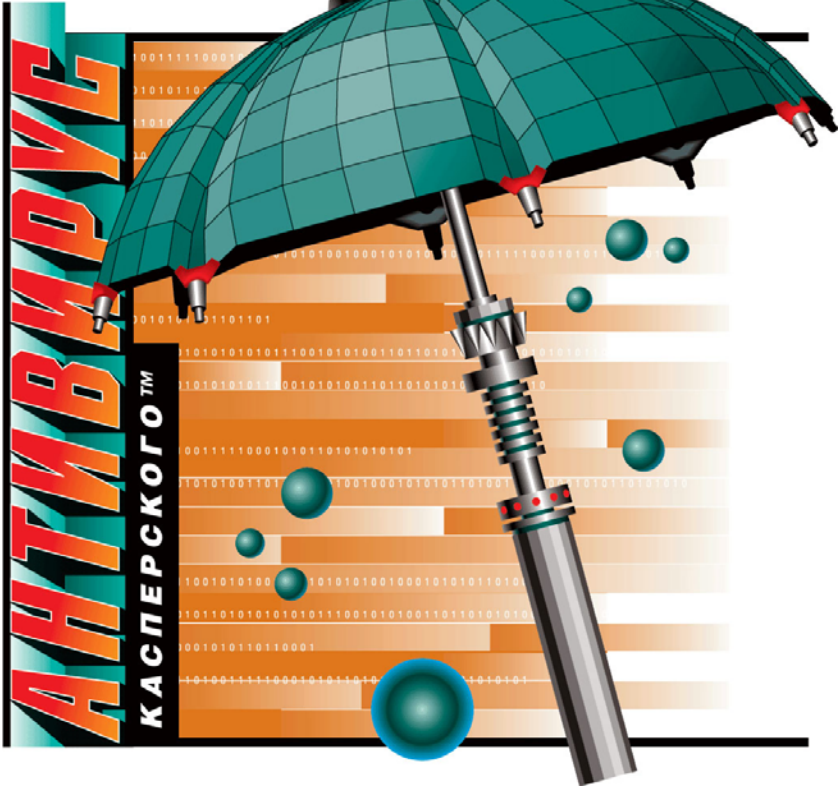


KASPERSKY LAB



**РЕАЛЬНАЯ
ЗАЩИТА**
ВИРТУАЛЬНОГО
ПРОСТРАНСТВА



Kaspersky Anti-Virus® 5.0 for Sendmail with Milter API

ADMINISTRATOR'S MANUAL

KASPERSKY ANTI-VIRUS® 5.0 FOR SENDMAIL
WITH MILTER API

Administrator's manual

© Kaspersky Lab
<http://www.kaspersky.com>

Revision date: March 2005

Contents

CHAPTER 1. KASPERSKY ANTI-VIRUS® FOR SENDMAIL WITH MILTER API.....	6
1.1. Hardware and software system requirements	7
1.2. Licensing policies.....	8
1.3. Distribution kit	8
1.4. Help desk for registered users	9
1.5. Adopted conventions.....	10
CHAPTER 2. TYPICAL DEPLOYMENT SCENARIOS	12
2.1. Installing Kaspersky Anti-Virus on the same server with your mail system	12
2.2. Installing Kaspersky Anti-Virus on a dedicated server	14
2.3. Installing Kaspersky Anti-Virus as a filter (single or additional).....	15
CHAPTER 3. INSTALLATION AND UNINSTALLATION OF KASPERSKY ANTI-VIRUS	16
3.1. Software installation on a server running Linux.....	16
3.2. Software installation on a server running FreeBSD or OpenBSD.....	17
3.3. Installation process	17
3.4. Post-install setup	19
3.5. Location of application files and directories	19
3.6. Software uninstallation on a server running Linux.....	21
3.7. Software uninstallation on a server running FreeBSD or OpenBSD.....	21
3.8. Uninstallation process	21
CHAPTER 4. RECOMMENDED OPERATION MODES	23
4.1. High overall security mode.....	23
4.2. Top reliability mode	24
4.3. Optimal operation mode.....	25
4.4. Top performance mode.....	25
CHAPTER 5. USING KASPERSKY ANTI-VIRUS FOR SENDMAIL WITH MILTER API.....	27
5.1. Delivering disinfected messages to recipients	27

5.2. Blocking infected messages	28
5.3. Delivering protected messages.....	30
5.4. Sending notifications to senders, recipients, and administrator.....	30
5.5. Filtering e-mail traffic by attachments	33
5.6. Updating the anti-virus database and application kernel	34
5.7. Backing up e-mail messages	34
CHAPTER 6. ADDITIONAL SETUP	36
6.1. Integrating the application into your mail system.....	36
6.2. Installing and uninstalling the remote management module	38
6.3. Defining an e-mail scan policy	40
6.4. Adjusting scan thoroughness.....	40
6.5. Selecting objects to scan.....	41
6.6. Message status	41
6.7. Assigning actions for objects.....	42
6.8. Selecting objects to be filtered and assigning actions.....	43
6.9. Configuring backup options.....	44
6.10. Configuring database and kernel module updates	45
6.11. Customizing notifications.....	46
6.11.1. Notification templates	48
6.11.2. Customizing notification templates	50
6.11.2.1. Macros.....	50
6.11.2.2. Iteration constructs.....	51
6.11.2.3. Scope of visibility for an iterative statement	52
6.11.2.4. Variables	53
6.11.2.5. Language syntax	54
6.11.2.6. Notification macros for the application	56
6.12. Reporting options	57
6.13. Parameters of update report generation	58
6.14. Statistics parameters.....	60
6.15. Restarting Kaspersky Anti-Virus for Sendmail with Milter API.....	60
6.16. Managing the application from the command line	62
6.17. Localization of displayed date and time format	63
6.18. Controlling the application performance	63
CHAPTER 7. USING LICENSES.....	65

7.1. License keys management	65
7.1.1. Viewing license key information	66
7.1.2. License extension	67
7.1.3. License key removal	69
CHAPTER 8. COMPATIBILITY WITH OTHER KASPERSKY LAB APPLICATIONS	70
CHAPTER 9. VERIFYING PROPER OPERATION OF THE ANTI-VIRUS	71
CHAPTER 10. FREQUENTLY ASKED QUESTIONS.....	73
APPENDIX A. ADDITIONAL INFORMATION.....	77
A.1. Application configuration file	77
A.2. Error return codes	84
APPENDIX B. MALWARE IN UNIX ENVIRONMENT	86
B.1. Viruses.....	86
B.2. Trojan software.....	87
B.3. Network worms	88
APPENDIX C. KASPERSKY LAB.....	91
C.1. Other Kaspersky Lab products.....	92
C.2. Our contact information	95
APPENDIX D. LICENSE AGREEMENT	96

CHAPTER 1. KASPERSKY ANTI-VIRUS[®] FOR SENDMAIL WITH MILTER API

Kaspersky Anti-Virus[®] for Sendmail with Milter API provides anti-virus protection for e-mail traffic handled by Sendmail with Milter API running on a Linux\Unix server.

Kaspersky Anti-Virus for Sendmail with Milter API running on a mail server will...

- Intercept incoming and outgoing e-mail messages handled by the server.
- Scan e-mail traffic for viruses using the anti-virus engine. The program scans the entire message as well as message objects, including the header, body, and attachment (depending on the anti-virus policy).
- Back up e-mail messages prior to performing any action related to anti-virus protection, including blocking and rejecting messages. The administrator can then restore original messages from these backup copies.
- Handle infected objects of e-mail messages detected during the scan.
- Filter e-mail messages. This version of the product filters messages by MIME type, size, and name of attachments.
- Notify the senders and administrators about the results of anti-virus treatment and message filtering. The program may also send detailed notifications using an external mail agent.
- Provide general statistics and reports on program performance.

For more details about these functions, please refer to the corresponding chapter in this Administrator's Guide.

The advanced features of Kaspersky Anti-Virus for Sendmail with Milter API allow the administrator to perform the following tasks:

- *Configure the application from a remote location* through the web interface of the Webmin application.
- *Customize templates for sending notifications* to senders, recipients, and administrators using a special language.

1.1. Hardware and software system requirements

For smooth operation of Kaspersky Anti-Virus for Sendmail with Militer API, your mail server must meet the following hardware and software requirements:

Minimum hardware requirements for program operation:

- Intel Pentium 133 MHz processor or higher
- 32 MB RAM
- 100 MB available space on your hard drive (this amount does not include space necessary for storing backup message copies).

Minimum hardware requirements for a mail server with about 800 MB of traffic per day¹ (250-300 mail accounts (addresses)):

- Celeron (Mendocino) 400 MHz processor
- 512 MB RAM
- 8 GB available space on your hard drive (this amount does not include space necessary for storing backup message copies).

Optimal hardware requirements:

- For a mail server with about 800 MB of traffic per day (250-300 mail accounts (addresses)):
 - 2xPentium Xeon 1,8 GHz processor
 - 1 GB RAM
 - 100 MB available space on your hard drive (for Kaspersky Anti-Virus operation).
- For a mail server with about 400 MB of traffic per day² (100-150 mail accounts (addresses)):
 - Pentium III 900 MHz processor
 - 512 MB RAM.

¹ The following scheme is used to calculate daily traffic: average message size is 60 KB, during 10-hour period, with 25 scan processes working in parallel, about 13200 messages are processed, which totals to 800 MB.

² The following scheme is used to calculate daily traffic: average message size is 60 KB, during 10-hour period, with 25 scan processes working in parallel, about 6666 messages are processed, which totals to 400 MB.

Software requirements:

1. One of the following operating systems:
 - Linux RedHat (v. 8 or 9), Linux SuSE (v.8.2, 9.0 or 9.1) or Linux Debian (v. 3.0)
 - FreeBSD, v. 4.9, 4.10 or 5.2.1
 - OpenBSD, v. 3.4
2. Sendmail version 8.11.x or higher with Milter API (installed)
3. Webmin program (www.webmin.com) (installed) to manage Kaspersky Anti-Virus from a remote location.

1.2. Licensing policies

The following licensing policies are available for Kaspersky Anti-Virus 5.0 for Sendmail with Milter API:

- Licensing by traffic
- Licensing by mail addresses

The type of the policy is defined by the license key you use.

Licensing by traffic covers a certain amount of mail traffic processed by the application during one day. This information is provided in the license key.

Licensing by mail addresses covers the mail addresses of the domains that are listed in the application configuration file and the mail addresses of the server on which Kaspersky Anti-Virus is installed (the latter do not belong to the domain structure).

Both types of licenses have a limited validity period, which is usually one or two years since the date of purchase. The license validity period depends on the license key you use.

1.3. Distribution kit

You can purchase Kaspersky Anti-Virus either from our distributors (retail box) or in our Internet-shop (www.kaspersky.com, **E-Store** section).

When purchasing a retail box, you will receive the following distribution kit:

- A sealed envelope with an installation CD (or a set of floppy disks) containing software product files

- Administrator's guide
- License key written on the installation CD or a floppy disk
- License agreement

Before you unseal the envelope containing the CD (or floppy disks), be sure to thoroughly review the license agreement.

When purchasing Kaspersky Anti-Virus in the Web-shop, you download the product from Kaspersky Lab's website. The distribution file contains the program and the license key.

The License Agreement (LA) is a legal agreement between you (either an individual or a single entity) and the manufacturer (Kaspersky Lab Ltd.) describing the terms under which you may use the anti-virus product which you have purchased.



Make sure to read the terms of the LA!

If you do not agree to the terms of this LA, Kaspersky Lab is not willing to license the software product to you and you should return the unused product to your Kaspersky Anti-Virus dealer for a full refund, making sure the envelope with CD (or diskettes) is sealed.

If you have unsealed the envelope, you have agreed to all the terms of the LA.

1.4. Help desk for registered users

Kaspersky Lab offers a large service package, enabling registered users to efficiently use Kaspersky Anti-Virus.

If you register and purchase a subscription, you will be provided with the following services for the period of your subscription:






- daily virus-definition database updates via e-mail;
- product upgrades;
- phone and e-mail advice on matters related to your software installation, configuration and performance;
- information about new Kaspersky Lab products and new computer viruses (for those who subscribe to our newsletter).



Kaspersky Lab does not give advice on the performance and use of your operating system or various other technologies.

1.5. Adopted conventions

The text in this document is formatted in accordance with its meaning. The table below lists the conventions adopted for use in the text.

Style	Purpose
Bold type	Menu titles, menu items, window titles, parts of dialog boxes, etc.
 Note.	Additional information, notes.
 Attention!	Information that should be paid special heed.
 In order to perform the action, <ol style="list-style-type: none"> 1. Step 1. 2. ... 	Description of procedure for user's steps and possible actions.
 Task, example	Statement of problem, example for using the software features.
 Solution	Solution to a defined problem.
[key] – key purpose.	Command line keys.
Text of information messages and the command line	Text of configuration files, informative messages, and the command line.

CHAPTER 2. TYPICAL DEPLOYMENT SCENARIOS

Kaspersky Anti-Virus for Sendmail with Militer API can be rolled out using the following methods, depending on the initial configuration of your mail system and specific needs of your organization:

- *on the same server your mail system is on*: this scenario is used by default if you have a configured Sendmail system on your server (see section 2.1 on page 12).
- *on a dedicated server*: use this method if your mail server is under a high load (see section 2.2 on page 14).

Note that in both cases the application will function identically, regardless of the deployment scenario you choose. They differ only in the method of interaction between Kaspersky Anti-Virus and Sendmail.

To configure Kaspersky Anti-Virus, consider other Militer filters integrated into your mail system. If you have such filters, you can install Kaspersky Anti-Virus as:

- *a single Militer filter*.
- *Together with other Militer filters*: if you have other mail filters, for example, Kaspersky Anti-Spam (see section 2.3 on page 15).

The sections below describe each scenario in detail.

2.1. Installing Kaspersky Anti-Virus on the same server with your mail system



When describing the operation and configuration of Kaspersky Anti-Virus in this guide, it is assumed that Kaspersky Anti-Virus has been installed on the same server as your mail system.

Kaspersky Anti-Virus for Sendmail with Militer API processes incoming and outgoing mail as follows:

1. Email traffic forwarded from other servers or from users arrives at Sendmail.
2. The mail system then forwards messages to Kaspersky Anti-Virus through Milter API for anti-virus processing.
3. Kaspersky Anti-Virus scans and handles email messages and, depending on the settings, sends them back through Milter API to the mail system. The anti-virus application can generate and send notifications using an external mail agent.
4. The mail system then routes mail traffic to either external mail servers or mailboxes of local users.

Therefore, you need to reconfigure socket options for Sendmail and Kaspersky Anti-Virus during the installation of Kaspersky Anti-Virus. In this deployment scenario, it is recommended that you use the local socket rather than the network socket to speed up data transfer between the applications. Configure Sendmail as follows:

- If you use `sendmail.cf`:

```
XKAVMilter, S=unix:socket_file_path,  
F=T,T=S:10m;R:15m;E:15m
```

or

```
XKAVMilter, S=local:socket_file_path,  
F=T,T=S:10m;R:15m;E:15m
```

- If you use `sendmail.mc`:

```
INPUT_MAIL_FILTER(`KAVMilter',  
`S=local:socket_file_path,  
F=T,T=S:10m;R:15m;E:15m')
```

or

```
INPUT_MAIL_FILTER(`KAVMilter',  
`S=unix:socket_file_path,  
F=T,T=S:10m;R:15m;E:15m')
```

- In the **[kavmilter.global]** section of the Kaspersky Anti-Virus configuration file, make the following changes:

```
ServiceSocket=unix:socket_file_path
```

or

```
ServiceSocket=local:socket_file_path
```

2.2. Installing Kaspersky Anti-Virus on a dedicated server

If your mail server's load is consistently high, it is more reasonable to install Kaspersky Anti-Virus on a dedicated server in order to avoid server malfunction, because anti-virus processing of mail traffic consumes considerable server resources.

- If Kaspersky Anti-Virus is installed on a dedicated server, it operates as follows:
- The email thread arrives at the mail server with Sendmail installed.
- Sendmail then forwards messages to Kaspersky Anti-Virus through a network socket.
- The processed mail thread, together with anti-virus notifications, is sent back to the mail system for further delivery.

If Kaspersky Anti-Virus is installed on a dedicated server, you must use a network socket for email traffic to be received and delivered via Sendmail.

Configure Sendmail as follows:

- If you use `sendmail.cf`:

```
XXKAVMilter, S= inet:1052@ip_address,  
F=T,T=S:10m;R:15m;E:15m
```
- If you use `sendmail.mc`:

```
INPUT_MAIL_FILTER(`KAVMilter',  
`S= inet:1052@ip_address,  
F=T,T=S:10m;R:15m;E:15m')
```
- In the **[kavmilter.global]** section of the Kaspersky Anti-Virus configuration file, make the following changes:

```
ServiceSocket= inet:ip_address
```

2.3. Installing Kaspersky Anti-Virus as a filter (single or additional)

Kaspersky Anti-Virus for Sendmail with Milter API can be installed as either a single filter or together with other filters. If other mail filters have been installed on your system, you should carefully define their sequence based on filter settings.

If you are installing Kaspersky Anti-Virus ahead of another filter, note that anti-virus processing can affect the contents of the email thread: some elements of email messages (headers, body, etc.) can be changed, notifications generated by the anti-virus software can be added to the thread, and some messages can be deleted or rejected for further processing. Therefore, another filter located behind Kaspersky Anti-Virus will deal with a processed, and therefore altered, email thread. Consider this factor when configuring filters behind the anti-virus application. For example, you may exclude notifications generated by Kaspersky Anti-Virus from filtering.

