Dr.WEB®

Anti-virus + Anti-spam
for UNIX Mail Servers

Defend what you create

Administrator Manual

Anti-virus Dr.Web for UNIX mail servers
Version 6.0.2
Administrator Manual
02.03.2012

Doctor Web Head Office
2-12A, 3rd str. Yamskogo polya
Moscow, Russia
125124

Web site: www.drweb.com
Phone: +7 (495) 789-45-87

Refer to the official web site for regional and international office information.

Doctor Web

Doctor Web develops and distributes Dr.Web® information security solutions which provide efficient protection from malicious software and spam.

Doctor Web customers can be found among home users from all over the world and in government enterprises, small companies and nationwide corporations.

Dr.Web antivirus solutions are well known since 1992 for continuing excellence in malware detection and compliance with international information security standards. State certificates and awards received by the Dr.Web solutions, as well as the globally widespread use of our products are the best evidence of exceptional trust to the company products.

We thank all our customers for their support and devotion to the Dr.Web products!

Table of Contents

Introduction

This Manual describes the following **Dr.Web®** solutions for mail processing and filtering in UNIX® based systems:

- **anti-virus + anti-spam Dr.Web for UNIX mail servers;**
- **anti-spam Dr.Web for UNIX mail servers;**
- **anti-virus Dr.Web for UNIX mail servers;**
- **anti-virus + anti-spam Dr.Web for UNIX mail gateways;**
- **anti-spam Dr.Web for UNIX mail gateways;**
- **anti-virus Dr.Web for UNIX mail gateways.**

In fact all these solutions differ from each other only by the combination of installed modules and plug-ins. Hereinafter all of them will be referred to as **Dr.Web for UNIX mail servers**. Depending on modules installed, interaction with different mail transfer systems can be established, successful mail protection from viruses and spam can be achieved and also the software can operate as mail gateway.

Each solution is also presented in three variations for major UNIX based operating systems ("*UNIX systems*" hereinafter): Linux, FreeBSD and Solaris x86.

As far as all these solutions for various UNIX based operating systems ("UNIX systems" hereinafter) differ from each other only slightly, then hereinafter all of them will be referred to as **Dr.Web for UNIX mail servers**. Critical differences will be described in separate chapters and paragraphs.

Manual is designed for the person responsible for anti-virus protection and security ("Administrator" hereinafter).

Features of the protection of electronic mail in UNIX systems:

- Examination of all incoming SMTP-traffic for viruses, their diagnostics and neutralization.

  In most cases, viruses are not made directly for UNIX systems. Through electronic mail ordinary Windows viruses are

distributed, including macro-viruses for Word, Excel and other office applications.

- Filtration of spam and other undesired correspondence.

Dr.Web for UNIX mail servers solution performs both tasks mentioned above.

A range of problems to be solved by **Dr.Web for UNIX mail servers** solution is only limited by the selection of installed plug-ins: special libraries responsible for the direct processing of messages.

Also two SDKs are available:

- SDK providing tools for the development of new modules performing functions of **Receiver**/**Sender** components and providing support to new MTAs.
- SDK providing tools for the development of new plug-ins for mail processing.

The following modules are included into the **Dr.Web for UNIX mail servers** solution:

- Console antivirus scanner **Dr.Web Scanner** used to detect and cure viruses on the local machine and shared directories;
- Background module **Dr.Web Daemon** used as an external antivirus filter;
- Auxiliary module **Dr.Web Monitor** used to run and terminate other **Dr.Web** modules in the necessary order;
- **Dr.Web Control Agent** used for gathering statistical information and integration with **Dr.Web Enterprise Security Suite**;
- Perl script **Dr.Web Updater** used to automatically update virus databases;
- **Dr.Web MailD** module is used for analyzing and processing the mail traffic and enables integration of other **Dr.Web** modules with Sendmail, Postfix, Courier, Qmail, CommuniGate Pro, ZMailer, Exim mail transfer systems. **Dr.Web MailD** can also operate as part of anti-virus network under control of **Dr.Web Enterprise Security Suite**.

In the present manual basic steps of setup, adjustment and startup procedures of **Dr.Web for UNIX mail servers** solution will be discussed. This manual contains information on the following topics:

- General product description;
- Installation of **Dr.Web for UNIX mail servers** solution;
- Running **Dr.Web for UNIX mail servers** solution;
- Usage of updating package **Dr.Web Updater**;
- Usage of **Dr.Web Agent**;
- Usage of console scanner **Dr.Web Scanner**;
- Usage of background on-demand scanner **Dr.Web Daemon**;
- Usage of **Dr.Web Monitor**;
- Configuration of **Dr.Web for UNIX mail servers** solution.

At the end of this Manual you will find technical support service contact information.

Doctor Web products are being constantly developed. Add-ons to virus databases are released daily or even several times a day. New versions of programs appear. Diagnostics techniques and methods of anti-virus protection, as well as integration with other applications of UNIX systems are improved regularly. Besides that, the list of applications compatible with **Doctor Web** products is constantly expanding, therefore some settings and functions described in this Manual will slightly differ from current program version. To get up-to-date program information please refer to documentation files included in delivery package.

Terms and abbreviations

This guide utilizes the following content conventions and signs:

Convention	Description
Bold	Names of buttons and other elements of the graphical user interface (GUI), and required user input that must be entered exactly as given in the guide.
Green and bold	Names of **Doctor Web** products and components.

Convention	Description
Green and underlined	Hyperlinks to topics and web pages.
`Monospace`	Code examples, input to the command line and application output.
Italics	Placeholders which represent information that must be supplied by the user. For command-line input, it indicates parameter values.
CAPITAL LETTERS	Names of keys and key sequences.
Plus sign ('+')	Indicates a combination of keys. For example, ALT+F1 means to hold down the ALT key while pressing the F1 key.
Exclamation mark	A warning about potential errors or any other important comment.

To define directories to which components of the software complex are installed, specific conventional symbols are used: `%bin_dir`, `%etc_dir` and `%var_dir`. Depending on the OS being used, these symbols refer to the following directories:

for Linux and Solaris:

```
%bin_dir = /opt/drweb/
%etc_dir = /etc/drweb/
%var_dir = /var/drweb/
```

for FreeBSD:

```
%bin_dir = /usr/local/drweb/
%etc_dir = /usr/local/etc/drweb/
%var_dir = /var/drweb/
```

System requirements

Dr.Web for UNIX mail servers is compatible with

- Linux distributions that meet requirements listed in Compatibility with Linux Distributions;
- FreeBSD version 6.x and higher for Intel x86 and amd64 platform;
- Solaris version 10 for Intel x86 and amd64 platform.

Platform must be fully compatible with x86 processor architecture in 32-bit or 64-bit modes. 64-bit systems must support execution of 32-bit applications.

For example:

To enable support for 32-bit applications in systems based on Debian/Ubuntu Linux `ia32-libs` library must be installed, for systems based on ALT Linux `i586-glibc-core` library must be installed.

Dr.Web for UNIX mail servers hardware requirements are the same as requirements for command line interface of compatible operating system.

Installation requires 190megabytes.

GUI installer requires X Window System. For automatic execution of interactive configuration script in graphical mode, `xterm` or `xvt` terminal emulator must be installed.

Also, following packages must be installed in your system:

- `base64`
- `unzip`
- `crond`

Depending on the range of problems to be solved by **Dr.Web for UNIX mail servers** solution and total system load during its operation, hardware requirements may vary widely.

Compatibility with Linux Distributions

Dr.Web for UNIX mail servers solution is compatible with x86 and x86-64 Linux distributions.

Requirements for versions of kernel and glibc library depend on type of the installation package:

- epm distribution package for UNIX systems (Linux x86) - kernel version 2.4.x, glibc 2.2 (not recommended) or higher, or: 2.6.x, glibc version 2.3 or higher;
- epm distribution package for UNIX systems (Linux x86-64) - kernel version 2.6.x, glibc version 2.3 or higher;
- rpm packages (rpm-apt, urpmi, yum, zypper) - kernel version 2.6.18 or higher, glibc 2.5 version or higher;
- deb packages (apt) - kernel version 2.6.26 or higher, glibc 2.7 version or higher;

Dr.Web for UNIX mail servers solution was tested to work on following distributions:

- ALT Linux versions 4 - 6 (32-bit), versions 5-6 (64-bit);
- Arch Linux (64-bit);
- ASPLinux versions 12 - 14 (32-bit);
- Debian versions 3.1 - 6 (32-bit), versions 4-6 (64-bit);
- Fedora 14 (64-bit);
- Gentoo *;
- Mandriva Linux versions 2009, CS4 (32-bit), 2010.x (64-bit);
- Mandrake 10;
- openSUSE versions 10.3-11 (32/64-bit);
- PCLinuxOS 2010;
- Red Hat Enterprise Linux (RHEL) versions 4 - 6 (32-bit), versions 5 - 6 (64-bit);
- Suse Linux Enterprise Server versions 9 - 11 (32-bit), versions 10-11 (64-bit);
- Ubuntu versions 7.04 - 11.04;

Other distributions that meet above requirements are also supported but were not tested. If you have any compatibility issues with your Linux distribution, please contact technical support at http://support.drweb.com/request/.

Package files location

Dr.Web for UNIX mail servers solution is installed by default to `%bin_dir`, `%etc_dir` and `%var_dir` directories. OS-independent directory tree is created in these directories:

- `%bin_dir` – executable modules of **Dr.Web for UNIX mail servers** and updating package **Dr.Web Updater** (perl script `update.pl`);
- `%bin_dir/lib/` – anti-virus engine as loadable library (`drweb32.dll`). In the same subdirectory various service libraries for packages of **Dr.Web for UNIX mail servers** solution can reside;
- `%etc_dir/agent/` – additional configuration files for **Dr.Web Agent** module;
- `%etc_dir/monitor/` – additional configuration files for **Dr.Web Monitor** module;
- `%var_dir/bases/*.vdb` – databases of known viruses;
- `%etc_dir` – configuration files of **Dr.Web for UNIX mail servers** solution: `drweb32.ini`, `agent.conf`, `monitor.conf`, `drwebd.enable` and `drweb-monitor.enable` (two last-mentioned are for adjustment of daemons' operation);
- `%bin_dir/lib/ru_scanner.dwl` – language file for **Dr.Web Scanner** package;
- `%bin_dir/web/` – a Webmin package with Web interface for **Dr.Web for UNIX mail servers** ;
- `%bin_dir/scripts/` – interactive configuration script, migration script for updating configuration from older versions;

- `%etc_dir/maild/templates/` — templates for notifications to be sent on detection of malicious objects or errors during scanning and file processing;
- `%bin_dir/doc/` — documentation. All documentation is presented in plain text files in English and Russian (KOI8-R and UTF-8 encodings) languages;
- `%var_dir/infected/` – quarantine directory to move infected or suspicious files to, if such reaction is specified in settings for **Dr.Web for UNIX mail servers** software system components.

Configuration files

All **Dr.Web for UNIX mail servers** settings are stored in configuration files which you can use to configure all software components. Configuration files are plain text files in the following format:

```
--- beginning of the file ---
[Section 1 name]
Parameter1 = value1, ..., valueK
...
ParameterM = value1, ..., valueK
...
[Section X name]
Parameter1 = value1, ..., valueK
...
ParameterY = value1, ..., valueK
--- end of the file ---
```

If a line begins with ";" or "#" characters, it is considered to be comment line. These lines are skipped when reading parameters from the configuration file.

If any parameter is commented out or not specified, it does not mean that this parameter has no value. In this case the hard coded

default value will be used. Only few parameters are optional or do not have default values. All such cases will be described separately.

If any parameter is incorrect, respective **Dr.Web** module would output error message to console (if module is running in foreground mode) and log file and terminate.

When any unknown parameter is found in configuration file, **Dr. Web for UNIX mail servers** modules continue execution and output warning to the log file.

Parameter value may be enclosed in quotation marks (and must be enclosed in quotation marks if it contains white spaces). Some parameters can have several values. These values can be delimited by comma, or each value can be set in a separate line of configuration file. To see if parameter can have multiple values check parameter description in this Manual

Examples:

Multiple values delimited by commas:

```
Names = XXXXX, YYYYY
```

Multiple values set in several strings:

```
Names = XXXXX
Names = YYYYY
```

All parameters in this Manual are described in the following way:

```
ParameterName = {parameter type | possible values}
```

Parameter description.

{possibility to have multiple values}.

Default value:

```
ParameterName = {value | empty}
```

Parameters are described in the order they are presented in the

respective configuration file.

Parameter value types:

- `numerical value` – parameter value is a positive integer;
- `time` – parameter value is set in time measurement units. Value is a positive integer followed by the time measurement unit type: `s` – for seconds, `m` – for minutes, `h` – for hours, `d` – for days. Types are case insensitive. If time measurement unit type is omitted, value is considered to be set in seconds.

  Examples: 30s, 15m
- `size` – parameter value is set in memory capacity measurement units (either disk space or memory capacity). Value is a positive integer followed by the memory capacity measurement unit type: `b` – for bytes, `k` – for kilobytes, `m` – for megabytes, `g` – for gigabytes. Types are case insensitive. If memory capacity measurement unit type is omitted, value is considered to be set in bytes.

  Examples: 20b, 15k
- `permissions` – numerical value indicating access privileges to files and folders. `4` denotes read access (`r`), `2` denotes write access (`w`), `1` denotes execution access (`x`). When privileges are specified, all these numbers are summed up for each category of users (file owner, group of file owners and all other users, which are neither file owners, nor members of the corresponding group).

  Examples: `755 (-rwxr-xr-x)`, `644 (-rw-r--r--)`
- `path to file/directory` – parameter value is a path to some file or directory in the file system.
- `actions` – actions to be performed with objects induced a reaction of **Dr.Web for UNIX mail servers**. Set of acceptable actions for different parameters may vary, and in this case it is clearly specified in the description of each parameter separately.There are two groups of action-type values: mandatory and optional. For the **Dr.Web MailD** you can specify one mandatory and up to three optional actions. Mandatory action always stands first on the list. For the **Dr.**

Web Scanner only one action can be specified.

Available mandatory actions:

- `Cure` – cure infected object and repack the message;
- `Remove` – remove infected object and repack the message;
- `Discard` – decline a message without sending a notification to the sender about it;
- `Pass` – pass the message;
- `Reject` – decline a message with a notification to the sender about it;
- `Tempfail` – notify the sender that the message temporary can not be delivered.

Available optional actions:

- `Quarantine` – move message to quarantine;
- `Redirect [(address[| address|...])]` – redirect a message to address(es) specified in brackets. If no address (es) are specified, the message is sent to address defined in **`RedirectMail`** parameter value in `[MailD]` Section. Several addresses can be specified, delimited by "`|`" symbol;
- `Notify` – send a report about detected threats. Message processing continues;
- `Score (SCORE)` – add `SCORE` to the message score. `SCORE` value may be negative;
- `Add-header (HEADER)` – add some `HEADER` to a message. `HEADER` is specified as `[NAME:]BODY`, where `NAME` is header name (`X-DrWeb-MailD` by default), and `BODY` is header value. If ";" character is used in header it has to be escaped; otherwise configuration can be interpreted incorrectly.

Escaping characters

To escape punctuation marks it is necessary to use 3 backslashes "\".

Example:

```
EmptyFrom = continue, add-header (header: Empty header\\\; spam)
```

Parentheses escaping inside the header is possible by using single backslash.

Example:

```
ProcessingErrors = tempfail,add-header (\(header:header\))
```

It is also possible to escape the whole expression by enclosing the entire header in double quotation marks: `"add-header (Header)"`.

Example:

```
ProcessingErrors = tempfail,"add-header (header:(spam))"
```

To escape double quotation marks triple backslash is used.

Examples:

```
EmptyFrom = continue,"add-header(header[X-Header]:new\\\"header\\\")"
EmptyFrom = continue,add-header(header\[X-Header\]:new\\\"header\\\")
```

Available actions for the **Dr.Web Scanner**:

- `Cure` – cure infected object;
- `Delete` – delete infected object;
- `Rename` – rename infected object;
- `Move` – move the file to quarantine;
- `Ignore` – skip the file;
- `Report` – only output information to log.

`address` – socket addresses of **Dr.Web for UNIX mail servers** components and external packages. These parameters are specified as `TYPE:ADDRESS`. The following address types are acceptable:

- `inet` – TCP sockets are used. `ADDRESS` is specified in `PORT@HOST_NAME` format, where `HOST_NAME` can be either direct IP-address or

host domain name.

Example:

```
Address = inet:3003@127.0.0.1
```

- `local` – local UNIX sockets are used. `ADDRESS` is a path to socket file.

  Example:

  ```
  Address = local:%var_dir/.daemon
  ```

- `PID` – real address of the process must be read from its PID-file. This address type is acceptable only in some cases, and in such a case this will be explicitly pointed out in parameter description.

- `text value` – parameter value is a text string, which can be enclosed in quotation marks (and must be enclosed in quotation marks when contain white spaces).
- `pool options` – threads pool settings.

  At first, number of threads in a pool is defined:

  - `auto` – number of threads in a pool is automatically detected, depending on the current system load;
  - `N` – non-negative integer. At least `N` threads in a pool will be active, and new threads will be created as required;
  - `N-M` – positive integers, and $M \geq N$. At least `N` threads in a pool will be active, and new threads will be created as required until the number of threads reaches `M` value.

  Further the following additional parameters can be specified:

  - `timeout = {time}` – if a thread does not become active during the specified period of time, it is closed. This parameter does not affect the first `N` threads, which are waiting for requests infinitely.

    Default value: `2m`

  - `stat = {yes|no}` – statistics for threads in a pool. It is saved each time `SIGUSR1` system signal is received, to the directory specified in the

value of **BaseDir** parameter from [General] section.

Default value: no

- log_level = {Quiet|Error|Alert|Info|Debug} – log verbosity level for threads in a pool. If the value is not explicitly specified, value of **LogLevel** parameter from [Logging] section is used.
- stop_timeout = {time} – maximum time for a working thread to stop (e.g. when program finishes its operation, or when it is necessary to decrease the number of threads in a pool.

Examples:

InPoolOptions = auto, timeout=1m, stat = yes

Number of threads is detected automatically, timeout for a thread to be considered inactive is set to one minute, statistics is collected.

- lookups – generalized interface for performing search through objects and receiving their values. Values are delimited by commas. Sometimes a prefix defining a specific type of lookup can be put before a value: [PREFIX1:]VALUE1, [PREFIX2:]VALUE2, ... If no prefix is specified, value is used without it. The following prefixes can be used in lookups:
  - file – path to file. Each value in the file must reside on the new line. This is the fastest search because it allows to use sorting and binary search in files.
  - regex – regular expression (Perl syntax) is looked up as a substring, full compliance is not required.
  - rfile – path to file. File contains a set of regular expressions (Perl syntax), and each regular expression must reside on the new line.
  - ldap – search path on LDAP-server.
  - odbc, oracle – SQL request to ODBC or Oracle database.

- `postgres` – SQL request to PostgreSQL database.
- `cdb` – request to CDB database, i.e. text name of the database key. CDB database does not support SQL syntax. It only supports `[text key]:[text value]` requests.
- `berkeley` – request to the Berkeley database.
- `firebird` – SQL request to Firebird database.
- `sqlite` – SQL request to SQLite database.
- `mysql` – SQL request to MySQL database.

- `LookupsLite` – this value type is similar to `lookups`, with one minor difference: only values themselves or `file` type of lookups can be used in this type.
- `storage` – objects to store data. Syntax is similar to `lookups` except for the different prefix types and impossibility to use `$s` macro.

  Prefix could be:

  - `odbc` – the syntax is the same as in requests to LDAP. In SQL request values can be specified in `:name<type>` format, where `name` - is the name of the stored object (for each parameter its own list of available names is used), and `type` - is the type of the parameter to be used when a record to the database is made.
  - `oracle` – the syntax is the same as in requests to ODBC.
  - `postgres`, `mysql`, `sqlite`, `firebird` – the syntax is the same as in requests to ODBC, with one minor difference: `char(length)` type is not supported, and for the string data `varchar_long` type must be used.

  Example:

  ```
  "odbc:insert into plugin_stat values (:plugin_name<varchar_long>, :size<int>, :num<int>)";
  ```

Please note that in this request quotation marks are necessary, because commas are used in it.

- `TLS Settings` – TLS and SSL encrypted connection settings. Settings format: `PARAMETER VALUE`. `PARAMETER VALUE` pairs are separated by commas. If the `VALUE` is the path to a file, it is case-sensitive. The current version supports the following settings:
  - **`use_sslv2`** `{yes | no}` – toggles use of the SSLv2 protocol. By default the use of SSLv2 is disabled, because this protocol is insecure.
  - **`use_sslv3`** `{yes | no}` – toggles use of the SSLv3 protocol. SSLv3 is enabled by default.
  - **`use_tlsv1`** `{yes | no}` – toggles use of the TLSv1 protocol. TLSv1 is enabled by default.
  - **`private_key_file`** `{path to the file}` – the absolute path to a private key file. The key must have the PEM format. Key encryption is supported. The parameter is required for server configuration. The parameter's value is unspecified by default.
  - **`private_key_password`** `{string}` – the password for the key specified by the **`private_key_file`** parameter. The parameter's value is unspecified by default.
  - **`certificate`** `{path to the file}` – the path to a certificate file with a signed public key. The value for this parameter must be specified together with the value of the **`private_key_file`** parameter. This parameter is required for server configuration. The parameter's value is unspecified by default.
  - **`verify_mode`** `{none | peer | client_once | fail_if_no_peer_cert}` – sets peer certificate verification mode.
    - ✓ `none` – skip peer certificate verification. This value is set by default;

- ✓ `peer` – verify a peer's certificate. The parameter is ignored in the client mode if the server doesn't send a certificate for anonymous encryption. This is the default value for client connections;
- ✓ `client_once` – make the server to request a certification only when connect for the first time. The value can only be used together with the `peer` value of this parameter.
- ✓ `fail_if_no_peer_cert` – set the server to treat the lack of the client certificate as an error. The value can only be used together with the the `peer` value of this parameter.

Examples:

```
verify_mode peer, verify_mode client_once
verify_mode none
```

If `peer` and `none` are found in one block of settings, the last specified value is used.

- **`verify_ca`** `{the path to a file | the path to a directory}` – the absolute path to a file or directory where CA certificates in the PEM format reside. These certificates are used to validate a peer's certificate.
- **`cipher_list`** `{string}` – a list of allowed encryption algorithms. Use the man ciphers command to get information about the format of encryption algorithms list (OpenSSL must be installed).

- `string` – set of text values delimited by commas. If

parameter value is set in `file:/path_to_file` format, then text values are taken from the file `path_to_file`. In this file each text value must be specified on a separate line. If it appears to be impossible to read values from the `path_to_file` file, a notification about an error is output to log, and execution of the program continues.

- `value` – some parameters can have types not described above.

Logging for **Dr.Web for UNIX mail servers** modules may be very detailed (when `Debug` value is specified) or may be omitted entirely (when `Quiet` value is specified, and no information is logged at all). Log verbosity settings may have the following values: `Quiet`, `Error`, `Info`, `Alert`, `Notice`, `Warning`, `Verbose`, `Debug`.

Dr.Web Daemon and **Dr.Web Scanner** components work with levels: `Error`, `Info`, `Notice`, `Warning`, `Alert`.

Dr.Web Updater component uses log levels from the following list: `Quiet`, `Error`, `Alert`, `Info`, `Debug`, `Verbose`.

Installation and Deinstallation

Below you can find detailed description of **Dr.Web for UNIX mail servers** solution installation and deinstallation procedures for UNIX systems. Administrator (root) privileges are necessary to perform all these operations.

You must carefully uninstall all packages of earlier product versions (delivered in rpm or deb formats) from any previous installations.

Dr.Web for UNIX mail servers solution distribution package for UNIX systems is delivered in EPM format (script-based distribution package with installation and removal scripts and standard install/uninstall GUIs) designed to use with ESP Package Manager (EPM). Please note that all these scripts belong only to EPM-package itself, not to any of the components of **Dr.Web for UNIX mail servers**.

Installation, deinstallation and upgrade procedures for **Dr.Web for UNIX mail servers** solution can be carried out in the following ways:

- via install/uninstall GUIs;
- via install/uninstall console scripts.

During installation dependencies are supported, i.e. if for successful installation of any component some other components must be previously installed (e.g., `drweb-daemon` package requires `drweb-common` and `drweb-bases` packages to be previously installed), then they will be installed automatically.

If you install **Dr.Web for UNIX mail servers** solution to the computer, where some other **Dr.Web** products have been previously installed from EPM-packages, then at every attempt to remove some modules via uninstall GUI you will be prompted to remove absolutely all **Dr.Web** modules, including those from other products.

Please, pay special attention to the actions you perform and selections you make during deinstallation to avoid accidental removal of some useful components.

Installation from Distribution Package for UNIX systems

Dr.Web for UNIX mail servers solution is distributed as a self-extracting package `drweb-mail-[product name]_[version number]~[OS name].run`. The following components are included into this distribution:

- `drweb-common`: contains main configuration file `drweb32.ini`, libraries, documentation and directory structure. During installation of this component `drweb` user and `drweb` group will be created;
- `drweb-bases`: contains anti-virus search engine (**Engine**) and virus databases. It requires `drweb-common` package to be previously installed;
- `drweb-libs`: contains common libraries for all the components of the software solution;
- `drweb-epm6.0.2-libs`: contains libraries for graphical installer and uninstaller. It requires `drweb-libs` package to be previously installed;
- `drweb-epm6.0.2-uninst`: contains files of graphical uninstaller. It requires `drweb-libs` package to be previously installed;
- `drweb-boost147`: containes common libraries for **Dr.Web Agent** and **Dr.Web Monitor**. It requires `drweb-libs` package to be previously installed;
- `drweb-updater`: contains update utility (**Updater**) for **Engine** and virus databases. It requires `drweb-common` and `drweb-libs` packages to be previously installed;
- `drweb-agent`: contains **Dr.Web Agent** executable files

and its documentation. It requires `drweb-common` and `drweb-boost147` packages to be previously installed;

- `drweb-agent-es`: contains files required to run **Dr.Web Agent** in central protection mode. It requires `drweb-agent`, `drweb-updater` and `drweb-scanner` packages to be previously installed;
- `drweb-daemon`: contains **Dr.Web Daemon** executable files and its documentation. It requires `drweb-bases` and `drweb-libs` packages to be previously installed;
- `drweb-scanner`: contains **Dr.Web Scanner** executable files and its documentation. It requires `drweb-bases` and `drweb-libs` packages to be previously installed;
- `drweb-monitor`: contains **Dr.Web Monitor** executable files and its documentation. It requires `drweb-agent`, `drweb-common` and `drweb-boost147` packages to be previously installed;
- `drweb-maild`: contains **Dr.Web MailD** executable files and its documentation. It requires `drweb-maild-common` package to be previously installed;
- `drweb-maild-common`: contains libraries for **Dr.Web Agent**, **Dr.Web Monitor** and **Dr.Web MailD**. It requires `drweb-common`, `drweb-gperftools0`, `drweb-agent` and `drweb-monitor` packages to be previously installed;
- `drweb-maild-plugin-drweb`: contains library of `drweb` plug-in, its configuration file, documentation and configuration script. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-web`: contains Web interface of **Dr.Web for UNIX mail servers**
- `drweb-maild-plugin-headersfilter`: contains library of `headersfilter` plug-in, its configuration file, documentation and configuration script. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-plugin-modifier`: contains library of modifier plug-in, its configuration file, documentation and configuration script. It requires `drweb-maild` package to be previously installed;

- `drweb-maild-plugin-vaderetro`: contains configuration file of vaderetro plug-in, documentation and configuration script. It requires `drweb-maild` and `drweb-libvaderetro` packages to be previously installed;
- `drweb-libvaderetro`: contains library of `vaderetro` plug-in;
- `drweb-maild-smtp`: contains executable files of **Sender** and **Receiver** modules which enable operation of **Dr.Web for UNIX mail servers** solution as a proxy-server for SMTP and LMTP protocols, **Dr.Web MailD** configuration file with corresponding settings, documentation and configuration script for **Dr.Web Monitor**. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-cgp`: contains executable files of **Sender** and **Receiver** modules which enable interaction with Communigate Pro mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Communigate Pro for interaction with **Dr.Web MailD**. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-courier`: contains executable files of **Sender** and **Receiver** modules which enable interaction with Courier mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Courier for interaction with **Dr.Web MailD**. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-exim`: contains executable files of **Sender** and **Receiver** modules which enable interaction with Exim mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Exim for interaction with **Dr.Web MailD**. It requires `drweb-maild` package to be previously installed;
- `drweb-maild-postfix`: contains executable files of **Sender** and **Receiver** modules which enable interaction with Postfix mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Postfix for interaction with **Dr.Web MailD**. It requires `drweb-maild` package to be previously installed;

- drweb-maild-qmail: contains executable files of **Sender** and **Receiver** modules which enable interaction with Qmail mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Qmail for interaction with **Dr.Web MailD**. It requires drweb-maild package to be previously installed;
- drweb-maild-sendmail: contains executable files of **Sender** and **Receiver** modules which enable interaction with Sendmail mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of Sendmail for interaction with **Dr.Web MailD**. It requires drweb-maild package to be previously installed;
- drweb-maild-zmailer:contains executable files of **Sender** and **Receiver** modules which enable interaction with ZMailer mail transfer agent, **Dr.Web MailD** configuration file with settings for the specific MTA, documentation and configuration script for adjustment of ZMailer for interaction with **Dr.Web MailD**. It requires drweb-maild package to be previously installed;
- drweb-gperftools0: contains Google Performance Tools library used by **Dr.Web MailD**. It requires drweb-libs package to be previously installed;
- drweb-mail-servers-gateways-doc: contains **Administrator manual** in english and russian languages.

In distributions for 64-bit systems an additional package with libraries is included: drweb-libs64 - containing libraries for 64-bit components.

To install all the components of **Dr.Web for UNIX mail servers** solution automatically you may use either console (CLI) or the default file manager of your GUI-based shell. In the first case allow the execution of the corresponding self-extracting package with the following command:

```
# chmod +x drweb-mail-[product name]_
[version number]~[OS name].run
```

and then run it:

```
# ./drweb-mail-[product name]_
[version number]~[OS name].run
```

As a result `drweb-mail-[product name]_[version number]~[OS name]` directory will be created, and install GUI will be initialized. If startup has been performed without root privileges, install GUI will try to gain appropriate privileges by itself.

If install GUI fails to start, then interactive console installer will be initialized.

If you want only to extract the content of the package without starting install GUI, use `--noexec` command line parameter:

```
# .drweb-mail-[product name]_
[version number]~[OS name].run --noexec
```

After you extract the content, you may initialize install GUI and continue setup with the following command:

```
# drweb-mail-[product name]_
[version number]~[OS name]/install.sh
```

To initialize console installer use the following command:

```
# drweb-mail-[product name]_
[version number]~[OS name]/setup.sh
```

During the installation the following processes take place:

- Original configuration files are recorded to the `%etc_dir/software/conf/` directory with the following names: `[configuration_file_name].N`.
- Operational copies of configuration files are placed to the corresponding directories of the installing software.
- Other files are installed. If in the corresponding directory file with the same name already exists (e.g. after inaccurate removal of previous versions of the packages), it will be overwritten with the new file, and its copy will be saved as

`[file_name].O`. If some `[file_name].O` file already exists in this directory, it will be replaced with the new file of the same name.

- If you select a **Run interactive postinstall script** check-box in the corresponding window of the graphical installer, then after installation of the components the post-install script will be initialized for basic adjustment of **Dr.Web for UNIX mail servers**.

Using GUI Installer

To install with GUI

1. Execute the following command:

   ```
   # drweb-mail-[product name]_
   [version number]~[OS name]/install.sh
   ```

   The setup program launches. On any step, click **Back** or **Next** to navigate, or click **Cancel** to abort installation.

   On the Welcome screen, click **Next**.

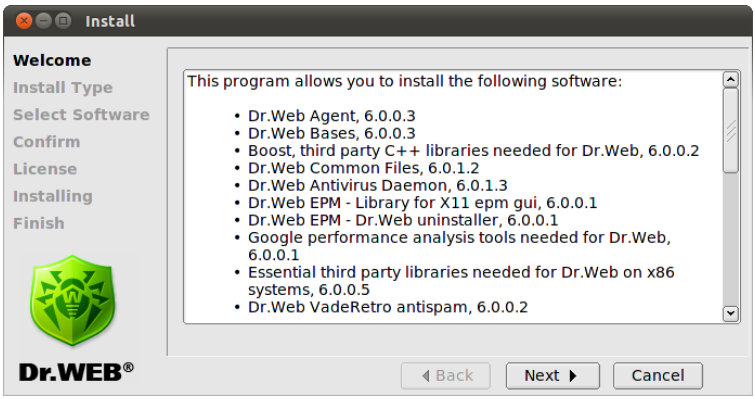


Figure 1. Welcome window

2. On the **Install Type** screen, select installation type: typical configuration for your mail gateway **Dr.Web for Mail Gateways** or preferred MTA **Dr.Web for <MTA> (Full installation)** with all the necessary components selected by default or custom configuration.

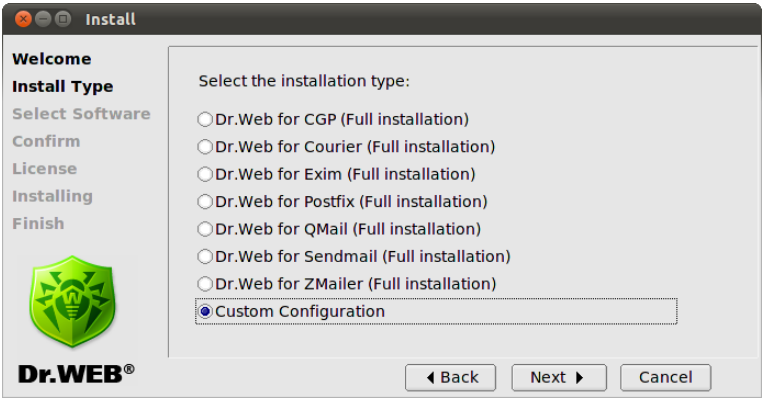


Figure 2. Install Type screen for MTA

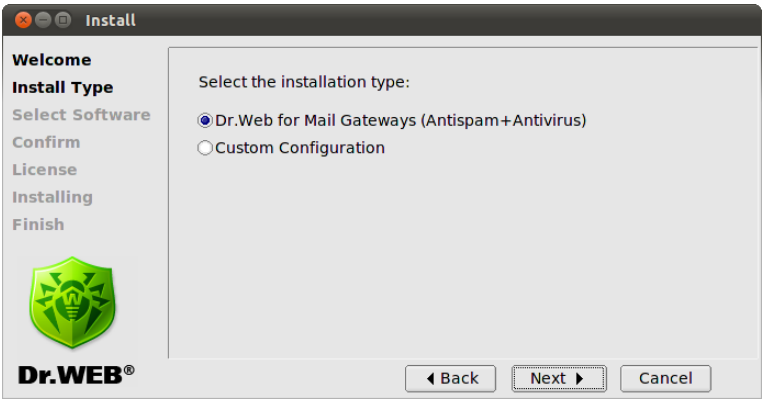


Figure 3. Install Type screen for Dr.Web for Mail Gateways

If you selected **Custom Configuration**, then select

necessary components on the **Select Software** screen:

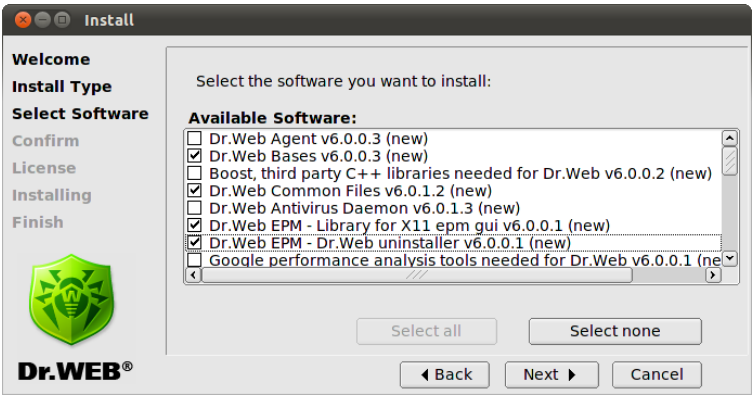


Figure 4. Select Software screen

If installation of a component requires some other components to be previously installed, all corresponding dependencies are selected for installation automatically. For example, if you select to install **Dr.Web Antivirus Daemon**, then **Dr.Web Bases** and **Dr.Web Common Files** are selected and installed automatically.

At installation, packages for different mail transfer agents may conflict with each other (drweb-maild-smtp and various drweb-maild-MTA). For example, if you try to select to install **Dr.Web Mail Daemon – Exim Connector** and **Dr.Web Mail Daemon – Postfix Connector** simultaneously, you receive an error message and suggestion to select only one of them.

Click **Install None** to clear selection.

When you complete selection, click **Next**.

3. On the **Confirm** screen, review and confirm the list of components to install:

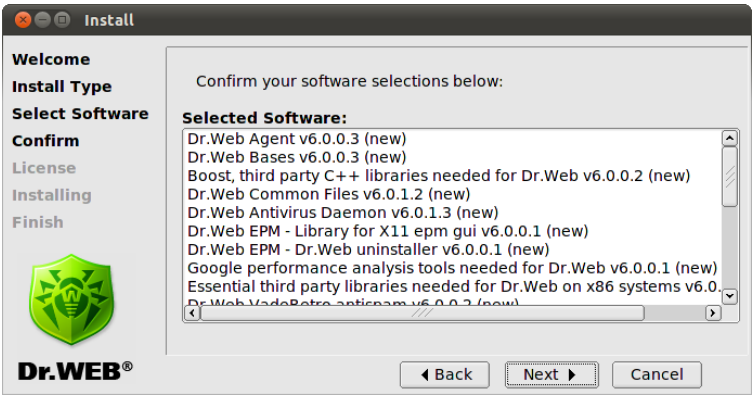


Figure 5. Confirm screen

Click **Next** to confirm selection, or click **Back** to make changes.

4. Review the license agreement. To proceed, you need to accept it. If necessary, use the **Language** list to select preferred language:

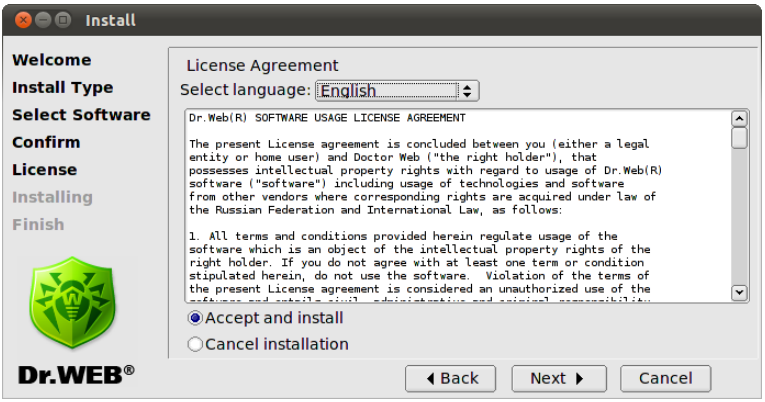


Figure 6. License Agreement screen

5. If you accepted the License Agreement, installation start. On the **Installing** screen, you can review the installation process in real-time:

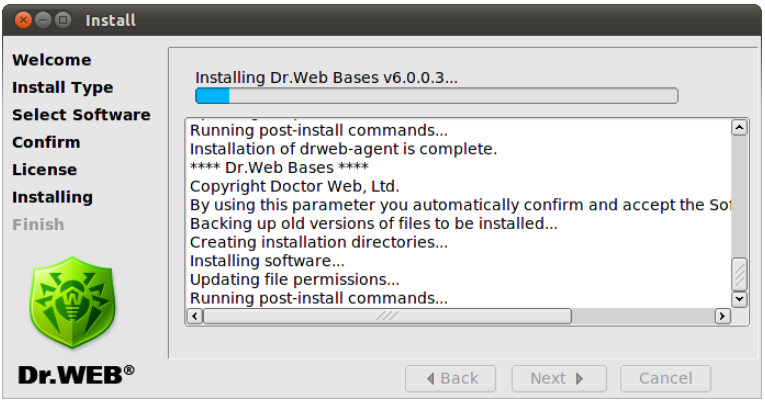


Figure 7. Installing screen

This report is logged at the same time in the `install.log` log file located at the `drweb-mail-[product name]_[version number]~[OS name]` directory. If you selected to **Run interactive postinstall script**, then after installation of the components is completed, the post-install script is initialized for basic configuration of **Dr.Web for UNIX mail servers**.

