will be rejected. You can use here a regular expression, not just a simple string.

filter_attachment variable_1 variable_2

Description: This parameter is used to filter the names of the e-mail attachments.

variable_1 is a regular expression defined in the *Regular Expression Declarations Section*.

variable_2 is a variable defined in the *Action Declarations Section*. If this parameter is not defined then the filter for attachment names is disabled.

Example:

```
filter_attachment file_regexp file_actions
```

Where:

```
file_regexp = .*.((vbs)|(exe)|(com))

file_actions = delete, reject
```

This filtering rule deletes all the attached files with extension ".vbs", ".exe" or ".com" from an e-mail.

If a file cannot be deleted then the whole e-mail will be rejected.

filter_content variable_1 variable_2

Description: This parameter is used to filter the e-mail body and the contents of the attachment. If this parameter is not defined, the content filtering is disabled.

variable_1 is a string defined in the *Regular Expression Declarations section*.

variable_2 is a variable defined in the *Action Declarations section*.

Example:

```
filter_content body_string_1 body_actions_1

filter_content body_string_2 body_actions_2

filter_content body_string_3 body_actions_3

filter_content body_string_4 body_actions_4

filter_content body_string_5 body_actions_5

filter_content body_string_6 body_actions_6
```

Where:

```
body_string_1 = confidential

body_string_2 = salaries

body_string_3 = balance sheet

body_string_4 = tax

body_string_5 = income

body_string_6 = revenue
```

and:

```
body_actions_1 = delete, reject

body_actions_2 = delete, reject

body_actions_3 = delete, reject

body_actions_4 = copy, ignore
```

```
body_actions_5 = copy, ignore

body_actions_6 = copy, ignore
```

In this case the content filtering module of **ravmd** is looking for user-specified strings. A priority is assigned to each rule. The rule with the highest priority is the first one you specify (`body_string_1`). This rule has a priority of 1. The next rules receive lower priority levels (2...n), depending on the order you are specifying them (`body_string_2` has a priority of 2, `body_string_3` has a priority of 3 and so on). In our example, **ravmd** will start searching for all the strings defined by the user (`confidential`, `salaries`, `balance sheet`, `tax`, `income` and `revenue`) in the same time. If the first match is found for one string having the highest priority level (`confidential`, in this example), the search stops and the actions from `body_actions_1` are executed. If the first match is found for one string having a lower priority level (`tax` for instance), the search will continue for the rest of the e-mail, looking for eventual matches for `confidential`, `salaries` and `balance sheet`, strings with higher priority. If a match is found, **ravmd** will execute `body_actions_1` if the match is for `confidential`. If the match is not for `confidential` but for `salaries` for instance, **ravmd** will keep looking for `confidential`. If a match with `confidential` is found, **ravmd** will execute `body_actions_1`. If no match with `confidential` is found, **ravmd** will execute `body_actions_2` (actions corresponding to `salaries`). **ravmd** will ignore the first match (for `tax`) in all these cases.

If after finding a match for `tax` no match with higher priority strings (`confidential`, `salaries` or `balance sheet`) is found, **ravmd** will execute `body_actions_4` (the actions corresponding to `tax`).

Because `body_actions_1`, `body_actions_2` and `body_actions_3` are identical (`delete, reject`) and `body_actions_4`, `body_actions_5` and `body_actions_6` are also identical (`copy, ignore`), you can significantly simplify you work using:

```
body_string_1 = confidential|salaries|balance sheet

body_string_2 = tax|income|revenue
```

and:

```
body_actions_1 = delete, reject

body_actions_2 = delete, reject
```

The behaviour of **ravmd** is in this case the same as explained above.

Note:	If you want to define content filtering rules for mail boddies containing strings that include the „|" character, use „||" (i.e. `body_string = confidential||salaries` will return matches for mail bodies containing the „confidential|salaries" expression).

warn_sender = enumeration

Description: Use this parameter to specify when to send notifications to the e-mail sender. See below for valid keywords.

warn_receiver = enumeration

Description: Use this parameter to specify when to send notifications to the e-mail receivers. See below for valid keywords.

warn_admin = enumeration

Description: Use this parameter to specify when to send notifications to the administrators.

You have to specify *who* is warned by **RAV AntiVirus for Mail Servers** and *when*. If one of these parameters is not defined then the respective user category will **not** receive notifications. The valid keywords are:

Keyword	Meaning
`found_virus`	Send alert to the defined recipients when a virus is found.
`found_subject`	Send alert to the defined recipients when the subject matches a content filtering rule.
`found_attach`	Send alert to the defined recipients when an attached file name matches a content filtering rule.
`found_content`	Send alert to the defined recipients when the mail body contains a string matched by a content filtering rule.
`always`	Send alert to the defined recipients in all the above-mentioned situations.
`never`	Never send alert.
`match_all_flags`	Send alert to the defined recipients only when *ALL* the previously defined rules (`found_virus`, `found_subject`, `found_attach` or `found_content`) are matched.