If Kaspersky Anti-Virus is installed behind another filter, set the first filter to forward the email thread to Kaspersky Anti-Virus via a socket.

In this case, Kaspersky Anti-Virus receives the email thread that has been processed and changed by the first filter.

Configure Milter filters installed on your mail server as follows:

- Configure Sendmail and Kaspersky Anti-Virus socket options as described in section 2.1 on page 12.
- Configure other mail filters installed on your mail server either behind or ahead of the anti-virus software to transmit the email thread via a socket.

CHAPTER 3. INSTALLATION AND UNINSTALLATION OF KASPERSKY ANTI-VIRUS

Prior to beginning the installation of Kaspersky Anti-Virus for Sendmail with Militer API, we recommend the following preparations for your system:

- Make sure that your system meets the hardware and software requirements for installation of the Kaspersky Anti-Virus (please see para. 1.1 on p. 7).
- Enter the system as superuser (**root**).

3.1. Software installation on a server running Linux

Kaspersky Anti-Virus is distributed as three different installation packages, depending upon distribution type.



In order to start the installation of Kaspersky Anti-Virus from an .rpm package, enter the following text in the command line:

```
rpm -i <package_file_name>
```



In order to start the installation of Kaspersky Anti-Virus from a .deb package, enter the following text in the command line:

```
dpkg -i <package_file_name>
```

3.2. Software installation on a server running FreeBSD or OpenBSD

The installation package for Kaspersky Anti-Virus is supplied in a .pkg package for servers running FreeBSD or OpenBSD operating systems.



In order to start installing Kaspersky Anti-Virus from a .pkg package, enter the following text in the command line:

```
pkg_add <package_name>
```

3.3. Installation process

The procedure for installing Kaspersky Anti-Virus is automatic and not interactive. If any of the installation steps cannot be performed, the administrator must perform it after the installation is complete.

The install process for Kaspersky Anti-Virus for Sendmail with Milter API performs the following steps automatically:

1. Creating a group and a user account named **kav** under which Kaspersky Anti-Virus will operate.
2. Adding application settings to the `/var/db/kav/applications.setup` file that is used to update the anti-virus database and program modules.
3. Registering the license key (only for alpha and beta application version).

If for any reason you do not have a license key in your distribution package (for example, you purchased the program at an Internet shop and have not received the key yet), the appropriate warning will appear on the console.

Immediately after the installation is complete, you will have to install the key by copying it to the special directory in the application location.

4. Defining domains (i.e., mailboxes of these domains) that will be protected by Kaspersky Anti-Virus. The default domain is the system domain, including all subdomains (if they exist). For example, if your domain is `domains.of.example.com`, the mail accounts on the following domains will also be protected: `example.com`, `of.example.com`, and `domains.of.example.com`.

5. Registering the *kavmilterd* service with the startup system (depending on your operating system).
6. Searching and automatically editing the Sendmail configuration to integrate it with the anti-virus filter.

Prior to making any configuration changes, you must back up the original Sendmail configuration. You can use this backup configuration if Kaspersky Anti-Virus is uninstalled.

After making configuration changes, Sendmail should be restarted so that the changes take effect. If Sendmail is not rebooted during the installation, the configuration changes will not be applied. The corresponding information will be displayed on the console. You will need to change the Sendmail configuration after Kaspersky Anti-Virus installation. Otherwise, the mail traffic will not be filtered on the server.

7. Running the *kavmilter* service (using *kavmilterd* init script) that initializes the anti-virus filtration of mail traffic.
8. Registering a cron task for automatically updating the anti-virus database and the anti-virus kernel modules. The database will be updated for the first time two minutes after the installation is complete. After this, it will be updated every hour.
9. Registering Kaspersky Anti-Virus module for Webmin, if you have Webmin installed.



Please ensure that your network / proxy settings are configured appropriately (see section 6.2 on page 38, configuration section **[updater.options]**).

Check if *cron* service is running, because it is required for automatic application updates.

10. Registering a cron task for hourly checks of the backup storage size. If the size exceeds the limit of 128 MB, 20% of the storage will be purged.
11. Forming links to reference information about Kaspersky Anti-Virus performance. To display the information, use the **man** command.



Note that the Kaspersky Anti-Virus module for Webmin is not installed during application installation.

If you want to manage the application from a remote location, manually install the module for Webmin (see section 6.2 on page 38).

3.4. Post-install setup

The installation of Kaspersky Anti-Virus for Sendmail with Milter API involves automatic configuration of the application and mail system.

However, you may need to perform some post-installation tasks:

1. Install the license key if this was not done during the installation. To install the license key, copy the key file to the special directory defined by the **LicensePath** parameter and restart the application (for details on restarting the application, see section 6.15 on page 60).
2. Configure the Sendmail system to integrate it with the anti-virus filter (if this was not done during the installation) (see section 6.1 on page 36) and restart Sendmail.
3. Configure proxy server settings in the Kaspersky Anti-Virus configuration file if you connect to the Internet through a proxy server (see Appendix A on page 77). This is required to update the database and kernel modules.
4. If necessary, perform additional configuration of the application (see Chapter 6 on page 36).
5. Install the Kaspersky Anti-Virus module for Webmin to enable remote management of the application, if that was not done automatically during the installation (see section 6.2 on page 38).

3.5. Location of application files and directories

In Linux, Kaspersky Anti-Virus files and directories have the following locations:

/etc/kav/5.0/kavmilter/ – Directory with application configuration files:

kavmilter.conf – Application configuration file.

kavmilter.setup – Configuration file added to *applications.setup* for retrieving and installing updates.

init.d/kavmilterd – Service script to control *kavmilter* operation.

profiles/ – Directory containing configurations of preset profiles.

/opt/kav/5.0/kavmilter/bin – Location of application executable files, such as *kavmilter*, *keepup2date*, and *licensemanager*.

/opt/kav/5.0/kavmilter/doc – Directory containing application documentation.

/opt/kav/5.0/kavmilter/man – Location of manual pages.

/opt/kav/5.0/kavmilter/web – directory containing the *kavmilter.wbm* remote management module for the Webmin web-interface.

/var/db/kav/5.0/kavmilter/ – Application directory that includes:

backup/ – Message backup storage directory.

bases/ – Directory storing the anti-virus database and kernel modules.

bases/backup/ – Directory for storing backup copies of the anti-virus database and kernel modules created prior to updating.

licenses/ – Directory containing license keys for the application.

patches/ – Location of program patches.

run/ – Directory that stores the file with the application ID.

templates/ – Directory for storing notification templates.

tmp/ – Directory with temporary files.

/var/log/kav/5.0/kavmilter – Directory that contains report files (if the application is configured to save reports to a file rather than the system log).

The location of Kaspersky Anti-Virus files on xBSD differs from those for Linux OS:

/etc/kav/5.0/kavmilter/ – Directory with the application configuration file for OpenBSD.

/usr/local/etc/kav/5.0/kavmilter/ – Directory with the application configuration file for FreeBSD.

/usr/local/share/kav/5.0/kavmilter/bin – Directory for the application executable files.

/usr/local/share/kav/5.0/kavmilter/doc – Location of application documentation.

/usr/local/share/kav/5.0/kavmilter/web – directory containing the *kavmilter.wbm* remote management module for the Webmin web-interface.

/usr/local/man – Location of manual pages.



When Kaspersky Anti-Virus is installed on a server running FreeBSD, the *kavmilterd* service script that controls the performance of the *kavmilter* executable file must be located in the */usr/local/etc/kav/5.0/kavmilter/rc.d/* directory.

3.6. Software uninstallation on a server running Linux

To uninstall Kaspersky Anti-Virus previously installed from a package you should issue the following command:



In order to remove Kaspersky Anti-Virus if installed from a .rpm package, enter the following text in the command line:

```
rpm -e <package_name>
```



In order to remove Kaspersky Anti-Virus if installed from a .deb package, enter the following text in the command line:

```
dpkg -r <package_name>
```

3.7. Software uninstallation on a server running FreeBSD or OpenBSD



In order to remove Kaspersky Anti-Virus if installed from a .pkg package, enter the following text in the command line:

```
pkg_delete <package_name>
```

3.8. Uninstallation process

The procedure for uninstalling Kaspersky Anti-Virus is automatic, not interactive and contains the following steps:

1. Removing the cron task of checking the backup storage from the list of tasks for the **kav** user.
2. Removing the cron task for updating the anti-virus database and anti-virus kernel modules from the list of tasks for the **kav** user.
3. Rolling back the Sendmail configuration changes you made to integrate it with the anti-virus filter. Restart the mail system to make the previous configuration effective.

4. Stopping the *kavmlter* service. From this moment, anti-virus filtration of mail traffic is disabled.
5. Rolling back the registration of the *kavmlterd* service in the system: in SySV systems, the links to the *rc.d* must be removed; in BSD-based systems, the links to a script corresponding to this service are removed, in OpenBSD-based systems, the *rc.local* file should be edited.
6. Rolling back the registration of Kaspersky Anti-Virus application with the system: the corresponding section is removed from */var/db/kav/applications.setup*.
7. Deleting the **kav** user from the system.
8. Removing the links to the reference information about the application.
9. Deleting temporary files or directories created during Kaspersky Anti-Virus performance.
10. Deleting the Kaspersky Anti-Virus package: all directories and files of the application are removed, excluding reports and configuration files.
11. Removing Kaspersky Anti-Virus module for Webmin, if it was installed.

CHAPTER 4. RECOMMENDED OPERATION MODES

In addition to *kavmilter.conf*, the Kaspersky Anti-Virus for Sendmail with Milter API distribution kit includes four configuration files that provide four protection levels for your mail server:

kavmilter-high-security.conf – Application configuration that delivers a high overall protection to your e-mail traffic (see section 4.1 on page 23).

kavmilter-high-accuracy.conf – Application configuration that provides maximum protection for your e-mail traffic (see section 4.2 on page 24).

kavmilter-default.conf – Configuration that provides an optimal balance between protection level and performance efficiency. When this configuration is enabled, the application runs seamlessly on your mail server, allowing you to work with other applications (see section 4.3 on page 25). The default configuration file, *kavmilter.conf*, is a copy of this file.

kavmilter-high-scanspeed.conf – With this configuration, the program scans e-mail traffic at its fastest, sacrificing functionality for speed (see section 4.4 on page 25).

The administrator can edit any of these files to tailor the application to fit your company needs and environment.

After installation, all these files are stored in the */etc/kav/5.0/kavmilter/profiles* directory.

These options are described in more detail below.

4.1. High overall security mode

This mode offers the most comprehensive protection of your mail traffic. In this mode, the program notifies senders, recipients, and administrator about scan results. This mode includes the following functions:

- The program scans e-mail messages using a combined scan policy: every letter is first scanned for viruses on the whole and then each message object is scanned separately, regardless if the infected objects found or not.
- E-mail messages are filtered by MIME type, attachment name and size, and also by virus name.

- A backup copy is created for every message that undergoes anti-virus filtering; an information file is created for each of these messages.
- All infected messages and their objects are subject to anti-virus processing. If disinfection fails, the message or part of it is deleted.
- All corrupted messages and their objects are replaced with appropriate notifications. All protected objects that cannot be scanned are deleted.
- Notifications about the actions applied to a message or object are sent to the sender, recipient, and administrator.
- All program messages and events are logged in the report.

4.2. Top reliability mode

Compared with High Overall Security mode, Top Reliability mode has less information recorded in the program report. However, anti-virus protection is generally enhanced by deleting corrupted and password protected objects. These objects cannot be scanned for viruses and are potentially hazardous to your computer.

- The program scans e-mail messages using a combined scan policy: every letter is first scanned for viruses on the whole and then each message object is scanned separately, regardless of whether infected objects are found or not.
- E-mail messages are filtered by MIME type and attachment name.
- A backup copy is created for every message that undergoes anti-virus processing; the information file is created for each of these messages.
- All infected messages and their objects are subject to anti-virus processing. If disinfection fails, the message or a part of it is deleted.
- All corrupted and password protected messages or their objects are deleted. Instead of the deleted messages, the program creates corresponding notifications.
- Notifications about the actions applied to a message or its object are sent to the sender, recipient, and administrator.
- All program messages and events, except for debugging information, are logged in the report.

4.3. Optimal operation mode

This mode provides an optimal balance between anti-virus protection level and scan speed:

- The program scans a bulk e-mail messages as a whole and then, if e-mail is identified as infected, each message object is scanned separately.
- E-mail messages are filtered by MIME type and attachment name.
- A backup copy is created for every message that undergoes anti-virus processing; an information file is not created.
- All infected messages and their objects are subject to anti-virus processing. If disinfection fails, the message or a part of it is deleted.
- All corrupted and password protected messages and their objects are deleted. Instead of the deleted messages, the program creates corresponding notifications.
- Notifications about the actions applied to a message or its object are sent to the sender and recipient. The program does not notify the administrator about this.
- All program messages and events, except for debugging information, are logged in the report.

4.4. Top performance mode

This mode provides for maximum application performance; however, the reliability of anti-virus protection is somewhat decreased:

- The program scans a bulk of e-mail messages as a whole and then, if e-mail is identified as infected, each message object is scanned separately.
- E-mail message filter is disabled.
- A backup copy is created for every message that undergoes anti-virus processing; the information file is not created.
- All infected messages and their objects are subject to anti-virus processing. If disinfection fails, the message or its part is deleted.
- All corrupted and password protected messages and their objects are skipped for scanning, and this information is logged in the report.

-
- Notifications about the actions applied to a message or object are sent only to the recipient. The program does not notify the administrator or the sender.
 - Critical events, information messages, and error messages are logged in the report.

CHAPTER 5. USING KASPERSKY ANTI-VIRUS FOR SENDMAIL WITH MILTER API

The main function of Kaspersky Anti-Virus for Sendmail with Milter API is to secure the mail traffic on your mail server against viruses. However, you can significantly extend the application functionality to better meet the needs of your company by using it for filtering e-mail by attachments, backing up e-mail traffic, etc.

This chapter describes the most important tasks that can be implemented using the application. For details on the advanced features of Kaspersky Anti-Virus for Sendmail with Milter API, please refer to Chapter 6 on page 36.



Note that the examples below consider only the configuration that is directly related to implementing the tasks described. The solutions provided for each task describe task configuration only by editing the configuration file. Remote management options using Webmin are not discussed in the documentation.

Most of the examples below require that the application be reconfigured and rebooted to apply recent changes (see section 6.15 on page 60).

5.1. Delivering disinfected messages to recipients

The main role of Kaspersky Anti-Virus is to scan and disinfect e-mail messages using the anti-virus database.

If the application detects an infected message (message object) and fails to disinfect it, we recommend sending an appropriate notification to the recipient of this message.



Task: Scan all incoming messages and attachments for viruses, and try to disinfect infected messages and their objects. If disinfection fails, delete the infected object, replacing it with a corresponding notification. Send the notification to the recipient. Log all information concerning messages in the system log. Record statistics on messages, viruses, and xml resources.



To implement this task, configure the application as follows:

```
[kavmilter.global]
ScanPolicy=combined

[kavmilter.engine]
ScanArchives=yes
ScanPacked=yes
ScanCodeanalyzer=yes

[kavmilter.actions]
DefaultAction=cure

[kavmilter.notifications]
EnableNotifications=on
NotifyRecipients=infected
MessageDir=/var/db/kav/5.0/kavmilter/templates/
MessageSubject="Anti-virus notification message"

[kavmilter.log]
LogFacility=syslog
LogOption=scan.all

[kavmilter.statistics]
TrackStatistics=all
DataFormat=xml
DataFile=/var/log/kav/5.0/kavmilter/statistics.data
```

5.2. Blocking infected messages

You can block messages using several methods: the administrator can delete an infected message without notifying the recipient beforehand and return an error code to the sender as if it were sent by a mail agent.



Task: Block infected e-mail messages, delete them, and notify the administrator of such.



To implement this task, configure the application as follows:

```
[kavmilter.global]
ScanPolicy=combined
```

```
[kavmilter.engine]
ScanArchives=yes
ScanPacked=yes
ScanCodeanalyzer=yes
```

```
[kavmilter.actions]
DefaultAction=drop
```

```
[kavmilter.notifications]
EnableNotifications=on
SendmailPath=/usr/sbin/sendmail
NotifyAdmin=infected
AdminAddresses=admin@localhost
UseCustomTemplates=on
AdminSubject="Anti-virus notification message"
```



Task: Reject infected messages from the sender, return an error code to the sender, and notify the administrator of the actions.



To implement this task, configure the application as follows:

```
[kavmilter.global]
ScanPolicy=message
```

```
[kavmilter.actions]
DefaultAction=reject
```

```
[kavmilter.notifications]
EnableNotifications=on
SendmailPath=/usr/sbin/sendmail
NotifyAdmin=infected
```

```
AdminAddresses=admin@localhost
UseCustomTemplates=on
AdminSubject="Anti-virus notification message"
```

5.3. Delivering protected messages

Sometimes an e-mail message cannot be scanned for viruses because it is password protected or encrypted. The administrator must be sure of the user's ability to disinfect the message if it turns out to be infected.



Task: Deliver protected messages even if they are infected; notify the administrator of such.