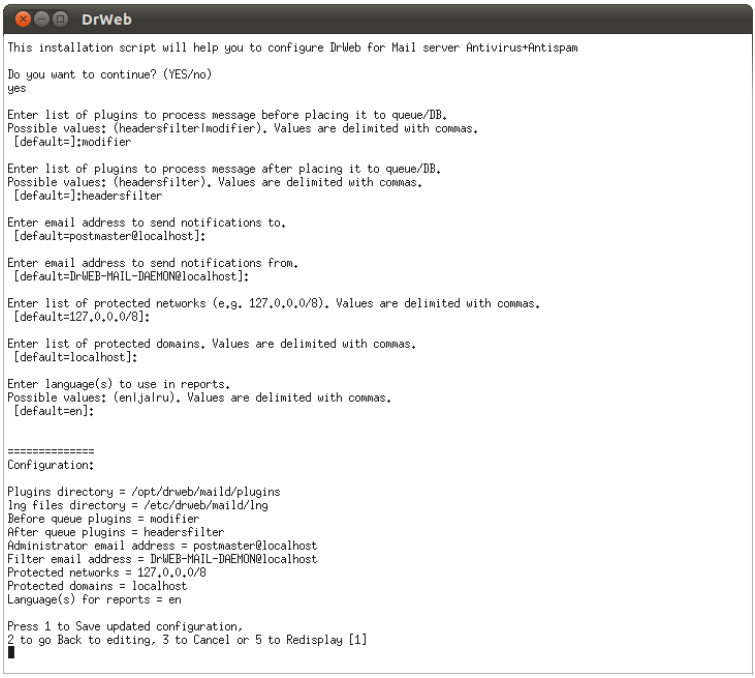


Figure. 8. Interactive post-install script

This script offers you to specify a path to the key file and to enable automatically the services necessary for proper operation of **Dr.Web for UNIX mail servers** (i.e., **Dr.Web Daemon**, **Dr.Web Agent**, **Dr.Web Monitor**):

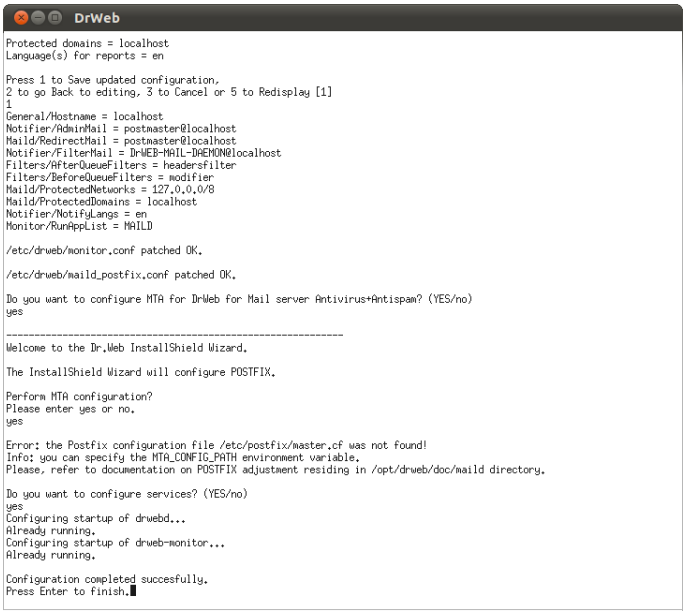


Figure 9. Configuring MTA and starting services automatically

6. On the **Finish** screen, click **Close** to exit setup:

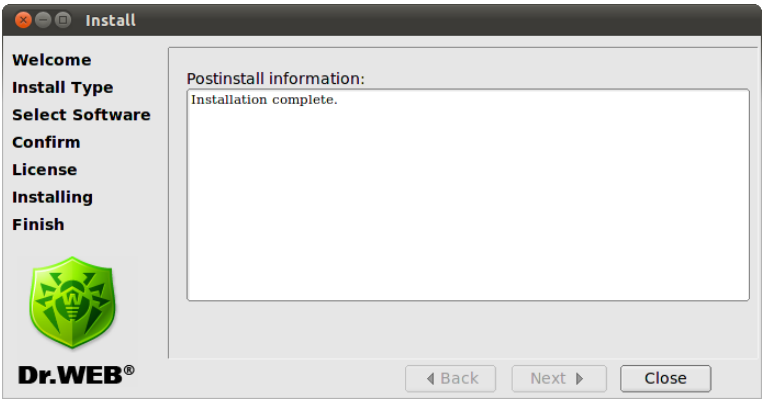


Figure 10. Finishing screen

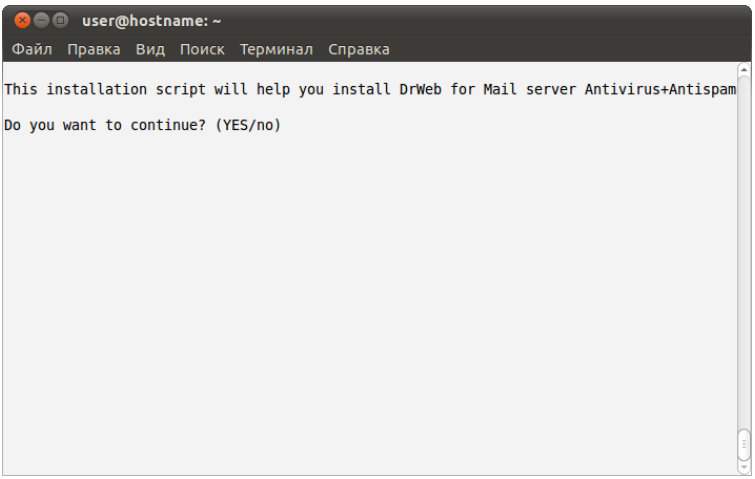
Using Console Installer

Console installer starts automatically is graphical installer fails to start. If console installer also fails to initialize (e.g., when it is impossible to gain necessary privileges), then you can try to run the following command with root privileges:

```
# drweb-mail-[product name]_
[version number]~[OS name]/setup.sh
```

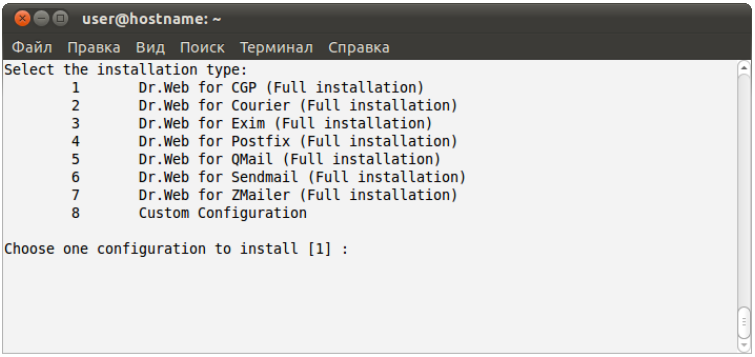
To install from console

1. Once the console installer start, a conversation window opens:



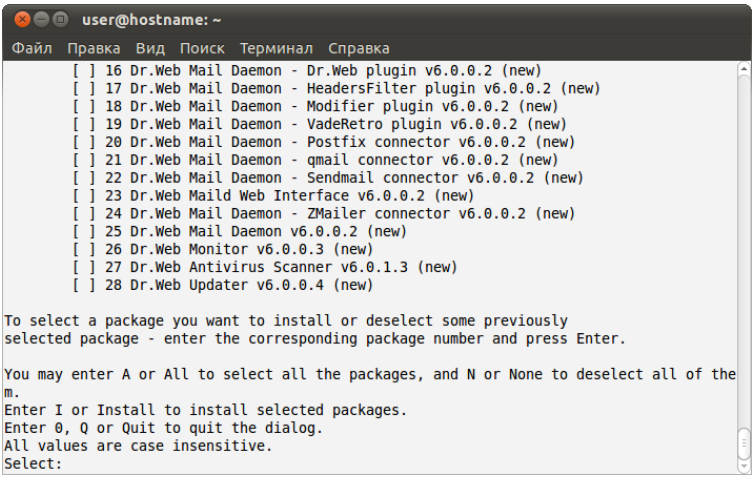
   If you want to install **Dr.Web for UNIX mail servers**, type **Y** or **Yes** (values are case insensitive), otherwise type **N** or **No**. Press ENTER.

2. If you selected to install **Dr.Web for UNIX mail servers**, installer suggests you to select the type of installation:



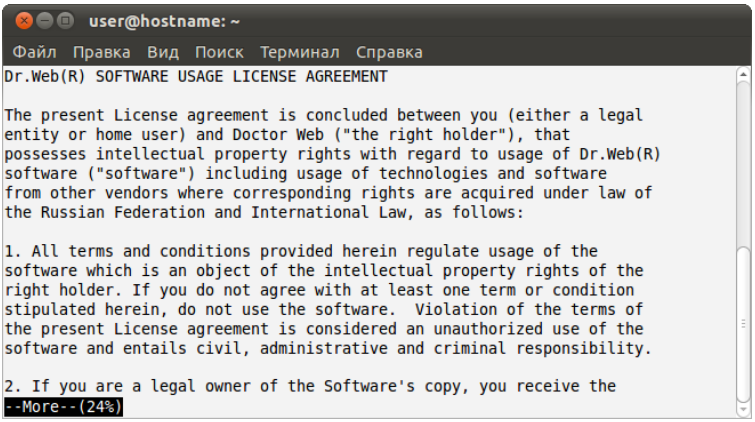
Type the number of corresponding mode and press ENTER.

3. If you selected **Custom Configuration**, specify components to install:



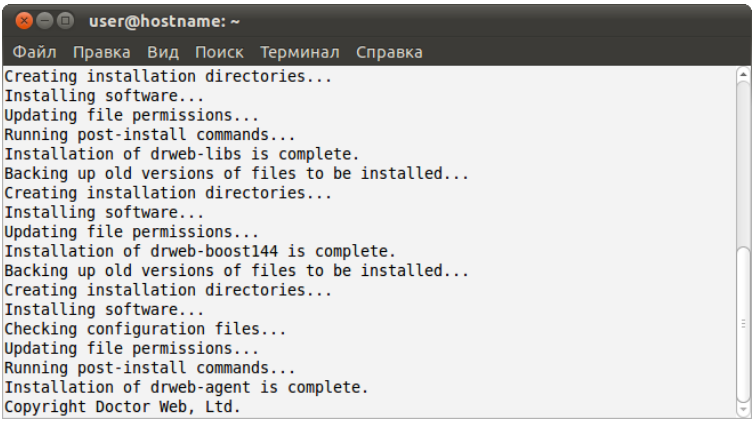
Type the number of a component from the list and press ENTER.

4. Review the Software License Agreement. To scroll the text, press SPACEBAR:



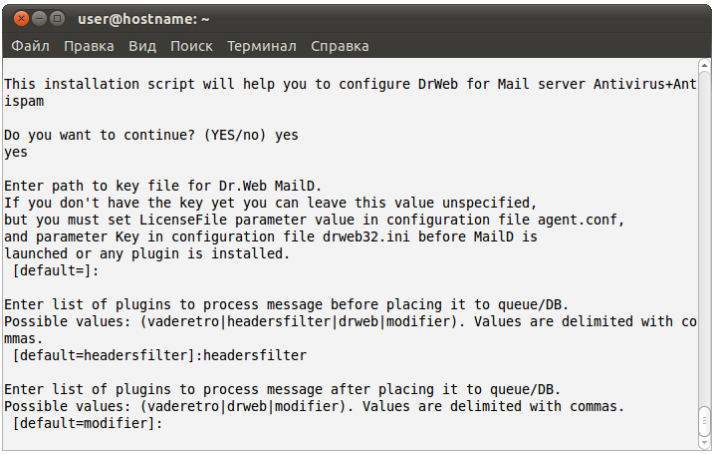
To continue the installation, you need to accept the License Agreement. If you agree to the terms, type **Y** or **Yes**.

5. The installation process starts immediately. You can review the installation process in console in real-time:



6. After installation of the components, the post-install script

runs automatically to set up basic configuration of **Dr.Web for UNIX mail servers**. This script offers you to specify path to the license key file and enable automatically all the services necessary for proper operation of **Dr.Web for UNIX mail servers** (i.e., **Dr.Web Daemon**, **Dr.Web Agent**, **Dr.Web Monitor**).



Removal of Distribution Package for UNIX Systems

To remove all the components of **Dr.Web for UNIX mail servers** solution via uninstall GUI, initialize it with the following command:

```
# %bin_dir/remove.sh
```

If startup has been performed without root privileges, uninstall GUI will try to gain appropriate privileges by itself.

If uninstall GUI fail to start, then interactive console uninstaller will be initialized.

After deinstallation you can also remove `drweb` user and `drweb` group from your system.

During the deinstallation the following actions are performed:

- Original configuration files are removed from the `%etc_dir/software/conf/` directory.
- If operational copies of configuration files were not modified by the user, they are also removed. If the user has made any changes to them, they are preserved.
- Other **Dr.Web** files are removed. If a copy of some old file has been created at installation, this file will be restored under the name it had before the installation. Usually, such copies are named `[file_name].O`.
- License key files and log files are preserved in corresponding directories.

Using GUI Uninstaller

To uninstall with GUI

1. Execute the following command:
   ```
   # %bin_dir/remove.sh
   ```
   The setup program launches. On any step, click **Back** or **Next** to navigate, or click **Cancel** to abort installation.

   On the Welcome screen, click **Next**:

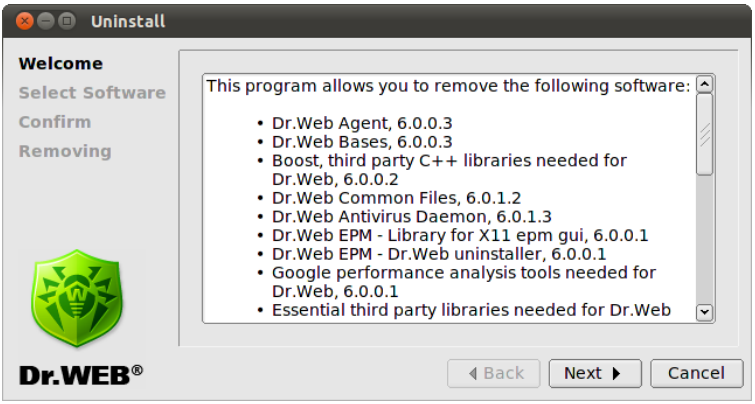


Figure 11. Welcome screen

2. On the **Select Software** screen, select components to remove:

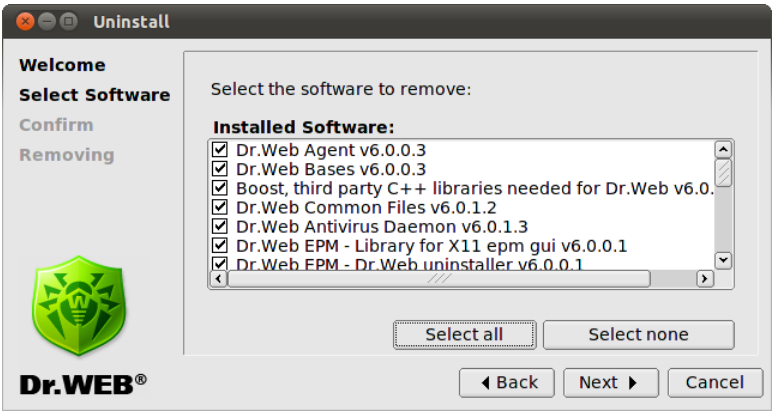


Figure 12. Select Software screen

All corresponding dependencies will be selected for de-installation automatically.

If you installed **Dr.Web for UNIX mail servers** solution on a computer with another **Dr.Web** product installed from EPM-packages, then setup lists all **Dr.Web** modules for both **Dr.**

Web for UNIX mail servers and the old product. Please, pay attention to the actions you perform and selections you make during de-installation to avoid accidental removal of useful components.

Click **Remove All** to select all components, or click **Remove None** to clear selection.

When you complete selection, click **Next**.

3. On the **Confirm** screen, review and confirm the list of components to remove:

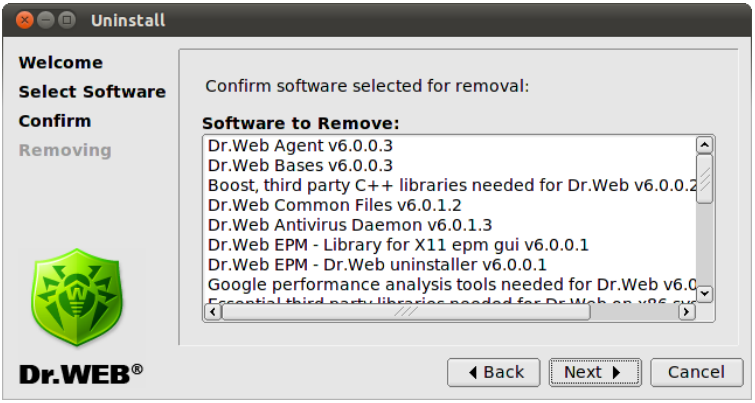


Figure 13. Confirm screen

Click **Next** to confirm selection, or click **Back** to make changes.

4. On the **Removal** screen, you can review removal process in real-time:

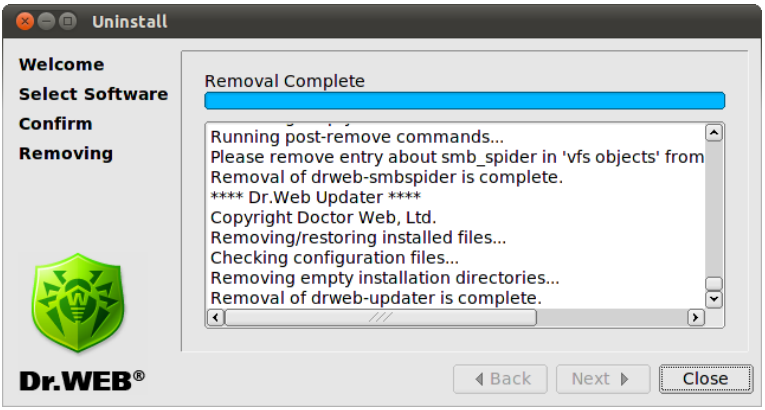


Figure 14. Removal screen

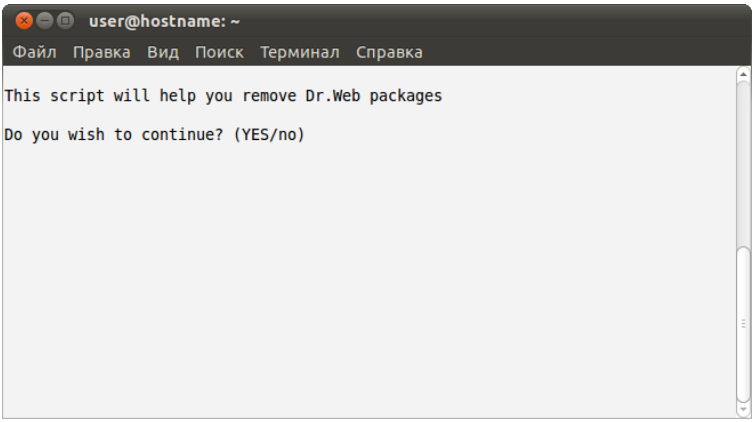
5. Click **Close** to exit setup.

Using Console Uninstaller

Console uninstaller starts automatically when graphical uninstaller fails to start.

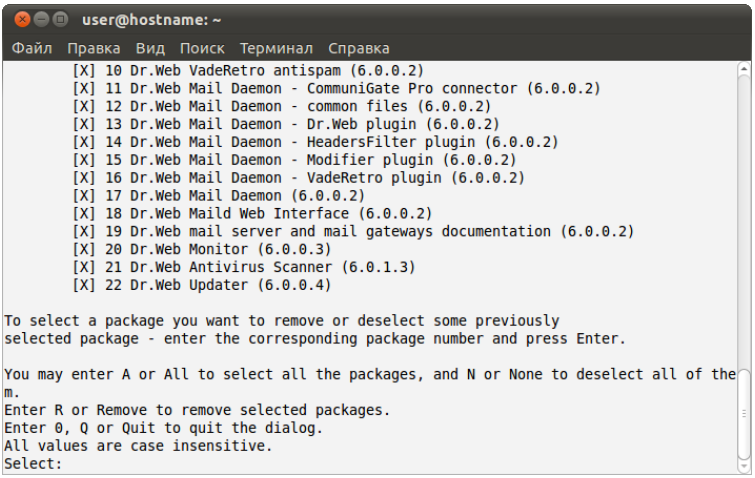
To uninstall from console

1. Once the console uninstaller start, a conversation window opens:



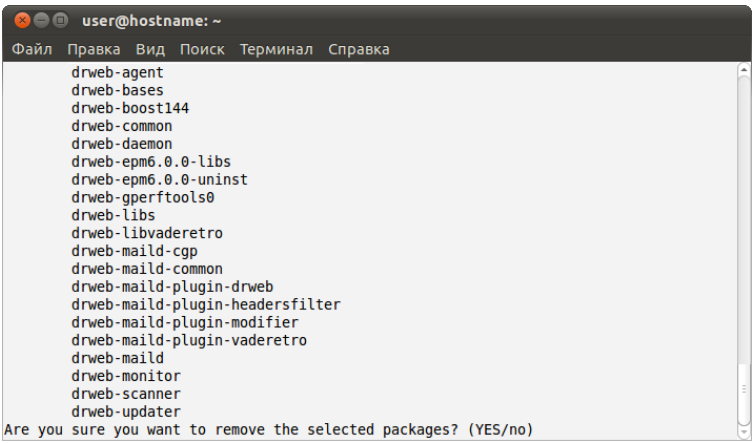
If you want to uinstall **Dr.Web for UNIX mail servers**, type **yes**, otherwise type **no**. Press ENTER.

2. Review the list of components available for removal:



3. Follow the prompts to select components to remove.
4. To start uninstall, confirm you selection by typing **Y** or **Yes**

and pressing ENTER (values are case insensitive):



5. You can review removal process in console in real-time.
6. Once the process completes, exit setup.

Installation from Native Packages

You can install **Dr.Web for UNIX mail servers** from native packages for common Linux distributions or Solaris and FreeBSD operating systems.

All packages are located in the **Dr.Web** official repository http://officeshield.drweb.com/drweb/. Once you have added the repository to the package manager of your system, you can install, update or remove necessary packages like any other program from repository. All dependencies will be resolved automatically.

After installing from repository, automatic post-install script for installing license key file will not be initiated. Licence key file must be manually copied to `%bin_dir`.

You need to restart all **Dr.Web** services after updating from repository for the updates to take effect.

Depending on desired solution you may install one of the following packages:

- `drweb-mail-gateways-as` - **Dr.Web Antispam for UNIX Mail Gateways**;
- `drweb-mail-gateways-av` - **Dr.Web Antivirus for UNIX Mail Gateways**;
- `drweb-mail-gateways-av-as` - **Dr.Web Antivirus and Antispam for UNIX Mail Gateways**;
- `drweb-courier-as` - **Dr.Web Antispam for Courier Mail Servers**;
- `drweb-courier-av` - **Dr.Web Antivirus for Courier Mail Servers**;
- `drweb-courier-av-as` - **Dr.Web Antivirus and Antispam for Courier Mail Servers**;
- `drweb-postfix-as` - **Dr.Web Antispam for Postfix Mail Servers**;
- `drweb-postfix-av` - **Dr.Web Antivirus for Postfix Mail Servers**;
- `drweb-postfix-av-as` - **Dr.Web Antivirus and Antispam for Postfix Mail Servers**;
- `drweb-qmail-as` - **Dr.Web Antispam for qmail Mail Servers**;
- `drweb-qmail-av` - **Dr.Web Antivirus for qmail Mail Servers**;
- `drweb-qmail-av-as` - **Dr.Web Antivirus and Antispam for qmail Mail Servers**;
- `drweb-sendmail-as` - **Dr.Web Antispam for Sendmail Mail Servers**;
- `drweb-sendmail-av` - **Dr.Web Antivirus for Sendmail**

Mail Servers;

- `drweb-sendmail-av-as` - **Dr.Web Antivirus and Antispam for Sendmail Mail Servers**;
- `drweb-cgp-as` - **Dr.Web Antispam for CommuniGate Pro Mail Servers**;
- `drweb-cgp-av` - **Dr.Web Antivirus for CommuniGate Pro Mail Servers**;
- `drweb-cgp-av-as` - **Dr.Web Antivirus and Antispam for CommuniGate Pro Mail Servers**;
- `drweb-exim-as` - **Dr.Web Antispam for Exim Mail Servers**;
- `drweb-exim-av` - **Dr.Web Antivirus for Exim Mail Servers**;
- `drweb-exim-av-as` - **Dr.Web Antivirus and Antispam for Exim Mail Servers**;
- `drweb-zmailer-as` - **Dr.Web Antispam for ZMailer Mail Servers**;
- `drweb-zmailer-av` - **Dr.Web Antivirus for ZMailer Mail Servers**;
- `drweb-zmailer-av-as` - **Dr.Web Antivirus and Antispam for ZMailer Mail Servers**.

Below you will find detailed instruction on how to add **Dr.Web** repository to supported package managers and install **Dr.Web for UNIX mail servers** using console.

All commands below for adding repositories, importing keys, installing and removing packages must be ran with administrator (root) privileges.

Debian, Ubuntu (apt)

Debian repository is signed by the digital key. For correct operation you need to import the key with command

```
wget -O - http://officeshield.drweb.com/drweb/drweb.key | apt-key add -
```

or

```
curl http://officeshield.drweb.com/drweb/drweb.key | apt-key
add -
```

To add the repository to you system, add the following line to

`/etc/apt/sources.list` file:

```
deb http://officeshield.drweb.com/drweb/debian stable non-free
```

To install **Dr.Web for UNIX mail servers** issue commands:

```
apt-get update
```

```
apt-get install <package name>
```

To remove **Dr.Web for UNIX mail servers** issue command:

```
apt-get remove <package name>
```

Alternatively, you can use graphical manager (e.g. Synaptic) to install or remove the packages.

Please note that after installation from native packages, `drwebd` enable file will be located in the following directories:

- `/etc/defaults` — for deb packages;
- `/etc/sysconfig` — for rpm packages.

ALT Linux, PCLinuxOS (apt-rpm)

To add the repository to you system, add the following line to `/etc/apt/sources.list` file:

32-bit version:

```
rpm http://officeshield.drweb.com/drweb/altlinux stable/i386
drweb
```

64-bit version:

```
rpm http://officeshield.drweb.com/drweb/altlinux stable/x86_64
drweb
```

To install **Dr.Web for UNIX mail servers** issue commands:

```
apt-get update
```

```
apt-get install <package name>
```

To remove **Dr.Web for UNIX mail servers** issue command:

```
apt-get remove <package name>
```

Alternatively you can use graphical manager (e.g. Synaptic) to install or remove the packages.

Mandriva (urpmi)

Download repository key from http://officeshield.drweb.com/drweb/drweb.key and save it on disk. Then, import the key with command

```
rpm --import <path to repository key>
```

Open the following file:

http://officeshield.drweb.com/drweb/drweb-i386.urpmi-media

or

http://officeshield.drweb.com/drweb/drweb-x86_64.urpmi-media

and you will be offered to add repository to the system.

Alternatively, you can add the repository using console with command

```
urpmi.addmedia drweb http://officeshield.drweb.com/drweb/mandriva/stable/
i386/
```

or

```
urpmi.addmedia drweb http://officeshield.drweb.com/drweb/
mandriva/stable/x86_64/
```

To install **Dr.Web for UNIX mail servers** issue commands:

```
urpmi.update drweb
```

```
urpmi <package name>
```

To remove **Dr.Web for UNIX mail servers** issue command:

```
urpme <package name>
```

Alternatively, you can use graphical manager (e.g. rpmdrake) to install or remove the packages.

Red Hat Enterprise Linux, Fedora, CentOS (yum)

Add the file with following content to `/etc/yum.repos.d` directory

32-bit version:

```
[drweb]
```

```
name=DrWeb - stable
```

```
baseurl=http://officeshield.drweb.com/drweb/
el5/stable/i386/
```

```
gpgcheck=1
```

```
enabled=1
```

```
gpgkey=http://officeshield.drweb.com/drweb/
drweb.key
```

64-bit version:

```
[drweb]
```

```
name=DrWeb - stable
```

```
baseurl=http://officeshield.drweb.com/drweb/
el5/stable/x86_64/
```

```
gpgcheck=1
```

```
enabled=1
gpgkey=http://officeshield.drweb.com/drweb/
drweb.key
```

To install **Dr.Web for UNIX mail servers** issue command:

```
yum install <package name>
```

To remove **Dr.Web for UNIX mail servers** issue command:

```
yum remove <package name>
```

Alternatively, you can use graphical manager (e.g. PackageKit, Yumex) to install or remove the packages.

Zypper package manager (SUSE Linux)

To add the repository, run the following command:

```
zypper ar -t YUM http://officeshield.drweb.com/drweb/el5/stable/i386/ drweb
```

or

```
zypper ar -t YUM http://officeshield.drweb.com/drweb/el5/stable/x86_64/
drweb
```

To install **Dr.Web for UNIX mail servers** issue commands:

```
zypper refresh
zypper install <package name>
```

To remove **Dr.Web for UNIX mail servers** issue command:

```
zypper remove <package name>
```

Alternatively, you can use graphical manager (e.g. YaST) to install or remove the packages.

FreeBSD operating system

You can install **Dr.Web** products from meta-ports for FreeBSD. Download archive `<package name>-meta_current-current~freebsd_all.tar.gz` from http://officeshield.

drweb.com/drweb/freebsd/ports/. Then, unpack the archive and run `make install` to compile and install **Dr.Web for UNIX mail servers**. If you install **Dr.Web for UNIX mail servers** in FreeBSD 6.1, use `-I` command line parameter to define path to `/usr/ports/Mk/` directory that contains port tree files.

Example:

```
tar -xzvf <package name>-meta_current-current~freebsd_all.tar.gz
make install -I /usr/ports/Mk/
```

In FreeBSD version 8 and later compat7x library is required by the **Dr.Web for UNIX mail servers** to work properly.

Solaris

Native packages for Solaris can be downloaded from the public FTP-server:

ftp://ftp.drweb.com/pub/drweb/unix/release/Solaris/packages

and installed using `pkgadd` utility.

Configuration Scripts

When all components are installed, you can use `configure.pl` configuration script to setup basic configuration of **Dr.Web MailD**. This script is located in the `%bin_dir/maild/scripts/` directory. When started, this script offers you to specify how to process e-mail with a certain plug-in (e.g., whether it should receive messages before or after they are put to the database), preferred language for notifications and an address where to send them, and paths to lists of protected networks and domains. This information is sufficient to start using **Dr.Web for UNIX mail servers**, but for a full-featured performance you will have to configure each

component and MTA manually.

For detailed information on parameters, methods, and techniques, refer to the corresponding chapters of this Manual: Adjustment and Startup, Plug-ins, Integration with Mail Transfer Systems.

The `configure_mta.sh` and `plugin_<name>_configure.pl` scripts from the `%bin_dir/maild/scripts/` directory are not designed to provide full-featured configuration for best performance of plug-ins and MTA. Use these scripts as reference sources only.

Startup of Dr.Web for UNIX mail servers

This section describes startup of **Dr.Web for UNIX mail servers** in Linux, Solaris or FreeBSD operating systems.

For Linux and Solaris

To run the **Dr.Web for UNIX mail servers** solution, do the following:

1. Register the software.
2. Place the key file to the directory for **Dr.Web for UNIX mail servers** executable files (default directory for UNIX systems is `%bin_dir`). Key file name may vary depending on the distribution kit (for the detailed information, see Software Registration chapter):
   - If **Dr.Web for UNIX mail servers** was purchased as a standalone product License key file is called `drweb32.key`. In this case, you should just copy file to `%bin_dir` directory without changing its name.
   - If **Dr.Web for UNIX mail servers** was purchased as a part of **Dr.Web Enterprise Security Suite** set, archive received during registration contains a key file for the **Enterprise Server** (`enterprise.key`) and a key file for workstations (`agent.key`). Rename `agent.key` as `drweb32.key` and copy it to the `%bin_dir` folder.

   If you want to use key file with a different location or name (for example, `agent.key`), you must specify its full path as a `Key` parameter value of configuration file `drweb32.ini`. While working in `Standalone` mode, this alternative path to the key file must be also specified in the value of **`LicenseFile`** parameter of the `agent.conf` configuration file of the **Dr.Web Agent** component.

3. Configure the software by making necessary changes to configuration files. Refer to the corresponding chapters of this Manual for the detailed information on configuration.
4. In `drwebd.enable` file from `%etc_dir` directory set `1` as a value of `ENABLE` variable to enable startup of **Dr.Web Daemon**. If it is not required to run **Dr.Web Daemon** (properly configured and working **Daemon** on some other computer in the network is used), `ENABLE` value must be set to `0` (it is also used as a default value).
5. In `drweb-monitor.enable` file from `%etc_dir` directory set `1` as a value of `ENABLE` variable to enable startup of **Dr.Web Monitor**.
6. Start initializing scripts for **Dr.Web Daemon** and **Dr.Web Monitor** either from console or from any file manager of your operation system. After startup **Dr.Web Monitor** will initialize all other components of the **Dr.Web for UNIX mail servers** solution (**Sender**, **Receiver**, **Notifier**, etc.). Also each component can be started independently, but **Dr.Web Agent** module must be started first, because all other components will receive their configuration through the **Agent**.

   In case of installation from native packages in Solaris:

   Through **Dr.Web for UNIX mail servers** installing, service management system SMF attempts to launch **Dr.Web Monitor** component. If **Monitor** can't find licence key file (for example in case of first **Dr.Web for UNIX mail servers** installing), it stops it's work and changes SMF to maintenance state.

   To launch **Monitor**, maintenance state should be reseted:

   - Enter the command

     ```
     # svcs -p <FMRI>
     ```

     where FMRI - unique identifier of controlled resource, in this case - **Dr.Web Monitor** component.
   - Forcibly cancel processes from `svcs -p` output list.

     ```
     # pkill -9 <PID>
     ```

     where PID — number of process, that listed above.
   - Restart **Dr.Web Monitor** with command

```
# svcadm clear <FMRI>
```

While installing **Dr.Web for UNIX mail servers** from native packages in Solaris, complex launches with service management system SMF:

```
# svcadm enable <drweb-monitor>
# svcadm enable <drweb-daemon>
```

To stop service enter:

```
# svcadm disable <service name>
```

Module `drwebd` can be launched in two modes:

1. Standard run through the `init` script
2. Using **Dr.Web Monitor**

While working in second mode, you need to set ENABLE parameter to 0 in .enable file.

Each of the components can be started manually as well, but note, that **Dr.Web Agent** component must be initialized beforehand in order to provide configuration information to all the other components.

For FreeBSD

To run the **Dr.Web for UNIX mail servers** solution, do the following:

1. Register the software.
2. Place the key file to the directory for **Dr.Web for UNIX mail servers** executable files (default directory for UNIX systems is `%bin_dir`). Key file name may vary depending on the distribution kit (for the detailed information, see Software Registration chapter):

- If **Dr.Web for UNIX mail servers** was purchased as a standalone product License key file is called `drweb32.key`. In this case, you should just copy file to `%bin_dir` folder without changing its name.
- If **Dr.Web for UNIX mail servers** was purchased as a part of Dr.Web Enterprise Security Suite set, archive received during registration contains a key file for the **Enterprise Server** (`enterprise.key`) and a key file for workstations (`agent.key`). Rename `agent.key` as `drweb32.key` and copy it to the `%bin_dir` directory.

If you want to use key file with a different location or name (for example, `agent.key`), you must specify its full path as a `Key` parameter value of configuration file `drweb32.ini`. While working in `Standalone` mode, this alternative path to the key file must be also specified in the value of **`LicenseFile`** parameter of the `agent.conf` configuration file of the **Dr.Web Agent** component.

3. Configure the software by making necessary changes to configuration files. Refer to the corresponding chapters of this Manual for the detailed information on configuration.
4. Add the following lines to the `/etc/rc.conf` file:
   - `drwebd_enable="YES"` – to enable startup of **Dr.Web Daemon**. If it is not required to run **Dr.Web Daemon** (properly configured and working **Daemon** on some other computer in the network is used), then you can just not include the specified line in the `rc.conf` file;
   - `drweb_monitor_enable="YES"` – to enable startup of **Dr.Web Monitor.**
5. Start initializing scripts for **Dr.Web Daemon** and **Dr.Web Monitor** either from console or from any file manager of your operation system. After startup **Dr.Web Monitor** will initialize all other components of the **Dr.Web for UNIX mail servers** solution (**Sender**, **Receiver**, **Notifier**, etc.). Also each component can be started independently, but **Dr.Web Agent** module must be started first, because all other components will receive their configuration through the **Agent**.

Each of the components can be started manually as well, but note, that **Dr.Web Agent** component must be initialized beforehand in order to provide configuration information to all the other components

SELinux

To set up successful operation of **Dr.Web Scanner** and **Dr.Web Daemon** components in OS protected by SELinux, you must compile politics for operation with corresponding modules `drweb-scanner` and `drweb-daemon` or set 1 as a value of allow_execheap variable.

Templates used in compilation of modules for politics may vary widely, depending on the type of Linux distribution, its version, set of SELinux politics and user settings. To receive more detailed information on compilation of politics you may refer to corresponding documentation on your Linux distribution.

To create necessary politics :

1. Create new SELinux policy source file (.te file). This file define the access rules related to described module. You can create necessary politics:
   - Using `policygentool` utility. To do this, specify two parameters: the name of the policy module (interaction with which has to be adjusted) and the full path to the corresponding executable.

Please note that `policygentool` utility which included in `selinux-policy` package in Red Hat Enterprise Linux and CentOS Linux, may not work correctly. In this case, use `audit2allow`.

Example:

```
# policygentool drweb-scanner /opt/drweb/drweb.real
```

– for **Scanner**.

```
# policygentool drweb-daemon /opt/drweb/drwebd.real
```

– for **Daemon**.

You will be prompted to enter a few common domain characteristics, and for each module three files will be created: `[module_name].te`, `[module_name].fc` and `[module_name].if`.

- Using `audit2allow` utility. This utility generates policy modules based on reports of denial of access from system log files. Reports can be searched automatically in system log files or you can set the path to log file manually.

In general, when using the `audit` daemon, audit log located in `/var/log/audit/audit.log` file. Otherwise, AVC messages are stored in `/var/log/messages` log file.

Example:

```
# audit2allow -M -i /var/log/audit/audit.log drweb
```

In this example, `audit2allow` search AVC messages in `audit.log` file.

Example:

```
# audit2allow -a -M drweb
```

In this example, `audit2allow` search AVC messages in system log files automaticaly.

In both cases, `audit2allow` creates two files:

SELinux source file of policy (`drweb.te`) and compiled policy module `drweb.pp`. If you want to make changes to the access rules of **Dr.Web for UNIX mail servers** components, then edit `drweb.te` and go to step 2. If you don't want to change policy file, go to step 4 to install `drweb.pp` policy module.

2. Using checkmodule utility, create a binary representation (.mod file) of the policy source file. Please note that for successful policy compilation a `checkpolicy` package must be installed on the system.

   Example:

   ```
   # checkmodule -M -m -o drweb.mod drweb.te
   ```

3. Create policy module (drweb.pp) by using `semodule_package` utility.

   Example:

   ```
   # semodule_package -o drweb.pp -m drweb.mod
   ```

4. To install the new policy module into the module store, use the `semodule` utility.

   Example:

   ```
   # semodule -i drweb.pp
   ```

It is also possible (but not reccomended) to set `1` as a value of `allow_execheap` environment variable to set up operation of **Dr.Web Scanner** and **Dr.Web Daemon** in SELinux. `allow_execheap` variable allow or deny execution of data in memory heap for all applications that runs in *unconfined domain*. To set value of `allow_execheap` variable, execute the following command:

```
# setsebool -P allow_execheap = 1
```

Software Registration. License Key File

User privileges for using **Dr.Web for UNIX mail servers** solution are controlled by special file called license key file.

License key file contains the following information:

- list of **Dr.Web for UNIX mail servers** components licensed to user;
- license expiration date;
- list of plug-ins licensed to user (some plug-ins don't require registration in the key file);
- other restrictions (for example, number of e-mails to be checked by plug-ins per day).

License key file has `*.key` extension and by default must be placed in a directory for **Dr.Web for UNIX mail servers** executable files.

License key file is digitally signed to prevent its editing. Edited license key file becomes invalid. It is not recommended to open your license key file in text editors to avoid its accidental corruption.

Users who have purchased **Dr.Web for UNIX mail servers** solution from **Doctor Web** certified partners obtain the license key file. The parameters of the key file are specified according to the license user has paid for. The license key file contains the name of the user (or a company name), and the name of the selling company.

For evaluation purposes users may also obtain a demo key file. It allows user to enjoy full functionality of the **Dr.Web for UNIX mail servers** solution, but has a limited term of use, and no technical support is provided.

License key file may be supplied as:

- a `drweb32.key` file license key for workstations, or as a zip archive containing license key file in case of purchasing **Dr.Web for UNIX mail servers** as a standalone product;

- a zip-archive, which contains a key file for the Server (`enterprise.key`) and a key file for workstations (`agent.key`) in case of purchasing **Dr.Web for UNIX mail servers** as a part of **Dr.Web Enterprise Security Suite**.

License key file may be received using one of the following ways:

- sent by e-mail as a ZIP-archive containing license key file with `*.key` extension (usually after registration on the web site). Extract license key file using the appropriate archiving utility and copy/move it to the directory for **Dr.Web for UNIX mail servers** executable files (default directory for UNIX systems is `%bin_dir`);
- included into the distribution package;
- supplied on a separate media as a file with `*.key` extension. In this case user must copy it manually to the `%bin_dir` directory.

License key file is sent to user via e-mail usually after registration on the web site (web site location is specified in registration card accompanying the product). Visit the site, fill in the web form with your customer data and submit your registration serial number (printed on the registration card). As a result of this procedure license is activated, and license key file is created for the serial number provided. Then it is sent to user on the e-mail address specified.

It is recommended to keep license key file until it expires, and use it when reinstalling or repairing **Dr.Web for UNIX mail servers** solution installation. If the license key file is damaged or lost, it can be recovered by the same procedure as during license activation. In this case you must use the same product serial number and customer data you have entered during the registration, only e-mail address can be changed (in this case license key file will be sent to the new e-mail address). If serial number matches any entry in **Dr.Web for UNIX mail servers** database, the corresponding key file will be dispatched to user by automatic system using e-mail address provided.

Registration with the same product serial number can be performed up to 25 times. If you need to recover lost license key file after 25th registration, you must make a request for license key file

recovery on http://support.drweb.com/request/, and also specify all data used during registration, valid e-mail address and detailed description of the situation. Request will be considered by **Dr.Web for UNIX mail servers** technical support service engineers, and after approval license key file will be provided to user via automatic support system or dispatched via e-mail.