Please note that these values are not inherited from the [`global`] group. This way you can specify different notification policies for different groups.

Example 1:

```
warn_sender = found_virus, found_subject, found_attach,
found_content

warn_receivers = found_virus

warn_admin = always
```

In this example, the sender is warned whenever a virus is found *or* the subject matches a content filtering rule *or* an attached file name matches a content filtering rule *or* the mail body contains a string matched by a content filtering rule. The receivers are warned whenever a virus is found. The

administrator is always warned.

Example 2:

```
warn_sender = found_virus, found_subject,
match_all_flags
```

In this example, the sender is warned whenever a virus is found *AND* the subject matches a content filtering rule.

Example 3:

```
warn_receivers = found_virus, found_subject,
found_content, match_all_flags
```

In this example, the receivers are warned whenever a virus is found *AND* the subject matches a content filtering rule *AND* the mail body contains a string matched by a content filtering rule.

do_not_scan = boolean

Description: This parameter is used to specify if the e-mail files for the current group are scanned or not. This way it is possible to exclude some e-mail addresses and/or domains from the scanning process.

Default value: **No**.

Example:

```
do_not_scan = yes
```

do_not_warn = enumeration

Description: This parameter is used to specify the e-mail address that will not be notified.

do_not_show = enumeration

Description: This parameter is used to specify the e-mail address that will be hidden in all e-mail notifications.

Why is this parameter required?

There may be cases when one user should not be notified or his e-mail address should not be displayed in the warning mails. If this is the case, these options will help you solve the problem.

Note that only the receiver's e-mail address is compared against the specified e-mail address.

These parameters have no *default values* (no comparisons will be made).

Example:

```
do_not_warn = user3@domain2.org

do_not_show = user1@domain1.com, user2@domain1.com
```

admin_addr = enumeration

Description: This parameter is used to specify the *e-mail addresses* of the administrators that will be notified when an infected or suspicious file has been detected. This warning mail contains messages created using the strings specified for each situation. This parameter has no *default value*.

Example:

```
admin_addr = postmaster@domain1.com,
postmaster@domain2.net, user1@domain1.com

admin_addr = ravmails@stats.ravantivirus.com
```

Note:	The option of forwarding the e-mail to ravmails@stats.ravantivirus.com is highly recommended. This will help the RAV Development Team to determine the level of spreading of this virus and new viruses found in the world or potential detection problems. The Technical Support team at GeCAD Software may also diagnose potential problems for the end-user, such as old updates of the virus signatures database or old product kits. In this case, the end-user will be informed about the best solution for solving his problem. GeCAD Software treats each mail in strict confidence.

antispam_configuration = enumeration

Description: Using the **antispam_configuration** parameter you can set the desired actions for the four different accuracy levels available (**low**, **medium**, **high** and **very high**). In the **enumeration** part you specify **name_conf_antispam1**, **name_conf_antispam2**, **name_conf_antispam3** and **name_conf_antispam4**, corresponding to the four different accuracy levels.

```
antispam_configuration = name_conf_antispam1,
name_conf_antispam2, name_conf_antispam3,
name_conf_antispam4
```

advertising_msg = variable

Description: Using the `advertising_msg` parameter you can append a personal message to each mail message scanned by **ravmd**. The length of the mail message will grow accordingly when using this feature.

The `advertising_msg` parameter must be declared in the group file or even in the [`global`] group. The value for **variable** must be declared in the

language file and the `language.equiv` file must be included in the group file.

Please note that **RAV AntiVirus for Mail Servers** cannot append the text to any e-mail due to the MIME format of the mail and the `advertising_msg` feature is not supported by **ravcgate**.

Example:

- In the group file (that might be even the [`global`] one) define:

```
_include /etc/opt/rav/languages/english.equiv

advertising_msg = my_advertising
```

- In `/etc/opt/rav/languages/english` define:

```
my_advertising = "COMPANY_NAME maintains e-mail
messages virus free"
```

update_executable= string

Description: This parameter is used to specify the name of an executable file used by **ravmd** to start the update process. You must specify the full path to the executable file. The update process is started only if **ravmd** is receiving an update e-mail sent by RAV Team. If you want to receive such e-mails, please send a request to:

updates-l-subscribe@lists.ravantivirus.com

Default value: **`/etc/opt/rav/bin/ravupdate.sh`**

Example:

```
update_executable = null
```

This line is disabling the **update executable** feature:

```
update_executable = /home/ravms/ravupdate.sh
```

This line executes the specified script file every time an update e-mail is processed by **ravmd**.