Make the following changes in the application configuration:

```
[kavmilter.global]
ScanPolicy=combined
```

```
[kavmilter.actions]
ProtectedAction=skip
```

```
[kavmilter.notifications]
EnableNotifications=on
SendmailPath=/usr/sbin/sendmail
NotifyAdmin=all
AdminAddresses=admin@localhost
UseCustomTemplates=on
AdminSubject="Anti-virus notification message"
```

5.4. Sending notifications to senders, recipients, and administrator

Kaspersky Anti-Virus can send notifications upon virus detection.

Recipient and sender addresses for sending notifications are inherited from the original e-mail message.

The administrator addresses must be specified for the **AdminAddresses** parameter of the **[kavmilter.notifications]** section.

To enable sending notifications, configure the application as follows:

```
[kavmilter.notifications]
EnableNotifications=on
NotifySender=infected
NotifyRecipients=infected
NotifyAdmin=infected
AdminAddresses=admin@localhost
MessageDir=/var/db/kav/5.0/kavmilter/templates/
MessageSubject="Anti-virus notification message"
```



You can customize the format of notifications. For more detail about this, see section 6.9 on page 44.

Below, we consider several examples of how to configure notifications.



Task: Notify the recipient and administrator about a rejected message containing a virus (action for infected objects – *reject*). The sender must receive an error code about an undeliverable mail as if it was sent by the mail agent.



To implement this task, make the following configuration changes:

```
[kavmilter.global]
ScanPolicy=combined

[kavmilter.actions]
DefaultAction=reject

[kavmilter.notifications]
EnableNotifications=on
NotifySender=infected
NotifyRecipients=infected
NotifyAdmin=infected
AdminAddresses=admin@localhost
```

```
MessageDir=/var/db/kav/5.0/kavmilter/templates/  
RejectReply="Message rejected because it contains  
malware"
```



Task: Notify the recipient and administrator that the message containing protected objects has skipped anti-virus processing (action for protected objects – *skip*).



To implement this task, configure the application as follows:

```
[kavmilter.global]  
ScanPolicy=combined  
  
[kavmilter.actions]  
ProtectedAction=skip  
  
[kavmilter.notifications]  
EnableNotifications=on  
NotifyRecipients=protected  
NotifyAdmin=protected  
AdminAddresses=admin@localhost  
MessageDir=//var/db/kav/5.0/kavmilter/templates/  
MessageSubject="This message was NOT scanned by KAV!"
```



Task: Inform the recipient, sender, and administrator about a filtered message. Insert an additional header with information about the application into any mail message scanned by Kaspersky Anti-Virus.



To implement this task, make the following configuration changes:

```
[kavmilter.global]  
ScanPolicy=combined  
AddxHeader=yes  
  
[kavmilter.actions]  
DefaultAction=cure  
  
[kavmilter.filter]  
IncludeSize=10
```

```
FilteredSizeAction=skip

[kavmilter.notifications]
EnableNotifications=on
NotifySender=filtered
NotifyRecipients=filtered
NotifyAdmin=filtered
AdminAddresses=admin@localhost
MessageDir=/var/db/kav/5.0/kavmilter/templates/
MessageSubject="Anti-Virus notification message"
SendmailPath=/usr/sbin/sendmail
UseCustomTemplates=on
```

5.5. Filtering e-mail traffic by attachments

The application can filter e-mail messages by attachment name, attachment MIME type, and attachment size.



Task: Deliver messages with attachments whose size is below 500 Kb without additional treatment. Delete messages with attached files called *loveletter*. Notify the recipient and administrator about the actions performed by the application.



To implement the task, configure the application as follows:

```
[kavmilter.global]
ScanPolicy=combined

[kavmilter.engine]
ScanArchives=yes
ScanPacked=yes
ScanCodeanalyzer=yes

[kavmilter.actions]
DefaultAction=cure
```

```
[kavmilter.filter]
IncludeSize=500
FilteredSizeAction=skip
IncludeName=loveletter\.*
FilteredNameAction=delete

[kavmilter.notifications]
EnableNotifications=on
NotifyRecipient=filtered
NotifyAdmin=all
AdminAddresses=admin@localhost
MessageDir=/var/db/kav/5.0/kavmilter/templates/
MessageSubject="Anti-virus notification message"
SendmailPath=/usr/sbin/sendmail
UseCustomTemplates=on
```

5.6. Updating the anti-virus database and application kernel

During the application installation, the cron task of updating the database and application kernel is registered on the server. Updating is performed every four hours after Kaspersky Anti-Virus for Sendmail with Militer API is installed on the server.

If you want to update the components earlier than at the scheduled time, use the *keepup2date.sh* script supplied with the distribution package. To configure updating manually, enter the following string:

```
keepup2date.sh -run
```

5.7. Backing up e-mail messages

Before applying any actions to messages or their objects, we strongly recommend that you back up messages before repairs are attempted as a data safety precaution.



Task: Scan e-mail traffic for viruses and disinfect all infected objects. Delete the objects that cannot be disinfectd. Upon every attempt to disinfect or delete a message, create backup copy of it with a full description. Notify the recipient and administrator about the actions performed.



To implement the task, make the following configuration changes:

```
[kavmilter.global]
ScanPolicy=combined
```

```
[kavmilter.engine]
ScanArchives=yes
ScanPacked=yes
ScanCodeanalyzer=yes
```

```
[kavmilter.actions]
DefaultAction=cure
```

```
[kavmilter.backup]
BackupPolicy=info
BackupOption=cured, deleted
BackupDir=/var/db/kav/5.0/kavmilter/backup
```

```
[kavmilter.notifications]
EnableNotifications=on
NotifyRecipient=infected
NotifyAdmin=all
AdminAddresses=admin@localhost
MessageDir=/var/db/kav/5.0/kavmilter/templates/
MessageSubject="Anti-virus notification message"
SendmailPath=/usr/sbin/sendmail
UseCustomTemplates=on
```

CHAPTER 6. ADDITIONAL SETUP

This section describes in detail additional setup of Kaspersky Anti-Virus functionality. Unlike the settings made during the installation process (please see para. 3.3 on p.17) which are required and essential for product functioning, additional setup is performed at the administrator's discretion. Those settings extend product functionality and allow its adjustment for operation within corporate framework of a specific enterprise.

6.1. Integrating the application into your mail system

If the application has not been integrated with Sendmail during installation, use *kavmilter-setup.sh*, a special utility for integrating Kaspersky Anti-Virus with your mail system. Sendmail needs to be restarted after you make necessary configuration changes. You can also roll back to the previous Sendmail configuration if needed.

Use the following command line options:

- add-filter** – Change the Sendmail configuration file;
- del-filter** – Roll back to the previous Sendmail configuration and cancel the latest changes;
- check-filter** – Check the Sendmail configuration for the *kavmilter* filter to make sure that the filter has been added. If the filter has been successfully added, the console will display **yes**; otherwise the value will be **no**.
- set-filter <action>** – Specify further actions to be performed by Sendmail if the *kavmilter* filter is unavailable (specified limits have been exceeded, a cold restart has occurred, etc.). These actions are recorded in the mail system configuration in the filter definition section. Here are some possible actions:
 - tempfail** – The client connection shall fail with error 451 (for example, *451 4.7.1 Please try again later*);
 - reject** – Any incoming messages shall be rejected. The error return code is 554 (for example, *554 Not accepting messages*);
 - pass** – Skip email messages (or forward it to another filter) even if they remained unfiltered by *kavmilter*. This action creates an additional risk for users;

- add-service** – Register *kavmilter* as a service (as SysV or start from *rc.local*);
- del-service** – Cancel registration of *kavmilter* as a service and roll back the changes in configuration files;
- check-service** – Check whether *kavmilter* is registered as a service and was started at operation system startup. If the filter has been registered and started, the console will display **yes**; otherwise, the value will be **no**;
- add-product** – Add the application configuration file *kavmilter.setup* to the */var/d/kav/applications.setup* that is used to retrieve updates;
- del-product** – Delete the application configuration file *kavmilter.setup* from */var/d/kav/applications.setup*;
- check-product** – Check whether the application configuration file *kavmilter.setup* has been added to */var/d/kav/applications.setup*. If the file has been added, you will see **yes** on the console; otherwise, you will see **no** on the console;
- add-webmin-module** – Add *kavmilter* module (included in the package) to the Webmin modules directory and grant the access to it for the superuser (**root**);
- del-webmin-module** – Remove *kavmilter* module from the Webmin modules directory, and rollback all changes in the Webmin configuration concerned with *kavmilter*;
- default-domains** – Specify the domain name and add the domain and all its subdomains to the application configuration file as the value of the **LicensedUsersDomains** parameter. This command line option is available only if you use the licensing policy by mail addresses (about licensing policies, see section 1.2 on page 8).

Sendmail can use the generated *sendmail.cf* file or the *sendmail.mc* file as an application configuration file. Therefore, the file in which to add information about the *kavmilter* filter will be selected automatically, from the following considerations and under the following conditions:

- If the *sendmail.mc* file does not exist or the value of the `USE_SENDMAIL_CF` environment variable is *sendmail.cf* or the binary `m4` file has not been found, the */etc/mail/sendmail.cf* file will serve as the configuration file.
- If the value of the `USE_SENDMAIL_MC` environment parameter is *sendmail.mc*, the application will use *sendmail.mc* as the mail system configuration file. The `INPUT_MAIL_FILTER` directive that defines the use of *kavmilter* as a mail filter is added to the configuration file.
- If both of these configuration files exist and the environment variable does not strictly specify the use of one of these files, the *sendmail.mc* file is used as the configuration file.

If you are running OpenBSD, the Sendmail default configuration file is *localhost.cf*. Kaspersky Anti-Virus makes changes to this configuration file.



Note that if you work under OpenBSD and run Sendmail using another configuration file (**-C** option) or run Sendmail using command line options or only **-bd** option, Sendmail will be started using *sendmail.cf* as a configuration file.

In order to define strictly the Sendmail configuration file use the following command line options:

- sendmail-cf <path_to_file>** – Specify the different *sendmail.cf* file to add and delete the modification concerned with using *kavmltcr* filter or checking it's status.
- sendmail-mc < path_to_file >** – Specify the different *sendmail.mc* file to add or delete the modification concerned with using *kavmltcr* filter or checking it's status.

Two command line options above are used only with **-add-filter**, **-del-filter** and **-check-filter** options.

For example, to use different configuration file *sendmail.cf* and add into it the modification concerned with using *kavmltcr* filter enter the following in the command line:

```
-sendmail-cf </path_to_sendmail.cf> -add-filter
```

If the specified configuration file is not found, the application will return the error code and add, delete or check operation will be canceled.

If you specified both **-sendmail-cf** and **-sendmail-mc** options, the application will use mc-file.

6.2. Installing and uninstalling the remote management module

You can configure Kaspersky Anti-Virus settings and stop/start anti-virus tasks from a remote location using Webmin, a web-based interface. To enable remote management, you should install the Webmin application, install the Kaspersky Anti-Virus module for Webmin, and configure the application.



For instructions on how to install Webmin, refer to the documentation for this product.

To install the Kaspersky Anti-Virus module for Webmin, follow these steps:

1. Open a Webmin page in your browser window.
2. Select **Webmin Configuration** and open the **Webmin Modules** configuration section.
3. In the **Install Module** section, select installation from a file (**from local file**) and specify the full path to *kavmilter.wbm*, the Kaspersky Anti-Virus module for Webmin in the corresponding field.

For Linux, the default path to the module is:

/opt/kav/5.0/kavmilter/web/kavmilter.wbm; for FreeBSD and OpenBSD, the module is located in:

/usr/local/share/kav/5.0/kavmilter/web/kavmilter.wbm.

4. Click **Install Module From File**.

As the result, the KAV for Sendmail module will be added to the **Others** tab.

After installation, open the module (**Others → KAV for Sendmail**), switch to the **Module Config** tab and check whether the paths to the main Kaspersky Anti-Virus files and directories are specified correctly.

Then, you can set up joint operation of the Anti-Virus with the Webmin package. For example, using Webmin, you can limit access to the application by setting up user passwords (about Webmin settings, see the documentation for this product).



Note that this document describes configuration options for Kaspersky Anti-Virus only by editing the configuration file. Configuration and launch of tasks using the Webmin module are not discussed, as the module interface structure is similar to the order of sections and options in the application configuration file.

To get help on configuration options available in Webmin, refer to the Webmin help system. Click the ? button in the upper right corner of the Webmin configuration section to open the help system.

To uninstall the Kaspersky Anti-Virus module for Webmin, follow these steps:

1. Run the Webmin application.
2. Select **Webmin Configuration** and switch to the **Webmin Modules** configuration section.
3. In the **Delete Module** section, select **KAV for Sendmail** and click the **Delete Selected Modules** button.

To reinstall the Kaspersky Anti-Virus module for Webmin, first uninstall it, and then install it again.



If you are reinstalling the module, all paths to the main Kaspersky Anti-Virus files and directories listed on the **Module Config** tab will be saved automatically.

6.3. Defining an e-mail scan policy

Using Kaspersky Anti-Virus, the mail server administrator can customize the anti-virus protection of incoming and outgoing e-mail messages by defining scan policies.

There are two types of policies:

- **message** – Scan the entire message for viruses, regardless of its separate objects (header, body, attachment). This policy also aims to detect viruses that infect and corrupt MIME messages.

If a message is flagged as clean during the scan, its separate objects won't be analyzed. It will be delivered to the recipient. This policy guarantees faster scanning of the clean message than a combined policy (see below).

If a message is flagged as infected and the preset action for such messages is **cure** or **delete**, the program will subsequently analyze all message objects.

- **combined** – Scan both the entire message and then, regardless of the scan results, analyze all message objects for viruses (header, body, and attachment).

To analyze separate message objects, the application first breaks the message down into individual components, scans each component separately, and then restores the message integrity.

The **message** policy is less strict, and, hence, requires less time and resources. The **combined** policy provides the most thorough analysis of e-mail messages.

The type of the policy is defined by the **ScanPolicy** parameter in the **[kavmilter.global]** section.

6.4. Adjusting scan thoroughness

The mail server administrator can adjust the level of anti-virus protection, including the following issues:

- Enable or disable the *heuristic code analyzer* for scanning messages.

The heuristic analyzer is a powerful tool for detecting *modified malicious code* that is similar to a known virus signature, i.e., it recognizes new viruses that are not yet in the database. The use of heuristic technology is defined by the **ScanCodeAnalyser** parameter in the **[kavmilter.engine]** section.

- How long the application should scan a message or a message object.

The maximum scan time for a message or a message object is specified by the **MaxScanTime** parameter and is ten seconds by default. If the application fails to scan the object within this time, it is skipped from scanning.

- How many objects can be scanned for viruses.

The administrator can limit the number of simultaneous scan requests by specifying the **MaxScanRequests** parameter. The default value is zero (unlimited). Use this limitation only if anti-virus scanning has a significant impact on the server.

6.5. Selecting objects to scan

During anti-virus scanning of server mail traffic, the application searches all attachments for viruses.

Since scanning archives and compressed executables requires significant time and server resources, the administrator can decide whether to enable or disable the analysis of such files for viruses.

Scan options for archives and compressed executables are defined by the **ScanArchives** and **ScanPacked** in the **[kavmilter.engine]** section. By default, the application scans these types of files.



Attention! The application is unable to detect viruses in password protected archives! This attachment is flagged as **Protected** and further actions applied to it are defined by the **ProtectedAction** parameter of the **[kavmilter.actions]** section.

6.6. Message status

The scan result generated by the anti-virus kernel is a code (it is the internal application code which is not displayed on the console) that defines the status of analyzed objects:

Clean – The message (or part of it) is free from viruses.

Error – The message (or part of it) is corrupted and an error occurred while scanning it.

Protected – The message (or part of it) is protected with a password or other means of protection. Therefore, it was skipped for anti-virus scanning.

Infected – The message (or part of it) contains malicious code (code sample is available in the anti-virus database or it was detected by the heuristic code analyzer).

If disinfection of an infected object has failed, the object is assigned the **CureFailed** status.

Depending on the message status, the application applies an appropriate action to this message.

6.7. Assigning actions for objects

The following stage of anti-virus protection entails handling messages according to their scan status (see section 6.5 on page 41).

Select one of the following actions to be applied to infected messages:

warn – Replace the infected message with a notification that this object contains a virus.

cure – Disinfect the infected object in the message. If disinfection fails, delete the object and add the corresponding notification to the message.

drop – Accept the message but do not deliver it to the recipient. The application will delete the infected message.

reject – Reject the message and return the corresponding error message to the sender.

skip – Deliver the message to the recipient without treatment.

delete – Delete the infected object and add the corresponding notification to the message.

The action to be applied to infected objects is defined by the **DefaultAction** parameter in the **[kavmilter.action]** section. By default, the application tries to disinfect all infected messages or their objects.

As actions to be applied to protected or encrypted messages, select **skip** or **delete**.

For messages that generated a scan error, select **warn**, **skip**, or **delete**.

The **warn**, **cure**, and **delete** options involve creating a notification that replaces the infected object. The notification describes actions applied to the object. By

default, the original mail message (for the **warn** and **delete** options) or treated message (for **cure**) is attached to the notification sent to the sender, recipient, and administrator. You can customize the notification messages by editing the corresponding templates (see section 6.11 on page 46).

6.8. Selecting objects to be filtered and assigning actions

In addition to processing e-mails and scanning them for viruses, you can filter them. The filtering procedure analyzes message objects and can be performed according to MIME type, name, and size of attachments.