Path to license key file of the certain component must be specified as a **Key** parameter value in corresponding configuration file (`drweb32.ini`).

Example:

```
Key = %bin_dir/drweb32.key
```

If license key file specified as a **Key** parameter value is failed to read (wrong path, permission denied), expired, blocked or invalid, the corresponding component terminates.

When less than two weeks left until the license expiration, **Dr.Web Scanner** outputs warning message at start and **Dr.Web Daemon** notifies user via e-mail. Messages are sent at every startup, restart or reload of the **Daemon** for every license key file installed. To enable this option you must set up **MailCommand** parameter in `[Daemon]` section of `drweb32.ini` configuration file.

If you want to use key file from the different location, you must specify full path to it in the value of **LicenseFile** parameter from the `[StandaloneMode]` of the **Dr.Web Agent** configuration file (refer to the [StandaloneMode] Section description).

In **Dr.Web for UNIX mail servers** solution there is a possibility to use several license key files simultaneously. List of plug-ins licensed to user is made of all plug-ins mentioned in key files (or at least in one of them). Limitations on operation of certain plug-in are set according to information from all the key files used.

During the operation of the whole software complex, various plug-ins must have the same limitations. In case there are several key files with different limitations on operation of plug-ins, then the lowest value is used for the operation of **Dr.Web for UNIX mail servers** solution in whole.

Example:

Three license key files are used. In the first one limitation for `drweb` plug-in is set to 10,000 letters per day. In the second one limitation for vaderetro plug-in is set to 15,000 letters per day. In the third one once again limitation for `drweb` plug-in is set to 10,000 letters per day. Then **Dr.Web for UNIX mail servers** solution will be able to work with both plug-ins mentioned in key files, but total limitation on operation of these plug-ins will be set to 15,000 letters per day (as for vaderetro), in spite of the fact that `drweb` plug-in can actually process 20,000 letters per day.

Command Line Dr.Web Scanner

Command line **Dr.Web Scanner** serves for detection and neutralization of malware on the local machine.

Command Line Parameters

You can run **Dr.Web Scanner** with the following command:

```
$ %bin_dir/drweb <path> [parameters]
```

where `<path>` - is the path or paths to scanned directories or the mask for checked files. If in startup path is specified with following prefix: `disk://<path to device file>` then boot sector of appropriate device will be checked and cured, if necessary. When **Scanner** is started only with `<path>` argument without any parameters specified, it scans the specified directory using the default set of parameters.

In the following example user home directory is being checked:

```
$ %bin_dir/drweb ~
```

When scanning is finished **Scanner** displays all found infected and suspicious files in the following manner:

```
/path/file infected [virus] VIRUS_NAME
```

After presenting information about infected or suspicious files, **Scanner** outputs summary report in the following manner:

```
Report for "/opt/drweb/tmp":
Scanned         : 34/32     Cured       : 0
Infected        : 5/5       Removed     : 0
Modifications   : 0/0       Renamed     : 0
Suspicious      : 0/0       Moved       : 0
```

```
Scan time      : 00:00:02 Scan speed : 5233
KB/s
```

Numbers divided by slash "/" mean: the first one – total number of files, the second one – number of files in archives.

You can use `readme.eicar` file included in the distribution package to test **Dr.Web Scanner**. Open this file in your text editor of choice and follow the instructions contained in the file to transform it into `eicar.com` program. When you check it with **Dr.Web Scanner**, it should output the following message:

```
      %bin_dir/doc/eicar.com infected by Eicar
Test File (Not a Virus!)
```

This program is not a virus and is used only for testing of anti-virus programs.

Scanner has many command-line parameters. In accordance to UNIX conventions they are separated from path by whitespace character and start with a hyphen("-"). To get complete list of parameters run **Scanner** with either `-?`, `-h`, or `-help` parameters.

Main command line parameters include

- scan area parameters;
- diagnostic parameters;
- actions parameters;
- interface parameters.

Scan area parameters indicate where checking for viruses should be performed:

- `path` – this arbitrary parameter may be used to specify path for scan. Several paths can be specified in one parameter;
- `@[+]<file>` – check objects listed in specified file. Plus symbol "+" instructs Scanner not to delete files from the list of objects after scan is completed. List file may contain paths to directories that must be scanned regularly, or list of files to be checked only once;

- `sd` – recursive scanning of files in all subdirectories starting from the current directory;
- `fl` – follow links, both to files and directories. Links causing loops are ignored;
- `mask` – ignore masks for file names.

Diagnostic parameters determine what type of objects will be checked for viruses:

- `al` – check all files;
- `ar[d|m|r][n]` – check archive files (ARJ, CAB, GZIP, RAR, TAR, ZIP etc.). `d` – delete, `m` – move, `r` – rename archives containing infected files, `n` – do not output archiver program names. Archives can be in simple (`*.tar`) or compressed forms (`*.tar.bz2`, `*.tbz`);
- `cn[d|m|r][n]` – check files in containers (HTML, RTF, PowerPoint etc.). `d` – delete, `m` – move, `r` – rename containers with infected objects, `n` – do not output container name;
- `ml[d|m|r][n]` – check e-mail files. `d` – delete, `m` – move, `r` – rename e-mail files with infected, `n` – do not output e-mail file types;
- `upn` – scan executable files packed with LZEXE, DIET, PKLITE, EXEPACK with compression type output disabled;
- `ex` – check files matching **`FilesTypes`** parameter in the configuration file;
- `ha` – enable heuristic analysis.

Actions parameters determine what actions must be performed if infected or suspicious files are detected:

- `cu[d|m|r]` – cure infected files or `d` – delete, `m` – move, `r` – rename infected files;
- `ic[d|m|r]` – actions for incurable files: `d` – delete, `m` – move, `r` – rename;
- `sp[d|m|r]` – actions for suspicious files: `d` – delete, `m` – move, `r` – rename;
- `adw[d|m|r|i]` – actions for adware: `d` – delete, `m` – move,

r – rename, i – ignore;

- dls[d|m|r|i] – actions for dialer programs: d – delete, m – move, r – rename, i – ignore;
- jok[d|m|r|i] – action for joke programs: d – delete, m – move, r – rename, i – ignore;
- rsk[d|m|r|i] – actions for potentially dangerous programs: d – delete, m – move, r – rename, i – ignore;
- hck[d|m|r|i] – actions for programs used for hacking: d – delete, m – move, r – rename, i – ignore;

Interface parameters determine Scanner summary output:

- v, version – show **Scanner** and anti-virus engine version numbers;
- ki – show information about license key file and its owner (only in UTF8);
- foreground[yes|no] – run **Scanner** in foreground or background mode;
- ot – output information to stdout stream;
- oq – disable output;
- ok – display "Ok" for not infected files;
- log=<path to file> – write log to specific file;
- ini=<path to file> – use alternative configuration file;
- lng=<path to file> – use alternative language file. If English interface has been chosen during installation, you may specify ru_scanner.dwl file to display reports in Russian;
- -a=<Agent address> – start **Scanner** in central protection mode;
- --only-key – at start **Scanner** receives from **Agent** only license key file.

These parameters could be disabled if you use them with "–" postfix:

```
-ar -cu -ha -ic -fl -ml -ok -sd -sp
```

Example:

This command will scan <path> directory with heuristic analysis disabled (it is enabled by default)

```
$ drweb <path> -ha-
```

By default **Scanner** starts with the following parameters:

```
-ar -ha -fl- -ml -sd
```

These parameters are optimal for thorough anti-virus protection and can be used in most typical cases. If any parameters is not required, you can disable it with "-" postfix as described above.

Disabling scan of archives and packed files will significantly decrease anti-virus protection level, because viruses are often distributed in archives (especially, self-extracting), enclosed in e-mail attachments. Office documents potentially susceptible to infection with macro viruses (Word, Excel) are also dispatched via e-mail in archives and containers.

When you run **Scanner** with default parameters, no cure actions and no actions for incurable and suspicious files are taken. For these actions to be performed, you must specify corresponding command line parameters explicitly.

Following actions are recommended:

- `cu` – cure infected files and system areas without deleting, moving or renaming infected files;
- `icd` – delete incurable files;
- `spm` – move suspicious files;
- `spr` – rename suspicious files.

When **Scanner** is started with `Cure` action specified, it will try to restore the original state of infected object. It is possible only if detected virus is a known virus, and cure instructions for it are available in virus database, though even in this case cure attempt may fail if infected file is seriously damaged by a virus.

If infected files are found inside archives they will not be cured,

deleted, moved or renamed. To cure such files you must manually unpack archives to the separate directory and instruct **Scanner** to check it.

When **Scanner** is started with action `Delete` specified, it will remove all infected files from disk. This option is suitable for incurable (irreversibly damaged by virus) files.

`Rename` action makes **Scanner** replace file extension with another extension (`*.#??` by default, i.e. first extension character is replaced with "#" character). Enable this parameter for files for other operating systems detected heuristically as suspicious. Renaming helps to avoid accidental execution of such files in these operating systems and therefore prevents infection.

`Move` action makes **Scanner** move infected or suspicious files to the quarantine directory (`%var_dir/infected/` by default). This option actually has a little value since infected and suspicious files for other operating systems can not infect or damage UNIX system. Moving of suspicious files for UNIX system may cause system malfunction or failure.

Recommended command for day-to-day scanning:

```
$ drweb <path> -cu -icd -spm -ar -ha -fl-
-ml -sd
```

You can save this command to the text file and convert it into simple shell script with command:

```
# chmod a+x [file name]
```

Scanner default settings could be changed in the configuration file.

Configuration File

Dr.Web Scanner can be used with default settings, but it could be convenient to configure it according to your specific requirements. **Scanner** settings are stored in configuration file (`drweb32.ini`

by default) which is located in `%etc_dir` directory. To use another configuration file, specify full path to it with command line parameter, for example:

```
$ %bin_dir/drweb -ini=%bin_dir/etc/drweb.ini
```

`[Scanner]`

EnginePath = {path to file}	Location of `drweb32.dll` module (anti-virus engine). This parameter is also used by **Updater**. Default value: **EnginePath** = `%bin_dir/lib/drweb32.dll`
VirusBase = {list of paths to masks}	Masks for loading virus databases. This parameter is also used by **Updater**. Multiple values are allowed. Default value: **VirusBase** = `%var_dir/bases/*.vdb,%var_dir/bases/*.VDB`
UpdatePath = {path to directory}	This parameter is used by **Updater** (`update.pl`) and is mandatory. Default value: **UpdatePath** = `%var_dir/updates/`
TempPath = {path to directory}	Directory where anti-virus engine puts temporary files. It is used when system has insufficient memory or to unpack certain types of archives. Default value: **TempPath** = `/tmp/`
LngFileName = {path	Language file location.

	Default value: **LngFileName** = %bin_dir/lib/ru_scanner.dwl
Key = {path to file}	Key file location (license or demo). Default value: **Key** = %bin_dir/drweb32.key
OutputMode = {Terminal | Quiet}	Output mode: Terminal – console output, Quiet – no output. Default value: **OutputMode** = Terminal
HeuristicAnalysis = {Yes | No}	Enable or disable heuristic detection of unknown viruses. Heuristic analysis can detect previously unknown viruses which are not included in the virus database. It relies on advanced algorithms to determine if scanned file structure is similar to the virus architecture. Because of that heuristic analysis can produce false positives: all objects detected by this method are considered suspicious. Please send all suspicious files to **Dr.Web** through http://vms.drweb.com/sendvirus/ for checking. To send suspicious file, put it in password protected archive, include password in message body and attach **Scanner** report. Default value: **HeuristicAnalysis** = Yes
ScanPriority = {value}	**Scanner** process priority. Value must be within -20 (highest priority) to 19 (Linux) or 20 (other UNIX-like operating systems) range. Default value:

	ScanPriority = 0
FilesTypes = {list of file extensions}	File types to be checked "by type", i.e. when **ScanFiles** parameter (explained below) has ByType value. "*" and "?" wildcard characters are allowed. This parameter can have multiple lines. Default value: FilesTypes = EXE, COM, SYS, OV?, BAT, BIN, DRV, PRG, BOO, SCR, CMD, VXD, 386, DLL, FON, DO?, XL?, WIZ, RTF, CL*, HT*, VB*, JS*, INF, AR?, ZIP, R??, PP?, OBJ, LIB, HLP, MD?, INI, MBR, IMG, CSC, CPL, MBP, SHS, SHB, PIF, SO, CHM, REG, XML, PRC, ASP, LSP, MSO, OBD, THE*, NWS, SWF, BMP, MPP, OCX, DVB, CPY, MSG, EML
FilesTypesWarnings = {Yes | No}	Notify about files of unknown types. Default value: **FilesTypesWarnings** = Yes
ScanFiles = {All | ByType}	Scan only files with extensions specified in **FileType** parameter. This parameter can be used only in local scan mode. Files in mailboxes are always scanned regardless of extension. Default value: **ScanFiles** = All
ScanSubDirectories = {Yes | No}	Enable/disable scanning in subdirectories. Default value: **ScanSubDirectories** = Yes

CheckArchives = {Yes | No}	Enable/disable extracting files archived with (WinZip, InfoZIP etc.), RAR, ARJ, TAR, GZIP, CAB and others. Default value: **CheckArchives** = Yes
CheckEMailFiles = {Yes | No}	Enable/disable checking mailbox files. Default value: **CheckEMailFiles** = Yes
ExcludePaths = {list of path (masks)}	Masks for files to be skipped during scanning. Default value: **ExcludePaths** = /proc,/sys,/dev
FollowLinks = {Yes | No}	Determine if **Scanner** will follow symbolic links during scanning. Default value: **FollowLinks** = No
RenameFilesTo = {mask}	Mask for renaming infected or suspicious files if action Rename is specified. For example, if you specify mask "#??", first character of file extension will be replaced with "#" character, and all other subsequent characters will be preserved. If file has no extension, it will be assigned with ".#" extension. Default value: **RenameFilesTo** = #??
MoveFilesTo = {path to directory}	Path to quarantine directory. Default value:

	MoveFilesTo = %var_dir/infected/
EnableDeleteArchiveAction = {Yes | No}	Enable/disable action Delete for multipart objects (archives, mailboxes, html pages) if they contain infected files. Please note: with this parameter enabled whole multipart object will be deleted (archive, mailbox, etc.), not just infected file or message. Use this option carefully! Default value: **EnableDeleteArchiveAction** = No
InfectedFiles = {Report | Cure | Delete | Move | Rename | Ignore}	Sets **Scanner** action when infected file is found: • Report – only output report to log file; • Cure – attempt to cure the file (only for infected files); • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Delete, Move and Rename actions, specified for archives, containers and mailboxes containing infected files, are applied to the whole archive, container or mailbox! Default value: **InfectedFiles** = Report

SuspiciousFiles = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when suspicious file is found: • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **SuspiciousFiles** = Report
IncurableFiles = {Report |Delete | Move | Rename | Ignore}	Sets **Scanner** action when infected file cannot be cured (should be used only if **InfectedFiles** = Cure): • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **IncurableFiles** = Report
ActionAdware = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when adware is found: • Report – only output report to log file; • Delete – remove the file;

	• `Move` – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • `Rename` – rename the file using mask specified in **`RenameFilesTo`** parameter; • `Ignore` – skip the file. Default value: **`ActionAdware`** = `Report`
`ActionDialers` = `{Report | Delete | Move | Rename | Ignore}`	Sets **Scanner** action when dialer is found: • `Report` – only output report to log file; • `Delete` – remove the file; • `Move` – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • `Rename` – rename the file using mask specified in **`RenameFilesTo`** parameter; • `Ignore` – skip the file. Default value: **`ActionDialers`** = `Report`
`ActionJokes` = `{Report | Delete | Move | Rename | Ignore}`	Sets **Scanner** action when joke program is found: • `Report` – only output report to log file; • `Delete` – remove the file; • `Move` – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • `Rename` – rename the file using mask specified in **`RenameFilesTo`** parameter; • `Ignore` – skip the file.

	Default value: **ActionJokes** = Report
ActionRiskware = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when potentially dangerous program is found: • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **ActionRiskware** = Report
ActionHacktools = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when hacking program is found: • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **ActionHacktools** = Report

ActionInfectedMail = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when infected file is found in mailbox: • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **ActionInfectedMail** = Report
ActionInfectedArchive = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when infected file is found in archive: • Report – only output report to log file; • Delete – remove the file; • Move – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • Rename – rename the file using mask specified in **RenameFilesTo** parameter; • Ignore – skip the file. Default value: **ActionInfectedArchive** = Report
ActionInfectedContainer = {Report | Delete | Move | Rename | Ignore}	Sets **Scanner** action when infected file is found in container: • Report – only output report to log file; • Delete – remove the file;

	• `Move` – move the file to the quarantine directory specified in **MoveFilesTo** parameter; • `Rename` – rename the file using mask specified in **`RenameFilesTo`** parameter; • `Ignore` – skip the file.
	Default value: **`ActionInfectedContainer`** = `Report`
`LogFileName` = {file name}	Log file name. You can specify `syslog` as log file name and logging will be carried out by `syslogd` system service. In this case you must also specify **`SyslogFacility`** and **`SyslogPriority`** parameters. As `syslogd` uses several files for logging various events of different importance, these two parameters and `syslogd` configuration file (usually `/etc/syslogd.conf`) determine location where information is logged to.
	Default value: **`LogFileName`** = `syslog`
`SyslogFacility` = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Logging type to be used by `syslogd` system service.
	Default value: **`SyslogFacility`** = `Daemon`
`SyslogPriority` = {Alert | Warning | Notice | Info | Error}	Logging priority when `syslogd` system service is used.
	Default value: **`SyslogPriority`** = `Info`

LimitLog = {Yes | No}	Enable/disable limit for log file size. If **LogFileName** = syslog this parameter is ignored. With this parameter enabled, **Scanner** will be checking log file size at startup. If log file size exceeds **MaxLogSize** parameter value, log file content will be erased and logging will start from scratch. Default value: **LimitLog** = No
MaxLogSize = {value in Kbytes}	Maximum log file size. Used only with **LimitLog** = Yes. Set this parameter value to 0 if you do not want log file to be unexpectedly modified at start up. Default value: **MaxLogSize** = 512
LogScanned = {Yes | No}	Enable/disable logging of information about all scanned objects regardless whether they are infected or not. Default value: **LogScanned** = Yes
LogPacked = {Yes | No}	Enable/disable logging of additional information about files packed with DIET, PKLITE and other utilities. Default value: **LogPacked** = Yes
LogArchived = {Yes | No}	Enable/disable logging of additional information about files archived with various archiving utilities. Default value:

	LogArchived = Yes
LogTime = {Yes | No}	Enable/disable logging of time for each record. Parameter is not used if **LogFileName** = syslog. Default value: **LogTime** = Yes
LogStatistics = {Yes | No}	Enable/disable logging of scan statistics. Default value: **LogStatistics** = Yes
RecodeNonprintable = {Yes | No}	Non-printable characters output mode for a given terminal. Default value: **RecodeNonprintable** = Yes
RecodeMode = {Replace | QuotedPrintable}	Decoding mode for non printable characters if **RecodeNonprintable** = Yes When **RecodeMode** = Replace all non-printable characters are substituted with **RecodeChar** parameter value (see below). When **RecodeMode** = QuotedPrintable all non printable characters are converted to quoted printable encoding. Default value: **RecodeMode** = QuotedPrintable
RecodeChar = {"?" | "_" | ...}	Sets character for replacing non-printable characters if **RecodeMode** = Replace. Default value: **RecodeChar** = "?"

Following parameters can be used to reduce scanning time in archive files (some objects in archives will not be checked).

MaxCompressionRatio = {value}	Maximum compression ratio, i.e. ratio of unpacked file size to packed file size. If the ratio exceeds specified value, file will not be extracted and therefore will not be checked. Parameter can take only natural values. E-mail message with such file is considered "mail bomb". Default value: **MaxCompressionRatio** = 5000
CompressionCheckThreshold = {value in Kbytes}	Minimum size of file inside archive for which compression ratio check will be performed (if it is specified by **MaxCompressionRatio** parameter). Default value: **CompressionCheckThreshold** = 1024
MaxFileSizeToExtract = {value in Kbytes}	Maximum unpacked size for file in archive. If unpacked size exceed specified value it will not be scanned. E-mail letter with such file is considered "mail bomb". Default value: **MaxFileSizeToExtract** = 500000
MaxArchiveLevel = {value}	Maximum archive nesting level. If archive nesting level exceeds specified value, it will not be scanned. E-mail message with such file is considered "mail bomb". Default value: **MaxArchiveLevel** = 8

MaximumMemoryAllocationSize = {value in Mbytes}	Size of maximum amount of memory consumption allowed for scanning one file (in megabytes). If value is set to 0, memory allocation will not be limited. Default value: **MaximumMemoryAllocationSize** = 0
ScannerScanTimeout = {time in seconds}	Maximum time period allowed for scanning one file (in seconds). If value is set to 0, scanning time will not be limited. Default value: **ScannerScanTimeout** = 0
MaxBasesObsolescencePeriod = {time in hours}	Period after last update during which virus databases are considered up-to-date. When this period is over, notification that databases are obsolete will be output to console. If value is set to 0, database obsolescence will not be checked. Default value: **MaxBasesObsolescencePeriod** = 24
ControlAgent = {Agent socket address}	**Agent** socket address in TYPE:ADDRESS format, where TYPE is for type of socket: inet - TCP socket, local or unix - UNIX socket. **Example**: **ControlAgent** = inet:4040@127.0.0.1,local:%var_dir/ipc/.agent **Scanner** receives from **Agent** license key file and configuration information (if **OnlyKey = No**). Default value:

	ControlAgent = local:%var_dir/ipc/.agent
OnlyKey = {Yes | No}	Enable receiving only license key file from **Agent** without configuration information. **Scanner** will use local configuration file. If this parameter value is set to No **Scanner** will also be sending to **Agent** statistics on scanned files. Default value: **OnlyKey** = No

Running Dr.Web Scanner

You can run **Dr.Web Scanner** with command

```
$ %bin_dir/drweb
```

If %bin_dir directory is added to PATH environment variable, you can run **Dr.Web Scanner** from any directory only by typing "drweb". However, doing so (as well as making a symbolic link to **Dr.Web Scanner** executable file in directories like /bin/, /usr/bin/, etc.) is not recommended for security reasons.

Dr.Web Scanner can be run with either root or user privileges. In the last case virus scanning can be only performed in directories, where user has read access, and infected files will be cured only in directories, where user has write access (usually it is user home directory, $HOME). Also, there are other restrictions when **Scanner** is started with user privileges, for example, on moving and renaming infected files.

When **Scanner** is started, it displays program name, platform name, program version number, release date and contact information. It also shows user registration information and statistics, list of virus databases and installed updates:

```
Dr.Web (R) Scanner for Linux, v6.0.1 (February 19, 2010)
```

```
Copyright (c) Igor Daniloff, 1992-2010
Support service: http://support.drweb.com/
To purchase: http://buy.drweb.com/
Program version: 6.0.0.10060 <API:2.2>
Engine version: 6.0.0.9170 <API:2.2>
Loading  /var/drweb/bases/drwtoday.vdb  -  Ok,
virus records: 1533
Loading  /var/drweb/bases/drw60012.vdb  -  Ok,
virus records: 3511
-------------------------------------------
Loading  /var/drweb/bases/drw60000.vdb  -  Ok,
virus records: 1194
Loading  /var/drweb/bases/dwn60001.vdb  -  Ok,
virus records: 840
Loading  /var/drweb/bases/drwebase.vdb  -  Ok,
virus records: 78674
Loading  /var/drweb/bases/drwrisky.vdb  -  Ok,
virus records: 1271
Loading  /var/drweb/bases/drwnasty.vdb  -  Ok,
virus records: 4867
Total virus records: 538681
Key file: /opt/drweb/drweb32.key
Key file number: XXXXXXXXXX
Key file activation date: XXXX-XX-XX
Key file expiration date: XXXX-XX-XX
```

After displaying this report **Scanner** terminates. In order to scan for viruses or neutralize detected threats you must specify additional command-line parameters.

Dr.Web Daemon

Dr.Web Daemon is a background antivirus module designed to perform scanning for viruses on request from other **Dr.Web** software components. It can scan files on disk or data transferred through socket. Requests for scanning are sent using special protocol via UNIX sockets or TCP sockets. **Dr.Web Daemon** uses the same antivirus engine and virus databases as **Scanner** and is able to detect and cure all known viruses.

Dr.Web Daemon is always running and has simple and straightforward protocol for sending scanning requests. Because of that, it is a perfect solution to be used as antivirus filter for

Command-line Parameters

As any other UNIX program **Dr.Web Daemon** supports command line parameters. They are separated from specified path by white space and are prefixed by hyphen "-". To get complete list of parameters, run `drwebd` with `-?`, `-h` or `-help` parameters.

Dr.Web Daemon has the following command line parameters:

- `-ini=<path to file>` – use alternative configuration file;
- `-lng=<path to file>` – use alternative language file. Specify `ru_daemon.dwl` if you want display program messages in Russian language;
- `--foreground=<yes|no>` – set **Daemon** running mode. Specify "`Yes`" if you want to run **Daemon** in foreground mode or "`No`" to run it in background(daemon) mode;
- `--check-only <command line parameters for checking>` – check validity of **Daemon** configuration parameters and specified command line parameters (if any);
- `-a=<Agent address>` – run **Daemon** in central protection mode (with configuration and key file received

from **Agent**);

- `--only-key` – receive only license key file from **Agent**, **Daemon** will use local configuration file.

Running Dr.Web Daemon

When **Daemon** is started with default settings, the following actions are performed:

- Configuration file is located and loaded. If configuration file is not found, then loading process terminates. Path to configuration file can be specified at startup by the command line parameter `-ini: {path/to/your/drweb32.ini}` or default value (`%etc_dir/drweb32.ini`) can be used. At start several configuration parameters get validated, and if parameter value is incorrect, default value is applied;
- Log file is created. User account used by **Daemon** must have appropriate privileges to write to the log file directory. Users do not have write permission for the default log directory (`/var/log/`). If **`User`** parameter is specified, you must also redefine **`LogFileName`** parameter and provide alternative log file location;
- Key file is loaded from the location specified in configuration file. If the key file is not found, then loading process terminates;
- If **`User`** parameter is specified, **Daemon** will offer to create an appropriate user account (default value is `drweb`) and to use it with the permissions provided;
- **Engine** (`drweb32.dll`) is loaded. If **Engine** is damaged or not found (because of some errors in configuration file), then loading process terminates;
- Virus databases are loaded in arbitrary sequence from the location specified in configuration file. If virus databases are damaged or absent, loading process proceeds;
- **Daemon** enters daemon mode, so all information about loading problems can not be output to console and is written to log file;
- Socket for interaction between **Daemon** and other **Dr.Web**

for UNIX mail servers solution modules is created. When TCP-sockets are used, there can be several connections (loading continues if at least one connection is established). When UNIX socket is used, **Daemon**'s user account must have appropriate privileges to read from the directory containing this socket and write to it. User accounts for modules must have execution access to the directory itself and write and read access to the socket file. Users do not have write permission for the default socket directory (`/var/run/`). If **`User`** parameter is specified, you must also redefine **`Socket`** parameter and provide alternative path to socket file. If UNIX socket was not created, then loading then loading process terminates;

- PID-file with **Daemon** PID information and transport addresses is created. User account used by **Daemon** must have appropriate privileges to write to the directory containing PID-file. Users do not have write permission for the default socket directory (`/var/run/`). If **`User`** parameter is specified, you must also redefine **`PidFile`** parameter and provide alternative path to PID-file. If PID-file was not created, then loading then loading process terminates.

Dr.Web Daemon Testing and Diagnostics

If no problems have occurred during initialization, **Daemon** is ready to work. To make sure that daemon have initialized correctly, issue command

```
$ netstat -a
```

and check whether necessary sockets have been created.

TCP sockets:

```
--- cut ---
Active Internet connections (servers and
established)
Proto  Recv-Q  Send-Q  Local Address  Foreign
```

```
Address  State
tcp  0  0  localhost:3000  *:*  LISTEN
raw  0  0 *:icmp  *:*  7
raw  0  0 *:tcp  *:*  7
Active UNIX domain sockets (servers and
established)
Proto  RefCnt  Flags  Type  State  I-Node  Path
unix  0  [ ACC ]  STREAM  LISTENING  384  /dev/
gpmctl
unix  0  [ ]  STREAM  CONNECTED  190  @0000001b
unix  1  [ ]  STREAM  CONNECTED  1091
@00000031
unix  0  [ ACC ]  STREAM  LISTENING  403 /tmp/.
font-unix/fs7100
unix  4  [ ]  DGRAM  293  /dev/log
unix  1  [ ]  STREAM  CONNECTED  1092  /dev/
gpmctl
unix  0  [ ]  DGRAM  450
unix  0  [ ]  DGRAM  433
unix  0  [ ]  DGRAM  416
unix  0  [ ]  DGRAM  308
--- cut ---
```

Unix socket:

```
--- cut ---
Active Internet connections (servers and
established)
Proto  Recv-Q  Send-Q  Local Address  Foreign
Address  State
raw  0  0 *:icmp  *:*  7
raw  0  0 *:tcp  *:*  7
```

```
Active UNIX domain sockets (servers and
established)
Proto  RefCnt  Flags  Type  State  I-Node  Path
unix  0  [ ACC ]  STREAM  LISTENING  384 /dev/
gpmctl
unix  0  [ ]  STREAM  CONNECTED  190 @0000001b
unix  1  [ ]  STREAM  CONNECTED  1091 @00000031
unix  0  [ ACC ]  STREAM  LISTENING  1127 %
var_dir/.daemon
unix  0  [ ACC ]  STREAM  LISTENING  403 /tmp/.
font-unix/fs7100
unix  4  [ ]  DGRAM  293  /dev/log
unix  1  [ ]  STREAM  CONNECTED  1092 /dev/
gpmctl
unix  0  [ ]  DGRAM  450
unix  0  [ ]  DGRAM  433
unix  0  [ ]  DGRAM  416
unix  0  [ ]  DGRAM  308
--- cut ---
```

If necessary sockets are missing from this list, there were problems with **Daemon** initialization.

To run functional test and obtain service information use console client for **Daemon** (`drwebdc`).

TCP sockets:

```
$ drwebdc -nHOSTNAME -pPORTNUM -sv -sb
```

Unix socket:

```
$ drwebdc -uSOCKETFILE -sv -sb
```

It should output report to the console similar to this:

```
--- cut ---
- Version: DrWeb Daemon 6.00
```

```
- Loaded bases:
Base /var/drweb/bases/drwtoday.vdb
contains 5 records.
Base /var/drweb/bases/drw60003.vdb
contains 409 records.
Base /var/drweb/bases/drw60002.vdb
contains 543 records.
Base /var/drweb/bases/drwebase.vdb
contains 51982 records.
Base /var/drweb/bases/drw60001.vdb
contains 364 records.
Total 53303 virus-finding records.
--- cut ---
```

If that did not happen, run extended diagnostics.

For TCP-socket:

```
$ drwebdc -nHOSTNAME -pPORTNUM -sv -sb -v
```

For UNIX socket:

```
$ drwebdc -uSOCKETFILE -sv -sb -v
```

More detailed report can identify the problem:

```
dwlib: fd: connect() failed - Connection
refused
dwlib: tcp: connecting to 127.0.0.1:3300 -
failed
dwlib: cannot create connection with a DrWeb
daemon
ERROR: cannot retrieve daemon version
Error -12
```

You can test **Daemon** with special `eicar.com` program included in the installation package. Use your text editor of choice to transform `readme.eicar` into `eicar.com` (see instructions within the file).

For TCP-socket:

```
$ drwebdc -nHOSTNAME -pPORTNUM eicar.com
```

For UNIX socket:

```
$ drwebdc -uSOCKETFILE eicar.com
```

It should output the following result:

```
Results: daemon return code 0x20
(known virus is found)
```

If that did not happen, check **Daemon** log file to see whether the file has been scanned. If the file has not been scanned, run extended diagnostic (see above).

If file was scanned successfully, **Daemon** is ready to operate.

When scanning very large archives, some issues with timeout expiration may occur. To fix this, increase values of `FileTimeout` and `SocketTimeout` parameters.

Please note that **Daemon** cannot scan files larger than 2 gigabytes. Such files will not be sent to **Daemon** for scanning by clients.

Scanning Modes

Dr.Web Daemon can scan for viruses

- chunks of data received from socket (remote scanning mode);

- files on disk (local scanning mode).

In the remote mode client send data to be scanned to **Daemon** through socket. **Daemon** can scan both anonymous memory and memory mapped to file system, the only difference will be logging information. This mode enables scanning files without read access but is less efficient than local scanning.

Local scanning mode is easier to use and provides better performance since client sends to **Daemon** only path to file to be scanned, not the whole file. Because clients can be located on different computers,the path must be specified with regard to the actual location of **Daemon**.

Local scan mode requires careful management of user privileges. **Daemon** must have read access to each file to be scanned. To perform **Cure** and **Delete** actions for files in mailboxes, you must also permit write access.

Usage of **Daemon** with mail servers requires special attention, because mail filters usually act on behalf of the mail system and use its privileges. In local scan mode mail filter usually creates a file with the message received from the mail system and provides **Daemon** a path to it. At this point you must carefully specify access permissions to the directory, where filters create appropriate files. We recommend either to include user whose privileges are used by **Daemon** into the mail subsystem group, or to run **Daemon** with the privileges of the mail system user.

Properly configured system does not require **Daemon** to use root privileges.

Signal processing

Dr.Web Daemon can receive and process the following signals:

- `SIGHUP` – reload configuration file;
- `SIGUSR1` – output process pull statistics to log file;

- SIGTERM – request **Daemon** termination;
- SIGKILL – force **Daemon** termination (if any problems have emerged).

Log Files

Since **Dr.Web Daemon** is a background program, information on its operation can only be obtained through log file. Log file contains details on processing of all scanning request sent to **Daemon**. You can set the log file location Alternatively, you can use syslog service to handle the logging.

Also, logging can be split to different files depending on **Daemon**'s client by setting ClientsLogs parameter. You can use this option to set up different **Daemon** log files for different clients that use the same **Daemon** to process their scanning requests.

Regardless of ClientsLogs parameter, if **Daemon** recognizes its client, scanning results will begin with prefix that identifies the client. Following prefixes are possible:

- <web> - **Dr.Web ICAPD**;
- <smb_spider> - **Dr.Web Samba SpIDer**;
- <mail> - **Dr.Web MailD**;
- <drwebdc> - console client for **Dr.Web Daemon**;
- <kerio> - **Dr.Web for Kerio Internet Gateways**;
- <lotus> - **Dr.Web for IBM Lotus Domino**.

On FreeBSD operating system information output to console by **Daemon** may be intercepted by syslog service and logged character-by-character. This is an issue of FreeBSD logging service that manifest itself if in syslog.conf configuration file logging level is set as *.info.

Pool statistics

Statistics on pool used for processing scanning request is output to

the log file on receiving SIGUSR1 signal (signal must be sent only to parent process, if a child process receives SIGUSR1, it will terminate) and on termination of **Daemon**.

Example of pool statistics output:

```
Fri Oct 15 19:47:51 2010 processes pool statistics: min = 1 max = 1024 (auto) freetime = 121 busy max = 1024 avg = 50.756950 requests for new process = 94 (0.084305 num/sec) creating fails = 0 max processing time = 40000 ms; avg = 118646 ms curr = 0 busy = 0
```

where:

- `min` - minimal number of processes in the pool;
- `max` - maximal number of processes in the pool;
- `(auto)` - displayed if limits on number of processes in the pool are determined automatically;
- `freetime` - maximum idle time for process in the pool;
- `busy max` - maximum number of simultaneous busy processes, `avg` - average number of simultaneous busy processes;
- `requests for new process` - number of request for new process creation (frequency of requests per second is displayed in parenthesis);
- `creating fails` - number of failed attempts of new process creation (failures are usually caused by insufficient resources);
- `max processing time` - maximum time for processing a single scanning request, `avg` - average time for processing a single scanning request;
- `curr` - current number of all processes in the pool;
- `busy` - current number of busy processes in the pool.

Configuration

Dr.Web Daemon can be used with default settings, but it could be convenient to configure it according to your specific requirements.

Daemon settings are stored in `[Daemon]` section of the configuration file (`drweb32.ini` by default) which is located in `%etc_dir` directory. To use another configuration file specify full path to it with command-line option.