Embedded messages

embed_clean_mail = boolean

Default value: **No.**

embed_cleaned_mail = boolean

Default value: **No.**

embed_unclean_mail = boolean

Default value: **No.**

use_embedded_msg = boolean

Default value: **No.**

use_embedded_warning = boolean

Default value: **No.**

Description: These parameters are used for specifying what messages you want to be send as embedded mails. Embedded mails are a new feature available in **ravmd** version 8.3.3. After being scanned, the original message can be attached to a new mail message, created by **ravmd**, and sent to its addressees. When the new mail is accepted by the MTA, the original mail is discarded (if the MTA is supporting the **discard** feature) or rejected (if the MTA does not support the **discard** feature, i.e. Courier or Dmail). When the new mail is not accepted by the MTA, the original message is sent instead.

You can specify the mail messages to be encapsulated:

- **embed_clean_mail**: encapsulate all clean mails (mail messages that were not infected/did not contain suspicious code),
- **embed_cleaned_mail**: encapsulate all cleaned mails (mail messages that were infected/did contain suspicious code, but they were cleaned by **RAV AntiVirus for Mail Servers**), or
- **embed_unclean_mail**: encapsulate all uncleaned mails (mail messages that were infected/did contain suspicious code and RAV AntiVirus was not able to disinfect them).

For each status described above the administrator can send a customized message using the `embedded_clean_msg`, `embedded_cleaned_msg` and `embedded_unclean_msg` parameters (described below).

- `use_embedded_msg`: when using this parameter, `embedded_clean_msg`, `embedded_cleaned_msg` and/or `embedded_unclean_msg` will be added to the encapsulated mail.
- `use_embedded_warning`: when using this parameter the notification mail will be added to the encapsulated mail only if the original mail is infected.

The rules for embedded mails are not inherited from the [`global`] group – you have to define the parameters for the suitable groups.

embedded_clean_msg=variable
embedded_cleaned_msg=variable
embedded_unclean_msg=variable

Definition: These three parameters are used for specifying the customized message send when embedding clean/cleaned/uncleaned messages.

The newly created message will therefore include, besides the original message:

- the customized message (when defined by the administrator); and
- the customized warning messages (when defined by the administrator).

embed_queue = string

Definition: The path on the local disk where **ravmd** is temporarily saving the embedded mails, before sending them.

Default value: `/var/spool/rav/embed_queue`.

BUGS

Please mail bug reports and suggestions to: mailto:ravteam@ravantivirus.com

Index of Group Parameters

Domain parameters

domain

Group members

sender
receiver
from_host
to_host

Engine actions

infected_actions
suspicious_actions

Engine parameters

use_heuristics
use_cf_inside_embedded_objects
scan_packed_executables
scan_archives
rename_ext
smart_scan

Warning messages

_virus_warning_messages
_subject_filter_warning_messages
_attachment_filter_warning_messages
_content_filter_warning_messages
warning_mail_subj
infected_msg
suspicious_msg
ignored_msg
rejected_msg
discarded_msg
cleaned_msg
moved_msg
copied_msg
deleted_msg

renamed_msg
saved_inf_msg
saved_sus_msg
cannot_clean_msg
cannot_move_msg
cannot_copy_msg
cannot_delete_msg
cannot_rename_msg
cannot_save_inf_msg
cannot_save_sus_msg

Specifying the sender of the warning mails

ravms_name
on_host
smtp_port
smtp_server
ravms_full_name
no_subject
mailer_daemon

Anti-spam parameters

quarantine
extra_header
extra_subject
embedded_msg
actions

Real-time Blackhole List (RBL) parameters

use_rbl
rbl_site
rbl_cache_file
rbl_cache_size
rbl_timeout
rbl_retry

White/Black List parameters

wbl_accept
wbl_reject

Miscellaneous

- warn_header_msg
- warn_footer_msg
- warn_txt_msg
- charset
- custom_msg
- max_processes
- timeout_per_file
- timeout_per_mega
- save_infected
- save_suspicious
- quarantine

RAV logging system

- log_file_name
- log_max_length
- log_rotate_after
- log_delete_after
- log_use_zip
- log_level

Group-specific parameters

- filter_subject
- filter_attachment
- filter_content
- warn_sender
- warn_receivers
- warn_admin
- do_not_scan
- do_not_warn
- do_not_show
- admin_addr
- antispam_configuration
- advertising_msg
- update_executable

Embedded messages

- embed_clean_mail
- embed_cleaned_mail
- embed_unclean_mail

embedded_clean_msg
embedded_cleaned_msg
embedded_unclean_msg
use_embedded_msg
use_embedded_warning
embed_queue