Note that this version of the application filters **message attachments only by headers!** The contents of attachments is not filtered.

Below, we discuss all filtering criteria in more detail:

- To enable message filtering, you MUST SPECIFY AT LEAST ONE NAME OR SIZE OF ATTACHMENTS as the value of the **IncludeMime**, **IncludeName** and **IncludeSize** parameters.
- The type of message objects from the mass of the **IncludeMime**, **IncludeName** and **IncludeSize** objects to be excluded from filtering (for example, those that cannot potentially contain viruses or other malicious code) must be specified as the values of the **ExcludeMime**, **ExcludeName** and **ExcludeSize** parameters in the **[kavmilter.filter]** section. All other types of attachments will be filtered and appropriately handled.

For filtered (blocked) objects, you can assign the following actions (the **FilteredMimeAction**, **FilteredNameAction**, and **FilteredSizeAction** parameters):

delete – Delete the object from the message and add a corresponding notification to the message.

skip – Leave the message as it is and forward it to the mail system for further delivery. In this case, the corresponding information will be recorded in the application report.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return the corresponding error message to the sender.

warn – Replace the whole message with a notification.

rename – Rename the attachment using the following rule: the last letter of the attachment extension is replaced with the "_" character. For example, the *exe* extension will be *ex_*, *com* will be *co_*, etc. This action cannot be applied to the attachments of MIME types.

6.9. Configuring backup options

Backing up messages is an advanced feature of Kaspersky Anti-Virus. Before applying any action to original messages, you can back them up in a special storage. This is a data safety precaution because you can always restore the original information if needed.

The following backup policies are available:

message – Create only a backup copy of the original message.

info – Create a copy of the original message and an information file (default policy).

none – Do not back up messages.

To define backup options, specify the action as the value of the **BackupPolicy** parameter of the **[kavmilter.backup]** section.

Messages with the following statuses can be backed up:

cured – Messages to be disinfected.

deleted – Messages containing at least one part to be deleted.

dropped – Messages that are accepted but will not be delivered.

rejected – Rejected messages.

warning – Messages containing at least one part replaced with a notification.

renamed – Filtered (MIME type) or renamed messages.

all – All the above types of messages.

To define the messages that will be backed up, specify the corresponding value for the **BackupOption** parameter.

All backup copies are stored in the directory defined by the **BackupDir** parameter, and, as was noted above, can also have an additional information file. This file contains information about the sender and recipient, the action applied to the original message, etc.

When Kaspersky Anti-Virus is active, the backup storage can be quickly filled. The storage needs to be periodically cleaned of old and unnecessary backups. This can be done using a special utility, *backup-sweeper.sh*, included into the distributions package. The utility registered with the system as a cron task after the installation can:

- Distribute backup copies in special folders within the storage named as *year-month-date*.
- Check the storage size and notify the administrator when it becomes critical.
- Delete the oldest folders with backup copies.

For this utility, the following command line options are available:

- install** – Create the cron task for this utility under a default user account.
- uninstall** – delete the cron task for this utility under a default user account.
- size** – Define the maximum size of the backup storage. The default size is 10 Mb.
- warn-only** – Ignore the specified maximum storage size; notify the administrator about the current storage size and warn him/her if the size is going to be exceeded.
- delete-oldest** – Delete folders with the earliest creation date if the storage size is critical or exceeded.



The **–warn-only** and **–delete-oldest** options cannot be used concurrently because they are mutually exclusive.

- path** – Change the location of the backup storage by specifying the full path to the new location.

6.10. Configuring database and kernel module updates

Updates are performed every four hours after Kaspersky Anti-Virus for Sendmail with Milter API is installed on the server. During the application installation, the cron task of updating the database and application kernel is created on the server.

As an updating source, the program uses Kaspersky Lab update servers defined by the **UpdateServerUrl** configuration parameter.



If you connect to the Internet using a proxy server, do not forget to specify its IP address as the value of the **ProxyAddress** parameter in the **[updater.options]** section.

If you want to use a local folder as an update source, set the **UseUpdateServerUrl** parameter to **yes**, **UseUpdateServerUrlOnly** to **yes** and specify the full path to the update storage folder (**UpdateServerUrl** parameter).

Before updating, the program always creates a back up copy of the database and kernel modules so that you can easily roll back to them if updating fails. The backup storage is defined by the **BackUpPath** parameter. Thus, you can always roll back to the previous version of the anti-virus database and restore earlier program modules.

If you want to configure general parameters, such as the user name under which updating starts, or perform it manually, use the *keepup2date.sh* script and the following command line options:

- install** – Create the cron task for the utility under the default user account.
- uninstall** – Delete the cron task for the utility under the default user account.
- run** – Start updating the anti-virus database and the kernel. If updating fails, the application will roll back to the previous anti-virus database and modules that were active before updating.
- user** – Specify another user account, differing from the default one, under which the utility will run on the server.

6.11. Customizing notifications

Notification is an e-mail message containing a description of the processed message that is sent to the recipient, sender, and server administrator.

In addition to the description of e-mail messages, the notification also contains descriptions of objects that were deleted for any reason from the message.

You can also attach the original email message to the notification. New email notifications containing only notification text must be created for the administrator and sender.

All notifications that can be customized by the administrator fall into one of the following two groups:

Standard notification is based on a unified template or on different templates. This notification is sent:

- *to the recipient* using Milter API. A new message is not created in this case; the notification text is simply added to the processed message.
- *to the administrator and sender* by the external mail agent, Sendmail. A new notification message is created and the original message can be attached to it. Usually this kind of notification is used to inform the administrator of a **drop** or **reject** action.

Special notification for the administrator is sent to the administrator in case of emergency, for example, if a critical error occurs during Anti-Virus performance. This kind of notification is also sent by the external mail agent, Sendmail.



See section 6.11.2 on page 50 on how to customize notification templates.

The **[kavmilter.notifications]** section of the application configuration contains all notification options.

Notifications can be created upon the following events:

Infected – Give notice about a message that was flagged as **Infected** and one of the following actions was applied to it: **reject**, **drop**, **warn**, **cure**, or **delete**.

Protected – Give notice about a message that is protected, and, hence, skipped from scanning. Because of the message status, the following actions are performed: **delete** or **skip**.

Error – Send notifications about a message that generated a scan error or is corrupted. One of the following actions could be performed: **warn**, **delete**, or **skip**.

Filtered – Give notice about a filtered message that underwent one of the following actions: **delete**, **skip**, or **rename**.

All – Give notice about all the above events.

None – Disable the notification of the party.

If you want to send notifications after any of the above events, assign the appropriate values to the **NotifySender**, **NotifyRecipients**, and **NotifyAdmin** parameters.

Special notifications to the administrator are generated if any of the following events occurs:

Discard – detection of an e-mail message, which has been assigned the **Infected** status upon anti-virus scanning with subsequent application of **reject** or **drop** action to it.

Fault – a critical error in the operation of the application.

Update – receipt of updates to the anti-virus databases.

License expiration – one week (three days or one day) remains before the license validity period expires.

License terms violation – a violation of the license agreement terms has occurred (the limitations on daily traffic volume or the number of e-mail accounts have been exceeded).

The application informs about license expiry or violation of license agreement automatically, no additional setup is required for the notifications; they cannot be disabled by administrators.

In order to enable sending of special notifications to administrators about **Discard**, **Fault** and **Update** events, assign a corresponding value to the **NotifyAdmin** parameters.

The language of notification depends on the encoding specified in the configuration file (**Charset** parameter of the **[kavmilter.notifications]** section). You can also encrypt notifications using the **TransferEncoding** parameters.

To create an English notification text, perform the following steps:

1. Assign the following values to the parameters below:

```
[kavmilter.notifications]
Charset=us-ascii
TransferEncoding=8bit
```

2. Create a notification template in the English language.

6.11.1. Notification templates

The following templates can be used to create notifications (the templates are stored in a directory defined by the **MessageDir** parameter of the configuration file):

Template for notifications about deleted objects – Text added to the original message if one of the message parts is deleted during anti-virus processing or filtering. This text might contain a macro describing the reasons for deletion. The following templates are available:

- *part_infected_deleted* – Text replacing the object that was deleted after an unsuccessful disinfection attempt.
- *part_filtered_deleted* – Text replacing the MIME object that was deleted based on MIME object filtration results.
- *part_filtered_rename* – Text that replaces an original email object, renamed as the result of filtering;
- *part_protected_deleted* – Text replacing a protected object that was deleted because it could not be scanned for viruses;
- *part_error_deleted* – Text replacing the object that generated a scan error and was therefore deleted.

Standard notification template – Text of the notification that is sent to the sender, recipient, and administrator using Milter API. This text might

contain a macro describing the reasons for deletion. The following templates are available:

- *message_default_notify* – Text sent by default to the recipient, sender, and administrators about the actions applied to the message.
- *message_infected_warn* – Text that replaces the infected message.
- *message_filtered_warn* – Text that replaces the filtered e-mail message;
- *message_error_warn* – Text that replaces a message that generated a scan error.
- *message_disclaimer* – Text, added to all processed and generated messages. By default this template includes the following notification: "This message has been scanned by Kaspersky Anti-Virus. For more information please see <http://www.kaspersky.com>".

Detailed notification template – Text notifying a person interested in knowing more about the anti-virus processing of an e-mail message. There are separate templates for notifications sent to the recipient, sender, and administrator. Set the **UseCustomTemplates** parameter to **on** in order to use these templates. The following templates are available:

- *message_sender_notify* – Text of the notification sent to the sender about actions applied to the original message.
- *message_recipients_notify* – Text of the notification sent to the recipient about actions applied to the original message.
- *message_admin_notify* – Text of the notification sent to the administrator about actions applied to the original message.

Special administrator notification template – Text added to special notifications sent upon critical events that require administrator's special attention. The following templates are available:

- *message_admin_discard* – Text notifying the administrator that the original message will not be delivered (**reject** or **drop**);
- *message_admin_update* – the text used to notify the administrator about receipt of updates to the anti-virus databases for the application;
- *message_admin_fault* – Text notifying the administrator that a critical error has occurred while scanning the message.

- *Text notifying the administrator about the license expiration date.* Notifications are sent three times: a week before the license expiration, in three days, and on the expiration date. The notification text or sending options cannot be customized.
- *Administrator notification about a violation of the license agreement* (the limitations on daily traffic volume or the number of e-mail accounts have been exceeded) will be generated and sent automatically. Administrators cannot edit the notification text or control its dispatch.



When the application is started, the presence of all the above templates is verified. If even one of these templates is missing, the application will return an error.

The application also verifies that the size of each template that cannot exceed 8KB.

6.11.2. Customizing notification templates

Kaspersky Anti-Virus gives users the flexibility to customize the default notification templates that will be sent to administrators, senders, and recipients. The templates are customized using a special notification language.

The template language is a set of control statements and macros.

Below, we consider the rules of this language, its syntax and examples of use in detail.

6.11.2.1. Macros

A macro is a substitution element used in email notification templates. In a notification text created using a template, the macro is replaced with a certain value.

The syntax for macros is `%macro_name%`.

If a macro name contains '%', it should be screened (see section 6.11.2.5 on page 54).

Several values can be assigned to a macro. In this case, the simple input of `"%macro_name%"` will output the last assigned value.

To assign several values to one macro, use *iterative statements*.

6.11.2.2. Iteration constructs

An iteration construct (IC) is the main element of the template language.

The syntax for an iteration construct is

```
<FOR INAME IOP IVALUE>BODY</FOR>
```

where:

<FOR – the beginning of IC definition. The < symbol that is not the beginning of an IC definition should be screened (see section 6.11.2.5 on page 54);

INAME – IC name in the format **1*(nchar)*(nchar)**; the maximum length is 64 bytes;

IOP – comparison operation in the format **== | !=**; the maximum length is 2 bytes;

IVALUE – value of IC in the format **1*(vchar)*(vchar)**; the maximum length is 4096 bytes. IC values only work in double quotes. When comparing with a value that contains a quotation mark, use the relevant escape symbol (see section 6.11.2.5 on page 54). Example:

```
<FOR _macro_name_parent_ == "\" value 1\"">
```

> – end of IC definition and the beginning of iterator body. The < symbol that is not the end of IC definition must be hidden (see section 6.11.2.5 on page 54);

BODY – iterator body in the format ***(char)**;

</FOR> – end of the iterator body definition. The < symbol that is not the end symbol of the iterator body definition must be screened (see section 6.11.2.5 on page 54);

... – separator in the format ***()*(t)**

nchar – characters from set a-z, A-Z, 0-9, -, _

vchar – symbols from set nchar, *, ?

char – symbols from the set of values 32 – 255

Example of an iteration construct:

```
<FOR _macro_name_ == "*" >%_macro_name_%</FOR>
```

When executing this construct, the parser transforms the above command into the condition constructs:

```
<FOR _macro_name_ == " value 1">%_macro_name_%</FOR>
```

```
<FOR _macro_name_ == " value 2">%_macro_name_%</FOR>
```

```
<FOR _macro_name_ == " value 3">%_macro_name_%</FOR>
```

```
<FOR _macro_name_ == " value N">%_macro_name_%</FOR>
```

These condition constructs are parsed sequentially.

Thus, iteration constructs are used to distinguish both the single and multiple values of a macro.

For example, if the macro `%FILTERNAME%` has the values of `KAVFilter1`, `KAVFilter2`, `KAVFilter3`, and `SimpleFilter`, then

the construct:

```
<FOR FILTERNAME == "KAVFilter1">%FILTERNAME%</FOR>
```

will produce the text:

```
KAVFilter1
```

the construct:

```
<FOR FILTERNAME ^= "KAVFilter?">%FILTERNAME%, </FOR>
```

will produce the text:

```
KAVFilter1, KAVFilter2, KAVFilter3
```

the construct:

```
<FOR FILTERNAME != "KAVFilter2">%FILTERNAME%, </FOR>
```

will produce the text:

```
KAVFilter1, KAVFilter3, SimpleFilter
```

the construct:

```
<FOR FILTERNAME != "KAV*">%FILTERNAME%, </FOR>
```

will produce the text:

```
SimpleFilter,
```

6.11.2.3. Scope of visibility for an iterative statement

Any iteration construct can have sub-macros, whose values are defined within the scope of visibility for the parent construct only. Iterative statements can be used not only to output particular values of particular macros, but also to define the scope of visibility of sub-macros.

The scope of visibility of a sub-macro is defined by the start and end tags of the condition construct:

```
<FOR _macro_name_parent_ ==
  "_value 1">%_macro_name_child_%</FOR>
```

In the above example, the scope of the macro `_%_macro_name_parent_%` includes all sublevels (between the **FOR** tags) if the macro value is overridden.

6.11.2.4. Variables

Variables provide better flexibility in customizing templates using the Template language.

A variable can be defined within the specified scope of flexibility as follows:

```
<DEF _var_name_ = "_const_value_"/>
```

This variable can be used further as a usual macro without any limitations.

The syntax for a variable definition statement is as follows:

```
<DEF VNAME VOP VVALUE/>
```

where:

`<DEF` – beginning of variable definition statement. The `<` symbol that is not the beginning of the statement must be screened (see section 6.11.2.5 on page 54);

`VNAME` – variable name in the format **1*(nchar)*(nchar)**; the maximum length is 64 bytes;

`VOP` – assignment operation in the format `=`; the length is 1 byte;

`VVALUE` – variable value in the format **1*(vchar)*(vchar)**; the maximum length is 4096 bytes. The value only works in double quotes. If compared with a value that has a quote mark inside, use the escape symbol (see section 6.11.2.5 on page 54). Example:

```
<DEF _value_name_ = "\"_value 1\""/>
```

`>` – end of the variable definition statement. The `>` symbol that is not the end of the variable definition must be hidden (see section 6.11.2.5 on page 54). Unlike the **FOR** statement, the **DEF** statement has no body. Therefore, the tag end bracket should notify the parser that the end tag is missing.

`...` – separator in the format `*()*(\t)`

`nchar` – symbols from set a-z, A-Z, 0-9, -, _

`vchar` – symbols from set nchar, *, ?

If a variable is redefined in its scope, a new value will be substituted after each redefinition. Thus, the statement:

```
<DEF __NAME__ = "NAME_1"/>Now you will see the first
value: %__NAME__%.
<DEF __NAME__ = "NAME_2"/>Now you will see the sec-
ond value: %__NAME__%.
```

will be output as:

```
Now you will see the first value: NAME_1.
Now you will see the second value: NAME_2.
```

A variable can have a macro as its value.

```
<DEF _var_name_ = "% macro name %"/>
```

In this case, the parser will first substitute a macro for a variable and then it will replace the macro with its value in the current scope.

6.11.2.5. Language syntax

Special symbols

- %** marks a macro. The macro should be between two symbols "%". Example: %VIRUSNAME%
- <** opening bracket of a tag.
Example: <FOR FILTERNAME == "KAVFilter1">
- >** closing bracket of a tag.
Example: <FOR FILTERNAME == "KAVFilter1">
- </** opening bracket of an end tag.
Example: </FOR>
- />** closing bracket of the end tag for a construct without a body.
Example: <DEF __NAME__ = "NAME_1"/>
- ** escape symbol. Instructs the parser to treat the following special character as a plain one. Example: \%VIRUSNAME\%
- ==** equal sign: a coincidence in mask or value.
Example: <FOR FILTERNAME == "KAVFilter1">
Example: <FOR FILTERNAME == "KAVFilter*">
- !=** unequal sign: a non-coincidence in mask or value.
Example: <FOR FILTERNAME != "KAVFilter1">

Example: `<FOR FILTERNAME != "KAVFilter*">`

- * Unlimited length of all possible values. It is used only inside tags in comparison with templates.