EnginePath = {path to file}	Location of `drweb32.dll` module (anti-virus engine). This parameter is also used by the **Updater**. Default value: **EnginePath** = `%bin_dir/lib/drweb32.dll`
VirusBase = {list of files or masks}	Masks for loading virus databases. This parameter is also used by the **Updater**. Multiple values are allowed. Default value: **VirusBase** = `%var_dir/bases/*.vdb,%var_dir/bases/*.VDB`
UpdatePath = {path to directory}	This parameter is used by the **Updater** (`update.pl`) and is mandatory. Default value: **UpdatePath** = `%var_dir/updates/`
TempPath = {path to directory}	Directory where anti-virus engine creates temporary files. It is used when system has insufficient memory or when it is necessary to unpack certain types of archives. **TempPath** = `%var_dir/spool/`

Key = {path to file}	Key file location (license or demo). Please note that **Daemon** and **Scanner** can have different license key files. In this case you must change the value of this parameter correspondingly. **Daemon** can use several license key files simultaneously. For each of them **Key** parameter value in [Daemon] section of drweb32.ini file must be specified. In this case **Daemon** tries to combine all license permissions from all available license key files. Default value: **Key** = %bin_dir/drweb32.key
MailAddressesList = {path to file}	This parameter is used only if you have e-mail license for 15 or 30 addresses. Specified file must contain a list of e-mail addresses (15 or 30 as specified by the license, one e-mail address per line), for which both incoming and outgoing messages will be checked. Aliases are treated as separate addresses. Default value: **MailAddressesList** = %etc_dir/email.ini
OutputMode = {Terminal | Quiet}	Output mode: Terminal - output to console, Quiet - do not output. Default value: **OutputMode** = Terminal
RunForeground = {Yes | No}	Allows to disable or enable daemon mode for the **Dr.Web Daemon**. With Yes value specified **Daemon** will run as foreground process. This parameter can be used for certain monitoring utilities (for example, daemontools). Default value:

	RunForeground = No
User = {user name}	User which privileges will be used by the **Daemon**. It is strongly recommended to create separate **drweb** user account, which will be used by the **Daemon** and filters. It is not recommended to run **Daemon** with root privileges, even though it may take less time to set up. This parameter cannot be changed when reloading configuration using SIGHUP. Default value: **User** = drweb
PidFile = {path to file}	File to store **Daemon**'s PID and UNIX socket (if it is enabled by **Socket** parameter) or port number (if TCP socket is enabled by **Socket** parameter). If more than one **Socket** parameter is specified, this file will contain information on all the sockets (one per line). This file is created every time **Daemon** starts. Default value: **PidFile** = %var_dir/run/drwebd.pid
BusyFile = {path to file}	File where **Daemon**'s busy flag is stored. This file is created by a **Daemon** child process upon a receipt of the scan command and is removed after successful execution of the command. Filenames created by each **Daemon** child process are appended by a dot and ASCIIZ representation of PID (for example, /var/run/drwebd.bsy.123456). Default value: **BusyFile** = %var_dir/run/drwebd.bsy

ProcessesPool = {process pool settings}	Dynamic process pool settings. First number of processes in the pool must be specified: • auto – number of processes will be set automatically depending on system load; • N – unsigned integer number. Pool will have at least N active processes, additional processes will be created if necessary; • N-M – integer unsigned numbers, M>=N. Pool will have at least N active processes, additional processes will be created if necessary, but maximum total number of processes cannot exceed M. Then, optional secondary parameters may be specified: • **timeout** = {time in seconds} — timeout for closing an inactive process. This parameter does not affect first N processes which await requests continually. • **stat** = {yes|no} — statistics for processes in a pool. It is saved each time SIGUSR1 system signal is received, to the directory specified in the value of BaseDir parameter from General section. • **stop_timeout** = {time in seconds} — maximum waiting period for stopping a working process.
	Default value: **ProcessesPool** = auto,**timeout** = 120, **stat** = no, **stop_timeout** = 1

OnlyKey = {Yes | No}	Only license key file will be received from the **Agent**. Local configuration file will be used for all the settings. Default value: **OnlyKey** = No
ControlAgent = {socket address}	Agent socket address in TYPE:ADDRESS format, where TYPE is for type of the socket: • inet – TCP socket, • local or unix – UNIX socket. **Example**: **ControlAgent** = inet:4040@127.0.0.1,local:%var_dir/ipc/.agent **Daemon** receives from the **Agent** license key file and configuration information (if **OnlyKey** = No). Default value: **ControlAgent** = local:%var_dir/ipc/.agent
MailCommand = {command}	Command used by the **Daemon** and the **Updater** for sending notifications and information bulletins on new updates to the user (administrator) via e-mail. If less than two weeks left until the key file (or one of the key files) expires, **Daemon** starts sending out notifications every time system starts, restarts or reboots. Default value: **MailCommand** = "/usr/sbin/sendmail -i -bm -f drweb -- root"

NotifyPeriod = {value}	This parameter value specifies how many days should be left before license expiration for the **Daemon** to start sending notifications of license renewal. If parameter value is set to 0, **Daemon** starts sending out notifications immediately after key file expires. Default value: **NotifyPeriod** = 14
NotifyFile = {path to file}	File with a timestamp of last notification of license renewal. It is send to the administrator after the key file expires. Default value: **NotifyFile** = %var_dir/.notify
NotifyType = {Ever | Everyday | Once}	Frequency of license expiration notifications. Once - notification is sent only once. Everyday - notification is sent daily. Ever - notification is sent with every **Daemon** restart and every database update. Default value: **NotifyType** = Ever
FileTimeout = {value in seconds}	Maximum time allowed for the **Daemon** to perform a scan of one file. Default value: **FileTimeout** = 30
StopOnFirstInfected = {Yes | No}	Enables or disables stopping file scan upon detection of the first virus. Default value: **StopOnFirstInfected** = No

ScanPriority = {value}	Priority of **Daemon** process. Value must be in the following range: -20 (highest priority) to 19 (lowest priority for Linux) or 20 (lowest priority for FreeBSD and Solaris). Default value: **ScanPriority** = 0
FilesTypes = {list of file extensions}	File types to be checked "by type", i.e. when **ScanFiles** parameter (explained below) has ByType value. "*" and "?" wildcard characters are allowed. This parameter can have multiple lines. Default value: **FilesTypes** = EXE, COM, SYS, OV?, BAT, BIN, DRV, PRG, BOO, SCR, CMD, VXD, 386, DLL, FON, DO?, XL?, WIZ, RTF, CL*, HT*, VB*, JS*, INF, AR?, ZIP, R??, PP?, OBJ, LIB, HLP, MD?, INI, MBR, IMG, CSC, CPL, MBP, SHS, SHB, PIF, SO, CHM, REG, XML, PRC, ASP, LSP, MSO, OBD, THE*, NWS, SWF, BMP, MPP, OCX, DVB, CPY, MSG, EML
FilesTypesWarnings = {Yes | No}	Notify about files of unknown types Default value: **FilesTypesWarnings** = Yes
ScanFiles = {All | ByType}	Scan only files with extensions specified in **FileTypes** parameter. This parameter can be used only in local scan mode. Files in mailboxes are always scanned regardless of extension. Default value: **ScanFiles** = All

CheckArchives = {Yes | No}	Enables or disables extracting files archived with (WinZip, InfoZIP etc.), RAR, ARJ, TAR, GZIP, CAB and others. Default value: **CheckArchives** = Yes
CheckEMailFiles = {Yes | No}	Enables or disables checking mailbox files. Default value: **CheckEMailFiles** = Yes
ExcludePaths = {list of paths or masks}	Masks for files to be skipped during scan. Default value: **ExcludePaths** = /proc,/sys,/dev
FollowLinks = {Yes | No}	Determine if **Daemon** will follow symbolic links during scanning. Default value: **FollowLinks** = No
RenameFilesTo = {mask}	Mask for renaming infected or suspicious files if action Rename is specified. For example, if you specify mask "#??", first character of file extension will be replaced with "#" character, and all other subsequent characters will be preserved. If the file has no extension, it will be assigned with ".#" extension. Default value: **RenameFilesTo** = #??
MoveFilesTo = {path to directory}	Path to the quarantine directory. Default value: **MoveFilesTo** = %var_dir/infected/

BackupFilesTo = {path to directory}	Directory for backup copies of infected files made if requested action was Cure. Default value: **BackupFilesTo** = %var_dir/infected/
LogFileName = {file name}	Log file name. You can specify syslog as log file name and logging will be carried out by syslogd system service. In this case you must also specify **SyslogFacility** and **SyslogPriority** parameters. As syslogd uses several files for logging various events of different importance, these two parameters and syslogd configuration file (usually /etc/syslogd.conf) determine location where information is logged to. Default value: **LogFileName** = syslog
SyslogFacility = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Logging type to be used by syslogd system service. Default value: **SyslogFacility** = Daemon
SyslogPriority = {Alert | Warning | Notice | Info | Error}	Logging priority when syslogd system service is used. Default value: **SyslogPriority** = Info

LimitLog = {Yes | No}	Enables or disables limit for log file size. This parameter is not used if LogFileName = syslog. If limit is enabled, **Daemon** will check the size of log file on startup or receiving the SIGHUP signal. If log file size is greater then MaxLogSize value, log file will overwritten with empty file and logging will begin from scratch. Default value: **LimitLog** = No
MaxLogSize = {value in Kbytes}	Maximum log file size. Used only with **LimitLog** = Yes. Default value: **MaxLogSize** = 512
LogScanned = {Yes | No}	Enable/disable logging of information about all scanned objects regardless whether they are infected or not. Default value: **LogScanned** = Yes
LogPacked = {Yes | No}	Enable/disable logging of additional information about files packed with DIET, PKLITE and other utilities. Default value: **LogPacked** = Yes
LogArchived = {Yes | No}	Enable/disable logging of additional information about files archived with various archiving utilities. Default value: **LogArchived** = Yes

LogTime = {Yes | No}	Enable/disable logging of time for each record. Parameter is not used if LogFileName = syslog. Default value: **LogTime** = Yes
LogProcessInfo = {Yes | No}	Enable/disable logging of every scanning process PID and filter address (host name or IP) from which scanning has been activated. This data is put before each record. Default value: **LogProcessInfo** = Yes
RecodeNonprintable = {Yes | No}	Non-printable characters output mode for a given terminal. Default value: **RecodeNonprintable** = Yes
RecodeMode = {Replace | QuotedPrintable}	Decoding mode for non printable characters if RecodeNonprintable = Yes. When RecodeMode = Replace, all non-printable characters are substituted with RecodeChar parameter value (see below). When RecodeMode = QuotedPrintable, all non printable characters are converted to quoted printable encoding. Default value: **RecodeMode** = QuotedPrintable
RecodeChar = {"?" | "_" | ...}	Character to replace non-printable characters if RecodeMode = Replace. Default value: **RecodeChar** = "?"

Socket = {socket address}	Socket to be used for communication with **Daemon**. Sockets can be specified in several ways. If it is necessary to specify several socket addresses in one string, you should use TYPE:ADDRESS format, where TYPE is the type of socket: inet - TCP socket, local or unix - UNIX socket. **Example**: Socket = inet:3000@127.0.0.1, local:%var_dir/.daemon Also you can specify socket address in PORT [interfaces] | FILE [access] format. For a TCP socket, specify decimal port number (PORT) and the list of interface names or IP addresses for incoming requests (interfaces). **Example:** **Socket** = 3000 127.0.0.1, 192.168.0.100 For UNIX sockets, specify socket name (FILE) and access permissions in octal form (access). Example: **Socket** = %var_dir/.daemon Number of Socket parameters is not limited. **Daemon** will work with all correctly described sockets. To enable connections on all available interfaces set 3000 0.0.0.0 as a value for this parameter.
	Default value: **Socket** = %var_dir/run/.daemon

SocketTimeout = {value in seconds}	Maximum time allowed for transferring data through socket (file scanning time is not included). Default value: **SocketTimeout** = 10
ClientsLogs = {list}	Enables splitting the log files. If during communication with **Daemon** client uses the option to transfer its ID, log file will be substituted with the file specified in this parameter. Descriptions of log files are delimited by commas or whitespaces. If more than six values are set, configuration file is considered invalid. The log files are defined in the following way: <client name1>:<path to file>, <client name2>:<path to file> Client name may be one of the following: • web — **Dr.Web ICAPD**; • smb_spider — **Dr.Web Samba SpIDer**; • mail — **Dr.Web MailD**; • drwebdc — console client for **Dr.Web Daemon**; • kerio — **Dr.Web for Kerio Internet Gateways**; • lotus — **Dr.Web for IBM Lotus Domino**. Example: drwebdc:/var/drweb/log/drwebdc.log, smb:syslog,

	`mail:/var/drweb/log/drwebmail.log` Default value:
MaxBasesObsolescencePeriod = {time in hours}	Period after last update during which virus databases are considered up-to-date. When this period is over, notification that databases are obsolete will be output to console. If value is set to 0, database obsolescence will not be checked. Default value: **MaxBasesObsolescencePeriod** = 24

The following parameters can be used to reduce scanning time in archive files (some objects in archives will not be checked). Actions applied to skipped archives are determined in `ArchiveRestriction` parameter of the corresponding modules.

MaxCompressionRatio = {value}	Maximum compression ratio, i.e. ratio of unpacked file size to packed file size. Parameter can take only natural values. If the ratio exceeds specified value, file will not be extracted and therefore will not be checked. Default value: **MaxCompressionRatio** = 500
CompressionCheckThreshold = {value in Kbytes}	Minimum size of the file inside an archive for which compression ratio check will be performed (if it is specified by the **MaxCompressionRatio** parameter). Default value: **CompressionCheckThreshold** =

	1024
MaxFileSizeToExtract = {value in Kbytes}	Maximum unpacked size for the file in an archive. If unpacked size exceeds specified value the archive will not be scanned. Default value: **MaxFileSizeToExtract** = 40960
MaxArchiveLevel = {value}	Maximum archive nesting level. If archive nesting level exceeds specified value, the archive will not be scanned. Default value: **MaxArchiveLevel** = 8
MessagePatternFileName = {path to file}	Path to template for message about license expiration. You can define expiration message according to your requirements. You can use variables that will be substituted for the following values: • $EXPIRATIONDAYS — number of day left until the license would expire; • $KEYFILENAME — path to license key file; • $KEYNUMBER — license number; • $KEYACTIVATES — license activation date; • $KEYEXPIRES — license expiration date. If there is no user-defined template, standard message in English will be used. Default value: **MessagePatternFileName =**

	%etc_dir/templates/drwebd/msg.tmpl
MailTo = {email address}	Administrator email address to send messages about license expiration, virus databases obsolescence, etc. Default value: **MailTo =**

Dr.Web Updater

You can use **Dr.Web Updater** to update automatically virus databases and content-specific black and white lists of Internet resources for the **Dr.Web for UNIX mail servers** solution. Updating module is implemented as a console script `update.pl` written in Perl, and you can find it in the directory containing **Dr.Web for UNIX mail servers** executable files.

Dr.Web Updater settings are stored in `[Updater]` section of the `drweb32.ini` configuration file in `%etc_dir` directory. If you want to use alternative configuration file, specify the full path to it with command line parameter at start.

To run the script use the following command:

```
$ %bin_dir/update.pl [parameters]
```

Updating

To ensure reliable protection **Dr.Web for UNIX mail servers** solution requires regular updates of virus databases.

Dr.Web for UNIX mail servers virus databases are stored in files with *.`vdb` extension. Update servers may also store them in lzma-archives. When new viruses are discovered, small files (only several KBytes in size) with database segments describing these viruses are released to provide quick and effective countermeasures.

Updates are the same for all supported platforms. There are daily "hot" updates (`drwtoday.vdb`) and regular weekly updates (`drwXXXYY.vdb`), where `XXX` is antivirus engine version number, and YY is a sequential number, beginning from 00 (for example, the first regular update for version 6.0 will be named `drw60000.vdb`).

"Hot" updates are released daily or even several times a day to provide effective protection against new viruses. These updates are

installed over the old ones: i.e. previous `drwtoday.vdb` file will be overwritten. When new regular update is released, all records from `drwtoday.vdb` are copied to `drwXXXYY.vdb`, and new empty `drwtoday.vdb` file is issued.

If you want to update virus databases manually, you must install all missing regular updates first, and then overwrite `drwtoday.vdb` file.

To add the update to the main virus databases, place the corresponding file to the directory for **Dr.Web for UNIX mail servers** executable files (`/var/drweb/bases/` by default) or to any other directory specified in the configuration file.

Signatures for virus-like malicious programs (adware, dialers, hacktools, etc.) are supplied in two additional files - `drwrisky.vdb` and `drwnasty.vdb` - with the structure similar to virus databases. These files are also updated regularly: `dwrXXYYY.vdb` and `dwnXXYYY.vdb` are for regular updates, and `dwrtoday.vdb` and `dwntoday.vdb` are for "hot" updates.

From time to time (as new antivirus techniques are developed), new versions of the antivirus package are released, containing the updated algorithms, implemented in the antivirus engine (**Engine**). At the same time, all released updates are brought together, and the new package version is completed with the updated main virus databases with descriptions of all known viruses. Usually, when upgrading the package to the new version the databases remain portable: i.e. new bases can be linked up to the old **Engine**. Please note that this does not guarantee detection or curing of new viruses, as it requires upgrading of algorithms in the **Engine**.

With regular updating virus databases have the following structure:

- `drwebase.vdb` – general virus database, received with the new version of the package;
- `drwXXXYY.vdb` – regular weekly updates;
- `drwtoday.vdb` – "hot" updates released daily or several times a day;
- `drwnasty.vdb` – general database of other malware,

received with the new version of the package;

- `dwnXXXYY.vdb` – regular weekly updates for other malware;
- `dwntoday.vdb` – "hot" updates for other malware;
- `drwrisky.vdb` – general database of riskware, received with the new version of the package;
- `dwrXXXYY.vdb` – regular weekly updates for riskware;
- `dwrtoday.vdb` – "hot" updates for riskware.

Virus databases can be automatically updated using **Dr.Web Updater** module (`%bin_dir/update.pl`). After installation user crontab file `/etc/cron.d/drweb-update` will be created to run **Updater** every 30 minutes to ensure regular updates and maximum protection. You can modify this file to change update period.

Cron Configuration

For Linux: a special file with user settings will be created in the `/etc/cron.d/` directory during installation of the software complex. It will enable interaction between `cron` and **Dr.Web Updater**.

In the task created for `crond`, vixie cron syntax is used. If you use different `cron` daemon, such as `dcron`, it is necessary to manually create a task to automatically start the **Dr.Web Updater** module.

For FreeBSD and Solaris: manual configuration of `cron` is required to enable its interaction with **Dr.Web Updater**.

For example, when you use FreeBSD you may add the following

string to `crontab` of `drweb` user:

```
*/30 * * * * /usr/local/drweb/update.pl
```

If you work with Solaris, the following set of commands is used:

```
# crontab -e drweb
# 0,30 * * * * /opt/drweb/update.pl
```

By default `cron` daemon launch Dr.Web Updater module every 0 and 30 minutes of every hour. This can cause increased load on the update servers of **Doctor Web** and cause update delays. To avoid such situation, it is recomended to change default values to arbitrary.

Command Line Parameters

--help parameter is used to show brief usage summary.

To use another configuration file, specify full path to it with --ini command line parameter. If the name of the configuration file is not specified, `%etc_dir/drweb32.ini` is used.

Example:

```
$ /opt/drweb/update.pl -ini=/path/to/conf_file
```

--what command line parameter allows to temporarily override value of Section parameter on Updater's launch. Parameter will take effect until next start of the script. Possible values: scanner or daemon.

Example:

```
$ /opt/drweb/update.pl --what=Scanner
```

--components parameter is used to view a list of all product components available for update.

Example:

```
$ /opt/drweb/update.pl --components
```

`--what` command line parameter allows to temporarily override value of **`Section`** parameter on Updater's startup. Parameter will take effect until next start of the script. Possible values: `scanner` or `daemon`.

You can also use command line parameter `--not-need-reload`:

- without this parameter **Dr.Web Daemon** will be restarted after `update.pl` script finishes its work. (Note: **Daemon** will be restarted only if some of its components have been updated, removed or added);
- if `--not-need-reload` parameter specified without any value, **Daemons** will not be restarted after `update.pl` script finishes its work;
- `not-need-restart` parameter can specified with names of the daemons that would not be restarted.

Example:

```
$ /opt/drweb/update.pl --not-need-reload=drwebd
```

Blocking Updates for Selected Components

You can configure **Updater** to block updates for selected components of your **Dr.Web** solution.

To view the list of available components, use `--components` command-line parameter:

Example:

```
# ./update.pl --components

Available Components:
    agent
    drweb          (frozen)
    icapd          (frozen)
    vaderetro_lib
```

If updates for any component are blocked, that component will be marked as frozen. Frozen components will not be updated when **Updater** is ran.

Blocking updates

To block updates for specific component use `--freeze=<components>` command-line parameter, where `<components>` is a comma-delimited list of names of components to be frozen.

Example:

```
# ./update.pl --freeze=drweb
Updates for component 'drweb' are frozen.
Run command './updater --unfreeze drweb' to start updates again.
```

Unblocking updates

To once again enable updates for a frozen component, use `--unfreeze=<components>` command-line parameter, where `<components>` is a comma-delimited list of names of components to be unfrozen.

Example:

```
# ./update.pl --unfreeze=drweb
Updates for component 'drweb' are no longer frozen.
```

Unfreezing will not update the component.

Restoring Components

When updating components of your **Dr.Web** solution, back-up

copies will be saved in **Updater** working directory. It enables you to restore any component to its previous state in case there are some problems with the update.

To restore component to a previous state, use `--restore==<components>` command-line parameter, where `<components>` is a comma delimited list of components to be restored.

Example:

```
# ./update.pl --restore=drweb
Restoring backup for component 'drweb'...
Updates for component 'drweb' are frozen.
Run command './updater --unfreeze drweb' to start updates again.

Backup for component 'drweb' has been restored!
Dr.Web (R) restore details:

Following files has been restored:
      /var/drweb/bases/drwtoday.vdb
      /var/drweb/bases/dwntoday.vdb
      /var/drweb/bases/dwrtoday.vdb
      /var/drweb/bases/timestamp
      /var/drweb/updates/timestamp
```

On restoring component will be automatically frozen. To enable updates for a restored component you need to unfreeze it.

Configuration File

Dr.Web Updater settings are stored in `Updater` section of

configuration file (`drweb32.ini` by default) which is located in `%etc_dir` directory:

`[Updater]`

`UpdatePluginsOnly` = `{Yes | No}`	With `Yes` value specified **Dr.Web Updater** will not update **Daemon** and **Scanner**. It will update only plug-ins. Default value: **`UpdatePluginsOnly`** = `No`
`Section` = `{Daemon | Scanner}`	Specifies from which section of configuration file **Updater** will take settings to determine path to key file, paths to virus databases, etc. Possible values: `Scanner, Daemon`. Value of this parameter can be temporarily overriden by `--what` command line parameter. Parameter will take effect until next start of the script. Default value: **`Section`** = `Daemon`
`ProgramPath` = `{path to file}`	Path to **Daemon** or **Scanner**. It is used by **Dr.Web Updater** for getting the product version and API information of the installed executable file. Default value: **`ProgramPath`** = `%bin_dir/drwebd`
`SignedReader` = `{path to file}`	Path to program used to read digitally signed files. Default value: **`SignedReader`** = `%bin_dir/read_signed`
`LzmaDecoderPath` = `{path to file}`	Path to program used for unpacking of lzma-archives.

	Default value: **LzmaDecoderPath** = %bin_dir/
LockFile = {path to file}	Path to lock file used to prevent sharing of certain files during their processing by **Doctor Web Updater**. Default value: **LockFile** = %var_dir/run/update.lock
CronSummary = {Yes | No}	If you specify Yes, **Dr.Web Updater** will output update report for each session to stdout. This mode can be used to send notifications to administrator by email, if **Updater** is run by the cron daemon. Default value: **CronSummary** = Yes
DrlFile = {path to file}	Path to the file containing list of accessible updating servers. **Dr.Web Updater** randomly selects a server from this list to download updates. This file is signed by **Doctor Web** and should not be modified by the user. It is updated automatically. Default value: **DrlFile** = %var_dir/bases/update.drl
CustomDrlFile = {path to file}	Path to the alternative *.drl file with the list of update servers. This file is signed by **Doctor Web** and should not be modified by the user. It is updated automatically. Default value: **CustomDrlFile** = %var_dir/bases/custom.drl

FallbackToDrl = {Yes | No}	Determines which *.drl file will be used first. If specified value is Yes, **Updater** will attempt to use the file specified in **CustomDrlFile** and then, if it fails, will attempt to use the file specified in **DrlFile**. Default value: **FallbackToDrl** = Yes
DrlDir = {path to directory}	Path to the directory containing drl files with lists of update servers for each plug-in. These files are signed by **Doctor Web** and should not be modified by the user. Default value: **DrlDir** = %var_dir/drl/
Timeout = {numerical value in seconds}	Maximum waiting period for downloading updates. Default value: **Timeout** = 90
Tries = {numerical value}	Number of attempts to be made by **Dr.Web Updater** to establish a connection with update server. Default value: **Tries** = 3
ProxyServer = {proxy server name or IP}	If you use proxy server for Internet access specify its name or IP-address. Default value: **ProxyServer** =
ProxyLogin = {proxy server user login}	Specify login if proxy requires authentication. Default value: **ProxyLogin** =

ProxyPassword = {proxy server user password}	Specify password if proxy requires authentication. Default value: **ProxyPassword** =
LogFileName = {file name}	Log file name. You can specify `syslog` as log file name and logging will be carried out by `syslogd` system service. In this case you must also specify **SyslogFacility** and **SyslogPriority** parameters. As `syslogd` uses several files for logging various events of different importance, these two parameters and `syslogd` configuration file (usually `/etc/syslogd.conf`) determine location where information is logged to. Default value: **LogFileName** = syslog
SyslogFacility = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Log type to be used by `syslogd` system service. Default value: **SyslogFacility** = Daemon
LogLevel = {Debug | Verbose | Info | Warning | Error | Quiet}	Log verbosity level. Default value: **LogLevel** = Info
LotusdPidFile = {path to file}	Path to **Lotus Daemon** PID file. Default value: **LotusdPidFile** = %var_dir/run/drweblotusd.pid
MaildPidFile = {path to file}	Path to `drweb-maild` PID file. Default value:

	MaildPidFile = %var_dir/run/drweb-maild.pid
IcapdPidFile = {path to file}	Path to drweb-icapd PID file. Default value: **IcapdPidFile** = %var_dir/run/drweb_icapd.pid
BlacklistPath = {path to directory}	Path to directory with .dws files. Default value: **BlacklistPath** = %var_dir/dws
AgentConfPath = {path to file}	Path to **Agent** configuration file. Default value: **AgentConfPath** = %var_dir/agent.conf
PathToVadeRetro = {path to file}	Path to libvaderetro.so library. Default value: **PathToVadeRetro** = %var_dir/lib/libvaderetro.so
ExpiredTimeLimit = {number}	Number of days before license expiration during which **Updater** will be attempting to update license key file. Default value: **ExpiredTimeLimit** = 14
ESLockfile = {path to file}	Path to lock file. If the lock file exists, **Dr.Web Updater** will not be automatically initialized by cron daemon. Default value: **ESLockfile** = %var_dir/run/es_updater.lock

Updating Process

Updating is done in following stages:

- **Dr.Web Updater** reads its configuration file.
- **Dr.Web Updater** uses not only parameters located in `[Updater]` section of main configuration file but also **`EnginePath`** (serves both to determine the **Daemon** version and to specify the directory to put `drweb32.dll` file), **`VirusBase`** (serves to specify the directory to put updated databases), **`UpdatePath`** (serves to specify the directory to put all other updated files) and **`PidFile`** (serves to specify path to file with `drwebd` process identifier to use for **Daemon** restarting).
- **Dr.Web Updater** requests the list of available updates from the server, then tries to download corresponding lzma-archives. If no lzma-archives are found, it downloads necessary bases in `*.vdb` and `*.dws` formats. To extract files from lzma-archives decompression utility is used, path to which is specified by **`LzmaDecoderPath`** parameter in the `[Updater]` section of main configuration file.
- Downloaded updates are put to the corresponding directories as described in Updating chapter.

Dr.Web Control Agent

Dr.Web Control Agent module (**Agent**) is a resident module used to manage settings of various modules of **Dr.Web for UNIX mail servers** solution, define antivirus policy depending on available licenses and collect virus statistics. When separate modules of **Dr.Web for UNIX mail servers** are started, or settings are changed, **Agent** sends to these modules all necessary configuration information. **Dr.Web Agent** can interact with other modules by exchanging control signals.

Since all the components of **Dr.Web for UNIX mail servers** solution (except for **Monitor**) receive their settings via `drweb-agent` module, it must be ran before all these modules, but after the `drweb-monitor` module.

When several parameters with the same name are specified in configuration file, **Dr.Web Control Agent** unites them in one string with comma as delimiter. You can also use backslash "\" to define parameter value in several lines. New line after backslash will be added to the previous line when **Agent** reads configuration information.

Example:

```
GlobalRules = select message, append_html
"lookup:file:/maild-files/somehtml.html"
```

This rule can also be specified in the following way:

```
GlobalRules = select message
GlobalRules = append_html "lookup:file:/maild-
files/somehtml.html"
```

Example:

```
to:user@host cont \
modifier/LocalRules=select mime.headers "X-
```

```
Spam-Level" "\\*\\*\\*", \
# 3 or more stars
if found,\
select mime.headers Subject ".*",\
replace "[SPAM] " "^",\
endif
```

Operation Mode

If necessary, **Doctor Web** solutions can be used to connect to corporate or private anti-virus networks managed by **Dr.Web Enterprise Security Suite** (**Dr.Web ESS**). To operate in such central protection mode, you do not need to install additional software or uninstall your **Dr.Web** solution.

To provide you with this option, **Agent** can operate in one of the two following modes:

- Standalone mode when a protected computer is not included in an anti-virus network or managed remotely. In this mode, configuration files and key files reside on local drives, and **Agent** is controlled in full from the protected computer.
- Enterprise mode (or central protection mode), when protection of the computer is managed from a central protection server. In this mode, some features and settings of **Dr.Web for UNIX mail servers** may be modified and blocked for compliance with a general (e.g., company) security policy. Licence key file for operation in this mode is received from the central protection server. Your personal key file on the local computer is not used.

To use central protection mode

1. Contact the anti-virus network administrator of your company for a public key file and parameters of connection to the central protection server.
2. In the **Agent** configuration file (by default, `%etc_dir/agent.conf`), set up the following parameters in the `[EnterpriseMode]` section:

- Set the `PublicKeyFile` parameter to location of a public key file received from anti-virus network administrator (usually, `%var_dir/drwcsd.pub`).This file includes an encryption public key for the access to **Dr.Web ESS**. If you are the anti-virus network administrator, you can locate the file in the corresponding directory on the **Enterprise Server**.
- Set the `ServerHost` parameter to IP-address or host name of the **Enterprise Server**.
- Set the `ServerPort` parameter to the **Enterprise Server** port number (usually, 2193).

3. To connect to central protection server, set the `UserEnterpriseMode` parameter to **Yes**.

   In the central protection mode, some features and settings of **Dr.Web for UNIX mail servers** may be modified and blocked in compliance with the general security policy. A key file for operation in this mode is received from central protection server. Your personal key file on a local computer is not used.

To run **Dr.Web Agent** in central protection mode `drweb-agent-es` package must be installed.

For **Dr.Web for UNIX mail servers** to fully support central protection mode, you must also set **Monitor** to operate in enterprise mode. For more details, see Operation Mode for **Dr.Web UNIX Monitor**.

If MTA drweb-courier, drweb-cgp-receiver settings were updated on the central protection server, you will need to manually send SIGHUP or STOP-START signal to them for the updates to take effect.

To use standalone mode

1. Make sure that all parameters in the `[StandaloneMode]` section of the **Agent** configuration file (by default, `%etc_dir/agent.conf`) are set properly.

2. In the `[EnterpriseMode]` section of the **Agent** configuration file, set the `UseEnterpriseMode` parameter to **No**.

   On switching to this mode, all settings of **Dr.Web for UNIX mail servers** are unlocked and restored to their previous or default values. You can once again access all features of **Dr.Web for UNIX mail servers** solutions and configure them in full.

For correct operation in standalone mode, **Dr.Web for UNIX mail servers** requires a valid personal key file. The key files received from central protection server cannot be used in this mode.

Joint usiage of Dr.Web for UNIX mail servers and Dr.Web Anti-virus for Linux solutions in central protection mode

You can safely use all server UNIX **Dr.Web** solution on the single host in central protection mode. **Dr.Web Anti-virus For Linux** however will be in conflict with server solutions. To run **Dr.Web for UNIX mail servers** or other **Dr.Web** server solutions in the central protection mode on the same host with **Dr.Web Anti-virus For Linux**, you will need to rename amc files for **Dr.Web Anti-virus For Linux** (`drweb-cc.amc`, `drweb-spider.amc`).

Due to implementation details, it is not possible to run **Dr.Web for UNIX mail servers** and **Dr.Web Anti-virus for Linux** in central protection mode on one host simultaneously. To enable central protection mode in **Dr.Web for UNIX mail servers** you should turn **Dr.Web Anti-virus for Linux** to standalone mode and delete or move to another directory files `%etc_dir/agent/drweb-cc.amc` and `%etc_dir/agent/drweb-spider.amc`.

It is recommended to keep this files as a backup in directory other than `%etc_dir/agent` if you are going to run **Dr.Web Anti-virus for Linux** in central protection mode later. In this case, set **Dr.Web for UNIX mail servers** into standalone mode, copy back-ups of `drweb-cc.amc` and `drweb-spider.amc` files to

`%etc_dir/agent/` and follow the instructions given in **Dr.Web Anti-virus for Linux** User Guide.

Command Line Parameters

Agent supports the following command line parameters:

- `-h, --help` – outputs information on command line parameters;
- `-v, --version` – outputs information on current Agent version;
- `-u, --update-all` – updates all modules;
- `-f, --update-failed` – updates modules that failed to update in standard mode;
- `-C, --check-only` – checks module configuration. This command line parameter cannot be used when the **Agent** is running.
- `-p, --newpwd` – changes username and password to access **Dr.Web ESS**;
- `-d, --droppwd` – discards **Enterprise Server** registration (user name and password) to perform workstation registration on **Enterprise Server** all over again;
- `-c <path to file>, --conf <path to file>` – specifies path to alternative configuration file;
- `-s <path to file>, --socket <path to file>` – specifies alternative socket address;
- `-P <path to file>, --pid-file <path to file>` – specifies path to PID-file of the **Agent**;
- `-e <application name>, --export-config <application name>` – exports configuration of application specified in argument to **Enterprise Server**. The argument is the name of the application specified in `Application "application name"` section of corresponding .amc file. This command line parameter cannot be used when the **Agent** is running. Also this parameter cannot be used to export configuration of **Dr.Web Antivirus**

for Linux workstations to **Enterprise Server**.

Configuration File

Setup of **Dr.Web Agent** is performed using its configuration file `%etc_dir/agent.conf`.

[Logging] Section

In [**Logging**] section parameters responsible for logging information about operation of **Dr.Web Agent** are collected:

`[Logging]`

Level = {Quiet | Error | Alert | Info | Debug}	Agent log verbosity level. Default value: **Level** = Info
IPCLevel = {Quiet | Error | Alert | Info | Debug}	Log verbosity level of IPC library. Default value: **IPCLevel** = Error
SyslogFacility = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Syslog facility type generating notifications on **Dr.Web** events if `syslogd` is used for logging of activity of **Dr.Web** and its components (please refer to `syslogd` documentation for further details). Default value: **SyslogFacility** = Daemon

FileName = {path}	Path to log file. You can specify `syslog` as log filename and logging will be carried out by syslogd system utility. As `syslogd` uses several files for logging various events of different importance, these two parameters and `syslogd` configuration file (usually `/etc/syslogd.conf`) determine location where information is logged to. Default value: **FileName** = syslog

[Agent] Section

This section contains general settings of **Dr.Web Agent**:

[Agent]

MetaConfigDir = {path to directory}	Directory name where meta-configuration files of `drweb-agent` reside. These files contain settings defining **Agent** interaction with other modules of **Dr.Web** software complex. Meta-configuration files are supplied by **Dr.Web** developers and do not need to be modified. Default value: **MetaConfigDir** = %etc_dir/agent/
UseMonitor = {Yes | No}	`Yes` value tells `drweb-agent` that **Monitor** is used as a part of **Dr.Web for UNIX mail servers** solution. Default value: **UseMonitor** = Yes
MonitorAddress = {address}	Socket used by **Agent** to interact with **Monitor** (parameter value must be the same as **Address** parameter value from **Monitor** configuration file).

	Default value: **MonitorAddress** = local:%var_dir/ipc/.monitor
MonitorResponseTime = {time in seconds}	Maximum time to get a response from drweb-monitor module. If **Monitor** doesn't respond during this period of time, **Agent** considers drweb-monitor not running and stops trying to establish connection with **Monitor**. Default value: **MonitorResponseTime** = 5
PidFile = {path to file}	Filename where **Agent** PID is written when **Agent** is run. Default value: **PidFile** = %var_dir/run/drweb-agent.pid

[Server] Section

In [**Server**] section parameters defining interaction of **Dr.Web Agent** with other modules of **Dr.Web for UNIX mail servers** solution are collected:

[Server]

Address = {socket address}	Socket used by **Agent** to interact with other modules of software complex. Multiple sockets can be specified, with comma used as a delimiter. Default value: **Address** = local:%var_dir/ipc/.agent, inet:4040@127.0.0.1

Threads = {numerical value}	Number of drweb-agent simultaneous threads. This parameter controls maximum number of simultaneous connections to modules reporting virus statistics to **Agent**. Value of this parameter cannot be changed using SIGHUP signal. Default value: **Threads** = 2
Timeout = {time in seconds}	Maximum time for establishing connection between **Agent** and other **Dr.Web** modules. Default value: **Timeout** = 15

[EnterpriseMode] Section

[**EnterpriseMode**] section contains parameters defining **Agent** operation in **Enterprise** mode:

[EnterpriseMode]

UseEnterpriseMode = {Yes | No}	With Yes value specified drweb-agent works in Enterprise mode, with No value specified it works in Standalone mode. Default value: **UseEnterpriseMode** = No
ComputerName = {text value}	Computer name in anti-virus network. Default value: **ComputerName** =

VirusbaseDir = {path to directory}	Path to directory where virus databases are located. Default value: **VirusbaseDir** = %var_dir/bases
PublicKeyFile = {path to file}	Path to file with public key to access **Enterprise Server**. Default value: **PublicKeyFile** = %bin_dir/drwcsd.pub
ServerHost = {IP address}	**Enterprise Server** server IP address. Default value: **ServerHost** = 127.0.0.1
ServerPort = {port number}	Port number to access **Enterprise Server**. Default value: **ServerPort** = 2193
CryptTraffic = {Yes | Possible | No}	Encryption of traffic between **Enterprise Server** and **Agent**. Default value: **CryptTraffic** = possible
CompressTraffic = {Yes | Possible | No}	Compression of traffic between **Enterprise Server** and **Agent**. Default value: **CompressTraffic** = possible

CacheDir = {path to directory}	Path to directory, where different utility files are stored: configuration files, files with access privileges for applications managed by **Enterprise Server**, files with registration information on **Enterprise Server**, etc. Default value: **CacheDir** = %var_dir/agent

[StandaloneMode] Section

In [**StandaloneMode**] section parameters defining **Agent** operation in **Standalone** mode are collected:

[StandaloneMode]

StatisticsServer = {server address}	Virus statistics server IP address or domain name Default value: **StatisticsServer** = stat.drweb.com:80/update
StatisticsUpdatePeriod = {time in minutes}	URL statistics server IP address or domain name Default value: **StatisticsUpdatePeriod** = 10
StatisticsProxy = {proxy server address}	Virus statistics proxy-server IP address or domain name. Please note that if the value is not explicitly specified, value of http_proxy environment variable is used. **Example:** StatisticsProxy = localhost:3128 Default value:

	StatisticsProxy =
StatisticsProxyAuth = {text value}	Username and password to access proxy-server. **Example:** StatisticsProxyAuth = test:testpwd Default value: **StatisticsProxyAuth** =
UUID = {identifier}	Unique user identifier for virus statistics server http://stat.drweb.com/. Please note that this parameter is mandatory for statistics transfer – so if you want to enable this function, you must specify personal UUID as a value of this parameter (md5 sum of license key file is usually used for this purpose). Default value: **UUID** =
LicenseFile = {path to file}	Location of **Dr.Web** license key files or demo key files. Default value: **LicenseFile** = %bin_dir/drweb32.key
ProtectedEmails = {lookups}	List of protected e-mail addresses. They can be specified explicitly, or path to file containing these addresses can be specified, or lookups can be used. Default value: **ProtectedEmails** = file:%etc_dir/email.ini

[Update] Section

`[Update]` section contains parameters that define how to perform update of **Dr.Web for UNIX mail servers** components via **Enterprise Server**:

`[Update]`

`CacheDir` = {path to directory}	Directory where **Agent** temporarily stores downloaded update files. Default value: **`CacheDir`** = `%var_dir/updates/cache`
`Timeout` = {time in seconds}	Maximum time for **Agent** to process downloaded update files. Default value: **`Timeout`** = `120`
`RootDir` = {path to directory}	Path to root directory. Default value: **`RootDir`** = `/`

Refer to *Administrator Manual* for **Dr.Web ESS** for more information.

Running Dr.Web Unix Control Agent

Please note that if you select "Configure Services" option in the conversation with the post-install script, all services including **Dr.Web Agent** will be started automatically.

When **Agent** starts with default settings, the following actions are performed:

1. **Agent** searches and loads its configuration file. If the configuration file is not found, **Agent** terminates.
2. If the parameters in the `[EnterpriseMode]` section are set correctly and **Dr.Web for UNIX mail servers** solution is operating within anti-virus network, then **Agent** starts in enterprise mode. Otherwise, if parameters in the `[Standalone]` section are set correctly, **Agent** starts in the standalone mode. If the parameters in the `[Standalone]` section are not set, **Agent** terminates.

3. Socket for interaction of **Agent** with other **Dr.Web** modules is created. If a TCP socket is used, then there can be several connections (loading continues if at least one connection is established). If a UNIX socket is used, it can only be created if the user, whose privileges are used to run drweb-agent, has read and write access to its directory. If socket cannot be created, **Agent** terminates.
   Further loading process depends on the selected operation mode.

If **Agent** operates in enterprise mode:

1. **Agent** connects to **Enterprise Server**. If the server is unavailable or authorization process fails during first time connection, **Agent** terminates. If **Agent** had worked previously with this server, but it's temporary unavailable (for example, in the event of connection problems), **Agent** use backup copies of configuration files received from the server earlier.
2. If connection is established, **Agent** receives key files and settings from **Enterprise Server**. After all setting and key files are received, **Agent** is ready for work.

If **Agent** operates in the standalone mode, then meta-configuration files that define **Agent** interaction with other **Dr.Web** modules are loaded. Location of meta-configuration files is set in the `MetaConfigDir` parameter in the `[Agent]` section of the **Agent** configuration file. When meta-configuration files are successfully loaded, **Agent** is ready for work.

Interaction with other Software Modules

Interaction with other software modules is performed by **Agent** metaconfiguration files (amc-files). These files describe configuration parameters, which values will be received by respective **Dr.Web** modules from **Agent**.

Description of each module can be found in `Application` section named after this module. At the end of the section **`EndApplication`** must be specified.

The following parameters must be present in the description of the module:

- **`id`**: identifier of the module in **Dr.Web ESS**.
- **`ConfFile`**: path to the configuration file of the module.
- **`Components`**: description of the component. At the end of this section `EndComponents` must be specified. For each component its name, list of the sections in the configuration file and the parameters in these sections necessary for proper operation of the component are specified. The list of sections and parameters is comma separated. To describe individual parameters properly you must specify full path to them (e.g. `Quarantine/Path`). In description of sections only their names must be specified (e.g. `General`). Back slash (\) in descriptions of sections and parameters is used to denote line breaks.

Example of amc-file from Dr.Web MailD package for Linux:

```
Application "MAILD"
  id                40
  ConfFile          "/etc/drweb/maild_smtp.conf"
  Components
    lookup_ldap       LDAP
    lookup_regex      REGEX
    drweb-maild       General, Logging, MailBase,
```

```
Stat, Maild, Filters,\
                      Quarantine, /_Rules=Rule*:
Rules, /Reports/Send,\
                           /Reports/SendTimes, /
Reports/Names, /Reports/MaxPoolSize,\
                                        /Reports/
MaxStoreInDbPeriod,                     /Reports/
CheckForRemovePeriod,\
                         /Notifier/FilterMail, /
Notifier/NotifyLangs,\
                     /Notifier/LngBaseDir
    drweb-notifier  General, Logging, Notifier,
/Sender/Method, /_Rules,\
                               Reports, /Filters/
BeforeQueueFilters,\
                     /Filters/AfterQueueFilters,
/Quarantine/AccessByEmail,\
                     /Quarantine/StoredTime
    drweb-sender    General, Logging, Sender
     drweb-receiver  General, Logging, /Maild/
ProtectedNetworks,\
                      /Maild/ProtectedDomains, /
Maild/IncludeSubdomains,\
                     SASL, Receiver
  EndComponents
EndApplication
```

Integration with Dr.Web Enterprise Security Suite

There are two possible situations which require integration of **Dr.Web for UNIX mail servers** solution with **Dr.Web Enterprise Security Suite** from system administrator:

- Setup and initial configuration of UNIX mail server in existing **Dr.Web ESS** environment;
- Embedding of successfully functioning UNIX mail server with already installed and configured **Dr.Web for UNIX mail servers** solution in **Dr.Web ESS** environment.

To make **Dr.Web for UNIX mail servers** solution work in **Dr.Web ESS** environment, set up **Dr.Web Agent** and **Dr.Web Monitor** components for operation in `Enterprise` mode, and register the whole software complex on **ESS** server. According to connection policy for new working stations (for more details refer to administrator manual for **Dr.Web Enterprise Security Suite)**, mail server can be connected to **Dr.Web ESS** in two different ways:

- when new account is created by central protection server automatically;
- when corresponding account is created by administrator manually.

Setup of Components

To start up in `Enterprise` mode after installation it is necessary to specify the changes in local configuration files of **Agent** and **Monitor**.

For Agent

In `[EnterpriseMode]` section of **Agent** configuration file `%etc_dir/agent.conf` set the following parameter values:

- **`UseEnterpriseMode`** = `Yes`;

- **PublicKeyFile** = %var_dir/drwcsd.pub (encryption public key for the access to central protection server. Take this file from the corresponding directory of **Enterprise Server** and move it to the specified path);
- **ServerHost** = IP-address or host name of **Enterprise Server**;
- **ServerPort** = **Enterprise Server** port (2193 by default).

For Monitor

In [Monitor] section of the **Monitor** configuration file %etc_dir/monitor.conf set the following parameter values:

- **UseEnterpriseMode** = Yes.

Automatic Creation of New Account by ES-server

When new account is created automatically:

1. When **Agent** is first started in Enterprise mode, it sends a request for the account details (station ID and password) to **ESS** server;
2. If **Enterprise Server** is set to Approve access manually mode (used by default, for more details refer to administrator manual for **Dr.Web Enterprise Security Suite**), system administrator must confirm registration of new station via web interface during one minute from an emergence of corresponding request;
3. After first start **Agent** records hash of station ID and password into file named pwd. This file is created in the directory that specified in **CacheDir** parameter of the [EnterpriseMode] section (default value is %var_dir/agent/);
4. Data from this file is used every time **Dr.Web for UNIX mail servers** solution connects to ES-server;
5. If you delete password file, repeated registration request will be made to **ESS** server after next start of the **Agent**.

Manual Creation of New Account by Administrator

When new account is created manually:

1. Create new account on **Enterprise Server**: station ID is generated automatically and password must be specified manually (for more details refer to administrator manual for **Dr.Web Enterprise Security Suite**).
2. Start **Agent** using command line parameter **--newpwd** (or **-p**) and type in the station ID and password. **Agent** records hash of station ID and password into file named `pwd`. This file is created in the directory that specified in **CacheDir** parameter of the [EnterpriseMode] section (default value is `%var_dir/agent/`).
3. Data from this file is used every time **Dr.Web for UNIX mail servers** solution connects to ES-server.
4. If you delete password file, the registration must be performed once again.

Configuring Components via Dr.Web Enterprise Security Suite Web Interface

Configuration of **Dr.Web for UNIX mail servers** and **Dr.Web Daemon** (antivirus plug-in, included in standard installation package) can be performed via **Dr.Web Control Center**.

In **Dr.Web Enterprise Security Suite** standard installation package the basic configuration files for **Dr.Web for UNIX mail servers** and **Dr.Web Daemon** components for Linux, FreeBSD and Solaris are included. When you configure certain components via web interface, values of corresponding parameters are changed in these configuration files on **Enterprise Server**. After that every time the components start, **Agent** requests and receives configuration from **Enterprise Server**.

Export of Existing Configuration to ES Server

Automatic export of configuration settings from local computer to **Enterprise Server** is possible via **Agent** operating in `Enterprise` mode. To export configuration use command line parameter **`--export-config`** (or **`-e`**).

You must specify the name of the component (DAEMON, `MAILD`).

Example:

```
# %bin_dir/drweb-agent --export-config MAILD
```

Starting up the System

To start up the system:

1. In **Dr.Web Control Center** interface open the page with **Monitor** settings and check **Daemon** and **Maild** boxes to enable configuration of the corresponding components;
2. Start **Monitor** on local computer:
   - `# /etc/init.d/drweb-monitor start` – for Linux and Solaris;
   - `# /usr/local/etc/rc.d/00.drweb-monitor.sh start` – for FreeBSD.

Collection of Virus Statistics

Agent receives statistics on computer threats from controlled modules and sends it to the official **Doctor Web** website devoted to statistics: http://stat.drweb.com/ (if Internet connection is available) or to **Dr.Web ESS** (if **Agent** is operating in enterprise mode). **Agent** needs the *unique user identifier* (UUID) to connect to this website. By default, license key file MD5 sum is used as a UUID. Also you can get a personal UUID from **Doctor Web**

Technical Support. In this case, your UUID must be specified explicitly in the **Agent** configuration file.

Statistics are gathered only for **Dr.Web** modules that receive settings from **Agent**. Information on how to set up interaction with **Agent** can be found in chapters describing these modules.

On the statistics website, you can find the aggregate statistics for computer threats for a given server or for all servers supported by **Dr.Web anti-virus for UNIX systems** or by **Dr.Web for UNIX mail servers** solution with anti-virus plug-in. **Agent** can simultaneously process statistics for computer threats from several different **Dr.Web** products which are able to interact with **Agent**.

Statistics processing results contain information on the most frequently detected threats (number of detections and overall percentage) for a given period.

Statistics is available in both HTML and XML formal. The second option is especially convenient when this data is going to be published on another web site, since it can be transformed according to web site's concept and design.

To get aggregate statistics on computer threats for all supported servers, visit http://stat.drweb.com/. You can view a list of detected threats for all supported servers (in descending order) with number and overall percentage of detections.

This web page may render differently depending on used browser.

The following illustration shows threats statistics.

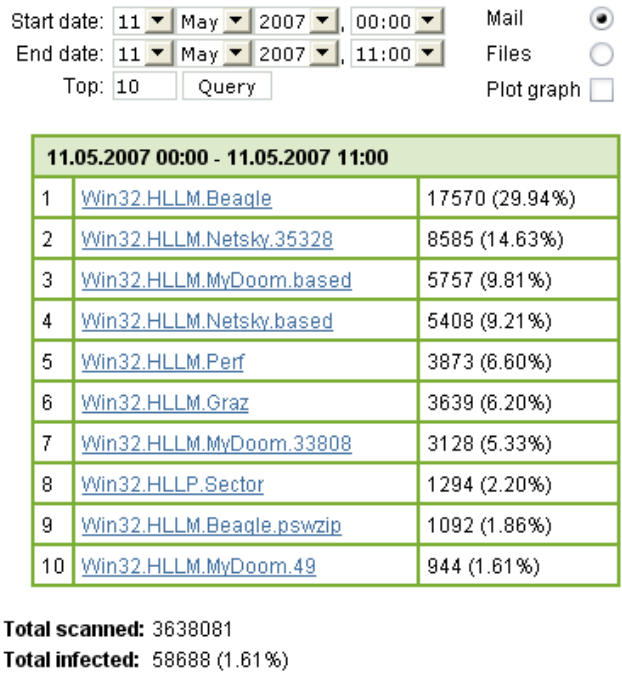


Figure 15. Computer threats statistics

To alter search parameters and to repeat search

1. Select either **Mail** or **Files** flags to get the statistics about the computer threats detected in the e-mails or in files.
2. In the drop-down lists for **Start date** and **End date**, select choose **start/end date** and **time** for the period of interest.
3. In the **Top** field, enter the required number of rows in the statistics table (most frequently the detected threats will be shown).
4. Select **Plot graph** if you want to view statistics in graphical form.
5. Click **Query**. The file with aggregate statistics in the XML form can be found at http://info.drweb.com/export/xml/top .

Example:

```
<drwebvirustop        period="24"        top="5"
vdbaseurl="http://info.drweb.com/
virus_description/"      updatedutc="2009-06-09
09:32:02">
    <item>
        <vname>Win32.HLLM.Netsky</vname>
        <dwvlid>62083</dwvlid>
        <place>1</place>
        <percents>34.201062139103</percents>
    </item>
    <item>
        <vname>Win32.HLLM.MyDoom</vname>
        <dwvlid>9353</dwvlid>
        <place>2</place>
        <percents>25.1303270912579</percents>
    </item>
    <item>
        <vname>Win32.HLLM.Beagle</vname>
        <dwvlid>26997</dwvlid>
        <place>3</place>
        <percents>13.4593034783378</percents>
    </item>
    <item>
        <vname>Trojan.Botnetlog.9</vname>
        <dwvlid>438003</dwvlid>
        <place>4</place>
        <percents>7.86446592583328</percents>
    </item>
    <item>
```

```
        <vname>Trojan.DownLoad.36339</vname>
        <dwvlid>435637</dwvlid>
        <place>5</place>
        <percents>7.31494163115527</percents>
    </item>
</drwebvirustop>
```

In this file, the following XML attributes are used:

- period – duration (in hours) of the statistics collection process;
- top – number of the most frequently detected threats in the statistics table (number of rows);
- updatedutc – last statistics' update time;
- vname – threat name;
- place – virus place in the statistics;
- percents – percentage of the detections.

The value of the period parameter and the sample size cannot be changed by user.

To get personalized threat statistics

Visit one of the following Web pages:

- For the statistics in HTML, go to http://stat.drweb.com/view/<UUID>. Personalized threat statistics page is similar to the aggregate threat statistics page.
- For the file with the personalized threat statistics in XML form, go to http://stat.drweb.com/xml/<UUID>.

The <UUID> in both cases stands for the MD5 sum of your license key file (unless you have a personal UUID received from **Doctor Web Technical Support**).

Example:

```
<drwebvirustop period="24" top="2" user="<UID>"
lastdata="2005-04-12 07:00:00+04">
```

```
    <item>
        <caught>69</caught>
        <percents>24.1258741258741</percents>
        <place>1</place>
        <vname>Win32.HLLM.Netsky.35328</vname>
    </item>
    <item>
        <caught>57</caught>
        <percents>19.9300699300699</percents>
        <place>2</place>
        <vname>Win32.HLLM.MyDoom.54464</vname>
    </item>
</drwebvirustop>
```

In this file the following XML attributes are used:

- period – duration (in hours) of the statistics collection process;
- top – number of the most frequently detected threats in the statistics table (number of rows);
- user – user identifier;
- lastdata – last time user sent the data to the server;
- vname – threat name;
- place – threat place in the statistics;
- caught – a number of the detections of the certain threat;
- percents – percentage of the detections.

The value of the period parameter and the sample size cannot be changed by user.

Dr.Web Monitor

Dr.Web Unix Monitor module (**Monitor** hereafter) is a memory resident module.

It is used to increase fault-tolerance of the whole **Dr.Web for UNIX mail servers** software complex. It ensures correct startup and termination of operation of software modules and their components as well as restart of any component due to its abnormal operation. **Monitor** starts all modules and loads, if necessary, some extra components of these modules. If **Monitor** fails to start a module, it repeats an attempt later. Number of attempts and a time period between them are defined by **Monitor** settings.

After all modules are loaded, **Monitor** permanently controls their operation. If any module or one of its components operates abnormally, **Monitor** restarts the stalled application. Maximum number of attempts to restart a component and a period of time between them are defined by **Monitor** settings. If any of the modules starts to operate abnormally, **Monitor** notifies the system administrator. **Dr.Web Monitor** can interact with **Dr.Web Control Agent** by exchanging control signals.

Operation Mode

If necessary, **Doctor Web** solutions can be used to connect to corporate or private anti-virus networks managed by **Dr.Web Enterprise Security Suite**. To operate in such central protection mode, you do not need to install additional software or uninstall your **Dr.Web** solution.

To provide you with this option, **Monitor** can operate in one of the two following modes:

- Standalone mode when a protected computer is not included in an anti-virus network or managed remotely. In this mode, configuration files and key files reside on local drives, **Monitor** is controlled in full from the protected computer, and modules start as set in **Monitor** configuration file.
- Enterprise mode (or central protection mode) when protection of local computer is managed from a central protection server. In this mode, some features and settings of **Dr.Web for UNIX mail servers** may be modified and blocked for compliance with a general (e.g., company) security policy. A key file for operation in this mode is received from the central protection server. Your personal key file on the local computer is not used.

To use central protection mode

1. Contact anti-virus network administrator of your company for a public key file and parameters of connection to the central protection server.
2. In **Monitor** configuration file (by default, `%etc_dir/monitor.conf`), set the `UseEnterpriseMode` parameter to **Yes**.

   In the central protection mode, some features and settings of **Dr.Web for UNIX mail servers** may be modified and blocked for compliance with the general security policy. A key file for operation in this mode is received from central protection server. Your personal key file on a local computer is not used.

For **Dr.Web for UNIX mail servers** to fully support central protection mode, you must also set **Agent** to operate in enterprise mode. For more details, see Operation Mode for **Dr.Web UNIX Agent**.

To use standalone mode

1. Make sure that all necessary modules that you want **Monitor** to start are listed in the `RunAppList` parameter under the `[Monitor]` section of **Monitor** configuration file (by default, `%etc_dir/monitor.conf`).
2. In the `[Monitor]` section of **Monitor** configuration file, set the `UseEnterpriseMode` parameter to `No`.

   On switching to this mode, all settings of **Dr.Web for UNIX mail servers** are unlocked and restored to their previous or default values. You can once again access all features of **Dr.Web for UNIX mail servers** solution and configure them in full.

For correct operation in standalone mode, **Dr.Web for UNIX mail servers** requires a valid personal key file. The key files received from central protection server cannot be used in this mode.

Command Line Parameters

Monitor supports the following command line parameters:

- `-h, --help` - outputs information on command line parameters;
- `-v, --version` - outputs information on current **Monitor** version;
- `-u, --update` - updating mode;
- `-C, --check-only` - checks module configuration;
- `-A, --check-all` - checks configuration of all modules;
- `-c <path to file>, --conf <path to file>` - specifies path to alternative configuration file.
- `-r, --run component1[,component2]` - starts components in the specified order.

Example:

```
-r AGENT, DAEMON
```

Configuration File

Setup of **Dr.Web Monitor** is performed using its configuration file `%etc_dir/monitor.conf`.

[Logging] Section

In [**Logging**] section parameters responsible for logging information about operation of **Dr.Web Monitor** are collected:

`[Logging]`

Level = {Quiet | Error | Alert | Info | Debug}	**Monitor** log verbosity level. Default value: **Level** = Info
IPCLevel = {Quiet | Error | Alert | Info | Debug}	Log verbosity level of IPC library. Default value: **IPCLevel** = Error
SyslogFacility = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Syslog facility type generating notifications on **Dr.Web** events if `syslogd` is used for logging of activity of **Dr.Web** and its components (please refer to `syslogd` documentation for further details). Default value: **SyslogFacility** = Daemon

FileName = {syslog | path to file}	Path to log file. You can specify `syslog` as log filename and logging will be carried out by `syslogd` system utility. Default value: **FileName** = syslog

[Monitor] Section

[Monitor] section contains all **Monitor** main settings:

[Monitor]

RunForeground = {Yes | No}	**Yes** value forces **Monitor** not to use `daemon` mode. It helps to control its state using special utilities (i.e., `daemontools`). Default value: **RunForeground** = No
User = {user name}	User name used to run **Monitor** with certain user privileges. Please note that when software complex operates in SMTP-proxy mode value of this parameter must be set to root. Default value: **User** = drweb
Group = {group name}	User group name used to run **Monitor** with certain user privileges. Please note that when software complex operates in SMTP-proxy mode value of this parameter must be set to root. Default value: **Group** = drweb

PidFileDir = {path to directory}	Path to directory where Monitor PID-file is stored when **Monitor** is started. Default value: **PidFileDir** = %var_dir/run/
ChDir = {path to directory}	Change of working directory when **Monitor** is started. If this parameter is set up, **Monitor** changes directory to the one specified in this parameter value. Otherwise working directory is not changed. Default value: **ChDir** = /
MetaConfigDir = {path to directory}	Path to directory where meta-configuration files reside. These files contain settings defining **Monitor** interaction with other modules of **Dr.Web** software complex. Meta-configuration files are supplied by **Dr.Web** developers and don't need editing. Default value: **MetaConfigDir** = %etc_dir/monitor/
Address = {address}	Socket used by **Monitor** to receive control signals. Default value: **Address** = local:%var_dir/ipc/.monitor
Timeout = {time in seconds}	Maximum time to establish connection between **Monitor** and other components of **Dr.Web** software complex. Default value: **Timeout** = 5

TmpFileFmt = {text}	Monitor temporary file names template. Template layout: path_to_file.XXXXXX, where X - random letters and digits used in temporary file names. Default value: **TmpFileFmt** = %var_dir/msgs/tmp/monitor.XXXXXX
RunAppList = {text}	List of modules started by **Monitor**, with comma used as a delimiter. Please note that this parameter will not be modified after uninstalling **Dr.Web** modules. You must manually remove uninstalled modules from this parameter. Otherwise **Monitor** will no be able to run and to execute other **Dr.Web** modules. Default value: **RunAppList** = AGENT
UseEnterpriseMode = {Yes | No}	Yes value makes **Monitor** receive the list of modules to be started from drweb-agent, not from **RunAppList** parameter value. Default value: **UseEnterpriseMode** = No
RecoveryTimeList= {time in seconds}	Time intervals between attempts to restart not responding modules. This parameter can have multiple values, delimited by commas. First attempt to restart a module is made after a period of time specified in first parameter value, second attempt – using second parameter value, and so on. Default value: **RecoveryTimeList** = 0,30,60

InjectCmd = {string}	Command to send reports. Please note that if you want to send reports to some other address (not only to root@localhost), you should specify it in the command. Default value: **InjectCmd** = "/usr/sbin/sendmail -t"
AgentAddress = {socket address}	Socket used by **Monitor** to interact with **Agent** (parameter value must be the same as **Address** parameter value from **Dr.Web Agent** configuration file). Default value: **AgentAddress** = local:%var_dir/ipc/.agent
AgentResponseTime = {time in seconds}	Maximum time to get a response from drweb-agent module. If **Agent** doesn't respond during this period of time, **Monitor** considers drweb-agent not working and tries to restart it. Default value: **AgentResponseTime** = 5

Running Dr.Web Unix Monitor

Please note that if you select "Configure Services" option in the conversation with the post-install script, all services including **Dr.Web Agent** will be started automatically.

When **Monitor** is ran with default settings the following actions are performed:

1. **Monitor** searches for and loads its configuration file. If configuration file is not found, loading stops;
2. Then it enters daemon mode, so all information about loading

problems cannot be output to console anymore and is written to log file;

3. Socket for **Monitor** interaction with other software modules is created. If TCP socket is used, there can be several connections (loading continues if at least one connection is established). If UNIX socket is used, it can be created only if the user whose privileges are used to run `drweb-monitor` has read and write access to the certain directory. If socket cannot be created, loading stops;
4. PID-file with Monitor PID information is created. If PID-file cannot be created, loading stops;
5. `drweb-monitor` module starts other software modules. If some module cannot load, **Monitor** tries to restart it. If all **Monitor** attempts to start the module are unsuccessful, **Monitor** unloads all previously loaded modules and terminates. All problems with the startup of modules **Monitor** reports using one of available methods (output to log file, notification via e-mail, startup of specific program). Notification methods used for various modules are set in **Monitor** meta-configuration file.

For successful startup of Dr.Web Monitor in automatic mode:

- value of ENABLE variable must be changed to `1` in the `%etc_dir/drweb-monitor.enable` file (for Linux and Solaris);
- or a `drweb_monitor_enable="YES"` line must be added to the `/etc/rc.conf` file (for FreeBSD).

Interaction with other Software Modules

Interaction with other software modules is performed via *Monitor configuration files* (mmc-files). These files are included in packages of those products which can interact with **Dr.Web Monitor**. In these files components' contents, location of binaries, their starting sequence and parameters of startup are described.

Description of each component can be found in `Application` section named after this component. At the end of the section `EndApplication` must be specified.

The following parameters must be present in the description of the component:

- **`FullName`** - full name of the component.
- **`Path`** - path to binary files.
- **`Depends`** - names of components which must be started before the described component is started. For example, `AGENT` component must be started before **Dr.Web Daemon** component, therefore in mmc-file for **Dr.Web Daemon** `Depends` parameter has `AGENT` value. If there are no dependencies, this parameter can be skipped.
- **`Components`** - list of binary files of modules started when component itself is started. Modules are started in order they are specified in this parameter. For each module command line parameters (may be enclosed in quotation marks), timeouts for startup and close down, notification type and startup privileges. *Notification type* - defines where to send notifications about component's failure. When `MAIL` value is specified, notifications are sent by mail, when `LOG` value is specified, information is output to log only. *Startup privileges* - define with privileges of which group and user, the component will be started.

Example of mmc-file for Dr.Web Daemon for Linux:

```
Application "DAEMON"
    FullName    "Dr.Web (R) Daemon"
    Path        "/opt/drweb/"
    Depends     "AGENT"
    Components
        # name   args   maxStartTime
maxStopTime     NotifyType  User:Group
        drwebd  "-a=local:/var/drweb/ipc/.agent
--foreground=yes"  30 10 MAIL drweb:drweb
```

```
    EndComponents
EndApplication
Application "MAILD"
 FullName    "Dr.Web (R) MailD"
 Path       "/opt/drweb/"
 Depends     "AGENT"
 Components
  # name       args    MaxStartTime MaxStopTime
NotifyType User:Group
  drweb-notifier local:/var/drweb/ipc/.agent 30
30 MAIL drweb:drweb
  drweb-sender   local:/var/drweb/ipc/.agent 15
30 LOG  drweb:drweb
  drweb-maild   local:/var/drweb/ipc/.agent 120
30 MAIL drweb:drweb
  drweb-receiver local:/var/drweb/ipc/.agent 15
30 MAIL root:drweb
 EndComponents
EndApplication
```

Dr.Web for UNIX Mail Servers

Dr.Web for UNIX mail servers solution is a group of interacting software modules. **Dr.Web for UNIX mail servers** modules can operate as a proxy-server for SMTP and LMTP protocols, or as filters for a wide range of supported mail transfer systems (Sendmail, Postfix, Exim, CommuniGate Pro, Courier, Zmailer and Qmail) with their own settings, statistics, reports and quarantine.

All components of **Dr.Web for UNIX mail servers** solution process mail messages cooperatively according to the following algorithm:

1. **Receiver** component receives incoming mail messages directly from mail systems or via SMTP/LMTP protocols and redirects them to `drweb-maild` component responsible for message check.

   Depending on the installed mail system and used protocols, functions of this component can be performed by different modules (`drweb-receiver`, `drweb-milter`, `drweb-cgp-receiver`, etc.). It should be mentioned that synchronous operation of several **Receiver** modules is fully supported, which allows to receive and process mail from several sources simultaneously. Some **Receiver** modules are able to modify and send collected messages upon the receipt of mail check results from `drweb-maild` component. For example, `drweb-milter` module has this feature, which allows to return check results to the Sendmail system before the SMTP session is closed.

2. `Drweb-maild` module is the main component of mail processing system. It performs MIME-processing of messages and forwards messages to plug-ins. It is also responsible for storing messages in the database. Check results are sent either to the **Receiver** component (if timeout for awaiting these results has not yet expired) or to the **Sender** component.
3. Messages are processed by plug-ins, connected to the `drweb-maild` module. User can initialize and disconnect plug-ins at any time without interrupting operation of

drweb-maild module. Messages are processed by plug-ins in order set by user. All messages generated by plug-ins are sent through **Sender** component. Some plug-ins require database support for proper operation.

4. **Sender** component is responsible for sending messages either directly to different mail systems or via SMTP/LMTP protocols. Depending on the installed mail systems and used protocols, functions of **Sender** component is performed by different modules (drweb-sender, drweb-cgp-sender, etc). **Sender** component can receive requests from drweb-maild, drweb-notifier and drweb-monitor to send messages.
5. drweb-notifier module is responsible for generation and dispatch of reports during the operation of the software system. Plug-ins as well as other system components can request for reports to be generated (for example, if virus is detected). Drweb-maild module can send a request for creation of general report with statistics on operation of all connected plug-ins. **Sender** component can send a request for DSN report regarding a failure in the dispatch of a message. Reports can be sent to senders, to receivers and to the system administrator as well.
6. drweb-agent module allows mail processing system to work in Standalone mode as well as in Enterprise mode in cooperation with **Dr.Web Enterprise Security Suite**. In Enterprise mode all system components (except for drweb-monitor) receive the configuration information via drweb-agent module. Therefore the latter must be started before any other components are loaded. drweb-agent also checks validity of licenses and collects statistics on operation of system components: names of blocked objects, total size of processed files, etc.
7. drweb-monitor module starts other system modules and terminates their operation in the order set by the user. It also provides control over their operation. If any system module begins to operate abnormally, drweb-monitor restarts it and notifies the system administrator (if this action is specified in settings).

The following plug-ins are available:

- drweb - this anti-virus plug-in performs anti-virus check of all incoming messages using **Dr.Web Daemon** module. Messages are transferred to **Daemon** in segments, so MIME-processing support is not required.
- headersfilter - this plug-in filters messages by their headers. Regular expressions (Perl syntax) can be used in filtration rules.
- vaderetro - this anti-spam plug-in uses its own **VadeRetro** library to filter out spam messages. This library is dynamically updated. It allows to maintain high quality of spam filtration. **VadeRetro** is notable for high speed of the message processing.
- modifier - this plug-in allows to modify the whole messages or only part of it according to the content of the message or its envelope. With this plug-in you will be able to add alphanumerical signature to checked messages or to delete images from messages marked as spam.

Command Line Parameters

Like every UNIX program software modules of **Dr.Web for UNIX mail servers** solution support command line parameters.

Command line for all these modules looks as follows:

```
module_name [parameters] agent_socket
```

where:

- module_name is the name of **Dr.Web for UNIX mail servers** module (for example, drweb-maild or drweb-notifier);
- parameters is for additional command line parameters;
- agent_socket is the socket through which **Dr.Web for UNIX mail servers** modules receive from **Dr.Web Agent** their configuration information.

To get complete list of parameters for any module start it with -h or --help parameters. In the current version of **Dr.Web for UNIX mail servers** modules support the following command line

parameters:

- `-v, --version` – current module version;
- `-l <level>, --level <level>` – log verbosity level for **Dr.Web for UNIX mail servers** module. Default value is `info`;
- `-t <value in seconds>, --timeout <value in seconds>` – maximum period of time to wait for a receipt of configuration information;
- `--component arg` – name of the component to be used in requests to **Agent** for providing configuration information;
- `--log-name arg` – name of the component to be used in logging;
- `--unique-id arg` – unique ID for **Receiver** and **Sender** components. This parameter enables operation of several **Receivers** or **Senders** at once. Each **Receiver** and **Sender** component receives a unique ID at startup. For the dispatch of the message a pair of **Receiver**/**Sender** components with the same ID is selected (if no suitable Senders are found, default **Sender** is used). A list of available **Senders** is reloaded using `SIGHUP`;
- `--check-only` – configuration check is performed when the module is started with this parameter. For proper functioning **Dr.Web Agent** must be working.

  If check turns out to be successful, the following message is output to console:

  ```
  Options OK
  ```

  If some errors occur during configuration check, the following message is output to console:

  ```
  Options ERROR
  ```

- `--check-all` – started with this parameter **Dr.Web Monitor** checks not only its own configuration, but also configuration of all other modules.

Example:

```
$ drweb-maild -t 30 local:%var_dir/ipc/.
```

agent

It starts **Dr.Web MailD** with 30 second timeout for receiving configuration data and with the **Agent**'s socket in `local:%var_dir/ipc/.agent`.

Signals

All program modules constantly residing in the memory support processing of the following signals:

- `SIGHUP` - forces modules to reread their configuration files. When **Dr.Web Monitor** receives this signal, it makes all the running components re-read their configuration.
- `SIGINT` and `SIGTERM` - after receipt of any of these signals, modules finish their operation.

Some modules can process additional signals:

- **Receiver** and **Sender** components check the internal directory structure for lost messages upon the receipt of `SIGALRM` signal. If such messages are found, an attempt to send them is made.
- **Sender** component makes an attempt to send all messages from the internal queue upon the receipt of `SIGUSR2` signal.
- **Receiver** module saves the address check statistics to a separate file upon the receipt of `SIGUSR1` signal.
- All the components save files with statistics on operation of dynamic thread pools and persistent connections to the directory specified in **`BaseDir`** parameter from `[General]` section of **Dr.Web MailD** configuration file upon the receipt of `SIGUSR1` signal.

Internal Statistics

Statistics on operation of thread pools and persistent connections linked to these pools is collected only when it is enabled explicitly in

thread pool settings (**InPoolOptions** and **OutPoolOptions** parameters of **Dr.Web MailD** configuration file) by specifying an additional parameter **stat** = yes.

Example:

```
InPoolOptions = auto, stat = yes
```

Names of files created upon a receipt of SIGUSR1 signal look like the following:

- name_[callback_](cli|srv)[.unique-id].txt – for statistics on connections;
- name_[callback_](thr[N])[.unique-id].txt – for statistics on pools.

where:

- name – name of the component without "drweb-" part.
- callback – callback of **Receiver** interface.
- cli – for client connections.
- srv – for server connections.
- unique-id – for modules started with unique identifier.
- thr – for thread pool.

If such file already exists, then statistics will be added to the end of this file.

Each entry begins with the following:

```
==================================
start:  Tue Oct  9 14:44:15 2008
curr:   Tue Oct  9 14:44:29 2008
period: 0d 0h 0m 14s
```

where date of the beginning of collecting statistics, current date and a period of time required for output are displayed.

For `srv` a number of created and closed connections and a maximum amount of elements in different queues are displayed.

```
closed: 0 (0 num/sec)
total created = 0 (0 num/sec)
max rea = 0 est = 0 don = 0 act = 0
```

For `cli` a number of connection created by request and closed on timeout, their average amount and current number are displayed.

```
created on request = 0 (0 num/sec)
closed by timeout = 0 (0 num/sec)
avg number = 0
current = 2
```

For `thr` output looks as follows:

```
min = 2 max = 2147483647 type = 0 freetime = 120
busy max = 0 avg = 0
requests for new threads = 0 (0 num/sec)
creating fails = 0
max processing time = 0 ms; avg = 0 ms
curr = 2 busy = 0
```

where:

- on the first line - maximum and minimum number of threads in one pool, type of the pool, maximum time (in seconds) for an additional thread to close upon inactivity are displayed;
- on the second line - maximum and average number of busy threads are displayed;
- on the third line - number and frequency of requests for creation of additional threads are displayed;
- on the fourth line - number of failed attempts to create additional threads is displayed;
- on the fifth line - maximum and average time of processing of these requests are displayed;

- on the sixth line - current number of threads in a pool and a number of busy threads are displayed.

Adjustment and Startup

Dr.Web for UNIX mail servers solution can be used with default settings, but if you want to ensure optimal performance, you may adjust it according to your specific requirements and situations.

All settings of **Dr.Web for UNIX mail servers** solution are stored in three configuration files which are located in `%etc_dir` directory. In `maild_MTA.conf` file general settings of **Dr.Web MailD** can be found. In `agent.conf` and `monitor.conf` files settings of **Dr.Web Agent** and **Dr.Web Monitor** are located.

If all files of **Dr.Web for UNIX mail servers** solution reside in their default directories, then its general setup can be performed via `configure.pl` script. This script can be found in `%bin_dir/maild/scripts/` directory by default. After startup it will prompt for values of some essential parameters and write them down to `maild_MTA.conf` configuration file. Other parameters necessary for interaction with mail transfer system must be set up manually, by editing **Dr.Web MailD** configuration file.

Configuration File

Dr.Web MailD settings are stored in configuration file `%etc_dir/maild_MTA.conf`.
Description of configuration file structure and parameter types can be found in the Configuration_Files chapter. Parameters are described in the order they are presented in configuration file.

[General] Section

In `[General]` section general settings of **Dr.Web MailD** are stored:

BaseDir = {path to directory}	Main operational directory. It contains sockets, databases and other files. In the current version of **Dr.Web MailD** value of this parameter can not be changed during system reload on SUGHUP signal. Default value: **BaseDir** = %var_dir/
MaxTimeoutForThreadActivity = {time}	Maximum time for a thread to close. This parameter is used at restart or shutdown of a system. Total amount of time for a system to sign off can be calculated in the following way: number of pools and the value of **MaxTimeoutForThreadActivity** parameter are multiplied, and then certain time constant is added to the result. Default value: **MaxTimeoutForThreadActivity** = 30s
IpcTimeout = {time}	Timeout for establishing connection between components. Default value: **IpcTimeout** = 40s
Hostname = {string}	Name of the host used by **Dr.Web for UNIX mail servers**. Default value: **Hostname** =

[Logging] Section

In [Logging] section parameters responsible for logging are stored. Logging is performed for all main modules of **Dr.Web for UNIX mail servers** solution.

Level = {Quiet | Error | Alert | Info | Debug}	Value of this parameter defines log verbosity level. Default value: **Level** = Info
IPCLevel = {Quiet | Error | Alert | Info | Debug}	Log verbosity level of IPC library. Default value: **IPCLevel** = Alert
SyslogFacility = {Daemon | Local0 .. Local7 | Kern | User | Mail}	Log type to be used by syslogd system service. Default value: **SyslogFacility** = Mail
FileName = {syslog | path to file}	Log file name. You can specify syslog as log file name and logging will be carried out by syslogd system service. In this case you must also specify **SyslogFacility**, **IPCLevel** and **Level** parameters. As syslogd uses several files for logging various events of different importance, these two parameters and syslogd configuration file (usually /etc/syslogd.conf) determine location where information is logged to. Default value: **FileName** = syslog

Data can be output to file or to syslog if Daemon value is specified for **SyslogFacility** parameter. When syslogd is used, then every string looks as follows:

```
'['tid']' name[.sub] level [sid(/mta-id)] text
```

where:

- tid – identifier of thread, responsible for an output of a string;

- `name` – name of component which performs an output (e. g. plug-in name);
- `sub` – name of component's service to perform output.

The most important services are:

- `ipc` – inter-process communication service;
- `thrN` – thread pool support service;
- `report` – reports support service;
- `ldap`,`odbc`,`oracle`,`sqlite`,`mysql`,`postgres`, `cdb`,`berkeley`,`firebird` – lookups support service;
- `control` – interactive management service;
- `parser` – templates parser service;
- `MRS` – receiving messages via SMTP/LMTP service;
- `smtp` – sending messages via SMTP service;
- `lmtp` – sending messages via LMTP service;
- `pipe` – sending messages through PIPE service;
- `queue` – processing of internal queue service.
- `level` – log verbosity level. The following values may be used: FATAL, ERROR, WARN, INFO, DEBUG.
- `sid` – session identifier of a message to which that log line is related. The number must be specified in hexadecimal notation;
- `mta-id` – identifier of a message inside MTA, from which this message is received. It can be obtained only if **Dr.Web MailID** is integrated with some MTA, and it allows to get such information;
- `text` – text of a log message.

When any module is started, log verbosity level is set to `INFO` by default. After configuration is received from **Agent**, this level is changed accordingly. To view module initialization log on `DEBUG` level (e.g. to gain information about parameters received from **Agent**) you may use `--level` command line parameter (by setting its value to `debug`).

[MySQL] Section

In [MySQL] section settings for establishing and maintaining interaction between **Dr.Web MailD** and MySQL database are collected:

User = {string}	MySQL database user name. Default value: **User** =
Password = {string}	MySQL database password. Default value: **Password** =
DB = {string}	Name of the MySQL database. Default value: **DB** =
Host = {hostname}	Name of the host used by MySQL database. Default value: **Host** = localhost
Port = {port address}	Port used to connect to MySQL database. **Example**: When TCP-socket is used: **Port** = tcp://1234 When UNIX socket is used: **Port** = unix:///path/to/socket Default value: **Port** =

Connections = {integer}	Number of simultaneous connections to MySQL database. When parameter value is set to 0, connections will be created on demand each time request to database is made (it usually takes additional time). Connections opened in advance can handle database requests in turn without wasting the time on reconnection. Default value: **Connections** = 4
SizeLimit = {integer}	Maximum number of strings received in response to single database request. When parameter value is set to 0, maximum number of received strings is not limited. Parameter value can also be specified in local lookup settings. Default value: **SizeLimit** = 10
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =
Lib = {path to file}	Path to libmysqlclient.so library. Default value: **Lib** = /usr/lib/libmysqlclient_r.so

Please note, that when using `libmysqlclient_r.so` library in FreeBSD 6.4/amd64 the following error might appear:

```
Undefined symbol "gethostbyname_r"
```

[PostgreSQL] Section

In [PostgreSQL] section settings for establishing and maintaining interaction between **Dr.Web MailD** and PostgreSQL database are collected:

ConnectionsString = {string}	String with connection settings for PostgreSQL database. The string can be empty to use default parameters, or it can contain one or more parameter settings separated by white space. Each parameter setting is in the form `keyword = value`. Spaces around the equal sign are optional. To write an empty value or a value containing spaces, enclose it in single quotes. If empty string is specified, default parameters are used. Please refer to http://postgresql.org/docs/8.3/static/libpq-connect.html for more details. **Examples**: **ConnectionString** = host=localhost port=5432 user=ai password=qwerty dbname=drweb **ConnectionString** = hostaddr=127.0.0.1:5432 dbname=mailddb user=mailddbuser password=Str0ngPaSSw0rd
	Default value:

	ConnectionsString =
SizeLimit = {integer}	Maximum number of strings received in response to single database request. When parameter value is set to 0, maximum number of received strings is not limited. Default value: **SizeLimit** = 10
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =
Lib = {path to file}	Path to libpq.so library. Default value: **Lib** = /usr/lib/libpq.so

[Firebird] Section

In [Firebird] section settings for establishing and maintaining interaction between **Dr.Web MailD** and Firebird database are collected:

Host = {hostname}	Firebird database hostname. Default value: **Host** = localhost
Database = {string}	Name of the Firebird database. Default value: **Database** =

User = {string}	Firebird database user name. Default value: **User** =
Password = {string}	Firebird database password. Default value: **Password** =
Charset = {string}	Charset encoding used in Firebird database. Default value: **Charset** = us-ascii
SizeLimit = {integer}	Maximum number of strings received in response to a single database request. When parameter value is set to 0, maximum number of received strings is not limited. Default value: **SizeLimit** = 10
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =
Lib = {path to file}	Path to libFBclient.so library. Default value: **Lib** = /usr/lib/libFBclient.so

[CDB] Section

In [CDB] section settings for establishing and maintaining interaction between **Dr.Web MailD** and CDB database are collected:

Sources = {path to file}	List of paths to CDB databases. Default value: **Sources** =
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =

[Berkeley] Section

In [Berkeley] section settings for establishing and maintaining interaction between **Dr.Web MailD** and Berkeley database are collected:

Databases = {path to file}	List of paths to Berkeley databases Default value: **Databases** =
Environment = {path to directory}	Path to directory where Berkeley database temporary lock files are stored. Default value: **Environment** =

SizeLimit = {integer}	Maximum number of bytes received in response to single database request. Values must be within the following range: from 1024 to 65536. Other values will be clamped to the allowed range. Default value: **SizeLimit** = 1
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =
Lib = {path to file}	Path to libdb.so library. Default value: **Lib** = /usr/lib/libdb.so

[SQLite] Section

In [SQLite] section settings for establishing and maintaining interaction between **Dr.Web MailD** and SQLite database are collected:

Database = {path to file}	Path to SQLite database file. Default value: **Database** =
SizeLimit = {integer}	Maximum number of strings received in response to a single database request. When parameter value is set to 0, maximum number of received strings is not limited. Default value:

	SizeLimit = 1
SkipDomains = {LookupsLite}	List of domains for which request to the database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =
Lib = {path to file}	Path to libsqlite3.so library. Default value: **Lib** = /usr/lib/libsqlite3.so
BusyTimeout = {time in milliseconds}	Timeout for the **Dr.Web MailD** to make an entry to the database. Default value: **BusyTimeout** = 2000

[ODBC] Section

In [ODBC] section settings for establishing and maintaining interaction between **Dr.Web MailD** and ODBC database are collected:

Lib = {path to file}	Path to library that supports ODBC version 3.0 or higher. Library must be built with threads support. UnixODBC is recommended. Library is located using dlopen system call (please refer to corresponding documentation). In current version this parameter can not be changed with SIGHUP signal. Default value: **Lib** = /usr/lib/libodbc.so

ConnectData {string}	=	ODBC connection parameters. Two formats are supported: • USER/PASSWORD/@DSN – is Oracle-like style; • DSN=value;UID=value; PWD=value – is ODBC-like style. To start working with ODBC at least DSN value must be specified. Parameter value can also be specified in local lookup settings. Default value: **ConnectData** =
SizeLimit {string}	=	Maximum number of strings received in response to single database request. When parameter value is set to 0, maximum number of received strings is not limited. Parameter value can also be specified in local lookup settings. Default value: **SizeLimit** = 0
SkipDomains {LookupsLite}	=	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings. Default value: **SkipDomains** =

[Oracle] Section

In [Oracle] section settings for establishing and maintaining interaction between **Dr.Web MailD** and Oracle database are collected:

Lib = {path to file}	Path to library that supports Oracle OTL version 8 or higher. Library must be built with threads support. Library is located using dlopen system call (please refer to corresponding documentation). In current version this parameter can not be changed with SIGHUP signal. Default value: **Lib** =
ConnectData = {string}	Oracle connection parameters. Two formats are supported: • USER/PASSWORD/@DSN – is Oracle-like style; • DSN=value;UID=value; PWD=value – is ODBC-like style. To start working with Oracle at least DSN value must be specified. Parameter value can also be specified in local lookup settings. Default value: **ConnectData** =
SizeLimit = {string}	Maximum number of strings received in response to single database request. When parameter value is set to 0, maximum number of received strings is not limited. Parameter value can also be specified in local lookup settings. Default value: **SizeLimit** = 0
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings.

	Default value: **SkipDomains** =

[LDAP] Section

In [LDAP] section settings for establishing and maintaining interaction between **Dr.Web MailD** and LDAP server are collected:

Lib = {path to file}	Path to OpenLDAP library version 2.0 or higher. Library must be built with threads support (i.e. must have "_r" suffix in file name). Library is located using dlopen system call (please refer to corresponding documentation). In current version this parameter can not be changed with SIGHUP signal.
	Default value: **Lib** = /usr/lib/libldap_r.so

Please note, that when using libldap_r.so library in FreeBSD 6.4/amd64 the following error might appear:

```
Undefined symbol "gethostbyname_r"
```

Hostname = {string}	LDAP server hostname. If parameter value is not specified, localhost is used. Parameter value can also be specified in local lookup settings.
	Default value: **Hostname** =
Port = {number}	LDAP server port. If parameter value is not set, port 389 is used. Parameter value can also be specified in local lookup settings.
	Default value:

	Port = 389
Timeout = {time}	Timeout for LDAP requests. Parameter value can also be specified in local lookup settings. Default value: **Timeout** = 10s
Version = {string}	LDAP protocol version. To enable secure data transfer with TLS/SSL you must use LDAP protocol version 3 or higher. If parameter value is not set, LDAP protocol version 3 is used. Parameter value can also be specified in local lookup settings. Default value: **Version** = 3
Bind = {Yes | No}	Enables binding before making requests. For LDAP protocol version 3, binding is not necessary. Parameter value can also be specified in local lookup settings. Default value: **Bind** = No
BindDn = {string}	Unique name for binding. Parameter value can also be specified in local lookup settings. Default value: **BindDn** =
BindPw = {string}	Password used for binding. Parameter value can also be specified in local lookup settings. Default value: **BindPw** =
SearchBase =	Base DN to start search from (RFC2253).