Example: `<FOR FILTERNAME == "KAV*">`

- ? All possible one-character values. It is used only inside tags in comparison with templates.

Example: `<FOR FILTERNAME == "KAVFilter?">`

- # Comment; the parser ignores all characters after '#' till the end of line.

Reserved keywords

FOR Iteration construct definition.

Example: `<FOR FILTERNAME = "KAVFilter1">`

DEF Variable definition (statement without an end tag). Example: `<DEF`

`__NAME__ = "NAME_1"/>`

Predefined macros

%CRLF% Line feed macro (CR+LF)

%TAB% Tab macro

The processing is performed within a global section (no statement is needed) or within a condition construct:

```
<FOR KAV_LANGUAGE == "5.0" > ... </FOR>
```

Escape sequences

The following sequences can be used to present special characters in the template language:

- To output the '\ symbol in the template text, enter '\\ '.
- If a line is ended with '\, it will be interpreted as a string continued on the following line. Additionally, an escape symbol at the end of the line screens the following EOL, which otherwise would exist in the generated message. Such a line is concatenated with the following one during processing before any other actions performed by the parser. This situation is handled independently by either the escape sequence being met inside a tag or outside a

tag.

See item 1 above if you want to place a ‘\’ at the end of line.

- To output the ‘%’ symbol into the template text, use ‘\%’.
- To output the ‘/’ symbol into the template text, use ‘\V’.
- To output the ‘<’ symbol into the template text, use ‘\<’.
- To output the ‘>’ symbol into the template text, use ‘\>’.
- To output the ‘#’ symbol into the template text, use: ‘\#’.



The template language is case sensitive.

The number of spaces or tab symbols (either their presence or absence) between the language constructs is not regulated.

Reserved keywords must be separated either by white space characters or by the special symbols.

6.11.2.6. Notification macros for the application

Macros can be used in notification templates for either entire messages or their parts. Using macros, you can customize notifications to include additional information on the properties of an original message or object or about actions applied to them.

The administrator can use the following macro in notifications concerning entire messages:

%CLIENT_ADDR% – remote address of the mail client.

%SENDER_ADDR% – sender address.

%RECPT_ADDR% – recipient address.

%HEADERS% – message header.

%BK_ACTION% – actions applied to the message that caused a backup copy to be created (if the application is configured to back up messages).

%BK_LOCATION% – full path to the backup storage (if the storage exists).

%ACTION_LIST% – list containing information about the message and its object and a list of actions applied to them. The information is output as **status action information** for each processed part of the message.

In notifications related to deleted objects from a message, the following macro can be used:

%STATUS% – Object status assigned as the result of scanning or filtering.

%ACTION% – Action applied to the object based on its status.

%INFO% – Information related to the following actions performed:

- List of detected viruses (malicious software) – for infected objects.
- Error code description – for objects that generated a scan error.
- MIME type or attachment name – for filtered objects.

The macros must be specified in the text of notification templates.

6.12. Reporting options

Kaspersky Anti-Virus performance results are logged in the application report. You can store results either in the system log or a separate file (defined by the **LogFacility** parameter of the **[kavmilter.log]** section).

The report records:

Events related to application functionality – All events that occur during application performance, for example, results of message scans.

Events not related to the application functionality – All events that are not directly caused by application performance but provide important information. This information can be the size of the backup storage, program errors, license policy events, etc.

The administrator can decide what information will be included into the report and determine the detail level of the selected data for each.

The types of data and their detail levels are discussed below.

The following information can be logged in the report:

config – Records about the application configuration.

scan – Information about scan results and actions performed.

backup – Data related to backing up e-mail messages.

internal – System messages about application initialization, signals, and processes.

all – All the above types of data.

Each of these categories can be assigned a special detail level:

critical – Critical events that interrupt application operation.

error – Errors that can be fatal or non-fatal for application operation.

warning – Events that reflect unusual situations during application performance. It is useful for the administrator to be aware of such situations.

- notice** – Events related to the application business logics.
- info** – General information concerning the application functionality.
- debug** – Debugging messages.
- all** – All the above levels.

You can combine the information categories and their detail levels. For example, if you want to record all information related to backing up messages, enter the following string into the configuration file:

```
LogOption=backup.all
```

To log only configuration errors, type the following:

```
LogOption=config.error
```

To prevent some information from logging, type, for example, the following:

```
LogOption=-scan.debug
```

The minus before the combination means that this category will be excluded from logging. The remaining information will be logged.

When the **LogFacility** is **file**, all log messages are written in the same file. This file can grow rather large, making recording slower.

To avoid delayed writing, you should enable the rotation of log file (**LogRotate=on**). In this mode, when the report file grows and reaches **RotateSize**, it is copied to *kavmilter.<number>.log* and the initial log truncates to zero. The **RotateRounds** keyword specifies how many different rotation rounds can occur. The *<number>* in rotated log file name specifies what round that is.

Since one of the purposes of log rotation is to avoid logs taking up space, when the **RotateRounds** number or rotations is reached), it starts writing over the old ones.

For example, when **RotateSize=1048576** and **RotateRounds=10** then *kavmilter.1.log*, *kavmilter.2.log* etc. will be created, each **1048576** bytes. After *kavmilter.10.log* is created and it has reached 1048576 bytes, *kavmilter.1.log* is overwritten on the next rotation, and so on.

6.13. Parameters of update report generation

Updating results are logged in a report that can be saved to the system log or as a separate file (**ReportFileName** parameter of the **[updater.report]** section).

You can adjust the amount of output information by changing the *report detail level*.

The **detail level** is a number that defines the degree of specialization of information related to components' operation. Each next level includes data of all previous levels plus some additional information.

The report detail levels are listed in the table below.

Levels	Level name in Webmin	Value
	Fatal errors	Only information regarding critical errors (that cause program termination because a command cannot be executed). For example, the component is infected, the database cannot be loaded, or the license key failed to load.
1	Errors	Information about other errors, including those which are not the cause of components terminating.
2	Info	Important information messages, e.g. whether the component is running or not, the path to the configuration file, information regarding the anti-virus database, license keys, and resulting statistics.
3	Activity	Messages regarding updating the anti-virus database and kernel modules.
10	Debug	Debugging information, e.g. configuration file contents.

Information regarding fatal errors that occur during component operation is output regardless of the preset detail level. The default level is **10**.

To define the report level, set the **ReportLevel** parameter of the **[updater.report]** section to the desired value.

The reports of any detail level are displayed as:

```
[date time level_of_detail] STRING
```

where:

[date time level_of_detail] is the parameter generated by the system. It contains the date and time (in the format set by the administrator) and the report detail level (the first letter of the detail level).



The format of time and date representation can be changed in the **[locale]** section of the configuration file.

STRING – A line of the report.

6.14. Statistics parameters

The application saves the following statistics based on performance results:

E-mail statistics provides general information related to e-mail traffic, including the number of incoming messages scanned by the anti-virus program, the number of protected or corrupted messages, and the overall size of messages.

Resource statistics contains information about resources consumed by scanning and processing e-mail messages. Here the application records the total amount of mail traffic, average scan time for a single message, etc.

Virus statistics displays information on the last ten detected viruses and IP addresses from which most viruses were received.

To determine what type of statistics you want to receive, set the **TrackStatistics** parameter in the **[kavmilter.statistics]** section to one of the following values:

none – Do not save application statistics.

message – Create message statistics.

resources – Create resource statistics.

viruses – Provide virus statistics.

all – Save statistics for messages, resources, and viruses.

The statistics can be displayed in text format or xml format (defined by the **DataFormat** parameter of the **[kavmilter.statistics]** section).

The full path to the statistics file is defined by the **DataFile** parameter.

6.15. Restarting Kaspersky Anti-Virus for Sendmail with Milter API

There are situations when you need to restart the application. Depending on the situation, you can do it using the following methods:

- Configuration changes.

For new changes to take effect, you need to restart Kaspersky Anti-Virus using the *kavmilter* service script. The configuration file with the most recent changes will be reloaded.

To start/stop/restart the application, use the following command line options:

start – Check whether Kaspersky Anti-Virus is running (using the process ID). If the application is running, the *kavmilter* script is stopped. If the application is not running yet, the *kavmilter* script starts and checks for the necessary changes in the Sendmail configuration required for successful integration of the mail system with the anti-virus filter. If these configuration changes are made, the anti-virus filter is initiated. The return code of **0** means that the filter has successfully started.

stop – Check whether Kaspersky Anti-Virus is running (using the process ID). If the application is running, the SIGTERM signal is executed. If the application does not start within three seconds, SIGKILL is executed. The return code of **0** means that the application has been stopped.

restart – Stop and start the application again, according to the procedure initiated by the **stop** and **start** options.

reload – Restart the application configuration and the anti-virus database using the SIGUSR1 signal.

status – Check whether Kaspersky Anti-Virus is running (by the process ID) using the **0** signal and output the application status on the console. If the application is running, the return code is **0**; if it is not running, the return code is **1**.

stats – write *kavmilter* statistics data to the predefined file.

check – Check whether Kaspersky Anti-Virus is running. The procedure is similar to using the **status** option, but the application status is not output to the console. The return codes are the same.

- Problems encountered during program operation.

If you encounter problems when working with the application, for example, I/O errors, library errors, etc., use the *watchdog* utility included into the distribution kit. This utility is installed on your computer together with Kaspersky Anti-Virus.

The *watchdog* utility produces a descendant process to control the parent process. If the application encounters a conflict and stops, the *watchdog* utility restarts the application again.

The maximum number of restarts induced by *watchdog* is defined by the **WatchdogMaxRetries** parameter in the **[kavmilter.global]** section. To disable this parameter, set it to **-1**.

The usage of the *watchdog* utility is regulated by the **-f** command line option. If the application is loaded with this option, *watchdog* is disabled.



The anti-virus database is reloaded immediately after updating. No application restart is needed. The automatic restart of the application is defined by the **PostUpdateCmd** parameter in the **[updater.options]** section.

6.16. Managing the application from the command line

Kaspersky Anti-Virus is managed from the command line using the following command line options:

- h** – Display help on the command line options;
- v** – Display the application version on the console;
- t** – Check the application configuration and verify the configuration operability; display error messages on the console;
- f** – Run the application and work with the current console (do not switch to background mode after startup);
- s <socket>** – Define the socket for data transfer; the format of the **<socket>** parameter is as follows:
 - inet:port@ip-addr** – Use a network socket working via the port **port** and the address **ip-addr**.
 - local:/socket/file/path** – Use a local socket.
- u <user >** – Start the application with the rights of the user **<user>** (for example, with the **root** user rights). By default, the application is started with the rights of the **kav** user;
- g <group>** – Start the application with the rights of the user group **<user>** (for example, with the **root** user group rights). By default, the application is started with the rights of the **kav** user group;
- c <file>** - Use the file **<file>** as the configuration file (default configuration file is */etc/kav/5.0/kavmilter/kavmilter.conf*);
- r <command>** - Execute one of the following commands:
 - reload** – Reload the application configuration file and the anti-virus database; all changes and updates will take effect after restart;

- stats** – Write statistics on application performance to a file defined by the **DataFile** parameter;
- stop** – Stop the application (stop filtering).

6.17. Localization of displayed date and time format

While operating, Kaspersky Anti-Virus compiles reports for each of its components as well as various notifications for users and administrators. Such information is always supplemented with the date and time of its output.

By default Kaspersky Anti-Virus uses the date and time formats corresponding to the system utility date standard:

%H:%M:%S – format of time output (**hh.mm.ss**).
%d/%m/%y – format of date output (**dd.mm.yy**).

The administrator may change the date and time format. Localization of formats is performed in the **[kavmilter.locale]** section of the application configuration file. You can define the following formats:

%I:%M:%S %P – for time output in twelve-hour format (**TimeFormat** parameter).
%y/%m/%d and **%m/%d/%y** – for date output (**DateFormat** parameter) (**yy.mm.dd** and **mm.dd.yy** respectively).

6.18. Controlling the application performance

The application distribution kit includes the *troubleshooter.sh* script, which allows you to troubleshoot all problems encountered during application operation. Using this script, you can also report serious bugs and problems to the Kaspersky Lab' Technical Support.

The information you want to send to Technical Support is compressed and can be encrypted using an open part of the PGP key included into the application distribution kit. You can encrypt files to be sent using any third-party pgp utility (not supplied with the application).

Use the following command line options:

- h** – Display all command line options for the *troubleshooter.sh* script.

- report** – Enable non-interactive operation mode (the default mode is interactive). If there are any problems requiring assistance from the user, the application will use default values to generate the report.
- check** – Automatically check application operation, configuration, and related issues that may cause problems with Anti-Virus functionality.
- to email** – Send requests about encountered problems to another address other than Kaspersky Lab Technical Support.
- key id** – Determine the PGP/GnuPG key for encrypting the archive with information to be sent to the Technical Support.

CHAPTER 7. USING LICENSES

The license for Kaspersky Anti-Virus is issued for a certain period (as a rule, it is one year from the purchase date) and is limited by either the daily mail traffic processed by the application or the number of protected email addresses. In the latter case, the application scans email traffic for the domains specified in the application configuration file and for the servers on which the application is installed.

When the license to use Kaspersky Anti-Virus expires, the application will continue its operation, but it will be unable to further update its anti-virus databases. The Anti-Virus will continue to cure infected objects, but it will use its old databases.

7.1. License keys management

A license key entitles you to use the product and contains all the required data pertaining to the license which you have purchased, such as license type, expiration date, information about distributors, etc.

Besides the right to use the product during the period of license validity, you are entitled to the following:

- round-the-clock technical support;
- daily updates of anti-virus databases;
- software updates (patches);
- new software versions (upgrades);
- timely notifications about new viruses.

These benefits also expire with the license. Kaspersky Anti-Virus will continue scanning your server mail traffic, but it will only use the anti-virus database that was current at the time that the license expired. The updating function will be unavailable. If you try to update the database manually, the application will stop working.

Therefore, it is very important to periodically check the information contained in the license key and keep track of its expiry date.

If your license is limited by mail traffic, it provides protection only of a certain amount of daily mail traffic specified in the license key. If daily mail traffic exceeds the license limit, the administrator will be prompted for the need to purchase a license for the amount of extra traffic.

If your license is issued for the specified numbers of mail addresses, it will extend to all addresses of the domains listed in the application configuration file (**LicensedUsersDomains** parameter) and to all addresses of the server on which Kaspersky Anti-Virus is installed (the server addresses do not belong to the domain). If number of mail addresses exceeds license limit, the administrator will be prompted for the need to purchase a license for the amount of extra traffic.

You must specify the main domain as well as all subdomains of this domain. To list several domains and subdomains, you can use regular expressions with the following syntax:

```
re: domain-regex
```

where:

`re:` is a prefix that defines the regular expression;

`domain-regex` is a POSIX regular expression that specifies the sender's domain or the recipient's domain.

7.1.1. Viewing license key information

You can review information about installed license keys in the logs produced by the *kavmilter* component since both of them load the information from the license keys during start.

In addition, Kaspersky Anti-Virus contains a special *licensemanager* component, which enables you not only to review more detailed information about the keys but also retrieve some analytical data.

All the information may be output to a server console or viewed remotely from any computer on your network through the Webmin interface.



In order to review the information about all installed license keys:

enter the following in the command line:

```
#!/licensemanager -s
```

The following information will be output to the server console:

```
Kaspersky license manager. Version 5.0.0.0/RELEASE
```

```
Copyright (C) Kaspersky Lab. 1998-2003.
```

```
Active key info:
```

```
Product name: Kaspersky Anti-Virus 5 Business Optimal 1  
month
```

```
Key file 00053BC3.key
```

Type: Commercial

Expiration date: 17-11-2003, expires in 60 days

Serial: 02B1-000454-00053BC

Additional key info:

Product name: Kaspersky Anti-Virus 5 Business Optimal 1 month

Key file 00053E3D.key

Type: Commercial

Expiration date: expired

Serial: 02B1-000454-00053E3



In order to review information about an installed license key:

enter, for example, the following text in the command line:

```
#./licensemanager -k 00053E3D.key
```

The following information will be output to the server console:

```
Kaspersky license manager. Version 5.0.0.0/RELEASE
```

```
Copyright (C) Kaspersky Lab. 1998-2003.
```

```
Product name: Kaspersky Anti-Virus 5 Business Optimal 1 month
```

```
Creation date: 23-07-2003
```

```
Expiration date: 21-11-2003
```

```
Serial 02B1-000454-00053E3
```

```
Type: Commercial
```

```
Lifespan: 30
```

7.1.2. License extension

Extending your license to use Kaspersky Anti-Virus restores all the functions of the software. In addition, you will be granted further access to services listed in para. 7.1 on p. 65.

The period of license validity depends on the type of licensing that you selected when purchasing the software (typical period of validity for the license to use Kaspersky Anti-Virus is one year).



In order to extend your license to use Kaspersky Anti-Virus for Send-mail with Milter API, you will need to:

contact the company where you purchased the software and obtain an extension for your license to use Kaspersky Anti-Virus.

or:

extend the license duration directly through Kaspersky Lab by sending a message to the Sales Department (sales@kaspersky.com) or fill out an appropriate form at the **E-Store** section of our site (www.kaspersky.com). After payment you will receive a license key sent to the e-mail address, indicated in your order form.