{string}	Default value: **SearchBase** =
SizeLimit = {integer}	Maximum number of strings received in response to single database request. When parameter value is set to 0, maximum number of received strings is not limited. Parameter value can also be specified in local lookup settings. Default value: **SizeLimit** = 0
Dereference = {3 | 2 | 1 | 0}	Permissions for LDAP aliases: • 0 – never; • 1 – when searching; • 2 – when locating base object for search; • 3 – always. This parameter value can also be specified in local lookup settings. Default value: **Dereference** = 0
ChaseReferrals = {integer}	LDAP_OPT_REFERRALS setting. To set this parameter LDAP protocol version 3 or higher is required. Parameter value can also be specified in local lookup settings. Default value: **ChaseReferrals** = 0
SkipDomains = {LookupsLite}	List of domains for which request to database is not required. This parameter often allows to improve total performance and considerably reduce server load. Parameter value can also be specified in local lookup settings.

	Default value: **SkipDomains** =
CheckPeriod = {time}	Maximum period of inactivity for LDAP connection to be closed. Check for LDAP inactive connections is made using the same period of time.
	Default value: **CheckPeriod** = 2m

[MailBase] Section

In [MailBase] sections settings for **Dr.Web MailD** database are collected:

MaxStoredMessages = {integer}	Maximum number of messages to be stored in mail database. When parameter value is set to 0, maximum number of messages is not limited. If number of messages in database exceeds the amount set by this parameter, then database is cleaned up from old messages until the specified quantity is reached. Messages which are already sent are deleted immediately, unsent messages are sent first and then are also deleted.
	Default value: **MaxStoredMessages** = 100000
MaxStorageSize = {size in bytes}	Maximum mail database size in bytes. When parameter value is set to 0, maximum size is not limited. If database size exceeds the limits set by this parameter, then database is cleaned up from old messages until the specified size is reached.
	Default value: **MaxStorageSize** = 0

MaxPoolSize {integer}	=	Maximum mail database pool size. If parameter value is set to 0, pool size is set up automatically according to available physical memory capacity. In current version this parameter can not be changed with SIGHUP signal. Default value: **MaxPoolSize** = 0
SendTimeout {time}	=	Timeout for plug-in to perform an asynchronous scan of a message. When timeout is exceeded, it is assumed that error occurred during message check. Actions for such case are defined in **ProcessingErrors** parameter of [Maild] section. Default value: **SendTimeout** = 30s
FrozenTimeout {time}	=	Additional time for message processing. If plug-in can not process the message in the period of time specified in **SendTimeout** parameter, processing time can be extended by **FrozenTimeout** parameter. Default value: **FrozenTimeout** = 2h
DeleteTimeout {time}	=	Maximum period of time for a message to be stored in mail database. **DeleteTimeout** parameter value must be greater than **FrozenTimeout** parameter value. Default value: **DeleteTimeout** = 48h
BackupPeriod {time}	=	Time period for a database backup. When parameter value is set to 0, no backup is performed.

	Default value: **BackupPeriod** = 0
BackupName = {filename}	Mail database backup file name. If specified file name ends with question mark ("?"), each backup is stored in a separate file, and question mark in the file name is replaced with the current time value.
	Default value: **BackupName** = %var_dir/msgs/db/.maildb.backup
MaxBodySizeInDB = {size}	Maximum size of a message stored in mail database. Messages exceeding this parameter value are stored in separate external files.
	Default value: **MaxBodySizeInDB** = 1k
SyncMode = {Yes | No}	Synchronization mode used for internal database. If the value of this parameter is set to yes, then for each transaction a fsync is called. As a result the database stored on disk is always up-to-date, but system performance is decreased. If the value is set to no, then a OS buffering is used for database synchronization. As a result after drweb-maild crashes, some data from the last transactions can be lost, but DB will not be destroyed and system performance will increase. If there are no special requirements for a system reliability, then you may leave no as a value of this parameter.
	Default value: **SyncMode** = no

[Filters] Section

In [Filters] section general setting for **Dr.Web MailD** plug-ins are collected:

LibDir = {path to directory}	A directory for plug-ins. Default value: **LibDir** = %bin_dir/maild/plugins/
Settings = {list of plug-in settings}	Plug-ins startup settings. They are specified in the following format: **Settings** = [plugin_settings], [plugin_settings]... where [plugin_settings] is plugin_name: [PARAM1]|...|[PARAMN], and [PARAMN] is **setting_name** = setting_value. Example: **Settings** = vaderetro: **max_size** = 400k|**log_level**=debug, drweb: **max_size** = 10m For Vaderetro plug-in maximum size of a message for scan is set to 400 Kbytes, log verbosity level is set to debug, and for drweb plug-in maximum size of a message for scan is set to 10 Mbytes. All values (except paths to files) are case-insensitive. Default value: **Settings** =

In current version only the following parameters can be specified for plug-ins:

section = {text value}	Name of the section in configuration file, where plug-in parameters are kept.
max_size = {size}	Maximum size of a message to scan. When parameter value is set to 0, maximum size is not limited. Its default value depends on the queue in which plug-in resides, and is defined by the value of **MaxSizeBeforeQueueFilters** or **MaxSizeAfterQueueFilters** parameter. This parameter can also be specified in rules as: plugin_name/**max_size** = value.
	Client's rules with **max_size** must look like the following: [Rule:client1] ... plugin_name/**max_size** = {size} [Rules] **md-client**:client1 cont rule=client1
	Example: [Rule:Client1] **AdminMail** = root@client1.drweb.ru **SenderAddress** = inet:25@10.0.0.0 **ProtectedDomains** = client1.drweb.ru, client1 **ProtectedEmails** = regex:.*@client1.drweb.ru, regex:.*@client1 **ProtectedNetworks** =

	10.0.0.0/32 drweb/**max_size** = 100k [Rules] ... **md-client**:client1 cont **rule** =client1
log_level = {Quiet | Error | Alert | Info | Debug}	Log verbosity level. Default value of this parameter is the same as **Level** parameter value from [Logging] section.
log_ipc_level = {Quiet | Error | Alert | Info | Debug}	IPC library log verbosity level. Default value of this parameter is the same as **IpcLevel** parameter value from [Logging] section.
syslog_facility = {Daemon | Mail | Local0 .. Local7}	Syslog facility type. Default value of this parameter is the same as **SyslogFacility** parameter value in [Logging] section.
log_filename = {syslog | path to file}	Path to log file. You can specify syslog as log filename and logging will be carried out by syslogd system service.
path_to_lib = {path to file}	Path to plug-in library. Path can be absolute or relative. If relative path is specified, value of **LibDir** parameter from [Filters] section is used as a root.

	Default value of this parameter is defined according to the following algorithm: `LibDir+"/lib"+plugin_name+".so"`, where plugin_name is written in lower case. For example, to connect `vaderetro` plug-in to the **Dr.Web MailD** in Linux environment with default settings, the following path should be used: `%bin_dir/maild/plugins/libvaderetro.so`. Via **`Settings`** parameter third-party plug-ins can be connected to the **Dr.Web MailD**. In this case value of this parameter is specified in the same way as it was described above: **`Settings`** = `vaderetro: log_level=info|max_size=200k| path_to_lib = /opt/drweb/maild/plugins/libvaderetro.so`.
	Default value: **`path_to_lib`** = `LibDir+"/lib"+plugin_name+".so"`
`BeforeQueueFilters` = `{list of plug-ins}`	List of plug-ins which process a message before it is put to the queue or to the database.
	Default value: **`BeforeQueueFilters`** =
`MaxSizeBeforeQueueFilters` = `{size}`	Maximum size of a message to be processed by plug-ins defined in **`BeforeQueueFilters`** parameter value. Used only when **`max_size`** parameter value is not explicitly specified for plug-in. When parameter value is set to `0`, maximum size is not limited.
	Default value: **`MaxSizeBeforeQueueFilters`** =

AfterQueueFilters = {list of plug-ins}	List of plug-ins which process a message after it is put to the queue or to the database. Default value: **AfterQueueFilters** =
MaxSizeAfterQueueFilters = {size}	Maximum size of a message to be processed by plug-ins defined in **AfterQueueFilters** parameter value. Used only when **max_size** parameter value is not explicitly specified for plug-in. When parameter value is set to 0, maximum size is not limited Default value: **MaxSizeAfterQueueFilters** = 0
PluginsBaseDir = {path to directory}	Path to the directory where plug-ins' working files are stored. Default value: **PluginsBaseDir** = %var_dir/plugins/

[Stat] Section

In [Stat] section parameters regulating statistics gathering in **Dr.Web MailD** are collected:

Detail = {value}	There exist several log verbosity levels: • off – disables collection of statistics, which allows to increase performance of the software complex. As a result there will be no statistics for export and no reports to send.

	• `low` – enables collection of statistics on operation of the whole software complex. As a result there will be possible to export statistics and send reports. • `medium` – allows to collect statistics on groups, which have not yet disabled this function in their settings. Access to group statistics can be gained either via the control socket or via the web interface. • `high` – allows to collect statistics on all users listed in the internal database, which have not yet disabled this function in their settings. Access to user statistics can be gained either via the control socket or via the web interface. Access to statistics can be obtained via the control socket or via the web interface. Statistics collected on `low` level is also included in reports, if this function is enabled. This parameter can be specified in rules for each Client separately, under the name **`StatDetail`**.
	Default value: **`Detail`** = `low`
`Send` = {`Yes` | `No`}	Sending reports to statistics server (or **Dr.Web** central protection server if **Dr.Web MailD** is working as part of the anti-virus network).
	Default value: **`Send`** = `Yes`
`SendPeriod` = {time}	Period of time to send statistics to server.
	Default value:

	SendPeriod = 10m
Timeout = {time}	Timeout for response from statistics server. Default value: **Timeout** = 30s

It is possible to perform export of statistics by means of **Dr.Web MailD** via storage type.

To enable export of statistics via storage type, do the following:

1. Specify Yes as a value of **ExportStat** parameter from [Stat] section;
2. Specify statistics export command(s) as a value of any of the following parameters from the [Stat] section:

ExportStat = {Yes | No}	Export statistics to storages defined in corresponding parameters (see below). Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately. Default value: **ExportStat** = No
ExportBlockObjectsStorage = {query string}	List of parameters to include in exported statistics for blocked messages. Export will be done right after message blocking but only if Antivirus plugin was able to scan the message (i.e. statistics will not be exported if message was blocked because of processing errors). The names of the table and fields in the database are not important but fields must have appropriate type and be in the same order in the query as in the database. It is not necessary to use all available values in the query.

	Text fields (<varchar_long>) must be enclosed in single quotation marks('). List of values to use in the query: • :number<int> – unique message identifier; • :q_name<varchar_long> – path to the quarantine file where the message was saved (if it was saved in quarantine); • :virus_name<varchar_long> – name of the blocked object found in the message; • :virus_code<int> – code of the blocked object found in the message. The list of codes: ○ 1 – infected; ○ 2 – virus modification; ○ 3 – suspicious; ○ 4 – cured; ○ 5 – deleted; ○ 6 – filtered; ○ 7 – skipped; ○ 8 – archive restrictions; ○ 9 – errors; ○ 10 – read errors; ○ 11 – write errors; ○ 12 – adware; ○ 13 – dialer; ○ 14 – joke; ○ 15 – riskware; ○ 16 – hacktool.

	• : plugin_name<varchar_long> – name of plug-in which blocked the message; • :sender<varchar_long> – sender address in angle brackets; • :client_ip<varchar_long> – IP address of Client that loaded message into mail database (if available); • :date<timestamp> – timestamp of loading message into mail database; • :client_id<varchar_long> – The unique identifier of the user for which save into mail database is performed. **Example**: `ExportBlockObjectsStorage = "odbc:insert into viruses values (:number<int>, ': q_name<varchar_long>', ': virus_name<varchar_long>', : virus_code<int>, ': plugin_name<varchar_long>', ': sender<varchar_long>', ': client_ip<varchar_long>', : date<timestamp>, ': client_id<varchar_long>)'"` Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately.
	Default Value: **ExportBlockObjectsStorage** =
ExportStatStorage = {query string}	Export of statistics on number of processed messages. Export is performed: • on shutdown;

	• after a period of time specified in **SendPeriod** parameter value in [Stat] section. If statistics is empty (no messages were processed), nothing is exported. The names of the table and fields in the database are not important but fields must have appropriate type and be in the same order in the query as in the database. It is not necessary to use all available values in the query. Text fields (<varchar_long>) must be enclosed in single quotation marks('). List of values to use in the query: • :size<int> – total size of scanned messages in bytes; • :num<int> – total number of scanned messages; • :q_num<int> – total number of messages saved in quarantine; • :r_num<int> – total number of redirected messages; • :n_num<int> – total number of messages with notifications; • :pass_num<int> – total number of passed messages; • :reject_num<int> – total number of rejected messages; • :discard_num<int> – total number of discarded messages; • :tempfail_num<int> – total number of tempfailed messages; • :date<timestamp> – mail database timestamp; • :q_size<int> – total size of messages saved in quarantine;

	• :r_size<int> – total size of redirected messages; • :n_size<int> – total size of messages with notifications; • :pass_size<int> – total size of passed messages; • :reject_size<int> – total size of rejected messages; • :discard_size<int> – total size of discarded messages; • :tempfail_size<int> – total size of tempfailed messages; • :work_time<int> – plug-in opration period in milliseconds (ms). **Example:** **ExportStatStorage =** "odbc: insert into g_stat values(: size<int>, :num<int>, : q_num<int>, :r_num<int>, : n_num<int>, :pass_num<int>, : reject_num<int>, : discard_num<int>, : tempfail_num<int>, : date<timestamp>)"
	Default value: **ExportStatStorage** =
ExportPluginStatStorage = {query string}	Statistics is exported only for plug-ins specified in **Names** parameter value of [Reports] section (or for all working plug-ins if parameter value is not specified). Export is performed: • on shutdown; • on SIGHUP signal; • when a report to Administrator is sent;

	• after a specified period of time, if reports are not sent too often If statistics is empty (no messages were processed), nothing is exported. The names of the table and fields in the database are not important but fields must have appropriate type and be in the same order in the query as in the database.It is not necessary to use all available values in the query. Text fields (<varchar_long>) must be enclosed in single quotation marks('). List of values to use in the query: • the same as in the description of **ExportStatStorage** parameter; • : plugin_name<varchar_long> - plug-in name to export statistics for; **Example**: **ExportPluginStatStorage = "** odbc:insert into plugin_stat values(': plugin_name<varchar_long>', : size<int>, :num<int>, : q_num<int>, :r_num<int>, : n_num<int>, :pass_num<int>, : reject_num<int>, : discard_num<int>, : tempfail_num<int>, : date<timestamp>)"
	Default value: **ExportPluginStatStorage** =

For detailed information on the existing facilities for exporting of the statistics, see Export of Statistics chapter.

[Reports] Section

In [Reports] section parameters regulating creation and dispatch of reports on operation of plug-ins are collected:

Send = {Yes | No}	Whether to send reports or not. Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately under the name **ReportsSend**. Default value: **Send** = Yes
SendTimes = {time}	Reports schedule. The syntax looks like the following: • hour:minute:second[-period] - send a report in the specified time every day; • Nw/hour:minute:second[-period] - send a report in the specified time on Nth day of week (0 - Sunday, 1 - Monday, 2 - Tuesday, etc.); • Nm/hour:minute:second[-period] - send a report in the specified time on Nth day of month. If the period of time is specified ({time} format), report covers the specified period of time, otherwise period of time is assumed to be equal to 24 hours. **Example**: **SendTimes** = 00:00:00-24h, 1w/00:00:00-7d, 2M/21:23:32-31d

	In this case three reports are sent: daily report at midnight, weekly report on Monday midnight and monthly report on the second day of each month at 21:23:32. Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately under the name **ReportsSendTimes**. Default value: **SendTimes** = 24h
Mail = {e-mail address}	E-mail addresses to send reports to. If parameter value is not specified, e-mail addresses defined in **AdminMail** parameter value in [Notifier] section are used. Several e-mail addresses can be specified with comma used as a delimiter. Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately under the name **ReportsMail**. Please note that if the value of **ReportsMail** parameter is set, then reports will not be sent to the address specified in **AdminMail** parameter. **Mail** =
Names = {list of plug-ins}	List of plug-ins, report is created for. Format: plug-in_name1, plug-in_name2, ... If parameter value is not set, report is created for plug-ins defined in **BeforeQueueFilter** and **AfterQueueFilter** parameter values in [Filters] section. Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately under the name **ReportsNames**.

	Names =
TopListSize = {integer}	Outputs to the report the lists of frequently blocked objects and addresses, from which maximum amount of blocked objects was sent. Parameter value defines number of entries in each list. When parameter value is set to 0, lists are not created. When parameter value is set to -1, lists' size is not limited. Value of this parameter is used by default for all the Clients. This parameter can be specified for each Client separately under the name **ReportsTopListSize**. **TopListSize** = 20
MaxStoreInDbPeriod = {time}	Maximum period of time to store statistics in reports' database. When parameter value is set to 0, old entries will not be deleted. **MaxStoreInDbPeriod** = 31d
CheckForRemovePeriod = {time}	Period of time at the end of which old entries will be deleted from reports' database. **CheckForRemovePeriod** = 5m

Configuration file **Dr.Web MailD** contains additional parameters for creation and sending the reports about the operation of plug-ins to the **Super-Administrator**.

GeneralSend = {Yes | No}	Sending general reports to the Super-Administrator. In those reports statistics on every client and general statistics on all plug-ins is displayed by default. All settings for general reports has **General** prefix. **GeneralSend** = Yes
GeneralSendTimes = {time}	A timetable for sending general reports to the Super-Administrator. Syntax:

	• `hour:minute:second[-period]` - send reports at the specified time every day; • `Nw/hour:minute:second[-period]` - send reports at the specified time on N day of the week (0 - Sunday, 1 - Monday, 2 - Tuesday, etc); • `Nm/hour:minute:second[-period]` - send reports at the specified time on N day of the month. If time interval is specified, then report is made for the specified time interval. Otherwise, time interval is 24 hours by default.
	Default value: **`GeneralSendTimes`** = `00:00:00`
`GeneralClientFilter` = `{regular expression}`	Filtration of clients during report preparation. Non-empty value of the parameter is a regular expression which Client's identifier is compared to. If there is a match, information on certain Client is included into the report. Information on all clients is included in general statistics.
	`GeneralClientFilter` =
`GeneralTotalStat` = `{Yes | No}`	Information on each plug-in is included to the general report.
	`GeneralTotalStat` = `Yes`
`GeneralMail` = `{e-mail address}`	Address used to send reports to the Super-Administrator.
	`GeneralMail` =

GeneralNames = {list of plug-in modules}	List of plug-ins for which the report to the Super-Administrator is made. List format: plug-in_name1, plug-in_name2, ... If this parameter is not set, report is made for modules, listed in values of **BeforeQueueFilter** and **AfterQueueFilter** parameters from the [Filters] section. **GeneralNames** = drweb1, drweb2
GeneralTopListSize = {number}	Number of frequently blocked objects and addresses to be displayed in the report to the Super-Administrator. **GeneralTopListSize** = 20

[Quarantine] Section

In [Quarantine] section settings for proper **Quarantine** operation are collected:

Path = {path to directory}	Path to the quarantine directory. Default value: **Path** = %var_dir/infected/
FilesMode = {numeric value}	Permissions for files to be moved to quarantine. This value can also be specified in rules in **QuarantineFilesMode** parameter. Default value: **FilesMode** = 0660
FilenamesMode = {Std | Tai | Rand48}	Naming mode for files to be moved to quarantine: • Std - renaming quarantined files with mkstemp command.

	`%FilenamesPrefix.XXXXXX` template is used, where `%FilenamesPrefix` is the prefix specified in **`FilenamesPrefix`** parameter value and `XXXXXX` is a combination of random letters and digits; • `Tai` - renaming quarantined files according to TAI (International Atomic Time). `%sec.%usec.%FilenamesPrefix.XXXXXX` template is used; • `Rand48` - renaming quarantined files with `lrand48` command. `%FilenamesPrefix.XXXXXXXX` template is used.
	Default value: **`FilenamesMode`** = `Std`
`FilenamesPrefix` = `{text value}`	Prefix used to rename files which are moved to quarantine. Parameter value must not contain "`%`" and "`_`" characters. Parameter can be specified in rules in **`QuarantineFilenamesPrefix`** parameter.
	Default value: **`FilenamesPrefix`** = `maild`
`AccessByEmail` = `{Yes | No}`	Request to receive messages saved to quarantine using control messages. Special message is sent to the e-mail address specified in **`FilterMail`** parameter value (or in rules) with special `Subject` header: `q:relative_path_to_file`

	where `relative_path_to_file` is a relative path to file saved in quarantine directory (for example, `/drweb/drweb.quarantine.puYtWx`). Corresponding message will be sent in response to such request only if one of its recipients or its sender matches control message sender. Such control message is automatically generated by MUA when corresponding link in received report is clicked. Please note, whereas default value of `OnlyTrustedControlMails` parameter of the `[Maild]` section it set to `yes`, control letters has to be sent from protected network. Otherwise, they will be ignored. Default value: **`AccessByEmail`** = `Yes`
`StoredTime` = `{time}`	Period of time to store a message in quarantine. When parameter value is set to `0`, this period of time is not limited. Default value: **`StoredTime`** = `24h`
`MaxSize` = `{size in Kbytes}`	Maximum size of messages in quarantine. If value of this parameter is set to `0`, then this size is not limited. For each message only body size is calculated, not its actual size on disk. This parameter affects only the size of internal database and does not affect the DBI storage (if it is connected). Specified value is used by default for all the Clients. A separate value for each client can be set in rules. Default value:

	MaxSize = 0
MaxNumber = {numerical value}	Maximum number of messages in quarantine. If value of this parameter is set to 0, then this number is not limited. This parameter affects only the number of messages in the internal database and does not affect the DBI storage (if it is connected). Specified value is used by default for all the Clients. A separate value for each client can be set in rules. Default value: **MaxNumber** = 0
MoveToDBI = {Yes | No}	Transfer of messages saved in quarantine from file storage to the DBI storage. To move messages to the DBI storage Perl modules File::Temp and DBI must be installed. Default value: **MoveToDBI** = No
DBISettings = {string}	DBI storage connection parameters. **Example**: "dbi:Pg:dbname=emails_db" Database must be created using SQL-ASCII character set. Default value: **DBISettings** =
DBIUsername = {text value}	User name to connect to the DBI storage. Default value:

	DBIUsername =
DBIPassword = {text value}	User password to connect to the DBI storage. Default value: **DBIPassword** =
SQLInsertCommand = {string}	A command to add a message to the DBI storage. Set, type and order of elements are fixed: 1. number - message number; 2. string - relative path to file containing a message. Format: client/plug-in/id.prefix, where client is a Client's identifier or "def" word, if some message does not belong to any Client, plug-in is for plug-in name, id is for message number in hexadecimal notation (8 characters are used in output), prefix is for prefix defined by **FilenamesMode** and **FilenamesPrefix** parameter values; 3. timestamp - time of adding the message to the database; 4. string – From header from an envelope (in angle brackets); 5. string - list of recipients from an envelope. Values are in angle brackets and delimited by commas; 6. string - message body. Elements in request must be replaced by question marks ("?"). **Example**: **SQLInsertCommand** = "INSERT INTO mail_export values

	`(?,?,?,?,?,?)"` Default value: **SQLInsertCommand** =
SQLRemoveCommand = `{string}`	A command used to delete messages from DBI storage. It is used when time limit for storing messages in database is specified. The only parameter specified in request is `time`, all messages older than this time must be deleted. Element in request must be replaced with question mark ("`?`"). **Example**: **SQLRemoveCommand** = "DELETE FROM mail_export WHERE put_time<=?" Default value: **SQLRemoveCommand** =
SQLSelectCommand = `{string}`	A command used to access messages in DBI storage (if it is necessary, for example, to request a message from quarantine using control letter). The only parameter used in request is a relative file name in quarantine. Element in request must be replaced with question mark ("`?`"). Set, type and order of elements are fixed: 1. `number` - message number; 2. `timestamp` - time of adding message to database; 3. `string` - message body; 4. `string` - `From` header from an envelope (in angle brackets); 5. `string` - list of recipients from an envelope. Values are in angle brackets and delimited by commas;

	6. <relative path to file> with object. For more details you may refer to the description of **SQLInsertCommand** parameter above. **Example**: **SQLSelectCommand** = "SELECT id, put_time,body,sender,rcpts, filename FROM mail_export WHERE filename LIKE ?" Default value: **SQLSelectCommand** =
PulseTime = {time}	Period of time to delete old messages and move messages from file storage to the DBI storage. When parameter value is set to 0, program defined in **PathToDrwebQp** parameter value will not start. Default value: **PulseTime** = 5m
PathToDrwebQp = {path to file}	Path to drweb-qp program. Default value: **PathToDrwebQp** = %bin_dir/drweb-qp
MoveAll = {Yes | No}	Move all incoming messages directly to /Path_parameter_value/def/backup/ directory for archiving. Parameter must be used with **MoveToDBI** = yes, otherwise directory may be quickly filled up with incoming messages. Default value: **MoveAll** = No

[Maild] Section

In [Maild] section general setting for proper operation of **Dr.Web MailD** are collected:

ProtectedNetworks = {lookups}	List of networks protected by **MailD**. Values are specified in CIDR format. This parameter is used for determining trusted networks in some VadeRetro parameters and if trust_protected_networks is specified in SessionRestrictions parameter in [Receiver] section. **Example**: **ProtectedNetworks** = 10.0.0.0/24, 127.0.0.0/8, 192.168.0.68 Default value: **ProtectedNetworks** = 127.0.0.0/8
ProtectedDomains = {lookups}	List of domains protected by **MailD**. This parameter is used for determining trusted domains if trust_protected_domains is specified in SessionRestrictions parameter in [Receiver] section. **Example**: **ProtectedDomains** = example.ru, example.com Default value: **ProtectedDomains** =
IncludeSubdomains = {Yes | No}	Include subdomains in protected domains list. Default value: **IncludeSubdomains** = yes

InPoolOptions = {pool settings}	Settings for thread pool that processing messages before queue. Default value: **InPoolOptions** = auto
OutPoolOptions = {pool settings}	Settings for thread pool that processing messages after queue. Default value: **OutPoolOptions** = auto
RedirectMail = {e-mail address}	E-mail address to send messages to, when Redirect action is used. Default value: **RedirectMail** = root@localhost
OnlyTrustedControlMails = {Yes | No}	Send control messages (e.g. to receive message from quarantine) only from protected network. If **Receiver** component does not provide information about client's IP address, it is necessary to use **GetIpFromReceivedHeader** = Yes, to force MTA add a correct Received header before handling message to the **Dr.Web for UNIX mail servers**. To make control messages work, outgoing clients' mail traffic must be scanned by the **Dr.Web** . Default value: **OnlyTrustedControlMails** = Yes
MaxScore = {numerical value}	Maximum message score. If message score exceeds the value of this parameter, then actions specified in **MaxScoreAction** parameter will be applied to this message, and message check will be stopped. This parameter is checked before message is passed to plug-ins and after the check by each plug-in.

	Default value: **MaxScore** = 10000
MaxScoreAction = {actions}	Actions applied to the message when its score exceeds the threshold value specified in **MaxScore** parameter. If reject action is specified and value of **UseCustomReply** parameter is set to yes, then SMTP response is taken from **ReplyMaxScore** parameter. After all actions are applied, the message check is considered finished. Mandatory actions (they must be specified necessarily) are: pass, discard, reject, tempfail. Additional actions are: quarantine, redirect, add-header, score. Please note, that several actions can be specified in this parameter. Default value: **MaxScoreAction** = reject
MaxMimeParts = {numerical value}	Maximum number of MIME parts in a message. If value is set to 0, then check is not performed. If the number of MIME parts in some message exceeds the specified threshold value, then message processing is aborted and actions specified in **ProcessingError** parameter are applied to it. Default value: **MaxMimeParts** = 1000

MaxNestedMimeParts = {numerical value}	Maximum number of nested MIME parts in the message. If value is set to 0, then check is not performed. If the number of nested MIME parts in some message exceeds the specified threshold value, then message processing is aborted and actions specified in **ProcessingError** parameter are applied to it. Default value: **MaxNestedMimeParts** = 100
LicenseLimit = {actions}	Actions applied to messages that have not been scanned due to license limitations. Mandatory actions (they must be specified necessarily) are: pass, discard, reject, tempfail. Additional actions are: quarantine, redirect, notify, add-header, score. Please note, that several actions can be specified in this parameter. Default value: **LicenseLimit** = pass
EmptyFrom = {actions}	Actions applied to messages that have an empty From header in envelope. Such situation is possible when mail notifications are used; spammers also ignore this header. Mandatory actions (they must be specified necessarily) are: continue, discard and reject. Additional actions are: quarantine, redirect, add-header, score. Please note, that several actions can be specified in this parameter. Default value: **EmptyFrom** = continue

ProcessingErrors = {actions}	Action applied to messages which invoke scanning errors. Mandatory actions (they must be specified necessarily) are: pass, discard, reject, tempfail. Additional actions are: quarantine, redirect, notify, add-header, score. Please note, that several actions can be specified in this parameter. Default value: **ProcessingErrors** = pass
RulesLogLevel = {Quiet | Error | Alert | Info | Debug}	Rules processor log verbosity level. Default value: **RulesLogLevel** = Alert
PidFile = {path to file}	Path to the PID-file of drweb-maild process. Default value: **PidFile** = %var_dir/run/drweb-maild.pid

The following parameters define SMTP replies for blocked messages.

When message is blocked by any component of **Dr.Web for UNIX mail servers** solution, a 550 5.7.0 error code and a specific text message is used for SMTP-reply. Texts for the messages can be specified in values of parameters described below. They must be enclosed in quotation marks.

UseCustomReply = {Yes | No}	Enables usage of custom messages in SMTP sessions. These messages will be sent as SMTP reply when incoming message is rejected. Default value: **UseCustomReply** = No

ReplyEmptyFrom = {string}	Reply to send when **EmptyFrom** = reject is applied and **UseCustomReply** = Yes. Only text part of reply can be specified: "550 5.7.0 Text". Text must be enclosed in quotation marks if it contains white spaces. Default value: **ReplyEmptyFrom** = "DrWEB maild: Messages from <> are blocked by administrator."
ReplyProcessingError = {string}	Reply to send when **ProcessingError** = reject is applied and **UseCustomReply** = Yes. Only text part of reply can be specified: "550 5.7.0 Text". Text must be enclosed in quotation marks if it contains white spaces. Default value: **ReplyProcessingError** = "DrWEB maild: Message is rejected due to software error."
ReplyMaxScore = {string}	Reply to send when **MaxScoreAction** = reject is applied and **UseCustomReply** = yes. Only text part of reply can be specified: "550 5.7.0 Text". Text must be enclosed in quotation marks if it contains white spaces. Default value: **ReplyMaxScore** = "Dr.Web MailD: Message is rejected due to score limit exceed."
GetIpFromReceivedHeader = {Yes | No}	Use Received header value as client's IP address if this address is not identified by the **Receiver** component. Default value:

	GetIpFromReceivedHeader = Yes
Control = {Yes | No}	Enables drweb-maild interactive management module. Default value: **Control** = No
ControlAddress = {socket address}	Socket address used by drweb-maild interactive management module. Default value: **ControlAddress** = inet:3009@127.0.0.1
ControlPoolOption = {pool settings}	Thread pool settings for the control socket of the drweb-maild. Default value: **ControlPoolOption** = auto
SkipDSNOnBlock = {Yes | No}	Skip DSN dispatch when program failed to pass return code to **Receiver** component after performing Reject or Tempfail actions. Default value: **SkipDSNOnBlock** = No

[Receiver] Section

In [Receiver] section settings for **Receiver** component are collected. This component is used in **Dr.Web for UNIX mail servers** solution for Exim and Postfix mail systems and in **Dr.Web for UNIX mail gateways** solution.

Address = {socket address}	Address used by **Receiver** component to receive messages. As a value of this parameter address of a socket is specified (either TCP-socket in `inet:port@hostname` format, or UNIX socket in `local:path_to_socket_file` format). Default value: **Address** = inet:25@0.0.0.0
PoolOptions = {pool settings}	Threads pool settings. Default value: **PoolOptions** = auto
RealClients = {Yes | No}	Accept connections directly from clients or from MTA. Default value: **RealClients** = Yes
ProcessingErrors = {actions}	Actions applied to messages when processing errors occur. Mandatory actions are: `tempfail`, `discard`, `reject`. Only one action can be specified. Default value: **ProcessingErrors** = reject
StalledProcessingInterval = {time}	Timeout to process stalled messages. Stalled messages are messages received by plug-ins, but not processed in a reasonable time to be sent to **Checker** component. Such situation can happen when network or power supply problems occur. Default value: **StalledProcessingInterval** = 10m

OneCommandTimeout = {time}	Timeout to execute single command. Default value: **OneCommandTimeout** = 5m
OneMessageTimeout = {time}	Timeout to receive single message. Default value: **OneMessageTimeout** = 10m
AddReceivedHeader = {Yes | No}	Add Received header to all received messages. Default value: **AddReceivedHeader** = Yes
ReturnReject = {Yes | No}	**Receiver** component policy in case of Reject action. When Yes parameter value is specified, 5** error is returned, when No parameter value is specified, 2** error is returned and DSN report is sent to message sender. When working with Exim MTA and there are some plug-ins in **BeforeQueueFilters** list, it is recommended to use **ReturnReject** = No to avoid messages freezing in Exim queue. Default value: **ReturnReject** = Yes
GreetingString = {string}	Greeting string to be output when new **Client** is connected. "%host%" macro is replaced by the **Hostname** parameter value from [General] section, "%ver%" macro is replaced by the current version of drweb-receiver module. Default value: **GreetingString** = "%host% Dr.

	Web SMTP receiver v%ver% ready"
MaxRecipients = {integer}	Maximum number of recipients. When parameter value is set to 0, maximum number of recipients is not limited. If IP address from which connection was initialized is marked as trusted, then this restriction is not checked. Default value: **MaxRecipients** = 100
MaxConcurrentConnection = {integer}	Maximum number of SMTP connections from a single IP address. When parameter value is set to 0, maximum number of SMTP connections from a single IP address is not limited. Default value: **MaxConcurrentConnection** = 5
MaxMailsPerSession = {integer}	Maximum number of messages per single session. When parameter value is set to 0, maximum number of messages per single session is not limited. Default value: **MaxMailsPerSession** = 20
MaxReceivedHeaders = {integer}	Maximum number of Received headers. When parameter value is set to 0, maximum number of Received headers is not limited. Default value: **MaxReceivedHeaders** = 100

MaxErrorsPerSession = {integer}	Maximum number of errors per single session. When parameter value is set to 0, maximum number of errors per single session is not limited. Default value: **MaxErrorsPerSession** = 10
MaxMsgSize = {size}	Maximum message size. This restriction will always be checked by **Receiver** component, even if IP address from which connection was initialized is marked as trusted. Default value: **MaxMsgSize** = 10m
MaxJunkCommands = {integer}	Maximum number of RSET, NOOP and NTFY commands per session. If this number exceeds the specified value, then an error counter is activated. Current value of the error counter is set to zero each time message is successfully processed by the drweb-maild module. If parameter value is set to 0, then this restriction is ignored. Default value: **MaxJunkCommands** = 100
MaxHELOCommands = {integer}	Maximum number of HELO, EHLO and LHLO commands per session. If this number exceeds the specified value, then an error counter is activated. If parameter value is set to 0, then this restriction is ignored. Default value: **MaxHELOCommands** = 20

RelayDomains = {lookups}	List of domains allowed for message relaying. When specifying a usual domain list for which **Dr.Web MailD** will be a mail relay, their subdomains will be ignored. I.e. mail arriving from their subdomains will not be relayed.
	It is possible to specify a list of subdomains by using regular expression or `rfile`.
	Example:
	RelayDomains = `regex:.*.domain.com`
	Allows relaying to all `domain.com` subdomains.
	Example:
	RelayDomains = `rfile:/path`
	`rfile` contains a set of regular expressions (Perl syntax), and each regular expression must reside on the new line:
	`.*.domain.com`
	`.*.domain1.com`
	`.*.domain2.com`
	In the current version of **Dr.Web MailD** **RelayDomains** parameter doesn't support wildcard DNS records. Thus, expressions of this type are not allowed:
	RelayDomains = `*.domain`

	Default value: **RelayDomains** =

A number of parameters described below specify actions for validation of IP-addresses of connections not marked as trusted on various stages of SMTP-session.

By default only connections from `localhost` and UNIX sockets are considered trusted.

Verification actions to be applied to IP-addresses of connections are specified in values of corresponding parameters sequentially, with comma used as delimiter. Actions are applied in order they are specified.

SessionRestrictions = {string}	Checks performed immediately after connection was initiated. • trust_protected_network [SCORE] – [SCORE] - if connection IP-address is included in the list defined by **ProtectedNetworks** parameter, it is marked as trusted IP-address. If SCORE is specified, its value is added to the score of each message transferred in current session and to the score of IP address of the sender ; • trust_protected_domains [SCORE] – check if connection IP-address is in the list defined by **ProtectedDomains** parameter. Check is performed using double DNS request. PTR-request is made to check if received host name is in **ProtectedDomains** list. If it is in this list, A-request is made to check if connection IP-address is in received address list. If yes, address is marked as trusted IP-address. If SCORE is specified, its value is added to the score of each message transferred in current session and to the score of IP address of the sender ;

	• `trust_white_networks [SCORE]` – if connection IP-address is in the white list defined by **`WhiteNetworks`** parameter, address is marked as trusted IP-address. If `SCORE` is specified, its value is added to the score of each message transferred in current session and to the score of IP address of the sender; • `trust_white_domains [SCORE]` – check if connection IP-address is in the white list defined by **`WhiteDomains`** parameter. PTR request is made. If it is in this list, it is marked as trusted IP-address. If `SCORE` is specified, its value is added to the score of each message transferred in current session and to the score of IP address of the sender ; • `reject_dnsbl [SCORE]` – check if connection IP-address is in RBL/DNSBL black lists defined by **`DNSBLList`** parameter. PTR request is made. If it is in this list, session is terminated and error code is returned. If `SCORE` is specified, its value is added to the score of each message transferred in current session and to the score of IP address of the sender; • `reject_black_networks [SCORE]` – if connection IP-address is in the black list defined by **`BlackNetworks`** parameter, session is terminated. If `SCORE` is specified, an error is output to log, and score value is added to the score of each message transferred in current session and to the score of IP address of the sender;

	• `reject_black_domains [SCORE]` – check if connection IP-address is in the black list defined by **`BlackNetworks`** parameter. PTR request is made. If it is in this list, session is terminated and error code is returned. If `SCORE` is specified, an error is output to log, and score value is added to the score of each message transferred in current session and to the score of IP address of the sender.
	Default value: **`SessionRestrictions`** = `trust_protected_network, trust_sasl_authenticated`
`HeloRestrictions` = `{string}`	Checks performed on `HELO/EHLO` stage: • `reject_unknown_hostname [SCORE]` – if host name has neither DNS A nor DNS MX record, mail from this address is blocked. If `SCORE` is specified, an error is output to log, and score value is added to the score of each message transferred in current session and to the score of IP address of the sender . A-requests and sometimes MX-requests are made; • `reject_diff_ip [SCORE]` – if IP-address of the client does not coincide with any of the IP-address resolved for domain name given in the EHLO/HELO command, mail from this address is blocked. If `SCORE` is specified, then the letter passes, but an error is output to log and score value is added to the score of each message transferred in current session and to the score of IP address of the sender.