The license key purchased has to be installed using the *licensemanager* utility (**LicensePath** parameter of the program configuration file).



In order to install a new key, you will need to:

enter, for example, the following in the command line:

```
#./licensemanager -a 00053E3D.key
```

The following information will be output to the server console:

```
Kaspersky license manager. Version 5.0.0.0/RELEASE  
Copyright (C) Kaspersky Lab. 1998-2003.  
Key file 00053E3D.key is successfully registered
```

We recommend updating your anti-virus databases after the procedure.

If you wish to install a new license key before the current one expires, you can install it as a reserved key. A reserved key begins working when the subscription period of the active key expires. The period of a reserved key validity is calculated from the moment of its activation.

A reserved key is installed using the standard method, similar to the installation of the main key. After that, a license key information request will output data to the server console pertaining both to the active and the reserved keys.

7.1.3. License key removal



In order to remove your active key,

enter, for example, the following text in the command line:

```
#./licensemanager -da
```

The following information will be output to the server console:

```
Kaspersky license manager. Version 5.0.0.0/RELEASE  
Copyright (C) Kaspersky Lab. 1998-2003.  
Active key was successfully removed
```



In order to remove your additional key,

enter, for example, the following text in the command line:

```
#./licensemanager -dr
```

The following information will be output to the server console:

```
Kaspersky license manager. Version 5.0.0.0/RELEASE  
Copyright (C) Kaspersky Lab. 1998-2003.  
Additional key was successfully removed
```

CHAPTER 8. COMPATIBILITY WITH OTHER KASPERSKY LAB APPLICATIONS

Kaspersky Anti-Virus 5.0 for Sendmail with Militer API does not cause any compatibility problems when running concurrently with the following Kaspersky Lab applications for Unix/Linux platforms:

- Kaspersky Anti-Virus 5.0.1-0 for Samba Servers installed under Linux SuSE 9.0.
- Kaspersky SMTP Gateway 5.0.0.28 installed under Linux SuSE 9.0.

When using a Kaspersky Lab application for Unix/Linux that has the real-time protection component *kavmonitor*, note that the Sendmail message queue is stored on a hard disk and, when a queued message is accessed, *kavmonitor* intercepts this message. If the message is infected or contains suspicious code, the *kavmonitor* component will block this message and prevent it from delivering. To avoid this problem, we advise that you exclude the Sendmail queue directory from the *kavmonitor* scan area.

During installation of Kaspersky Anti-Virus for Sendmail with Militer API on the same server that Kaspersky Anti-Virus for Unix/Linux is on, the *kavmilter* module is registered in the *kavmonitor* module using a special script. After the registration, the *kavmilter* module receives "permission" from *kavmonitor* to filter Sendmail messages.

To filter email messages, Kaspersky Anti-Virus for Sendmail with Militer API creates a temporary file in the specified directory on the disk. The *kavmonitor* module intercepts this file to perform anti-virus processing. If the file is flagged as infected, the *kavmonitor* component will block this message and prevent Kaspersky Anti-Virus for Sendmail from filtering it (signal **mlfi_abort**).


To avoid this problem, we recommend that you exclude the directory for Kaspersky Anti-Virus for Sendmail temporary files from the *kavmonitor* scan area.



The temporary file directory is defined by the **TempDir** parameter in the **[kavmilter.global]** section.

CHAPTER 9. VERIFYING PROPER OPERATION OF THE ANTI-VIRUS

When the installation and setup of Kaspersky Anti-Virus are complete, we recommend checking the settings and correct operation of the program using a test "virus" and modifications thereof.

The test "virus" has been specifically developed by  (The European Institute for Computer Antivirus Research) for verifying the operation of anti-virus products.

The test "virus" IS NOT A VIRUS and it contains no code that can harm your computer; at that, most products of anti-virus vendors identify it as a virus.



Never use real viruses to test the operational integrity of anti-virus products!

The test "virus" can be downloaded from the official site of **EICAR** at: http://www.eicar.org/anti_virus_test_file.htm. If you have no Internet access, you can create a test "virus" manually. To do so enter the line below in any text editor and save it to a file under the name **eicar.com**:

```
X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-  
FILE!$H+H*
```

The file, which you have downloaded from **EICAR** site or created in a text editor as described above, contains the body of a standard test "virus". The Anti-Virus detects it, assigns the **Infected** incurable type to it and performs an action defined by the administrator for objects belonging to such type.

In order to test the Anti-Virus reaction to the detection of objects belonging to other types, you may modify the contents of the standard test "virus" by adding one of the prefixes below (please see table below).



You can only check correct operation of Kaspersky Anti-Virus using modifications of the EICAR "virus" if your anti-virus databases are dated Oct. 24, 2003 or later (cumulative update – October 2003).

Table 1. Test "virus" modifications

Prefix	Object type
No prefix, standard test "virus"	Infected. The object is not cured.
CORR-	Corrupted.
SUSP-	Suspicious (unknown virus code).
WARN-	Warning (modified code of a known virus).
ERRO-	Error.
CURE-	Cured. The object is cured and, the text in the "virus" body is changed to CURED.
DELE-	The object is deleted automatically.

The first column of the table contains the prefixes, which should be added to the line beginning the standard test "virus" (e.g. CORR-X5O!P%@AP[4\PZX54(P^)7CC)7}\$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!\$H+H*). The second column contains the types of objects identified by the anti-virus application as a result of their addition. Actions performed over each object are defined by Anti-Virus settings selected by the administrator.



It is recommended that you test the operation of your anti-virus program for both incoming and outgoing mail in message bodies and attachments. To test the detection of viruses in message bodies, paste the text of either the standard or modified "virus" into a message body.

CHAPTER 10. FREQUENTLY ASKED QUESTIONS

This chapter is devoted to the most frequently asked questions users have pertaining to the installation, setup, and operation of the Kaspersky Anti-Virus; here we shall try to answer them in detail.



Question: *why does Kaspersky Anti-Virus cause a certain decrease of server performance noticeably loading the CPU?*

Virus detection is a purely computational (mathematical) problem connected with structural analysis, checksum calculation and mathematical data conversions. Therefore processor time is the main resource consumed by the anti-virus software. At that each new virus added into the anti-virus database increases the overall scanning time. That is a forced compensation for security and safety of your data.

Unlike other anti-virus products, which expediate scanning by excluding less easily detectable or less frequent (in the geographic location of the anti-virus vendor) viruses from their databases, as well as file formats that require complicated analysis (e.g. PDF), Kaspersky Lab believes that the purpose of its anti-virus is to provide real anti-virus security for its users instead of imaginary, since you cannot be semi-protected. In reality, "partial protection" is even worse than no protection at all (because in the latter case users take personal precautions).

Kaspersky Anti-Virus makes its users feel maximum protection. Of course, Kaspersky Anti-Virus software package allows experienced users to accelerate anti-virus scanning at the cost of overall security by disabling scanning of various file types, but we do not recommend doing so for users who want the best protection.



Question: *why is a license key required? Can my Anti-Virus work without it?*

Kaspersky Anti-Virus will not work without a license key.

If you haven't yet made a decision whether you wish to buy Kaspersky Anti-Virus, we can provide to you a trial key, which will work for two weeks or one month. The key will be blocked when that period expires.



Question: *what happens when a license to use the product expires?*

When your license to use Kaspersky Anti-Virus expires the software will continue operation, however, you will have no access to anti-virus database updates. The anti-virus software will cure infected objects, but it will use its old databases only.

Downloading anti-virus databases from the Kaspersky Lab site using the *keepup2date* component will be impossible. If you download the databases by means other than the *keepup2date* component, Kaspersky Anti-Virus will not work.

Therefore, we cannot guarantee your protection against new virus infections.



Question: *my Anti-Virus does not function.*

What should I do?

First of all, check whether a solution to your problem is described in this document and, in particular, in this section or on our site.

We also recommend contacting the company from which you purchased your Kaspersky Anti-Virus or writing a e-mail to the Technical Support Service (support@kaspersky.com).

The following steps will facilitate prompt processing of your inquiry:

1. Please indicate the operating system of your server, the name of the component which you cannot setup, and the problem in the subject of your message . For example:
Linux, anti-virus database updating fails.
2. Please use plain text messages. Messages in HTML format are harder to read.
3. At the beginning of the message, indicate precisely the version of your operating system, Kaspersky Anti-Virus installation package, and the name of your license key file.
4. Describe your problem briefly but exactly. Please keep in mind that the Support team has no information about your problem when your letter arrives, and they will only be able to help you if they understand it completely and manage to reproduce the scenario.
5. Please send the following data (compressed into one archive) to the Technical Support Service:
 - logs of the Anti-Virus components;

- your license key.
6. Any of the following must be mentioned in your message:
- a SCSI controller;
 - a very old or a new CPU or a multiprocessor configuration;
 - Less than 64 MB or more than 2 GB of RAM.
7. Please indicate approximate amount of daily traffic and whether you have peak loads.



Question: *can an intruder deliberately replace anti-virus databases?*

An intruder may be able to download the anti-virus databases from the Kaspersky Lab site and copy them to the directory where they should be stored, but the Kaspersky Anti-Virus will not use such databases during its work!

All anti-virus databases have unique signatures verified by Kaspersky Anti-Virus while accessing the bases. If a signature does not correspond to the one assigned at the Kaspersky Lab and the databases are dated after the date of license expiry, Kaspersky Anti-Virus will not use such databases.



Question: *are the X architecture processors supported (PowerPC, SPARC, Alpha, PA-RISC etc.)?*

The current version of the product does not support processors of those types.



Question: *will the Kaspersky Anti-Virus work with my Linux distribution?*

Kaspersky Anti-Virus has been tested with RedHat, Debian and SuSE distributions and Kaspersky Anti-Virus packages have been compiled specifically for those distributions.



If your distribution is 100% compatible with a supported one (for example, ASPLinux is compatible with Red Hat Linux), then the probability of critical problems is very low.

Users of distributions that are not included into the list supported by Kaspersky Lab may experience incorrect product operation. This is determined first of all by the operating system specifics. For example, your OS distribution may use a different library version or its system initializa-

tion scripts may have a non-standard location. In such cases, Kaspersky Lab Technical Support will be unable to help you.



Question: how do I decompress a *.tgz* or *.tar.gz* archive?

Archives belonging to *.tgz* or *.tar.gz* types are decompressed using the following command:

```
tar zxvf <archive_name>
```

APPENDIX A. ADDITIONAL INFORMATION

A.1. Application configuration file

This appendix provides detailed explanation of every section of the *kavmilter.conf* file that is the default configuration file for Kaspersky Anti-Virus for Sendmail with Milter API on your server.

The default values given here are recommended by Kaspersky Lab experts.

The **[kavmilter.global]** section contains general parameters required for application startup and operation:

RunAsUid=kav – User identification number assigned to the account under which the program is running.

RunAsGid=kav – Group identification number assigned to the group under which the program is running.

ServiceSocket=innet:1052@127.0.0.1 – Socket type (local or network) on which Kaspersky Anti-Virus listens. The parameter can be in the following format: *socket_type:parth_to_socket*, for example:

inet:port@ip-address – listen on specified port;

local:/path/to/socket – listen on local Unix socket.

WatchdogMaxRetries=10 – Maximum number of retries to restart Kaspersky Anti-Virus using *watchdog*. The value of -1 corresponds to the unlimited number of retries.

ScanPolicy=message – E-mail scan policy. The following are possible values of this parameter:

message – Scan the entire message for viruses; if a virus is found, start scanning message objects (header, body, and attachment).

combined – Scan the entire message and each message part for viruses.

TempDir=/var/db/kav/5.0/kavmilter/tmp/ – Directory for storing temporary files.

LicensedUsersDomains=localhost – List of domains that contain accounts licensed for the use of Kaspersky Anti-Virus for Sendmail with Milter API. This option is available only if your license is issued for a certain number of mail addresses.

AddXHeaders=yes – Adds a header with application information to a filtered email message.

AddDisclaimer=no – Add a disclaimer text to each processed or generated message. You can customize this text by editing *message_disclaimer* template. The disclaimed text is added as a text part at the end of message, and does not affect or change the content of the original message.

The **[kavmilter.engine]** section includes parameters defining the scanning procedure:

MaxScanRequests=0 – Maximum number of requests for scanning messages. If the parameter is 0, the number of requests is unlimited.

MaxScanTime=10 – Maximum number of seconds during which the program scans one object (a message or a message object). If the value is exceeded, the program returns an error.

ScanArchives=yes – Scan archives. To disable this mode, set the parameter to **no**.

ScanPacked=yes – Scan packed executables. To disable this mode, set the parameter to **no**.

ScanCodeanalyzer=yes – Scan using an heuristic code analyzer to detect malicious programs, virus modifications, and unknown viruses. To disable this mode, set the parameter to **no**.

The **[kavmilter.actions]** section contains options to be applied to infected message objects:

DefaultAction=cure – Apply the default action to an infected message object. Set one of the following actions as the default:

warn – Replace the infected message with a notification about a virus found.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return the corresponding error code to the sender.

cure – Disinfect the infected message object. If disinfection is impossible, send a notification that this message contains a virus.

delete – Delete the infected object and send the corresponding notification.

ProtectedAction=skip – Apply the selected action to a password protected message object that could not be scanned for viruses. Select one of the following actions:

skip – Skip the protected object.

delete – Delete the protected object from the message and attach the corresponding notification.

ErrorAction=skip – Apply the selected action to the corrupted object in a message that could not be scanned because of an error. Select one of the following options:

warn – Replace the message with a notification message.

skip – Skip object for anti-virus processing.

delete – Delete the object from the e-mail message and attach the corresponding notification.

VirusNameList – List viruses that require special actions to be applied to objects infected with these viruses.

VirusNameAction=drop – Actions to be applied to a message or its object if it is infected with a virus listed in the **VirusNameList** parameter.

warn – Replace the infected message with a notification that this message contains a virus.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return an error code to the sender.

delete – Delete the infected object and add the corresponding notification to the original message.

UsePlaceholderNotice=yes – Attach a notification about the deleted object.

The **[kavmilter.backup]** contains options for creating backup copies before applying any actions to e-mail messages:

BackupPolicy=info – Backup policy. You can select one of the following actions:

none – Do not create a backup copy.

message – Create a backup copy of the message each time before applying an action to the message.

info – Create a backup copy and an information file with the message description.

BackupOption=all – Type of messages for which to create backups. Select one of the following values:

filtered – E-mail message filtered by at least one object of this message.

infected – Infected messages.

deleted – Messages containing at least one object that will be deleted.

warning – Messages containing at least object that will be replaced with a notification.

dropped – Messages that were accepted by the server but will not be delivered to the recipient.

rejected – Rejected e-mail messages.

error – Messages that cannot be scanned due to an error.

all – All the above types of messages.

BackupDir=/var/db/kav/5.0/kavmilter/backup – Directory for storing backup copies of messages.

The **[kavmilter.filter]** section defines rules for message filtering:

IncludeMime – MIME type of message attachments to be filtered.

ExcludeMime – MIME type of message attachments to be excluded from anti-virus scans.

IncludeName – Attachment name to be filtered.

ExcludeName – Attachment name to be excluded from virus scans.

IncludeSize – Size of e-mail attachments to be filtered.

ExcludeSize – Size of e-mail attachments to be excluded from virus scans.

FilteredMimeAction=skip – Action to be applied to the attachment of the **IncludeMime** type. Select one of the following options:

skip – Skip this object for scanning.

delete – Delete the object of this MIME type from the message and add a corresponding notification.

warn – Replace the message with a notification.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return an error code to the sender.

FilteredNameAction=skip – Action to be applied to the attachment name of the **IncludeName** type. Select one of the following values:

skip – Skip the object with this name.

replace – Rename the object.

delete – Delete the object with this name from the message and add a corresponding notification.

warn – Replace the message with a notification.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return an error code to the sender.

FilteredSizeAction=skip – Action applied to an attachment if its size complies with the value of the **IncludeSize** parameter. Select one of the following options:

skip – Do not process objects of this size; transfer them to the mail system for delivery.

delete – Delete objects of this size and add the corresponding notification to the original message.

warn – Replace the message with a notification.

drop – Accept the message but do not deliver it to the recipient.

reject – Reject the message and return an error code to the sender.

The **[kavmilter.notifications]** section contains standard notification options:

EnableNotifications=yes – Enable Notifications mode. To disable sending notifications, set the parameter to **no**.

NotifySender=none – Notify the sender upon detection of e-mail messages (message objects) with this status. Select one of the following values:

filtered – E-mail message or its object has been filtered.

infected – E-mail message or its object is infected with a virus.

protected – A part of the message is protected and cannot be scanned for viruses.

error – E-mail message or its object is corrupted or generated a scan error.

all – All the above types of messages.

none – Disable the notification.

NotifyRecipients=infected – Notify the recipients upon detection of e-mail messages (message objects) with this status. The status options are the same as for the **NotifySender** parameter.

NotifyAdmin=none – Notify the administrator of discarded messages or special conditions. The status options are:

discard – E-mail message was discarded through the **reject** or **drop** action.

fault – The program generated a fault or ended unexpectedly.

update – New anti-virus database update was downloaded.

none – Disable the notification.

AdminAddresses=postmaster@localhost – E-mail address of the mail server administrator. You can specify several addresses, separated with a space.

MessageDir=/var/db/kav/5.0/kavmilter/templates/ – Directory for storing notification templates.

MessageSubject=Anti-virus notification message – Header of a standard notification added to the **Subject** field.

RejectReply=Message rejected because it contains malware – Header of the notification about a rejected message.

Charset=us-ascii – Name of the character set for notifications.

TransferEncoding=7bit – Value of the notification encryption algorithm.

SendmailPath=/usr/sbin/sendmail – Full path to the binary Sendmail used to send additional notifications.

UseCustomTemplates=off – Enable using custom templates for generating notifications. To enable the mode, set the parameter to **on**.

SenderSubject – Subject of the sender notification.

ReceiverSubject – Subject of the recipient notification.

AdminSubject – Subject of the administrator notification.

The **[kavmilter.log]** section includes reporting options:

LogFacility=syslog – File that will store program reports. Select one of the following values:

syslog – Log results in the system log.

file – Log results in a special file. If this option is selected, specify the full name to the report file as the **LogFilepath** parameter.

LogFilepath=/var/log/kav/5.0/kavmilter/kavmilter.log – Path to the report file. This parameter is ignored if the system log is selected for logging reports.

LogOption=all – category of messages and events to be recorded in the report. Select one of the following values:

internal – System messages: application initialization, signals, and processes.

scan – Anti-virus scan results and applied actions.

config – Messages about application configuration.

backup – Messages related to backing up e-mail messages.

all – All the above types of messages.

Each category of messages logged in the report can have several detail levels: *debug*, *info*, *notice*, *warning*, *error*, *critical*, or *all*.

You can combine both message types and detail levels as follows:

```
LogOption=backup.all
```

```
LogOption=config.error
```

```
LogOption=scan.all
```

```
LogOption=-scan.debug
```

The prefix before each combination means that this information is skipped from recording.

LogRotate=on – Enable report file rotation mode (is used only if **LogFacility=file**). To disable this mode set this parameter to **off**.

RotateSize=1048576 – Report file size in bytes. When it is reached, new report file is created.

RotateRounds=10 – Number of report files created during rotation. When this number is reached, the application starts to overwrite the oldest one.

The **[kavmilter.statistics]** section includes statistics options:

TrackStatistics=none – Enable recording the following statistics:

none – Do not record any statistic information on program performance.

message – Log all statistics related to e-mail messages (total number of scanned messages, number of incoming messages, etc.).

resources – Log all statistics related to resources required by the program operation (total number of mail traffic, single message scan time, etc.);

viruses – Record statistics about detected viruses (last ten viruses detected in messages, etc.);

all – Log statistics of all the above types.

DataFormat=text – Display statistics in the following formats:

text – text format as *category.field=value*.

xml – root element is statistic, children elements are category and field, and value is body element.

DataFile=/var/opt/kav/log/statistics.data – Full path to the file that stores statistics.

The **[path]** section contains parameters that define the paths to critical directories.

BasesPath=/var/opt/kav/5.0/kavmilter/bases/ – Full path to the anti-virus database.

LicensePath=/var/opt/kav/5.0/kavmilter/licenses/ – Full path to the directory where license keys are stored.

The **[locale]** section contains options for displaying date and time in the reports and statistics.

DateFormat=%d-%m-%Y – Format of date displayed in the report.

TimeFormat=%H:%M:%S – Format of time displayed in the report.

The **[updater.path]** section contains parameters that define the paths to critical directories used for updating.

UploadPatchPath=/var/db/kav/5.0/kavmilter/patches/ – Full path to the directory storing kernel updates.

BackUpPath=/var/db/kav/5.0/kavmilter/bases/backup/ – Full path to the backup storage of the anti-virus database and kernel modules.

AVBasesTestPath=/opt/kav/5.0/kavmilter/bin/avbasesetest – Full path to the *avbasesetest* utility used to check the anti-virus database integrity. If updates are not corrupted, they are copied from the temporary folder to the directory storing the anti-virus database.

The **[updater.options]** section contains parameters that define the paths to critical directories used for updating.

UseUpdateServerUrl=yes – Do not use as an updating source the Kaspersky Lab server defined by the **UpdateServerUrl** parameter.

UpdateServerUrl=ftp://downloads1.kaspersky-labs.com – The address of a Kaspersky Lab server used as a source for updating the database and kernel modules.

PostUpdateCmd=/opt/kav/5.0/kavmilter/bin/kavmilter -r reload – Restart the application after updating is complete.

RegionSettings=Russia – region name.

ConnectTimeout=30 – Number of seconds within which the application can attempt to connect to the update source.

ProxyAddress – IP address of a proxy server if it is used for Internet connection. By default, the value is not set.

PassiveFtp – Use passive FTP mode when downloading updates via FTP. To enable the mode, set the parameter to **yes**. To disable it, set the parameter to **no**.

The **[updater.report]** section contains parameters that define the paths to critical directories used while updating.

Append=no – Create a new log every time the *keepup2date* component starts. All previous logs will be deleted. To enable the mode when messages are added to the existing report file, set the parameter to **yes**.

ReportFileName=/var/log/kav/5.0/kavmilter/keepup2date.log – Name of the *keepup2date* report file.

ReportLevel=10 – Report detail level.

A.2. Error return codes

Errors may occur during application performance. Listed below are possible error return codes.

Internal errors:

- 1 – invalid log option;
- 2 – daemon failed;

- 3 – not enough privileges to change uid
- 4 – not enough privileges to change gid;
- 5 – cannot spawn filter child
- 6 – maximum number of retries for restarting application exceeded;
- 7 – endpoint file already exists and is not a socket;
- 8 – failed to register the application as a kavmilter filter;
- 9 – failed to initialize the KAV engine;
- 10 – failed to start main kavmilter loop;
- 255 – unidentified error.

Engine errors:

- 51 – Error initializing the database manager;
- 52 – Database load error;
- 54 – Failure to start the local scan manager;
- 63 – Memory allocation error for engine.

Configuration errors:

- 100 – parameter not found;
- 101 – unknown parameter;
- 102 – bad parameter type;
- 103 – parameter already exists;
- 104 – end of parameters array;
- 105 – section not found;
- 106 – unknown section;
- 107 – error reading configuration file;
- 108 – parameters not set.

APPENDIX B. MALWARE IN UNIX ENVIRONMENT

Viruses are much less frequent in Unix systems than, for example, under Windows, due to some peculiarities of those platforms. Trojan horses and network worms are less rare.

Malicious programs spread through networks using various methods, including software "holes". Let us review in more detail the types of malware existing geared toward Unix and the methods used for system infection.

B.1. Viruses

A virus is a program (certain executable code and/or instructions), capable of producing its copies (which do not have to be totally identical to the original) and embedding them into different objects and/or resources of computer systems, networks, etc. without user's knowledge. Such copies also have the ability to spread further.

A study of the virus environment reveals that viruses in Unix belong as a rule to the file type and record their code to executable files or create phantom files.

The following subclasses are defined according to their functioning algorithm:

- *Memory-resident (TSR) virus* means a virus which leaves a resident part after infection in the RAM system; this residual part subsequently intercepts system calls to infectable objects and actually incorporates into them. Resident viruses remain in the memory and active until computer power-down or an operating system reboot.
- *non-resident virus* is a virus which does not infect computer memory and remains active for a limited time. Some viruses leave small resident programs in memory which do not spread the virus.

As a rule, viruses in Unix are not dangerous – their influence is limited to a decrease of free disk space, graphic, sound and other effects. Some of them are harmless altogether, since they do not alter computer functionality in any way, except for a decrease of free disk space caused by their spread.

Some examples of Unix viruses are described below.

ELF_SNOOPY is a virus that infect executable Unix files.

Algorithm of virus activity: it finds all executable files present on a workstation, renames them to files with an .X23 extension and moves to a created /E directory. Then the virus copies itself to the original

files and changes their attributes to **777**. At the same time it creates user **snoopy** with the rights **777** as well in the main password list of the infected workstation.

Linux.Bliss is a group of non-resident viruses that infect Linux executables; these viruses are written in GNU C and have ELF format.

Algorithm of virus activity: once commenced the virus searches for executable files on a workstation and infects them by shifting file contents lower and writing itself into the space thus freed, appending an identification string to the end of file. Virus activity is limited by the rights of the user who started it (only accessible files are infected). If a user has system privileges, the virus may spread to the whole computer.

Linux.Diesel is a harmless non-resident Linux virus that infects Linux executables.

Algorithm of virus activity: once started, the virus reads its code from the carrier file, searches for Linux executables in system subdirectories and writes itself into the middle of each file, thus increasing the size of the last section.

Linux.Silov is a harmless non-resident Linux virus that infects Linux executables in ELF format.

Algorithm of virus activity: it uses two methods of file infection: resident and non-resident. The resident method: the virus remains in system memory and infects files in background. Non-resident method: the virus searches for executable files on disk and infects them.

Linux.Winter is a harmless non-resident Linux virus. It is very small – just 341 bytes.

Algorithm of virus activity: after start the virus receives control, searches for ELF files (Linux executables) in the current directory and infects them.

B.2. Trojan software

Trojan horse software is a program that performs actions which the user has not authorized. Upon commencement, a Trojan horse installs itself in the system and then begins monitoring it; the user receives no notifications about actions of the Trojan in the system. Such a computer is open for remote control.

Trojan software spreads through networks.

One typical representative of the Trojan software family in Unix is **TROJ_IRCKILL** – a Trojan which is actually a collection of software tools for user disconnection from IRC channels. The collection integrates four attack

utilities: FLOOD, MCB (Multiple Collide BOTs), SUMO BOTs, and FLASH – a special “flood” type for use in UNIX.

The FLASH attack type is used for direct modem disconnection by sending a **ping** command with “incorrect” data in a certain sequence to a certain IP address. The user’s modem will interpret the data as a command to disconnect and the user will be disconnected from the Internet. However, that attack type is applicable to some modem types only.

MCB attacks are performed via IRC channels. At the moment when IRC servers are unable to synchronize with each other (net split), the Trojan imitates a duplicate user’s name (nickname). After IRC servers achieve synchronization the said name becomes invalid and such user gets disconnected from an IRC channel.

Attacks by FLOOD BOTS/SUMO BOTS are also used in IRC networks, “producing” numerous users with random nicknames. The attack is used to “flood” an IRC channel or a user, who sends or receives chat messages until user computer reaches its bandwidth limit. Then such user will also be disconnected from an IRC channel.

Root kit is a collection of tools used by hackers in order to receive root access to a remote computer. It uses standard Unix programs – ps and ls. The only efficient method of recovery for computers hacked using the Root kit is restoration of important data from a regular backup copy, complete deletion of hard disk contents and reinstalling the OS.

B.3. Network worms

A malicious program belonging to this category does not add itself to executable objects, but instead copies itself to network resources. The class title is based exactly on the ability of worms to “crawl” through networks and other informational channels.

They penetrate computer memory from computer networks, calculate network addresses of other computers and send their own copies to those addresses.

Programs of that class may sometimes have work files on system disks, but they can also use no resources on a computer at all (except for RAM).

Worm.Linux.Ramen is the first known worm that infects RedHat Linux systems. It infects remote Linux systems (RedHat Linux), exploiting the buffer overrun problem. The software “hole” allows the virus to send executable code to a remote computer and its execution there – unnoticed by an administrator (user).

Infection source: .tgz archive from a network.

Activity algorithm: using the buffer overrun problem, the worm sends a short portion of its code to a remote computer. When the main worm component (*start.sh* file) starts, it opens a connection that successively downloads other components; they determine the addresses of the systems being attacked, then using buffer overrun breach send a worm loader there, which in its turn completes loading and starts the main portion of the worm code. The main page of a server is replaced with an HTML file containing the following text: "RameN Crew – Hackers looooooooooooooove noodles". Finally, the worm sends an e-mail message to two addresses, restarts the system and begins scanning the Internet again.

The worm also adds a command for starting its main file to the */etc/rc.d/rc.sysinit* system initialization file. As a result, the worm is started during all subsequent starts of an infected system.

Worm.Linux.Lion is an Internet worm that attacks Linux servers. It uses a security breach in the BIND DNS service to penetrate computer systems.

Activity algorithm: the worm scans the Internet, searching for systems with root access vulnerability. When it discovers such a system, the worm infects it, collects information on it (IP address, logins, passwords) in the *mail.log* file and then sends it to *1iOnsniffer@china.com* e-mail address.

In addition, the worm attempts to contact the *www.51.net* site (51.net domain is registered in China) via the Internet and download the file *crew.tgz* from it. The archive is then uncompressed on the infected computer with subsequent installation of routines making the infected system scan in turn the global network resources in search of new victims.

mIRC.Acoragil and **mIRC.Simpsalapim** are the first known mIRC worms. Their names originate from the code words used by the worms: if the text sent to a channel by any user contains the *Acoragil* line, then all users infected with the **mIRC.Acoragil** worm will be automatically disconnected from that channel. The same happens with the **mIRC.Simpsalapim** worm – it reacts in a similar manner to the **Simpsalapim** line.

Infection source: through the network, using mIRC commands, the worms send their code in *SCRIPT.INI* file to each new user connecting to the channel.6

Activity algorithm: the worms contain a Trojan code portion. **mIRC.Simpsalapim** contains a code for IRC channel capture: if the mIRC channel owner is infected, input of the password (*ananas*) will enable a hacker to seize control of the channel.

mIRC.Acoragil sends DOS, Windows or UNIX system files according to received code words. Some code words are chosen in such a manner as to attract no attention of the victim – *hi* or *the*. One of worm modifications sends the UNIX password file to the hacker.

Worm.Linux.Adm is an Internet worm that infects Linux systems. The worm sends a small portion of its code to remote computers, runs it there, downloads its main portion, and executes it.

Infection source: via networks; the worm spreads by sending its copies (infecting remote Linux systems) using a hole in Linux protection (the so-called "buffer overrun" breach). The hole allows sending executable code to a remote computer and its execution there – unnoticed by an administrator (user).

APPENDIX C. KASPERSKY LAB

Kaspersky Lab Ltd. was founded in 1997. Now it is the best known Russian developer of a wide range of software products for data security, producing systems for protection from viruses, unsolicited e-mail (spam) and hacker attacks.

Kaspersky Lab is an international enterprise. The central office is located in Russia, and local representative offices have been opened in the UK, France, Germany, Japan, Benelux, China, Poland, Romania, and the USA (California). A new company branch, the European Anti-Virus Research Centre, was opened in France. Our partner network unites more than 500 companies all over the world.

The Kaspersky Lab team today is represented by more than 250 highly qualified professionals; nine of them have MBA diplomas, fifteen have PhD degrees, and two are members of the esteemed Computer Anti-virus Researcher's Organization (CARO).

The main corporate values at Kaspersky Lab are the unique knowledge and experience accumulated by its staff during more than fourteen years of constant struggle against viruses. Thanks to our non-stop analysis of virus activities, we can predict the tendencies in the development of malware and provide users with solid protection against new types of threats. That advantage serves as the basis for the products and services offered by Kaspersky Lab. We are always one step ahead of our competitors providing our customers with the best protection.

Years of hard work have made our company the leader in the development of anti-virus technologies. Kaspersky Lab was the first to develop many modern standards applied in the sphere of anti-virus software. Our main corporate product, Kaspersky Anti-Virus, ensures reliable control over all potential sources of computer viruses: workstations, file servers, mail gates, firewalls and handheld computers. An easy-to-use interface help users automate anti-virus protection of computers and corporate networks to a maximum degree. Many Western companies producing anti-virus software use the Kaspersky Anti-Virus kernel in their products, such as Nokia ICG (USA), F-Secure (Finland), Aladdin (Israel), Sybari (USA), G Data (Germany), Deerfield (USA), Alt-N (USA), Microworld (India), BorderWare (Canada).

Customers of Kaspersky Lab receive a wide range of services, which guarantee that its products will function uninterruptedly and conform exactly to any specific business requirements. We design, implement and perform maintenance of corporate anti-virus complexes. Our anti-virus database is updated twice daily. We also provide round-the-clock technical support in several languages to our users.

C.1. Other Kaspersky Lab products

Kaspersky Anti-Virus® Personal

Kaspersky Anti-Virus Personal provides anti-virus protection for home computers running Windows 98/ME, Windows 2000/NT, and Windows XP from all known virus types including Trojan software, Internet worms, script viruses, dangerous ActiveX and Java applets, etc. Kaspersky Anti-Virus Personal constantly monitors all the potential sources of virus penetration: e-mail, the Internet, floppy disks, CD-ROMs, etc. Kaspersky Anti-Virus Personal includes a program for daily updates via Internet. A unique second-generation system of heuristic analysis is capable of neutralizing unknown viruses efficiently. A simple and convenient user interface allows settings to be configured effortlessly and makes using the program convenient.

Kaspersky Anti-Virus Personal provides:

- **on-demand scanning** of local drives on the user's request;
- **automatic real-time scanning** for virus presence of all files being used;
- **a mail filter**, which performs background scanning of all incoming and outgoing e-mail messages.

Kaspersky Anti-Virus Personal supports more than 700 formats of archived and compressed files and ensures that their contents will be automatically scanned, as well as removing dangerous code from ZIP archives.