	Default value: **HeloRestrictions** =
SenderRestrictions = {string}	Checks performed on FROM stage. • reject_unknown_domain [SCORE] – if sender host name has neither DNS A nor DNS MX record, mail from this address is blocked. If SCORE is specified, an error is output to log, and score value is added to the score of each message transferred in current session and to the score of IP address of the sender. A-requests and sometimes MX-requests are made; • trust_sasl_authenticated [SCORE] – if SASL authentication was successful, adress is marked as trusted IP-address. If SCORE is specified, its value is added to the message score. Default value: **SenderRestrictions** = trust_sasl_authenticated
RecipientRestrictions = {string}	Checks performed on RCPT stage. All the receivers are checked in turn. • reject_unknown_domain [SCORE] – if sender host name has neither DNS A nor DNS MX record, mail to this address is blocked. If SCORE is specified, an error is output to log, and score value is added to the message score. A-requests and sometimes MX-requests are made;

	• reject_unauth_destination [SCORE] – if recipient domain is neither in **RelayDomains** list nor in **ProtectedDomains** list, mail to this address is blocked. If SCORE is specified, an error is output to log, and score value is added to the message score; • reject_unknown_rcpts [SCORE] – checks if recipient is specified in the **ProtectedEmails** list. If recipient's address is not in this list, mail to this address is blocked. If SCORE is specified, an error is output to log, and score value is added to the message score. It is recommended to use it together with **anti_dha** **Reputation IP Filter**. Default value: **RecipientRestrictions** = reject_unauth_destination
DataRestrictions = {string}	Checks performed on RCPT stage. All the receivers are checked in turn. • reject_spam_trap [SCORE] – check for spam trap. Recipient address must have <USER@HOST> format. If host name is in the list defined by **ProtectedDomains** parameter (unless the list is empty) and user name is in the list defined by **SpamTrap** parameter, message is blocked. If SCORE is specified, an error is output to log, and score value is added to the message score. Full e-mail address can be also specified in **SpamTrap** list;

	• `reject_multi_recipient_bounce [SCORE]` – block messages with empty `FROM` header and several recipients. If `SCORE` is specified, an error is output to log, and score value is added to the message score; Default value: **`DataRestrictions`** =
`RestrictionStat` = `{Yes | No}`	Statistics for operation with restrictions. To get statistics send `SIGUSER1` signal to the `drweb-receiver` process. Statistics is stored in `restrictions.txt` file in directory defined in **`BaseDir`** parameter from `[General]` section. Default value: **`RestrictionStat`** = `No`
`DelayRejectToRcpt` = `{Yes | No}`	Suspend blocking messages until `RCPT` stage. Setting this parameter allows to work with outdated e-mail clients and output list of blocked recipient addresses to the log file. Default value: **`DelayRejectToRcpt`** = `Yes`
`BlackNetworks` = `{lookups}` **`WhiteNetworks`** = `{lookups}`	Networks black and white lists. These lists are used in `trust_white_networks` and `reject_black_networks` actions. Refer to **`ProtectedNetworks`** parameter for more details. Default value: **`BlackNetworks`** = **`WhiteNetworks`** =

DNSBLList = {lookups}	DNSBL servers list. This list is used in reject_dnsbl action. Servers are checked in turn in the order they are specified in parameter value until the message is blocked or the list is over. Default value: **DNSBLList** =
PositiveDNSBLCacheTimeout = {time}	Timeout to cache positive responses from DNSBL servers. Default value: **PositiveDNSBLCacheTimeout** = 24h
NegativeDNSBLCacheTimeout = {time}	Timeout to cache negative responses from DNSBL servers. Default value: **NegativeDNSBLCacheTimeout** = 10m
NegativeDNSCacheTimeout = {time}	Timeout to cache negative responses from DNS servers. Parameter value is valid for all DNS responses except DNSBL responses. Default value: **NegativeDNSCacheTimeout** = 10m
BlackDomains = {lookups} **WhiteDomains** = {lookups}	Black and white lists of domains. These lists are used in trust_white_domains and reject_black_domains actions. Refer to **ProtectedDomains** parameter for more details. Default value: **BlackDomains** = **WhiteDomains** =

SpamTrap = {lookups}	Spam trap address list. This list is used in `reject_spam_trap` action. Default value: **SpamTrap** =
ReputationIPFilter = {list of filters}	**Reputation IP filter** allows to assign a score to the IP address according to the collected statistics on connections and to block this IP address temporarily if its total score is greater than some threshold value. The following filters are available: **anti_dha**, **errors_filter**, **score_filter**. Filters are listed using comma as a delimiter, and are checked in order they were specified. Default value: **ReputationIPFilter** =
ProtectedEmails = {lookups}	List of protected addresses. It is used in `reject_unknown_rcpts` restriction. It allows to discard messages with invalid recipients and to resist DHA attacks (when it is used with **anti_dha** filter in **Reputation IP Filter**). It is recommended to specify this parameter together with `reject_unknown_rcpts` restriction and use it with **anti_dha** filter. Default value: **ProtectedEmails** =
MaxSessionScore = {integer}	A threshold value for the general score of each session. If this score exceeds the threshold value, then the corresponding connection will be closed with a temporary error returned. If this value is set to 0, then this parameter is ignored.

	Default value: **MaxSessionScore** = 10000

Restrictions allow to filter out unwanted mail in `drweb-receiver` module on the stage of SMTP session, before messages are passed to the `drweb-maild` for check. It allows to save resources and adds the additional level of spam filtration which increases spam detection probability.

Restrictions are applied on the following stages of SMTP session:

- connection of the new client (**SessionRestrictions** parameter);
- receipt of `HELO/EHLO` command (**HeloRestrictions** parameter);
- receipt of `FROM` command - i.e. when the client specifies sender for the new message (**SenderRestrictions** parameter);
- receipt of `RCPT` command - i.e. when the client adds new recipient to the current message (**RecipientRestrictions** parameter);
- receipt of `DATA` command - i.e. when the client has already finished transfer of all the recipients and is ready to send the body of the message (**DataRestrictions** parameter).

Restrictions are set as values of ***Restrictions** parameters, with comma used as a delimiter. They are checked in order they were set - from the left to the right. Restrictions are checked only after all other checks are performed (sequencing of commands, validity of their parameters, etc.).

For the each connection a `Trusted IP` flag is checked. If it is set, then restrictions are not checked. `Trusted IP` flag is always set for connections established using UNIX sockets, and also it can be set for some restrictions.

It is possible to collect statistics on each restriction to define the quantity of blocked messages and its efficiency. To get the collected data, send the special signal to the `drweb-receiver`

process as described in the chapter Signals. To enable or disable collection of statistics use **RestrictionStat** parameter.

Blocking effects vary depending on the stage of the SMTP session. When blocking is performed according to restrictions from **SessionRestrictions** parameter - the whole session appears to be blocked - i.e. for every subsequent command from the user an error is returned. Blocking on all other stages affects only the certain command.

Each restriction can take optional parameter - score value [SCORE] (except for **set_score** and **add_score** restrictions, where the score value is the only mandatory parameter). Depending on the type of the restriction, score is processed in different ways:

- restriction can work if the current score of the message is less than the value specified in the parameter;
- restriction can work if the current score of the message is greater than the value specified in the parameter;
- if restriction is activated a value of corresponding parameter is added to the message score.

Depending on the stage of the SMTP session, restrictions can work with a score of each message in the current session (for **SessionRestrictions** and **HeloRestrictions** stages) or with an individual score of each processed message (on other stages).

For each stage of checking for restrictions there are the stage-specific restrictions, as well as restrictions which can be used at almost every stage. To the latter the following applies:

- `mark_trust [SCORE]` - set `Trusted IP` flag. All other restrictions after this parameter will be skipped. If the `SCORE` is specified, then `Trusted IP` flag is set only when the current message score is lower than the value of the specified score.
- `sleep SEC [SCORE]` - sleep for the period of time in seconds, specified in `SEC`. It may be useful for blocking of spammers, because the majority of them will not wait for a response from server even for a few seconds. If the `SCORE` is

specified, then this restriction is applied only to messages which current score is greater than the specified score.

- `tempfail [SCORE]` - return the temporary SMTP error (code `4*`). It may be useful when it is necessary to temporarily reject a client, if it has not passed some checks and can pass them some time later. If the `SCORE` is specified, then temporary error is returned only when the current message score is greater than the specified score.
- `reject [SCORE]` - return the permanent SMTP error (code `5*`). It may be useful when a client did not pass the check and will not be able to pass it later. If the `SCORE` is specified, then a permanent error is returned only when the current message score is greater than the specified score.
- `pass_sasl_authenticated [SCORE]`- skip all other checks on this stage of the SMTP session if the client has successfully passed SASL authentication. This restriction is useful only on **SenderRestrictions**, **RecipientRestrictions** and **DataRestrictions** stages, because authentication can be passed only after `HELO/EHLO` command is received. If the `SCORE` is specified, then checks are skipped only for messages which current score is less than the specified score. Please note, that only checks specified for this stage of SMTP session (i.e. after `pass_sasl_authenticated`) will be skipped. Checks on other stages will not be skipped.
- `set_score SCORE` - changes current message score to `SCORE` value. If it is used at **SessionRestrictions** or **HeloRestrictions** stages, then it affects the score of every message in the session, on other stages it affects the score of the current processed message.
- `add_score SCORE` - adds `SCORE` value to the current message score. If it is used on **SessionRestrictions** or **HeloRestrictions** stages, then it affects the score of every processed message in the session, on other stages it affects the score of the current processed message.

Examples:

SenderRestrictions = `trust_protected_networks, reject`

- allows to only receive mail from IP-addresses, specified in **ProtectedNetworks**, other IP-addresses are blocked;

SenderRestrictions = trust_protected_networks, trust_protected_domains, sleep 5, add_score 10

- allows to receive mail from IP-addresses, specified in **ProtectedNetworks**, and from domains, specified in **ProtectedDomains**. For other messages before continuation, it pauses for 5 seconds and increases the message score by 10 points.

[SASL] Section

In [SASL] section parameters for SASL autentification in **Dr.Web for UNIX mail gateways** solution (designed for operation as a proxy-server for SMTP-protocol) are collected:

Use = {Yes | No}	Enable SASL authentication. Default value: **Use** = No
Driver = {cyrus}	SASL authentication driver. In current version only cyrus driver is available. To use it, install and set up cyrus-sasl2 library. Default value: **Driver** = cyrus
BrokenAuthClients = {Yes | No}	Support for outdated SMTP clients which use non-standard AUTH protocol syntax. Default value: **BrokenAuthClients** = Yes
AuthenticatedHeader = {Yes | No}	Adds names of registered users to Received header. When the value is set to Yes, names of registered users are visible to all.

	Default value: **AuthenticatedHeader** = No

[Cyrus-SASL] Section

In [Cyrus-SASL] section parameters which enable proper operation of cyrus-sasl driver, are collected:

Lib = {path to file}	Absolute path to cyrus-sasl2 library. Default value: **Lib** = /usr/lib/libsasl2.so.2
Path = {string}	Configuration file name (.conf extension is added automatically). cyrus-sasl2 library receives its settings from this file. Default value: **Path** = maild
ServerHostname = {string}	Host name. If parameter value is not set, **Hostname** parameter value from [General] section is used. If **Hostname** value is also not specified, then value returned by gethostname function is used. Default value: **ServerHostname** =
ServerRealm = {string}	SASL realm the server belongs to. Default value: **ServerRealm** =
SecurityOptions = {string}	List of security settings, separated by commas. The following security settings are allowed:

	• noplaintext - prohibition of authentication mechanisms susceptible to simple passive attacks (e.g., PLAIN, LOGIN); • noactive - protection from active (non-dictionary) attacks during authentication exchange; • nodictionary - prohibition of authentication mechanisms susceptible to passive dictionary attacks; • noanonymous - prohibition of authentication mechanisms allowing anonymous login; • mutual_auth - require mutual authentication.
	Default value: **SecurityOptions** = noanonymous

[Sender] Section

In [Sender] section settings of the **Sender** component (which is responsible for sending messages) are collected. This section is not included to **Dr.Web** software distribution for operation with Communigate Pro mail transfer system.

UseSecureHash = {Yes | No}	Add **SecureHash** header to all outgoing messages. **UseSecureHash** and **SecureHash** parameters are not used in solutions designed for operation with Courier and Exim mail transfer systems, and also in **Dr.Web for UNIX mail gateways** solution. In **Dr.Web for UNIX mail servers** solution for Sendmail and Qmail MTAs `Yes` value must be specified for **UseSecureHash** parameter, if the same MTA is used to send and receive messages. It allows to avoid messages' infinite loop possibility and optimizes system operation. If diverse MTAs are used to send and receive messages, `No` must be specified to avoid increase of **SecureHash** header beyond system limits. Default value: **UseSecureHash** = Yes
	In **Dr.Web for UNIX mail servers** solution for Zmailer MTA `Yes` value must be specified only if `drweb-zmailer` is used at a routing stage (i.e. is started from `process.cf`). In this case all messages generated by `drweb-sender` are processed by `drweb-zmailer`. **SecureHash** header is added to avoid messages' infinite loops and double-check possibility. Default value: **UseSecureHash** = No

	In **Dr.Web for UNIX mail servers** solution for Postfix MTA `Yes` value must be specified only when interaction with Posxfix is performed via `milter` protocol (`drweb-milter` module is used). In this case all messages generated by `drweb-sender` are processed by `drweb-milter`. **SecureHash** header is added to avoid messages' infinite loops and double-check possibility. Default value: **UseSecureHash** = No
SecureHash = {string}	Content of **SecureHash** header. An arbitrary string of symbols (not less then 10 symbols) can be used as a value of this parameter. To increase security it is strongly recommended to change parameter default value! In **Dr.Web for UNIX mail servers** solution for Zmailer MTA value of this parameter must be the same as `--hash` parameter value used at startup of `drweb-zmailer`, when Zmailer is used on routing stage. Default value: **SecureHash** = !!!---------___EDIT_THIS___!!!
StalledProcessingInterval = {time}	Timeout for processing of stalled messages. Stalled messages are messages received by plug-ins, but not processed in reasonable time to be sent to **Checker** component. Such situation can happen when network or power supply problems occur. Default value: **StalledProcessingInterval** = 10m

SendingIntervals = {time}	Time periods between attempts to send stalled messages. When **Dr.Web for UNIX mail servers** is running in synchronous mode, **Sender** attempts to send processed letter regardless of the first interval specified in the parameter value. If transfer is unsuccessful, **Sender** goes to delayed sending after an interval specified in **SendingIntervals** parameter value. If zero is used as a first parameter value, it is ignored as there was an attempt to send a letter immediately. If **Dr.Web for UNIX mail servers** is running in asynchronous mode, **Sender** always attempts to send letters according to the parameter value. Default value: **SendingIntervals** = 0s, 30s, 60s, 10m, 30m, 2h, 8h, 1d, 1d
Method = {SMTP | LMTP | PIPE}	Method used by **Sender** component to deliver messages. • SMTP - messages are sent via SMTP protocol; • LMTP - messages are sent via LMTP protocol; • PIPE - messages are sent via PIPE to some external mail program. Default value: **Method** =
MailerName = {SMTP | Sendmail | Postfix | CommuniGate | Qmail | Exim | Zmailer |	Name of the MTA working with **Dr.Web for UNIX mail servers**. This parameter is used when **Method** = pipe. In current version this parameter can not be changed with SIGHUP signal.

	Default value: Default value of this parameter depends on the MTA for which a certain distribution of **Dr.Web for UNIX mail servers** is designed.
Address = {address}	Address used by the **Sender** component to send messages. If **Method** = pipe, full path to the MTA used to receive messages must be specified in this parameter value, otherwise address of a socket used for dispatch of messages is specified as a value of **Address** parameter. In **Dr.Web for UNIX mail gateways** solution besides standard types of addresses, mx:HOSTNAME type can be used, where HOSTNAME is the name of the host. When this type is used, software complex receives all MX records using host name and sends message according to them.
	This parameter can have multiple addresses to send messages, delimited by commas. **Example**: **Address** = inet:25@10.4.0.90, inet:25@10.4.0.91, inet:25@10.4.0.92 If MTA located at address 10.4.0.90 stop to response, **Sender** will attemp to send an email to 10.4.0.91. In case of unsuccessful transmission, the letter will be sent to 10.4.0.91. When the amount of addresses is much, it is recomended to increase to 5 minutes values of **MaxTimeoutForThreadActivity** and **IpcTimeout** parameters to allow **Sender** switch to the last address in case of the absence of a response from the previous address.
	Default value:

	Default value of this parameter depends on the MTA for which a certain distribution of **Dr.Web for UNIX mail servers** is designed.
Options = {string}	Optional parameters for the external mail program, which is initialized when PIPE method is used. Default value: **Options** =
InPoolOptions = {pool settings}	Threads pool settings for processing before queue. Default value: **InPoolOptions** = auto
OutPoolOptions = {pool settings}	Threads pool settings for processing after queue. Default value: **OutPoolOptions** = auto

Parameters described below are specified only in solutions for Exim and Postfix MTAs, and in **Dr.Web for UNIX mail gateways** solution.

HeloCmdTimeout = {time}	Timeout to execute HELO/EHLO commands. Default value: **HeloCmdTimeout** = 5m
MailFromCmdTimeout = {time}	Timeout to execute MAIL command. Default value: **MailFromCmdTimeout** = 5m
RcptToCmdTimeout =	Timeout to execute RCPT command.

{time}	Default value: **RcptToCmdTimeout** = 5m
DataCmdTimeout = {time}	Timeout to execute DATA/BDAT commands. Default value: **DataCmdTimeout** = 2m
DataBlockTimeout = {time}	Timeout to send a message. Default value: **DataBlockTimeout** = 3m
EndOfDataTimeout = {time}	Timeout to receive confirmation of message delivery. Default value: **EndOfDataTimeout** = 10m
OtherCmdsTimeout = {time}	Timeout to execute other commands via SMTP/LMTP. Default value: **OtherCmdsTimeout** = 2m
PipeTimeout = {time}	Timeout to receive response using PIPE. Default value: **PipeTimeout** = 2m
SendDSN = {Yes | No}	Send DSN report. Default value: **SendDSN** = No

Router = {string}	Message routing rules depending on recipients. Messages addressed to different recipients can be sent from different e-mail addresses. If message has several recipients and such message must be sent from different e-mail addresses, recipients list must be divided into groups so, that each group will receive its own copy of a message from individual e-mail address. Message copy is created for every group of recipients. Parameter values are specified in DOMAIN ADDRESS format, where: • DOMAIN is a string to check recipients envelopes. Envelope has <user@host> format. Search is case-insensitive. For example, if "@localhost" string is looked up, <test@localhost> and <yy@localhost.localdomain> envelopes will match, and if <@localhost> string is looked up, only <test@localhost> envelope will match; • ADDRESS is an address to send message to, if DOMAIN string is found in envelope. ADDRESS format is similar to format of **Address** parameter in this configuration file. It is possible to supply several e-mail addresses delimited by "|" symbol, then message will be delivered to the first address which was possible to connect to. **Example**: **Router** = @main.server.com> mx:main.server.com|inet:25@backup.server.com

	In this case messages addressed to recipients from `main.server.com` domain will be sent to addresses indicated in MX record for `main.server.com`. If delivery fails, system will try to deliver message to `backup.server.com` on port `25`.
	Default value: **`Router`** =

Router usage

`Router` parameter allows to use Lookups (except `regex`, `wildcards` and `rfile`).

Example:

```
Router = "mysql:select address from senders where user='$u'"
```

This query checks whether there is a local part of local part of recipients address in `user` row of `sender` table in MySQL database. If yes, than letter is sent to the address specified in `address` column of the matched string.

Example:

```
Router = "ldap:///description?sub?(cn='$d')",
domain1.com inet:25@example.com | inet:1025@example.com | inet:2025@example.com,
mail.com mx: | inet:25@mail.backup, domain2.com mx:mail.ru | inet:25@mail.backup, "file:/path/to/routers.list"
```

First query search recipient's domain name in `cn` atribute and return redirection parameters from `description` field. Messages addressed to recipients from `domain1.com` will be sent to `example.com` on port `25`. If delivery fails, system will try to deliver message to the same address on port `1025` and then on port `2025`. Messages addressed to `mail.com` will be sent to MX records corresponding to this address. If recipient's address does

not match any of the previously described addresses, compliance will be checked in `/path/to/routers.list` file.

Please note that only one address if assigned to each domain, therefore expressions of this type are not allowed:
Router = domain, domain2 25@host

If the letter can not be sent to any of matched addresses, then depending on the last MTA's return code:

- **Sender** goes to delayed sending after an interval specified in **SendingIntervals** parameter's value. Rules specified in **Router** parameter also applicable to delayed shipment. If delayed sending fails then depending on **SendDSN** parameter's value DSN report will be generated by **Sender**. If special routes doesn't specified in **Router** parameter or `maild` rules for sender's domain that specified in undelivered letter's envelope, then this DSN will be sent to address specified in **Address** parameter.
- if last MTA return 5** code then DSN will be generated immediately and the letter will be deleted from out-queues. Scheme of DSN sending is similar to described above. If DSN can not be delivered it will be deleted after the interval specified in **SendingIntervals** parameter.

When some routes are determined in **Router** parameter and `[Rules]` section for the same domain, only routes specified in `[Rules]` section will take effect.

Please note that even if you configure routing of all mail using **Router** parameter, **Address** parameter value should not be empty; otherwise **Sender** fails to start. It should be borne in mind that if no match found for then the letter will be sent to address specified in **Address** parameter value.

[Milter] Section

In `[Milter]` section parameters for managing operation of `drweb-milter` module are collected. `Drweb-milter` module is

responsible for interaction between **Dr.Web for UNIX mail servers** solution and Postfix and Sendmail MTAs via `milter` protocol. This section is included to **Dr.Web MailD** configuration files of packages for operation with Postfix and Sendmail mail transfer systems.

Address = {socket address}	Socket address to establish connection via `milter` protocol. It must comply with definition specified in settings of mail system (in `sendmail.cf` configuration file of Sendmail MTA and in `main.cf` configuration file of Postfix MTA). Path to PID file cannot be used as a value of this parameter. **Example**: **Address** = local:%var_dir/ipc/drweb-milter.skt Default value: **Address** = inet:3001@127.0.0.1
Timeout = {time}	Timeout for `drweb-milter` to connect to Sendmail or Postfix MTA. Specified value must be greater than any **Timeout** parameter value in configuration file of the mail system. Default value: **Timeout** = 2h
PendedConnections = {numeric value}	Queue length for pending connections (`drweb-milter` waits for MTA to process messages). Default value: **PendedConnections** = 64
CanChangeBody = {Yes | No}	Let MTA modify message body. Postfix MTA supports this function starting from version 2.4. In current version this parameter can not be changed with `SIGHUP` signal.

	Default value: **CanChangeBody** = Yes
ProcessingTimeout = {time}	Timeout for the drweb-milter module to wait for message to be scanned. It is recommended to set this parameter value greater than **SendTimeout** parameter value from [MailBase] section. Default value: **ProcessingTimeout** = 40s
ProcessingErrors = {actions}	Action applied to messages invoked scanning errors. Only one of these actions can be specified: tempfail, discard, pass, reject. Default value: **ProcessingErrors** = reject
MinPersistConnection = {numeric value}	Minimum number of connections to the drweb-maild module. Default value: **MinPersistConnection** = 2
UseStat = {Yes | No}	Statistics of connections to the drweb-maild module. Statistics is written to file when drweb-milter process receives SIGUSR1 signal. Default value: **UseStat** = No
MaxFreetime = {time}	Inactivity timeout to wait before closing all connections with the drweb-maild module. Default value: **MaxFreetime** = 2m

ReplyPoolOptions = {pool settings}	Pool settings for threads processing responses from the drweb-maild module. Default value: **ReplyPoolOptions** = auto

[CgpReceiver] Section

In [CgpReceiver] section settings enabling interaction of **Receiver** component with CommuniGate Pro mail transfer system are collected. This section is included in **Dr.Web MailD** configuration file of a package designed for operation with the above mentioned MTA.

ProcessingTimeout = {time}	Timeout for the **Receiver** component to wait for a message to be scanned. It is recommended to set this parameter value greater than **SendTimeout** parameter value from [MailBase] section. Default value: **ProcessingTimeout** = 40s
PoolOptions = {pool settings}	Threads pool settings. Default value: **PoolOptions** = auto
ProcessingErrors = {actions}	Action applied to messages invoked scanning errors. Only one of these actions can be specified: tempfail, discard, pass, reject. Default value: **ProcessingErrors** = reject

ChownToUser = {string}	Set owner for a message file received from CommuniGate Pro MTA. As drweb-cgp-receiver module runs with Administrator privileges (root), you can either leave this parameter value empty and start up the whole **Dr.Web for UNIX mail servers** system with Administrator privileges, or you can set as this parameter value specific user name used to run **Dr.Web for UNIX mail servers** (drweb by default).
	Default value: **ChownToUser** = drweb

[CgpSender] Section

In [CgpSender] section settings enabling interaction of **Sender** component with CommuniGate Pro mail transfer system are collected. This section is included in **Dr.Web MailD** configuration file of a package designed for operation with the above mentioned MTA.

UseSecureHash = {Yes | No}	Add **SecureHash** header to all outgoing messages. If No value is specified, drweb-cgp-receiver module will not check messages sent using PIPE method. If Yes value is specified, drweb-cgp-receiver module will let messages with **SecureHash** header pass without check. If diverse MTAs are used to send and receive messages, No should be specified to avoid increase of **SecureHash** header beyond system limits.
	Default value: **UseSecureHash** = No

SecureHash = {string}	Content of **SecureHash** header. An arbitrary string of symbols (not less then 10 symbols) can be used as a value of this parameter. To increase security it is strongly recommended to change parameter default value. Default value: **SecureHash** = !!!---------___EDIT_THIS___!!!
PoolOptions = {pool settings}	Threads pool settings. Default value: **PoolOptions** = auto
SubmitDir = {path to directory}	Directory where drweb-cgp-sender module submits messages for CommuniGate Pro MTA to send them. Default value: **SubmitDir** = /var/CommuniGate/Submitted
SubmitFilesMode = {permissions}	Permissions for created notifications or cured messages. Default value: **SubmitFilesMode** = 0600
SubmitFilenamesPrefix = {string}	Prefix for file names of submitted messages. File name format: %{SubmitDir}/%{SubmitFilenamesPrefix}XXXXXX It is possible to use "%s" macro which will be replaced by a message identifier given to the message by CommuniGate Pro MTA and based on the file name. Usage of this macro can simplify log files analysis. Default value:

	SubmitFilenamesPrefix = drweb_submit_%s_
SubmitFilenamesMode = {std | tai | rand48}	Naming convention for file names of submitted messages: • Std - renaming files with mkstemp command. drweb_submit_XXXXXX template is used; • Tai - renaming files according to TAI (International Atomic Time). %sec.%usec. drweb_submit_XXXXXX template is used; • Rand48 - renaming files with lrand48 command. drweb_submit_XXXXXXXX template is used. Default value: **SubmitFilenamesMode** = std

[Courier] Section

In [Courier] section settings enabling interaction of **Dr.Web MailD** with Courier mail transfer system are collected. This section is included in **Dr.Web MailD** configuration file of a package designed for operation with the above mentioned MTA.

ProcessingTimeout = {time}	Timeout for the drweb-courier module to wait for a message to be scanned. It is recommended to set this parameter value greater than **SendTimeout** parameter value from the [MailBase] section. Default value: **ProcessingTimeout** = 40s

ProcessingErrors = {actions}	Action applied to a messages invoked scanning errors. Only one of these actions can be specified: tempfail, discard, pass, reject. Default value: **ProcessingErrors** = reject
MainPoolOptions = {pool settings}	Options for threads pool processing requests. Default value: **MainPoolOptions** = auto
ReplyPoolOptions = {pool settings}	Options for threads pool processing responses from the drweb-maild module. Default value: **ReplyPoolOptions** = auto
BaseDir = {path to directory}	Courier MTA installation directory. Default value: **BaseDir** = /usr/lib/courier
SocketDirs = {path to directory}	List of paths used to create UNIX sockets for interaction with the Courier MTA. UNIX socket is created in the first directory of the list, other directories are checked for UNIX sockets with same names as the drweb-courier module. Such UNIX sockets are deleted when found. In current version this parameter can not be changed with SIGHUP signal. Default value: **SocketDirs** = /var/lib/courier/allfilters, /var/lib/courier/filters

SocketAccess = {permissions}	Permissions for files with UNIX sockets used in **Dr.Web MailD** and Courier MTA interaction. In current version this parameter can not be changed with SIGHUP signal. Default value: **SocketAccess** = 0660

[Qmail] Section

In [Qmail] section settings enabling interaction of **Dr.Web MailD** with Qmail mail transfer system are collected. This section is included in **Dr.Web MailD** configuration file of a package designed for operation with the above mentioned MTA.

ProcessingTimeout = {time}	Timeout for the drweb-qmail module to wait for a message to be scanned. It is recommended to set this parameter value greater than **SendTimeout** parameter value from the [MailBase] section. Default value: **ProcessingTimeout** = 40s
ReadingTimeout = {time}	Timeout to receive envelope and message body from the qmail-queue module. Default value: **ReadingTimeout** = 20m
ProcessingErrors = {actions}	Action applied to the messages invoked scanning errors. Only one of these actions can be specified: tempfail, discard, pass, reject. Default value: **ProcessingErrors** = reject

MainPoolOptions = {pool settings}	Options for threads pool processing requests. Default value: **MainPoolOptions** = auto
ReplyPoolOptions = {pool settings}	Options for threads pool processing responses from the drweb-maild module. Default value: **ReplyPoolOptions** = auto
ListenUnixSockets = {socket address}	List of UNIX sockets for the drweb-qmail module to receive requests from the qmail-queue module for message scan. Sockets in this list must be also specified in the list of files monitored by the qmail-queue module. This list can be viewed using qmail-queue --help command. Default value: **ListenUnixSockets** = local:%var_dir/ipc/.qmail
QmailQueue = {path to file}	Path to the original initial qmail-queue file. Default value: **QmailQueue** = /var/qmail/bin/qmail-queue.original

[Notifier] Section

In [Notifier] section settings of drweb-notifier module are collected. Drweb-notifier is responsible for creation and sending of the reports about operation of **Dr.Web for UNIX mail servers** solution's components.

PoolOptions = {pool settings}	Threads pool settings. At first, number of threads in a pool is defined: • auto – number of threads in a pool is automatically detected, depending on the current system load; • N – non-negative integer. At least N threads in a pool will be active, and new threads will be created upon request; • N-M – positive integers, and M ≥ N. At least N threads in a pool will be active, and new threads will be created upon request until the number of threads reaches M value; Also the following additional parameters can be specified: • **timeout** = {time} – if a thread does not become active during the specified period of time, it is closed. This parameter does not affect the first N threads, which are waiting for requests infinitely. Default value: 2m • **stat** = {yes|no} – statistics for threads in a pool. It is saved each time SIGUSR1 system signal is received, to the directory specified in the value of **BaseDir** parameter from the [General] section. Default value: no • **log_level** = {Quiet|Error|Alert|Info|Debug} – log verbosity level for threads in a pool. If the value is not explicitly specified, value of **LogLevel** parameter from [Logging] section is used; • **stop_timeout** = {time} – maximum time for a working thread to

	stop (e.g. when program finishes its operation, or when it is necessary to decrease the number of threads in a pool). Default value: **PoolOptions** = auto
TemplatesBaseDir = {path to directory}	Path to the directory where report templates are kept. Default value: **TemplatesBaseDir** = %etc_dir/maild/templates
LngBaseDir = {path to directory}	Path to the directory where report language files are stored. Language files have .lng extension. Target language (ru for Russian, en for English, etc.) is specified in the first uncommented line of the language file. Specified value is used in **NotifyLangs** parameter to define the language for generation of reports. Default value: **LngBaseDir** = %etc_dir/maild/lng
AdminMail = {e-mail address}	Postmaster e-mail address. It is possible to specify several addresses. In this case generated reports are sent to all of them, and message body contains all the specified addresses. It recommended to specify this parameter, or reports will not be sent. Default value: **AdminMail** = root@localhost
FilterMail = {e-mail address}	E-mail address specified in From header of messages with reports. Default value:

	FilterMail = root@localhost
NotifyLangs = {string}	Language(s) used in the process of generation of reports. Default value: **NotifyLangs** = en
TemplatesParserLogLevel = {quiet | error | alert | info | debug}	Log verbosity level of subsystem creating reports based on templates. Default value: **TemplatesParserLogLevel** = info
RulesLogLevel = {quiet | error | alert | info | debug}	Rules processor log verbosity level. Default value: **RulesLogLevel** = info
MsgIdMap = {string}	Mapping of message identifier set in the **Receiver** component to the **Sender** component identifier for sending reports generated for this message. If mapping is not found, all reports are sent to the **Sender** component by default (with empty identifier). **Example**: MsgIdMap = id[12] sender_notifications In this case reports for messages generated by **Receiver** components with id1 or id2 identifiers are sent to the **Sender** component with sender_notifications identifier. This parameter is used in case of simultaneous usage of several pairs of **Sender** and **Receiver** components. Default value:

	MsgIdMap =
QuarantinePrefix = {string}	Prefix added to the output of file path in quarantine. This parameter allows to access files in quarantine using the off-site server. For example, if you install HTTP-server on the same host where **Dr.Web for UNIX mail servers** runs and set it up using **QuarantinePrefix** = http://mailhost/quarantine/, you can see links in reports, for example, http://mailhost/quarantine/headersfilter/drweb.quarantine.2kqtvI.
	Default value: **QuarantinePrefix** =

When **Dr.Web MailD** is processing a message, any plug-in can request the sending of the notification report about any event (detection of the virus, processing error, message blocking, etc.). These reports are generated by the **Dr.Web Notifier** (drweb-notifier module) and sent via the **Sender** component.

All reports are presented as template files with .msg extension. **Dr.Web Notifier** looks for them in the directory path to which is specified in the **TemplatesBaseDir** parameter. These templates support macros, conditions, cyclic paths and embedding of external files (syntax of these files is described in notify.* files), therefore they can be easily modified.

Three types of reports are available:

- reports with information about a specific message;
- regular reports with information about general activity of **Dr.Web MailD** software complex;
- DSN-reports about message delivery failure.

In all three cases the component sends the name of the report to the drweb-notifier module. All templates, except DSN-

templates, are in html and plain text format. Selection of an appropriate format is made according to `html` setting from the corresponding section of the rules.

For the reports of the first type **Dr.Web Notifier** checks with the help of rules whether it is necessary to send a report for each participant described below (for the detailed information, see chapter [Rules] Section):

- to the sender;
- to recipients (if reports' settings differ for each recipient, then more reports are sent to ensure that every recipient receives the report in the desired form);
- to the administrator.

Name of the report is made by adding `sender_`, `rcpts_` and `admin_` prefixes to the name of the external module with `.msg` extension. If such file is not found, then an error is reported.

For the reports of the second type **Dr.Web Notifier** sends only one report to administrator with general statics on operation of the software complex. Template for this report is included to `report.msg` file.

Reports of the third type are DSN-reports about delivery failure. The template for these reports is stored in `dsn.msg` file.

Dr.Web Notifier uploads all files with templates which comply with the following regular expression: `(admin|rcpts|sender|report|dsn)_?(.*?)\.msg`.

There exists a possibility to modify template files. **`NotificationNamesMap`** parameter allows to map the name of the report transferred to the **Dr.Web Notifier** to the new name, using which the name of the new template file will be created. Mapping must be performed only to names known to the **Dr.Web Notifier**, because otherwise it will not be able to find a required file.

[ProxyClient] Section

In the `[ProxyClient]` section settings of the `drweb-proxy-client` module are collected:

ProxyServersAddresses = {list of sockets}	List of socket addresses used by `drweb-proxy-server` components. Addresses are specified as follows: `ADDRESS1 [WEIGHT1], ADDRESS2 [WEIGHT2] ..` where `ADDRESS` has a basic address type, and `WEIGHT` is an optional numeric value from a range of 0 to 100, defining a "weight" of this address. This `WEIGHT` determines a relative work load on a certain host in the network. The greater is the value - the greater is the load on a certain server. Mail received from the **Receiver** component working on the same host as `drweb-proxy-client` does, will be passed for scan to corresponding software components using these sockets. There must be at least one valid server address among these addresses. Addresses are used according to the algorithm described in Using Proxy chapter.
	Default value: **ProxyServersAddresses** = `inet:8088@SERVER-IP`
Address = {list of sockets}	List of socket addresses used by the **Sender** component to receive requests from `drweb-proxy-server` components to send mail.

	drweb-proxy-server components will send mail to these addresses according to the value set for the **ProxyClientsAddresses** parameter from [ProxyServer] section. Default value: **Address** = inet:8066@0.0.0.0
MailPoolOptions = {pool settings}	Settings of a threads pool processing requests from the **Receiver** component. Threads pool processes requests from the **Receiver** component and sends messages to remote drweb-proxy-server components for check. After the check the message is either returned to the **Receiver** component, or sent via the **Sender** component. Default value: **MailPoolOptions** = auto
SenderPoolOptions = {pool settings}	Settings of a threads pool processing requests from drweb-proxy-server components to send mail via the **Sender** component. Before the message is sent to the **Sender** component, a temporary directory is created, where this message is stored. Results of **Sender**'s operation are returned to the dwreb-proxy-server. Default value: **SenderPoolOptions** = auto

[ProxyServer] Section

In the [ProxyServer] section settings of the drweb-proxy-server module are collected:

Address = {list of sockets}	List of socket addresses used by the drweb-proxy-server component to receive requests from drweb-proxy-client components. drweb-proxy-client passes messages for check to the drweb-proxy-server component according to the value set for the **ProxyServersAddresses** parameter from [ProxyClient] Section. Default value: **Address** = inet:8088@0.0.0.0
ProxyClientsAddresses = {list of sockets}	List of socket addresses used by drweb-proxy-client components to receive requests about sending messages. Addresses are specified as follows: ADDRESS1 [WEIGHT1], ADDRESS2 [WEIGHT2] .. where ADDRESS has a basic address type, and WEIGHT is an optional numeric value from a range of 0 to 100, defining a "weight" of this address. This WEIGHT determines a relative work load on a certain host in the network. The greater is the value - the greater is the load on a certain server. Socket addresses specified in the value of this parameter must match socket addresses from the value of **Address** parameter from [ProxyClient] Section. Default value: **ProxyClientsAddresses** = inet:8066@CLIENT-IP
ReceiverPoolOptions = {pool settings}	Settings of a threads pool responsible for passing messages for check to the drweb-maild.

	Threads in a pool receive requests from `drweb-proxy-client` to check some message, create a unique ID for this message and pass the message to the `drweb-maild` for check. After the check is finished `drweb-proxy-client` receives original or modified message. Default value: **`ReceiverPoolOptions`** = `auto`
`SenderPoolOptions` = `{pool settings}`	Settings of a thread pool responsible for passing messages to the `drweb-proxy-client`, to send them via the **Sender** component. Threads in a pool accept requests for sending messages from various components, and then pass these requests for processing to the `drweb-proxy-client`. Processing results are returned to components which made requests for sending messages. Default value: **`SenderPoolSettings`** = `auto`

[POP3] Section

Dr.Web MailD can work with POP3 servers via the protocol filter program. The POP3 filter program is a proxy-server connecting `drweb-maild` with the POP3 server. It filters e-mails sent to a user by the server. The POP3 server can be run on a local or on a remote computer.

In `[POP3]` section setting for `drweb-pop3` module are collected.

`ServerAddress` = `{socket address}`	Address used by the POP3 filter to connect to the POP3 server. Default value:

	ServerAddress = inet:pop3@localhost
ListenAddress = {socket address}	A list of socket addresses used to receive requests from clients. The following types of addresses can be specified: inet:.. or inet-ssl:.. (if you use TLS/SSL). The latter type requires POP3S protocol to be used by the POP3 filter. Default value: **ListenAddress** = inet:5110@localhost
ServerTLSSettings = {TLS/SSL settings}	TLS/SSL settings for server communications over POP3. Settings are separated by commas. The server is used only if a certificate and private key (private_key_file) are specified and inet-ssl socket is used. A detailed description of available parameters can be found in the general configuration file description. **Example**: **ServerTLSSettings** = use_sslv2 no, private_key_file /path/to/pkey, certificate /path/to/certificate Please note, that the user which privileges are used by the POP3 filter (usually, drweb user), must have read access to the file with the certificate. Default value: **ServerTLSSettings** =
ClientTLSSettings = {TLS/SSL settings}	TLS/SSL settings for client communications over POP3.

	Settings are separated by commas. A detailed description of available parameters can be found in the general configuration file description. **Example**: **ClientTLSSettings** = use_sslv2 no, private_key_file /path/to/pkey, certificate /path/to/certificate Please note, that the user which privileges are used by the POP3 filter (usually, drweb user), must have read access to the file with the certificate.
	Default value: **ClientTLSSettings** =
IoTimeout = {time}	Timeout for all input-output operations with client socket, when operation is in progress.
	Default value: **IoTimeout** = 60s
ProcessingTimeout = {time}	Timeout for drweb-maild to process messages.
	Default value: **ProcessingTimeout** = 60s
MinFilterToMaildConnections = {numerical value}	Minimum number of connections between POP3 filter and drweb-maild.
	Default value: **MinFilterToMaildConnections** = 2
MaxFilterToMaildConnections = {numerical value}	Maximum number of connections between POP3 filter and drweb-maild module. When the value is set to 0, number of connections is not limited.

	Default value: **MaxFilterToMaildConnections** = 0
FilterToMaildKeepAliveTime = {time}	Maximum retention time for inactive connections between POP3 filter and drweb-maild. To interact with drweb-maild, POP3 filter maintains several connections with it, and each connection can handle one operation. If there are no more available connections, then new connections are created until their amount meets the threshold specified in a value of **MaxFilterToMaildConnection** parameter. After connections stay inactive for a period of time specified in **FilterToMaildKeepAliveTime** parameter, they will be closed. Total amount of opened connections cannot be less than the value of **MinFilterToMaildConnections** parameter.
	Default value: **FilterToMaildKeepAliveTime** = 30s
PoolOptions = {pool settings}	Settings of main threads pool . This threads handle connections from clients. Each connection requires a new thread, otherwise some clients will be waiting for a free thread and stay disconnected.
	Default value: **PoolOptions** = auto
CallbackPoolOptions = {pool settings}	Settings of auxiliary threads pool. Threads handle signals about finished message processing from drweb-maild.

	Default value: **CallbackPoolOptions** = auto
MaxConnections = {numerical value}	Maximum number of incoming connections. If 0 is specified as a value of this parameter the number of incoming connections is not limited. Default value: **MaxConnections** = 0
DoS_Blackhole = {Yes | No}	If there are too many simultaneous connections from one IP address, drop them immediately without sending any error messages to the client. Default value: **DoS_Blackhole** = no
DisablePlainText = {Yes | No}	Do not allow the client to send login and password as plain text. It requires OpenSSL to be configured in advance. Default value: **DisablePlainText** = no
MaxConnectionsPerIp = {numerical value}	Maximum number of simultaneous connections from one IP address. If 0 is specified as a value of this parameter the number of incoming connections is not limited. Default value: **MaxConnectionsPerIp** = 0
MaxCommandLength = {size}	Maximum size of a command for the POP3 protocol. Each command is a string, which is sent from the client to the server. Maximum possible size of this command is about 1000 bytes according to the current RFC.