Kaspersky Anti-Virus® Personal Pro

The package is designed specifically for full-scale anti-virus protection of home computers functioning under Windows 98/ME, Windows 2000/NT, and Windows XP with business applications included in MS Office 2000. Kaspersky Anti-Virus Personal Pro package includes a program for daily updates for its anti-virus database and application modules via Internet. A unique second-generation system of heuristic analysis is capable of neutralizing unknown viruses efficiently. An easy-to-use interface allows settings to be configured effortlessly and makes using the program convenient.

Apart from the functions of automatic real-time and on-demand scanning on the user's request and the mail filter, Kaspersky Anti-Virus Personal Pro includes a **behavioral blocker**, which guarantees 100% protection from macro viruses.

Kaspersky® Anti-Hacker

Kaspersky Anti-Hacker is a personal firewall which provides full-scale protection for computers running Windows from unauthorized access of its data and from network attacks of hackers from a LAN or the Internet.

Kaspersky Anti-Hacker monitors network activity through the TCP/IP protocol for all applications installed on your computer. If suspicious actions of some application are detected, the program will notify you thereof and, if necessary, block network access for that application. As a result, it establishes confidentiality of the information inside your computer.

Owing to the SmartStealth™ technology, it becomes seriously complicated to detect your computer from the outside. As a result, hackers lose your computer as a suitable object for an attack. At the same time, the invisibility mode does not have any adverse effect on your use of the Internet: the software provides standard transparency and information accessibility.

Kaspersky Anti-Hacker also blocks typical network hacker attacks and monitors port scan attempts.

The program supports simplified administration based on five security modes. By default it uses a self-training mode, which helps you tune the security system to your own reactions to various events. This mode allows "fine-tuning" of the firewall for an individual user and a specific computer.

Kaspersky® Security for PDA

Kaspersky Security for PDA provides reliable anti-virus protection for the data stored in handheld computers functioning under Palm OS or Windows CE as well as the information transferred from a PC or any extension card, ROM files and databases. The program includes an optimal collection of anti-virus tools:

- **anti-virus scanner**, which scans data (stored both in a PDA or on extension cards of any type) at user's option;
- **anti-virus monitor**, which intercepts virus programs in the data transferred during the process of synchronization using the HotSync™ technology or synchronization with other PDA.

The program also provides for protection against unauthorized access to the data stored in a handheld computer by means of encrypted access to the device itself with encryption of all the data stored in a PDA and its extension cards.

Kaspersky Anti-Virus® Business Optimal

The software complex is a unique configurable solution for anti-virus protection of small and medium businesses.

Kaspersky Anti-Virus Business Optimal provides full-scale antivirus protection³ for:

- workstations running Windows 98/ME, Windows NT/2000/XP Workstation, Linux;

³ Depending upon the package type.

- file servers running Windows NT 4.0 Server, Windows 2000 Server/Advanced Server, Novell Netware, FreeBSD and OpenBSD, Linux;
- Microsoft Exchange 5.5/2000/2003, Lotus Notes/Domino, Postfix, Exim, Sendmail and Qmail mail gateways.

Kaspersky Anti-Virus Business Optimal also includes a system for centralized deployment and management, Kaspersky Administration Kit.

You can select anti-virus software personally according to the operating systems and applications you use.

Kaspersky® Corporate Suite

Kaspersky Corporate Suite is an integrated system which provides data security for your corporate network, no matter how large or complex it is. Software components included in the suite are designed for protection of all the nodes in a corporate network. They are compatible with the majority of current operating systems and software applications; the components are integrated through a centralized control system and have a unified user interface. The software suite ensures that you will establish a security system completely compatible with the system requirements of your network.

Kaspersky Corporate Suite provides for full-scale antivirus protection of:

- *workstations* running under Windows 98/ME, Windows NT/2000/XP Workstation, and Linux;
- *file servers* running under Windows NT 4.0 Server, Windows 2000, 2003 Server/Advanced Server, Novell Netware, FreeBSD, OpenBSD, and Linux;
- Microsoft Exchange 5.5/2000/2003, Lotus Notes/Domino, Postfix, Exim, Sendmail and Qmail *mail gateways*;
- *data traffic through firewalls*;
- handheld computers.

Kaspersky Corporate Suite also includes a system for centralized deployment and management - Kaspersky Administration Kit.

You can select anti-virus software personally according to the operating systems and applications that you use.

Kaspersky® Anti-Spam

Kaspersky Anti-Spam is the first Russian software package for protection against unsolicited mail (spam) for medium and small businesses. The product combines the revolutionary technique of linguistic text analysis, all modern methods of e-mail filtering (including RBL lists and formal message signs) and a unique

collection of services, which allow identification and deletion of up to 95% of unwanted traffic.

Kaspersky Anti-Spam is a filter installed on the "entry" to the corporate network, and it scans the incoming mail for spam. The software is compatible with any e-mail system installed on the customer's network and it can be installed either on an existing mail server or on a specifically dedicated one.

The high efficiency of the program is achieved through daily automatic updates of the content filtration database, which uses samples provided by the linguistic laboratory experts.

C.2. Our contact information

If you have any questions, you may contact our distributors or Kaspersky Lab Ltd. directly. You will always receive detailed consultations over the phone or e-mail. Complete and comprehensive answers will be provided for all of your questions.

Technical support:	Please find technical support information at http://www.kaspersky.com/supportinter.html
General information:	WWW: http://www.kaspersky.com http://www.viruslist.com
E-mail:	sales@kaspersky.com

APPENDIX D. LICENSE AGREEMENT

Standard End User Licence Agreement

NOTICE TO ALL USERS: CAREFULLY READ THE FOLLOWING LEGAL AGREEMENT ("AGREEMENT") FOR THE LICENCE OF SPECIFIED SOFTWARE ("SOFTWARE") PRODUCED BY KASPERSKY LABS. ("KASPERSKY LABS").

IF YOU HAVE PURCHASED THIS SOFTWARE VIA THE INTERNET BY CLICKING THE ACCEPT BUTTON, YOU (EITHER AN INDIVIDUAL OR A SINGLE LEGAL ENTITY) CONSENT TO BE BOUND BY AND BECOME PARTY TO THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, CLICK THE BUTTON THAT INDICATES THAT YOU DO NOT ACCEPT THE TERMS OF THIS AGREEMENT, AND DO NOT INSTALL THE SOFTWARE.

IF YOU HAVE PURCHASED THIS SOFTWARE ON A PHYSICAL MEDIUM, HAVING BROKEN THE CD SLEEVE, YOU (EITHER AN INDIVIDUAL OR A SINGLE LEGAL ENTITY) HAVE CONSENTED TO BE BOUND BY THIS AGREEMENT. IF YOU DO NOT AGREE TO ALL OF THE TERMS OF THIS AGREEMENT, DO NOT BREAK THE CD SLEEVE, DOWNLOAD, INSTALL, OR USE THIS SOFTWARE. YOU MAY RETURN THIS SOFTWARE FOR A FULL REFUND. YOUR RIGHT TO RETURN AND REFUND EXPIRES 30 DAYS AFTER PURCHASE FROM AN AUTHORISED KASPERSKY LABS DISTRIBUTOR OR RESELLER. THE RIGHT TO RETURN AND REFUND EXTENDS ONLY TO THE ORIGINAL PURCHASER.

All references to "Software" herein shall be deemed to include the software activation key ("Key Identification File") with which you will be provided by Kaspersky Lab as part of the Software.

1. Licence Grant. Subject to the payment of the applicable licence fees, and subject to the terms and conditions of this Agreement, Kaspersky Lab hereby grants you the non-exclusive, non-transferable right to use one copy of the specified version of the Software and the accompanying documentation (the "Documentation") for the term of this Agreement solely for your own internal business purposes. You may install one copy of the Software on one computer, workstation, personal digital assistant, or other electronic device for which the Software was designed (each a "Client Device"). If the Software is licensed as a suite or bundle with more than one specified Software product, this licence applies to all such specified Software products, subject to any restrictions or

usage terms specified on the applicable price list or product packaging that apply to any such Software products individually.

1.1 Use. The Software is licensed as a single product; it may not be used on more than one Client Device or by more than one user at a time, except as set forth in this Section.

1.1.1 The Software is "in use" on a Client Device when it is loaded into the temporary memory (i.e., random-access memory or RAM) or installed into the permanent memory (e.g., hard disk, CD-ROM, or other storage device) of that Client Device. This licence authorizes you to make only as many back-up copies of the Software as are necessary for its lawful use and solely for back-up purposes, provided that all such copies contain all of the Software's proprietary notices. You shall maintain records of the number and location of all copies of the Software and Documentation and will take all reasonable precautions to protect the Software from unauthorised copying or use.

1.1.2 If you sell the Client Device on which the Software is installed, you will ensure that all copies of the Software have been previously deleted.

1.1.3 You shall not decompile, reverse engineer, disassemble or otherwise reduce any part of this Software to a humanly readable form nor permit any third party to do so. The interface information necessary to achieve interoperability of the Software with independently created computer programs will be provided by Kaspersky Lab by request on payment of its reasonable costs and expenses for procuring and supplying such information. In the event that Kaspersky Lab notifies you that it does not intend to make such information available for any reason, including (without limitation) costs, you shall be permitted to take such steps to achieve interoperability, provided that you only reverse engineer or decompile the Software to the extent permitted by law.

1.1.4 You shall not make error corrections to, or otherwise modify, adapt, or translate the Software, nor create derivative works of the Software, nor permit any third party to copy the Software (other than as expressly permitted herein).

1.1.5 You shall not rent, lease or lend the Software to any other person, nor transfer or sub-licence your licence rights to any other person.

1.1.6 You shall not use this 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.

1.2 Server-Mode Use. You may use the Software on a Client Device or on a server ("Server") within a multi-user or networked environment ("Server-Mode") only if such use is permitted in the applicable price list or product packaging for the Software. A separate licence is required for each Client Device or "seat" that may connect to the Server at any time, regardless of whether such licenced Client Devices or seats are concurrently connected to or actually accessing or using the Software. Use of software or hardware that reduces the number of Client Devices or seats directly accessing or utilizing the Software (e.g.,

"multiplexing" or "pooling" software or hardware) does not reduce the number of licences required (i.e., the required number of licences would equal the number of distinct inputs to the multiplexing or pooling software or hardware "front end"). If the number of Client Devices or seats that can connect to the Software exceeds the number of licences you have obtained, then you must have a reasonable mechanism in place to ensure that your use of the Software does not exceed the use limits specified for the licence you have obtained. This licence authorises you to make or download such copies of the Documentation for each Client Device or seat that is licensed as are necessary for its lawful use, provided that each such copy contains all of the Documentation's proprietary notices.

1.3 Volume Licences. If the Software is licensed with volume licence terms specified in the applicable product invoicing or packaging for the Software, you may make, use or install as many additional copies of the Software on the number of Client Devices as the volume licence terms specify. You must have reasonable mechanisms in place to ensure that the number of Client Devices on which the Software has been installed does not exceed the number of licences you have obtained. This licence authorizes you to make or download one copy of the Documentation for each additional copy authorized by the volume licence, provided that each such copy contains all of the Document's proprietary notices.

2. Duration. This Agreement is effective for one (1) year unless and until earlier terminated as set forth herein. This Agreement will terminate automatically if you fail to comply with any of the conditions, limitations or other requirements described herein. Upon any termination or expiration of this Agreement, you must immediately destroy all copies of the Software and the Documentation. You may terminate this Agreement at any point by destroying all copies of the Software and the Documentation.

3. Support.

(i) Kaspersky Lab will provide you with the support services ("Support Services") as defined below for a period of one year following:

(a) Payment of its then current support charge, and:

(b) Successful completion of the Support Services Subscription Form as provided to you with this Agreement or as available on the Kaspersky Lab website, which will require you to produce the Key Identification File which will have been provided to you by Kaspersky Lab with this Agreement. It shall be at the absolute discretion of Kaspersky Lab whether or not you have satisfied this condition for the provision of Support Services.

(ii) Support Services will terminate unless renewed annually by payment of the then-current annual support charge and by successful completion of the Support Services Subscription Form again.

(iii) By completion of the Support Services Subscription Form you consent to the terms of the Kaspersky Lab Privacy Policy, which is attached to this Agreement,

and you explicitly consent to the transfer of data to other countries outside your own as set out in the Privacy Policy.

(iv) "Support Services" means

(a) Daily updates of the anti-virus database;

(b) Free software updates, including version upgrades;

(c) Extended technical support via e-mail and phone hotline provided by Vendor and/or Reseller;

(d) Virus detection and disinfection updates 24 hours per day.

4. **Ownership Rights.** The Software is protected by copyright laws. Kaspersky Lab and its suppliers own and retain all rights, titles and interests in and to the Software, including all copyrights, patents, trademarks and other intellectual property rights therein. Your possession, installation, or use of the Software does not transfer any title to the intellectual property in the Software to you, and you will not acquire any rights to the Software except as expressly set forth in this Agreement.

5. **Confidentiality.** You agree that the Software and the Documentation, including the specific design and structure of individual programs and the Key Identification File, constitute confidential proprietary information of Kaspersky Lab. You shall not disclose, provide, or otherwise make available such confidential information in any form to any third party without the prior written consent of Kaspersky Lab. You shall implement reasonable security measures to protect such confidential information, but without limitation to the foregoing shall use best endeavours to maintain the security of the Key Identification File.

6. Limited Warranty

(i) Kaspersky Lab warrants that for 90 days from first download or installation the Software will perform substantially in accordance with the functionality described in the Documentation when operated properly and in the manner specified in the Documentation.

(ii) You accept all responsibility for the selection of this Software to meet your requirements. Kaspersky Lab does not warrant that the Software and/or the Documentation will be suitable for such requirements nor that any use will be uninterrupted or error free;

(iii) Kaspersky Lab does not warrant that this Software identifies all known viruses, nor that the Software will not occasionally erroneously report a virus in a title not infected by that virus;

(iv) Your sole remedy and the entire liability of Kaspersky Lab for breach of the warranty at paragraph (i) will be at Kaspersky Lab option, to repair, replace or refund of the Software if reported to Kaspersky Lab or its designee during the

warranty period. You shall provide all information as may be reasonably necessary to assist the Supplier in resolving the defective item;

(v) The warranty in (i) shall not apply if you (a) make or cause to be made any modifications to this Software without the consent of Kaspersky Lab, (b) use the Software in a manner for which it was not intended or (c) use the Software other than as permitted under this Agreement;

(vi) The warranties and conditions stated in this Agreement are in lieu of all other conditions, warranties or other terms concerning the supply or purported supply of, failure to supply or delay in supplying the Software or the Documentation which might but for this paragraph (v) have effect between the Kaspersky Lab and you or would otherwise be implied into or incorporated into this Agreement or any collateral contract, whether by statute, common law or otherwise, all of which are hereby excluded (including, without limitation, the implied conditions, warranties or other terms as to satisfactory quality, fitness for purpose or as to the use of reasonable skill and care).

7. Liability

(i) Nothing in this Agreement shall exclude or limit Kaspersky Lab's liability for (i) the tort of deceit, (ii) death or personal injury caused by its breach of a common law duty of care or any negligent breach of a term of this Agreement, (iii) any breach of the obligations implied by s.12 Sale of Goods Act 1979 or s.2 Supply of Goods and Services Act 1982 or (iv) any liability which cannot be excluded by law.

(ii) Subject to paragraph (i), the Supplier shall bear no liability (whether in contract, tort, restitution or otherwise) for any of the following losses or damage (whether such losses or damage were foreseen, foreseeable, known or otherwise):

(a) Loss of revenue;

(b) Loss of actual or anticipated profits (including for loss of profits on contracts);

(c) Loss of the use of money;

(d) Loss of anticipated savings;

(e) Loss of business;

(f) Loss of opportunity;

(g) Loss of goodwill;

(h) Loss of reputation;

(i) Loss of, damage to or corruption of data, or:

(j) Any indirect or consequential loss or damage howsoever caused (including, for the avoidance of doubt, where such loss or damage is of the type specified in paragraph (ii), (a) to (ii), (i).

(iii) Subject to paragraph (i), the liability of Kaspersky Lab (whether in contract, tort, restitution or otherwise) arising out of or in connection with the supply of the Software shall in no circumstances exceed a sum equal to the amount equally paid by you for the Software.

8. The construction and interpretation of this Agreement shall be governed in accordance with the laws of England and Wales. The parties hereby submit to the jurisdiction of the courts of England and Wales save that Kaspersky Lab as claimant shall be entitled to initiate proceedings in any court of competent jurisdiction.

9. (i) This Agreement contains the entire understanding between the parties with respect to the subject matter hereof and supersedes all and any prior understandings, undertakings and promises between you and Kaspersky Lab, whether oral or in writing, which have been given or may be implied from anything written or said in negotiations between us or our representatives prior to this Agreement and all prior agreements between the parties relating to the matters aforesaid shall cease to have effect as from the Effective Date. Save as provided in paragraphs (ii) - (iii), you shall not have any remedy in respect of an untrue statement made to you upon which you relied in entering into this Agreement ("Misrepresentation") and Kaspersky Lab shall not have any liability to the other than pursuant to the express terms of this Agreement.

(ii) Nothing in this Agreement shall exclude or limit Kaspersky Lab's liability for any Misrepresentation made thereby if aware that it was untrue.

(iii) The liability of Kaspersky Lab for Misrepresentation as a fundamental matter, including a matter fundamental to the maker's ability to perform its obligations under this Agreement, shall be subject to the limitation of liability set out in paragraph 7(iii).