	Please note, that if the value of this parameter is left very small - up to 10 bytes (or even set to zero), then clients' commands will not be processed. Default value: **MaxCommandLength** = 1000b
OnFilterErrors = {actions}	Action to be applied to the message, when some error emerges before the message is passed to the drweb-maild module. Possible values are reject or pass. Default value: **OnFilterErrors** = reject

When session begins the filter extracts name of the user from the POP3 command **USER username** and saves it for the whole session period. If authentication on the POP3 server is successful, filter passes messages from server to client. All the commands and data are passed unmodified, except for the server reply to the RETR command.

Server reply to the RETR command is passed to the drweb-maild for processing - and only after that it is passed to the user.

Please note, that when the POP3 filter blocks some message according to the settings of any plug-in, and redirect action should be applied to this message - redirect will not be performed. In current version of **Dr.Web for UNIX mail servers** POP3 filter cannot send messages to the **Sender** component, so all the generated messages will not be sent.

To set up interaction of the POP3 filter with the current MTA you should uncomment the following line in the maild_MTA.mmc file:

```
drweb-pop3 local:/var/drweb/ipc/.agent 15 30
  MAIL drweb:drweb
```

[IMAP] Section

Dr.Web MailD can work with IMAP servers (caching is supported) via the protocol filter program. The IMAP filter program is a proxy-server connecting `drweb-maild` with the IMAP server. It filters e-mails sent to a user by the server. The IMAP server can be run on a local or on a remote computer.

In `[IMAP]` section setting for `drweb-imap` module are collected.

ServerAddress = `{socket address}`	Address used by the filter to connect to the IMAP server. Default value: **ServerAddress** = `inet:imap@127.0.0.1`
ListenAddress = `{socket address}`	A list of socket addresses used to receive requests from clients. The following types of addresses can be specified: `inet:..` or `inet-ssl:..` (if you use TLS/SSL). The latter type requires IMAPS protocol to be used by the IMAP filter. Default value: **ListenAddress** = `inet:5200@0.0.0.0`
ServerTLSSettings = `{TLS/SSL settings}`	TLS/SSL settings for server communications over IMAP. Settings are separated by commas. The server is used only if a certificate and private key (`private_key_file`) are specified and `inet-ssl` socket is used. A detailed description of available parameters can be found in the general configuration file description. **Example**:

	ServerTLSSettings = use_sslv2 no, private_key_file /path/to/pkey, certificate /path/to/certificate Please note, that the user which privileges are used by the IMAP filter (usually, drweb user), must have read access to the file with the certificate. In current version of the program SSL sessions cannot be cached.
	Default value: **ServerTLSSettings** =
ClientTLSSettings = {TLS/SSL settings}	TLS/SSL settings for client communications over IMAP. Settings are separated by commas. A detailed description of available parameters can be found in the general configuration file description. **Example**: **ClientTLSSettings** = use_sslv2 no, private_key_file /path/to/pkey, certificate /path/to/certificate Please note, that the user which privileges are used by the IMAP filter (usually, drweb user), must have read access to the file with the certificate. In current version of the program SSL sessions cannot be cached.
	Default value: **ClientTLSSettings** =
IoTimeout = {time}	Timeout for all input-output operations with the client's socket, when operation is in progress.

	Default value: **IoTimeout** = 60s
ProcessingTimeout = {time}	Timeout for the drweb-maild to process messages. Default value: **ProcessingTimeout** = 60s
MinFilterToMaildConnections = {numerical value}	Minimum number of connections between IMAP filter and drweb-maild. Default value: **MinFilterToMaildConnections** = 2
MaxFilterToMaildConnections = {numerical value}	Maximum number of connections between IMAP filter and drweb-maild module. When the value is set to 0, number of connections is not limited. Default value: **MaxFilterToMaildConnections** = 0
FilterToMaildKeepAliveTime = {time}	Maximum retention time for inactive connections between IMAP filter and drweb-maild.

	To interact with the `drweb-maild`, IMAP filter maintains several connections with it, and each connection can handle one operation. If there are no more available connections, then new connections are created until their amount meets the threshold specified in a value of **MaxFilterToMaildConnection** parameter. After connections stay inactive for a period of time specified in **FilterToMaildKeepAliveTime** parameter, they will be closed. Total amount of opened connections cannot be less than the value of **MinFilterToMaildConnections** parameter. Default value: **FilterToMaildKeepAliveTime** = 60s
CallbackPoolOptions = {pool settings}	Settings of auxiliary threads pool . Threads handle signals about finished message processing from the `drweb-maild`. Default value: **CallbackPoolOptions** = auto
PoolOptions = {pool settings}	Settings of main threads pool . These threads handle connections from clients. Each connection requires a new thread, otherwise some clients will be waiting for a free thread and stay disconnected. Default value: **PoolOptions** = auto
MaxConnections = {numerical value}	Maximum number of incoming connections. If 0 is specified as a value of this parameter the number of incoming connections is not limited. Default value:

	MaxConnections = 0
MaxConnectionsPerIp = {numerical value}	Maximum number of simultaneous connections from one IP address. If 0 is specified as a value of this parameter the number of incoming connections is not limited. Default value: **MaxConnectionsPerIp** = 0
DisablePlainText = {Yes | No}	Do not allow the client to send login and password as plain text. It requires OpenSSL to be configured in advance. Default value: **DisablePlainText** = no
DoS_Blackhole = {Yes | No}	If there are too many simultaneous connections from one IP address, drop them immediately without sending any error messages to client. Default value: **DoS_Blackhole** = no
MaxCommandLength = {size}	Maximum size of a command for the IMAP protocol. Each command is a string, which is sent from the client to the server. Maximum possible size of this command is about 1000 bytes according to the current RFC. Please note, that if the value of this parameter is left very small - up to 10 bytes (or even set to zero), then clients' commands will not be processed. Default value: **MaxCommandLength** = 1000b

MaxCachedHeadersPerMail = {size}	Maximum amount of memory to be allocated to store frequently used headers. IMAP filter caches main message headers in random access memory to speed up access to them. If 0 is specified as a value of this parameter, then amount of allocated memory is not controlled. Default value: **MaxCachedHeadersPerMail** = 64k
MaxLettersPerUser = {numerical value}	Maximum number of messages to be cached during one session. IMAP filter maintains cache of checked messages because IMAP protocol allows the client to perform lots of partial requests to one message. In most cases requests are performed sequentially, but if the user calls on several records, it will be necessary to cache more than one message. If the value of this parameter is set to 0 (it is strongly NOT recommended), the number of cached messages is considered unlimited. Default value: **MaxLettersPerUser** = 6
MaxDiskPerUSer = {size}	Maximum amount of disk space to be occupied by cached messages. Default value: **MaxDiskPerUSer** = 10m
OnFilterErrors = {actions}	Action to be applied to the message, when some error emerges before the message is passed to the drweb-maild module. Possible values are reject or pass. Default value: **OnFilterErrors** = reject

IMAP filter caches main message headers in RAM memory to speedup access to them. In theory it is possible to exhaust all the available memory and slow down operation of the IMAP filter by flooding it with messages containing large number of headers.

IMAP filter has a special anti-flooding parameter **MaxCachedHeadersPerMail**, controlling maximum total size of cached headers. Please, note, that if the value of this parameter is too small, there may be difficulties in displaying names and types of MIME attachments.

To set up interaction of the IMAP filter with the current MTA you should uncomment the following line in the `maild_MTA.mmc` file:

```
drweb-imap local:/var/drweb/ipc/.agent 15 30
  MAIL drweb:drweb
```

[Rules] Section

[`Rules`] section contains rules for message processing. Rules allow to change parameters of **Dr.Web MailD** operation flexibly depending on user needs. Rules help to specify program reaction on messages with certain set of elements and change message processing procedure according to them. Among these elements are sender and recipient addresses, detected malicious objects, some additional characteristics (e.g. sender IP-address or message size). Rules are applied sequentially from the first specified to the last.

Each rule consists of two parts:

```
CONDITION stop|cont [SETTINGS]
```

where `CONDITION` is a condition which must be true to enable application of settings specified in `SETTINGS` part of the rule. Using `CONDITION` you can also download `SETTINGS` from some external source (`ldap`, `mysql`, `etc`.) if no `SETTINGS` are specified in Rules.

Rule conditions

Each condition consists of two parts:
`[prefix_name:][value]`

where `prefix_name` is parameter name, and `value` - is parameter value.

The following parameter names are allowed:

- **any** - sender or recipient; parameter value - `lookup`.
- **from** (**sender**) - sender; parameter value - `lookup`.
- **to** (**rcpt**) - recipient; parameter value - `lookup`.
- **block** - blocking object (viruses or other malicious objects); parameter value - `lookup`.
- **client-ip** - sender IP-address (if **Receiver** component is adjusted to prompt for information about sender IP-address); parameter value - list of protected networks.
- **client-port** - sender port number (if **Receiver** component is adjusted to prompt for information about sender port number); parameter value - port number.
- **server-unix-socket** - absolute path to UNIX socket used to receive connection (if **Receiver** component is adjusted to prompt for information about socket address); parameter value - path to UNIX socket.
- **server-ip** - IP-address of interface used by **Receiver** to receive a letter (if **Receiver** component is adjusted to prompt for information about interface IP-address); parameter value - list of protected networks.
- **server-port** - port number of a server, used to receive connection (if **Receiver** component is adjusted to prompt for information about server port number); parameter value - port number.
- **id** - unique identifier of **Receiver** that received a specific letter (if **Receiver** component is adjusted to prompt for information about ID); parameter value - string with **Receiver** identifier.
- **auth** - information about successful authorization of a sender (if **Receiver** component is adjusted to prompt for information about sender authorisation); parameter value - not specified.

- **size** - message size. Before the size value you can specify a comparison method {!= | == | < | > | <= | >=}. If comparison method is not specified, ≤ (less or equal to) method is used by default. When you use comparison methods, place corresponding values in quotation marks, because some service characters ("!" and "=") are used in comparison method syntax.

  Example:

  ```
  "size:>=10m" cont scan=no
  ```

  shows that all messages greater than 10 MB must be excluded from scan. Usage of quotation marks is obligatory in this case.

- **md-client** – unique identifier of the Client which receives its settings from Rules. Before srart **Dr.Web MailD** searches Rules for settings for every active Client.

  Example:

  ```
  "md-client:client1"
  ```

  this condition will be true, if the message is received for the `client1` Client.

- **score** – message score. Before score value comparison operator can be specified: `{!= | == | < | > | <= | >=}`. If these symbols are not specified, then default group of comparison operatorss ≤ is used. Values with comparison operators must be enclosed in quotation marks, because some service characters are used in them: "`!`" and "=".

If parameter name is not specified then **any** is used by default. If parameter value contains white spaces or "`|&)(!=`" symbols, it must be enclosed with quotation marks. To specify separate quote (""") symbol within quotation marks, put back slash ("\") before it.

Also, instead of `[prefix_name:][value]` expression, special key words may be used: `true` or `false` - which always have positive or negative value respectively.

Example:

```
true cont some_settings
```

This settings will be applied all the time (if this rule was processed during check).

Separate conditions can be combined by brackets and logical operators AND (&&), OR (||), NOT (!) (in brackets alternative syntax is specified).

Examples:

```
sender:test && "size:>=10k"
```

This `CONDITION` will be true if the sender of the message is "`test`" and message size is greater than 10 Kbytes.

```
!("rcpt:ldap:///??sub?(mail=$s)" OR auth:)
```

This `CONDITION` will be true if at least one recipient is not found in the "`mail`" fild of `ldap` and the sender is not authorized.

When message is processed by plug-in, plug-in can ask **Dr.Web MailD** for certain parameter value. In this case **Dr.Web MailD** checks the message on compliance with conditions specified in rules. The following algorithm is used:

1. First, each recipient is checked on presence in the database. If it exists in the database, then search for a parameter value is performed in settings stored in the database for this recipient and all the recipient's groups.
2. If no match is found in the database, search for parameter values is performed in rules specified in configuration file. Rules are checked from the top downwards, in order they were specified. First, `CONDITION` is checked: if the match is found, a search for corresponding parameter value is performed in `SETTINGS` part of a rule.
3. If required parameter value is not found, and `CONDITION` is followed by "`stop`", search proceeds in section with default parameter values of **Dr.Web MailD**. If required parameter value is not found, and `CONDITION` is followed by "`cont`", further check on compliance with other conditions is

performed.

`Stop` allows to reduce total search time for required parameter values when it is known that there is no match amongst other conditions.

Example:

```
rcpt (sender, any): [address or regular expression] stop|cont [settings]
```

This rule allows to specify some settings for certain user.

Rule settings

`SETTINGS` is a set of **Dr.Web MailD** parameters with certain values:

```
[plug-in_name/]param1 = value1, [plug-in_name/]param2 = value2 ...
```

where **`paramN`** - parameter name, and `valueN` - parameter value. If parameter is used by plug-in, then plug-in name must be specified before parameter name with slash symbol between them.

SETTINGS are processed only when the certain parameter value is required. So, errors in parameters' values may be detected only when the program is launched and used. To detect these errors before startup you may use check-only mode (with the corresponding command line parameter `--check-only`). When parameters' values are saved to the database their validity is checked immediately, and rules with invalid parameters are blocked at the attempt to add them to the database.

Example:

```
sender:a@drweb.com cont headersfilter/Action = pass, vaderetro/max_size = 100k
```

in this case for sender a@drweb.com **`Action`** = `pass` is specified

for `headersfilter` plug-in, and maximum message size (**`max_size`**) is set to 100 Kbytes for `vaderetro` plug-in.

If comma is used in **`valueN`**, backslash symbol "\" must be specified before it.

Example:

```
to:a@drweb.com cont drweb/ProcessingErrors = pass\, redirect(err@drweb.com)
```

You cannot put `pass, redirect(err@drweb.com)` value in quotation marks, because in this case parser treats this part of a string as a single value and does not parse it to substrings while processing of **`ProcessingErrors`** parameter.

If you do not specify parameters in `SETTINGS` section, they are requested directly from server with lookup command in `CONDITION` section. It can be useful when you're working with LDAP:

```
to:regex:.*@drweb.com && "ldap:///-drwebRules-sub-(mail=$s)" cont
```

In this example, if message recipient is from `drweb.com` domain and message's sender or all recipients comply with ldap condition "`mail=$s`", parameters from `drwebRules` field are used. Parameters are uploaded for each new message and then are stored in cache. It allows user to change settings without server restart. Lookup with LDAP is enclosed in quotation marks because of brackets.

If the rule string appears to be longer than line, the backslash symbol "\" is used at the end of the line and rule continues on the next line.

Processing of the rule is always performed from top to bottom, and from left to right. Therefore new parameters block old parameters. For example, if you specify **`html`**=`yes`,**`html`**=`no`, the last value (**`html`**=`no`) is set.

This algorithm is used almost for all the parameters in rules except for the few ones with the different semantics. When the rules are processed, each new value of the certain parameter from the group of exceptions is added to the previously found value, and search continues through all the rules in database and configuration file. At the end of the search all the found values are combined.

The following parameters are among the exceptions:

- **LocalRules** in the Dr.Web Modifier plug-in;
- **AcceptCondition** in the headersfilter plug-in;
- **RejectCondition** in the headersfilter plug-in;
- **AcceptPartCondition** in the headersfilter plug-in;
- **RejectPartCondition** in the headersfilter plug-in;
- **MissingHeader** in the headersfilter plug-in;
- **WhiteList** in the VadeRetro plug-in;
- **BlackList** in the VadeRetro plug-in;
- **RegexsForCheckedFilename** in the drweb plug-in.

All the differences in semantics are explicitly specified in descriptions of parameters.

Please note, that if all the address-, user- or domain-specific parameters are stored in the database, they should be specified in a single line.

Example:

Address	Rules
test1@drweb.com	VadeRetro/SubjectPrefix = \"spam\", modifier/localrules=select message\, append_text "Some Text"
test2@drweb.com	headersfilter/MissingHeader = Date, headersfilter/MissingHeader =From, headersfilter/MissingHeader = To

stop directive for these parameters is processed as usual: it stops the search and returns the cumulative value of the certain

parameter.

Example:

Let us assume that we have a database set up via ODBC, and it looks like the following:

Address	Rules
test@drweb.com	modifier/LocalRules = select message\, append_text "Scanned 3333!

Also the following rule is set in configuration file:

```
true cont modifier/LocalRules = select message\, append_text "Scanned 44444 - global rules!", modifier/LocalRules = quarantine
```

And the following set of rules is set in the database for the certain user:

```
> email-info test@drweb.com
  test@drweb.com A=1 S=1
name:
  aliases: alias_test@drweb.com
  groups: divine good evil
rules:
1: true cont modifier/LocalRules = select message\, append_text "Scanned!", modifier/LocalRules = quarantine
2: true cont modifier/LocalRules = select message\, append_text "Scanned 2222!"
3: "rcpt:odbc:select rules from maild where a='$s'" cont
custom:
```

Then for a message to the `test@drweb.com`, processed by the Dr.Web Modifier plug-in, the following values of **`LocalRules`**

parameter will be used:

```
select message, append_text "Scanned!", quarantine, select message, append_text "Scanned 2222!", select message, append_text "Scanned 3333!", select message, append_text "Scanned 44444 - global rules!", quarantine
```

Please take into consideration the specific order of these values: at first, values are taken from the database, after that - from the configuration file.

If some more recipients are specified for the message, and some other values of `modifier/`**`LocalRules`** parameter are specified for them in the database (or not specified at all), then all these values from the database will be ignored, and the following global value will be applied:

```
select message, append_text "Scanned 44444 - global rules!", quarantine
```

If any errors are found in rule string (in all cases except processing of lookups), they are output to log file, and the rule itself is ignored. Please note, that not all parts of the rule are processed at the same time - lookups values and values of certain variables are processed only before immediate use, so all possible errors in these values can emerge at the very moment of processing the mail message. To check configuration for all possible errors please run the `drweb-maild` component with the `--check-only` command line parameter.

[Rule] sections

If required, you can define frequently used groups of parameters in special custom sections to be used by different rules. Custom sections are defined as follows:

```
[Rule: <section name>]
```

where `<section name>` is a unique section name that may contain Latin letters, numbers and white spaces and is not case-sensitive. Each parameter is specified in a separate line. The end of the user section can be marked either by the beginning of the next

section or by the end of the configuration file.

Please note, that `[Rule]` section does not define any rules by itself, it contains only settings to be applied by actual rules defined in `[Rules]` section. Parameters defined in this section may be invoked in other rules by using `rule=<section name>`. Current version of **Dr.Web MailD** does not allow more than one rule statement in a single rule, but number of possible custom sections is not limited.

Configuration file contains a special section of user parameters - section of default parameters. It is named `default` and the `Rule` keyword in the header of the section can be skipped. In default section default values for all parameters used in rules are specified. You can use `rule=default` to apply default settings by the rule.

Example:

This lines declare custom section MySection that defines two parameters (block reports and disable moving to quarantine) to be applied in other rules:

```
[Rule:MySection]
quarantine = no
notify = block
```

The following two rules make use of this custom section to set `quarantine` and `notify` parameters accordingly:

```
[Rules]
Rcpt:regex:example\.com cont rule=MySection
Sender:lol@foo.com && block:vir1 cont notify.
Skip=allow, notify.Virus=allow, rule=MySection
```

Once these rules are defined, reports and moving to quarantine will be disabled for messages with recipient belonging to domain example.com. If message has been sent from lol@foo.com, and blocking object vir1 is found, only reports on detected viruses will be sent and moving files to quarantine will be disabled.

Parameters that can be defined in rules

Following types of parameters are available:

- parameters specified in rules only;
- parameters specified in configuration files of other modules;
- parameters used only by **Clients** and specified with one mandatory condition: **Client**'s unique identifier (`md-client`).

For each parameter the possibility to use it in rules is specified in the configuration files of other modules and must be denoted in parameter description in documentation.

In [`rules`] the following parameters can be set:

`html` = {`Yes` | `No`}	`Yes` instructs **Dr.Web MailD** to generate notification in html-format; otherwise reports are generated in plain text format.
	Default value: **`html`** = `Yes`
`quarantine` = {`Yes` | `No`}	`Yes` instructs to move the message to quarantine.
	Default value: **`quarantine`** = `Yes`
`scan` = {list of plug-ins}	This parameter indicates which **Dr.Web MailD** plug-ins should be used for message scan. Plug-in names are delimited by colon. When parameter value is set to `All`, all plug-ins check the message. With `No` set as parameter value no plug-ins is used.

	To separate the plug-in names colon ":" is used. To exclude any plug-in from the list, specify a minus "-" symbol in front of each of them. There is no white space between "-" symbol and plug-in name. Plug-in names without "-" symbol cannot be written after `All` value. **Examples**: **scan** = `all` – message is checked by all plug-ins; **scan** = `no` – no plug-ins are used; **scan** = `all:-foo` – message is checked by all plug-ins except `foo`; **scan** = `Foo:Bar` – message is checked only by `foo` and `bar` plug-ins; **scan** = `all:foo` – wrong parameter format, because you cannot specify plug-in names without "-" symbols after `All` parameter value; **scan** = `-foo:all` – wrong parameter format, because `All` parameter value must be set at the very beginning of the string; **scan** = `-foo` – wrong parameter format, because one cannot specify plug-in names with "-" symbol, if `All` value is missing.
	Default value: **scan** = `All`
notify[.{notification type}] ={allow|block}[({address types})][condition]	This parameter controls output of notifications of different types. `allow` parameter value enables output of corresponding notification, and `block` value - disables it. If notification type is not specified, this parameter value is applied to all notifications.

	Possible notification types depend on types of reports supported by `drweb-notifier` module. Additional plug-ins can add their own notification types. By default, the following notification types are supported: • **`notify.Virus`** - notifications on viruses detected in a message; • **`notify.Cured`** - notifications on viruses cured in a message; • **`notify.Skip`** - notifications on skipped messages; • **`notify.Archive`** - notifications on messages, which have not been checked due to restrictions for archives; • **`notify.Error`** - notifications on errors occurred when checking messages; • **`notify.Rule`** - notifications on messages blocked by certain rules; • **`notify.License`** - notifications on messages skipped due to license limitations; • **`notify.Malware`** - notifications on detected malware. Parameter value can be followed by optional qualifier, specified in brackets, which indicates types of addresses for parameter to be applied. Several address types can be specified, delimited by colon. Possible qualifier values: • `sender` - notifications for a sender; • `rcpt` - notifications for a recipient; • `admin` - notifications for an administrator; • `any` (or no qualifier at all) - notifications for all address types.

	Examples:
	Notify=block or notify=block (any) - all types of notifications are blocked;
	notify.Virus = block (sender: admin) - notifications on detected viruses are blocked for administrator and sender.
	If no rule is found for the given type of notification, and no general rule is found either (with no type specified), then the corresponding notification is considered disabled.
	Values of **Notify** parameter are combined according to the same rules that are used for **LocalRules** parameter in **Dr.Web Modifier** plug-in.
	Example:
	`true cont notify.virus = allow (sender)`
	`true cont notify.virus = allow (admin), notify = block`
	these rules are similar to the following rule:
	`true cont notify.virus = allow (sender:admin), notify = block`
	According to this rule only notifications about viruses will be sent to the administrator and message sender, and all other notifications will be blocked.

	Default value: Default notification parameters: • **notify** = block(any) • **notify.Virus** = allow(any) • **notify.Cured** = allow(admin: sender)

	• **notify.Skip** = allow(sender) • **notify.Archive** = allow(admin:sender) • **notify.Error** = allow(admin:sender) • **notify.Rule** = allow(admin) • **notify.License** = allow(admin) • **notify.Malware** = allow(any)
plugin_name/max_size = {size}	This parameter allows to specify maximum size of the message for scam for each plug-in.
plugin_name/use = {Yes | No}	With Yes value specified certain plug-in is instructed to scan the message. With No value specified certain plug-in is instructed to skip the scan of the message.
PoolOptions = {pool settings} **NotificationNamesMap** = notify_name1 file_name1, notify_name2 file_name2 ...	This parameter allows to set the maximum size of the checked message for each plug-in module. **NotificationNamesMap** = name1 file_name1, name2 file_name2 ... It allows the mapping of the name of report to the new one. For example, it can be used to assign other reports depending on the envelope. Parameters: • nameN - name of the notification for which new file is created. List of notification names can be found above in the description of **notify** parameter. Also for general reports report name can be specified, and for DSN reports - dsn as well;

	• file_nameN - part of the new report file name for notification. The following rule is used in composing the full name. At the beginning of file name one of prefixes is added: sender_, rcpts_, admin_, report_ or dsn_ - and file extension is changed to .msg. As the result we get the name of the file sender_file_nameN.msg, to be looked up in directory specified in **TemplatesBaseDir** parameter from [Notifier] section. **Example**: **NotificationNamesMap** = virus my-virus, archive my-arch
SenderAddress = {address1|address2|...}	Address passed to **Sender** component for a message to be sent. Several addresses can be specified, delimited by "|" symbol. When **SenderAddress** parameter is used in rules of the following type: "to:mysql:select * from adr" cont SenderAddress = address1|address2|address3 message which meets the following criteria: "to:mysql:select * from adr" - is sent to the first available address from the list (i.e. if address1 is not available, then an attempt to send to address2 will be made; if address2 is also not available, then address3 will be used, etc). If **Sender** supports the given parameter, it sends the message directly to the specified address. At this time **SenderAddress** parameter is supported only by drweb-sender module using SMTP/LMTP method.

rule = {section name}	Applies parameters defined in custom section [Rule:section name] (see above).

Let's examine the case with several recipients in the message. The following parameters:

plugin_name/max_size

NotifyLangs

AdminMail

html

scan

plugin_name/use

and also all the parameters of the third type, which apply to the settings of the Clients, are processed for each recipient separately: i.e. if two recipients for some parameter have different values specified, then a copy of the message with different applicable settings is made for each of the recipients.

For other parameters, if CONDITION is executed for every recipient, then parameter value is used from this rule. Otherwise, the default value is used.

Validity of the rules in drweb-maild module via command line parameters can be checked via special interface. Various attributes of the hypothetic message are specified with the help of these parameters. Module outputs to console all settings from the rules that are applicable to this message. The following attributes are available:

- -s [--sender] arg - message sender (from an envelope);
- -r [--rcpt] arg - message recipient (from an envelope;
- -b [--block] arg - blocking object (e.g. name of a virus);
- --client-ip arg - IP-address of a sender;
- --server-ip arg - IP-address of the server of recipient;

- `--client-port arg` - sender port number;
- `--server-port arg` - port number of the server of recipient;
- `--server-us arg` - UNIX-socket of the server of recipient;
- `--id arg` - Receiver's unique identifier;
- `--auth` - message is received from authorized user;
- `--size arg` - message size (this value has {size} type);
- `--score arg` - message score;
- `--md-client arg` – unique identifier of the Client.

Example:

```
$ ./drweb-maild --auth
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG    notify* :
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG    all : block
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG       archive : from=allow; admin=allow;
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG    cured : from=allow;
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG        error : from=allow; admin=allow;
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG
Thu May 29 16:03:44 2009 [3081324208] maild.rules DEBUG    license : admin=allow;
Thu May 29 16:03:44 2009 [3081324208]
```

```
maild.rules DEBUG
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG       malware : from=allow;
to=allow; admin=allow;
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG    rule : admin=allow;
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG    skip : from=allow;
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG        virus : from=allow;
to=allow; admin=allow;
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG    scan : all
      Thu May 29 16:03:44 2009 [3081324208]
maild.rules DEBUG    html : 1
```

Lookups

Lookups is a common interface for looking up for objects and their values in external sources. Values are delimited by commas. A special prefix defining a type of lookup that can be put before the value:

```
[prefix1:]value1, [prefix2:]value2, ...
```

If prefix is not specified, then values are used directly.

Also, special sets of symbols may be specified as part of the query to the databases . They will be changed to the required elements before sending a query:

- `$s` – will be changed to the requested element. For

example, if some address is looked up, then $s wil be changed to this whole address (without angle brackets), if some domain is looked up - then $s wil be substituted with domain name.

- `$d` – if some address is looked up, then `$d` wil be substituted with the domain part of this address. Otherwise full request is inserted.
- `$u` – if some address is looked up, then `$u` wil be substituted with the username from this address. If parameter is looked up by domain, then an empty string will be inserted.
- `$$` – wil be substituted with single `$`.

The following prefixes can be used:

- `value` – a looked-up value itself is specified after this prefix. It can be useful, when the value contains : symbol, for examle.
- `file` - its value is a path to file. Each value in the file must be set on a separate line. In this case search is performed rather quickly, because assortment and binary search can be used.
- `regex` - its value is a regular expression (compatible with Perl regular expressions) - during check a substring is looked for, absolute matching is not required.
- `rfile` - its value is a path to file. The file contains a set of regular expressions (compatible with Perl regular expressions), and each of them must be set on a separate line. During check a substring is looked for, absolute matching is not required.
- `ldap` - its value is a path to LDAP-server. It is set in the following format:

  `[`**`param1`**`=val1|`**`param2`**`=val2|...|] ldap_url`

  where `ldap_url` - is an URL of LDAP query, and **`param1`**, **`param2`**, etc. are local parameters from [LDAP] section of **Dr.Web MailD** configuration file for the given lookup. From this section only those parameters can be specified, for which it is explicitly stated that this is possible.

  LDAP URL looks like the following:

```
ldap://hostport/dn[?attrs[?scope[?filter[?exts]]]]
```

where:

- `hostport` – is a host name with an optional ":portnumber";
- `dn` – is the search database;
- `attrs` – is a comma separated list of attributes to request;
- `scope` – is one of these three strings: `base`, `one`, `sub`;
- `filter` – is filter name;
- `exts` – are recognized set of LDAP and/or API extensions.

Example:

```
ldap://ldap.example.net/dc=example,dc=net?cn,sn?sub?(cn=*)
```

Dr.Web MailD works with LDAP via OpenLDAP library (and it must be not earlier than v. 2.0).

- `odbc`, `oracle` - its value is an SQL-query to the ODBC or Oracle database. It can be made in the following format:

  `[`**`param1`**`=val1|`**`param2`**`=val2|...|] sql_request`

  where **`param1`**, **`param2`**, etc. are local parameters from [`ODBC`] and [`Oracle`] sections of **Dr.Web MailD** configuration file for ODBC lookup and Oracle lookup correspondingly. In this section only those parameters can be specified, for which it is explicitly stated that it is possible. The same special symbols as in `ldap` lookups may be used.

  New DSN settings are applied only after restart of the software (connections are not re-initialized after `SIGHUP` signal is received).

  Dr.Web MailD works with ODBC via any library which supports ODBC version 3.0 or later. This library must be

compiled with the support of threads. It is recommended to use `UnixODBC 2.0` or later version.

Dr.Web MailD works with Oracle via `libclntsh` library, which supports version `OTLv8` or later and is supplied with Oracle client.

To connect to Oracle you may need to specify user name, password and the name of connection as the value of **`ConnectionString`** parameter (**`ConnectionString`** = `user/password@connectionname`).

Name of the connection can be set in two different ways:

1. If Dr.Web MailD is installed to the same computer Oracle does, then environment variable `ORACLE_HOME` must be set first for **Dr.Web MailD** according to Oracle documentation. After that you must specify one of TNS names as a connection name in `$ORACLE_HOME/network/admin/tnsnames.ora` file.
2. You can also copy the description of setup (without line breaks) directly from `$ORACLE_HOME/network/admin/tnsnames.ora`.

Example:

`tnsnames.ora` file:

```
CONNECTIONNAME =
  (DESCRIPTION =
     (ADDRESS = (PROTOCOL = TCP)(HOST =
localhost)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = CONNECTIONNAME)
    )
  )
```

So you can specify as a connection string the following:
`user/password@ CONNECTIONNAME`

or:

```
user/pasword@(DESCRIPTION = (ADDRESS = (PROTOCOL = TCP)(HOST = localhost)(PORT = 1521))(CONNECT_DATA = (SERVER = DEDICATED)(SERVICE_NAME = CONNECTIONNAME)))
```

- `postgres` - its value is an SQL-query to the PostgreSQL database. The syntax is the same as in requests to ODBC.
- `cdb` - its value is an alphanumerical name of the key in database. CDB database itself is a read-only storage of pairs `[alphanumerical key]:[alphanumerical value]`. You may use `tinycdb` package to create a database file. To use this lookup specify the list of files of CDB database. Each file is represented as a single table, which name is the same as file name (without specifying the full path to `file: /path/to/table.cdb --> table.cdb`).

  Use the following commands to read an entry from the table:

  ```
  select * from table.cdb where key='123'
  select * from table.cdb where key='$'
  ```

  CDB database does not support SQL query input language, that is why driver emulates to SQL the single command for operating with lookups:

  ```
  select * from @tablename where key='@string'
  ```

  where `@tablename` must be changed to the name of any file specified in `[CDB]` section of **Dr.Web MailD** configuration file as source items. In `cdb` lookups the same special symbols as in `ldap` lookups may be used.

  Example:

  ```
  cdb:skipdomains=regex:^inbox|select * from my_file where key='$s'
  ```

- `berkeley` - enables interaction with Berkeley DB. Format of

the query is similar to that of `cdb` prefix. During setup a symbolic link `/usr/lib/libdb.so` is created, pointing out to the current library. If no symbolic link is created, you must specify the correct library version (i.e. `/usr/lib/libdb-4.5.so`). This lookup can work only with libraries v. 4.3-4.6. The same special symbols as in `ldap` lookups may be used.

- `firebird` - its value is an SQL-query to the Firebird database. The syntax is the same as in requests to ODBC. To connect to the database specify address of the database server in the Host field:

```
Host=somehost              #somehost:3050
Host=somehost/1234         #somehost:1234
```

- `sqlite` - its value is an SQL-query to the SQLite database. The syntax is the same as in requests to ODBC. This lookup must be used only with SQLite databases v. 3.x.

  `SQLite` has a specific feature: each time any program writes to the database, the file becomes blocked for a period of time required for a program to make a record. So as far as several programs can work with single SQLite database, it is possible that some day a recording process will not be able to get exclusive access to database for a certain amount of time (specified in **`BusyTimeout`** parameter from `[SQLite]` section of **Dr.Web MailD** configuration file). Then the recording process aborts with the following error message: `"Database is locked"`.

  You must avoid to use GUIs for the SQLite database, because they may block the database in reserve. If some third-party process has blocked the database for a long time, or if the **Dr.Web MailD** itself is set up to output different types of statistics to one file with a tiny timeout, some errors during the export of statistics may occur.

  In `sqlite` lookups the same special symbols as in `ldap` lookups may be used.

  If SQLite database, which is used to store values for a number of parameters, had been unavailable for some period of time, but then connection to it was restored - a `HUP`

signal must be sent to the `drweb-maild` module to reinitialize its connection to the SQLite.

- `mysql` - its value is an SQL-query to the MySQL database. The syntax is the same as in requests to ODBC.

After the prefix a list of arbitrary local parameters can be specified for each lookup in the following format:

```
NAME1 = VALUE1 | NAME2 = VALUE2 | ... |
```

where:

- `NAMEM` – case-insensitive parameter name;
- `VALUEN` – parameter value.

The following local parameters can be used:

- **`SkipDomains`** – the list of domains which can be skipped from lookup requests. Detailed description of this parameter can be found in lookup settings in configuration file.
- **`OnError`** = `{ignore|exception}` – sets an error handling procedure for lookups.

  "`ignore`" value is used by default. It instructs to ignore the error and only output information about it to log.

  "exception" value instructs to generate an exception, which will be processed as a common error (e.g. using the **`ProcessingError`** parameter). This value is used when the corresponding safety policy is implemented in the company, and each message must be processed despite all the possible errors in any other components (e.g. in databases used by lookups).

`OnError` parameter can be specified in every lookups section (LDAP, ODBC, ...), so that corresponding actions will be applied to errors emerging during processing of lookups of those types.

Please note, that this error handling procedure with **`OnError`** parameter can be used only when the search for parameter values is performed, but not when the **Dr.Web MailD** is initialized. So, when some lookups cannot be properly processed at startup, such error is considered fatal, regardless of the **`OnError`** parameter value for those lookups.

Examples of Lookups Usage

With this query all messages from domain found in `domain` column of `maild` table in ODBC storage will be marked as belonged to the protected domain:

```
ProtectedDomains = "odbc:select domain from maild where domain='\$s'"
```

With this query the following addresses will be marked as protected: all the addresses from `%etc_dir/email.ini`, localhost address and all addresses found after ldap query `ldaps:///??sub?(mail=$s)`, excluding addresses with `fake.com`:

```
ProtectedEmails = file:%etc_dir/email.ini, localhost, ldap:skipdomains=regex:.*fake.com$|ldaps:///--sub-(mail=$s)
```

With this query the check is performed whether the certain address is listed in `email` column of the `maild` table in MySQL database. If the address is found, the message must be dispatched to the address found in `routerinfo` column, otherwise it is sent to all recipients with `foo` in the address:

```
Router = mysql:select routerinfo from maild where email='\$s', foo inet:234@foo.ru
```

Lookups can be used in rules.

This query allows to receive `rules` field for all `mail` ldap-fields containing receiver address. `rules` field contains settings to be applied to this recipient. Please note that all `CONDITIONs` must be enclosed in quotation marks, because of some special symbols are used (e.g. round brackets).

```
"rcpt:ldap:///-rules-sub-(mail=$s)" cont
```

For example, if you write:

```
rcpt:"ldap:///-rules-sub-(mail=$s)" cont,
```

then a compilation error occurs:

```
Mon Jun 29 18:53:01 2009 [3081262768] maild.rules ERROR '(' can not follow' "ldap:///-rules-sub-'
Mon Jun 29 18:53:01 2009 [3081262768] maild.rules ERROR error in parse condition: 'rcpt:"ldap:///-rules-sub-(mail=$s)" cont'
```

This query enables checking of the addresses. If sender or recipient addresses are contained in `skipaddr` field of `domain` table of SQLite database, then `drweb` plug-in is not used for them:

```
"any:sqlite:select skipaddr from domain where skipaddr = '$s'" cont scan=all:-drweb
```

Limitations on Usage of Lookups

There exist some limitations on usage of certain types of `lookups`.

`LookupsLite` value type is similar to `lookups` with one minor difference: only values themselves or `file` type of `lookups` can be used in this type. `LookupsLite` is used:

- in settings of lookups (e.g. **`SkipDomains`** parameter for each lookup);
- in settings of plug-ins.

At any attempt to specify a forbidden lookup, the following message is output to log:

```
Wed Jun 10 14:02:20 2009 [4160149200] Modifier ERROR Error in init lookup [cdb:select * from /root/mail/base_file_for_CDB.txt where key='domain']: can't use this lookup here
```

Validation of Lookups

To check the validity of lookups you can use a special `drweb-lookup` utility. Validation request can be made by the following command:

```
$ %bin_dir/drweb-lookup [parameters] query
```

where `query` stands for different types of lookups, where search is performed, and `parameters` stand for command line parameters.

The following parameters are available:

- `-h [ --help ]` - display help on command line parameters;
- `-v [ --version ]` - display current version;
- `-l [ --level ] arg` - log verbosity level;
- `--syslogfacility arg` - syslog facility;
- `-i [ --ipc-level ] arg` - IPC log verbosity level;
- `--log-filename arg` - log file name;
- `-a [ --agent ] arg` - path to **Agent** to receive additional configuration for lookups;
- `-t [ --timeout ] arg` - timeout for receiving configuration from **Agent**;
- `-q [ --query ] arg` - query string, which value we are looking for. If "-" is specified, then standard input must be used;
- `-e [ --exist ]` - only existence of the element in lookups will be checked without receipt of its value.

Examples:

```
$ ./drweb-lookup -q q -e e,w
q NOT FOUND

$ ./drweb-lookup -q q -e q,q
```

```
FOUND q

$ ./drweb-lookup -q test@drweb.com -e
'ldap:///-displayName-sub-(mail=$s)'
FOUND test@drweb.com

$ ./drweb-lookup -q test@drweb.com
'ldap:///-displayName-sub-(mail=$s)'notify.
virus=block, notify.virus=allow(rcpt), drweb/
ProcessingErrors = pass

$ ./drweb-lookup -q test@drweb.com "odbc:
select rules from maild where a='\$s'" scan =
all:-drweb
```

Statistics

During operation of the **Dr.Web MailD** two types of statistical information can be collected: general statistics and statistics on blocked messages.

General statistics contains information about general performance of the **Dr.Web** software: number and size of checked messages, number of spam messages, etc. Statistics on blocked messages contains information on certain blocked messages and their malicious content.

All statistics is saved to the internal database. General statistics is collected in the internal cache and every 5 minutes is saved to the DB. If `drweb-maild` module fails, some part of statistical information may be lost. Statistics on blocked messages is saved directly to the DB and can be exported if necessary. Detailed information about statistics can be received using special commands of the control socket.

There exist several log verbosity levels, which can be set via the **`Detail`** parameter from the [Stat] section and **`StatDetail`**

parameter, which can be specified in rules for each Client.

- `Stat off` value disables collection of statistics, which allows to increase performance of the **Dr.Web for UNIX mail servers**. As a result there will be no statistics for export and no reports to send.
- `Stat low` value enables collection of statistics on operation of the whole software complex. Collection of statistics on Clients operation is also enabled. As a result, there will be possible to export statistics and send reports.
- `Stat medium` value allows to collect statistics on groups, which have not yet disabled this function in their settings. Access to group statistics can be gained either via the control socket or via the web interface.
- `Stat high` value allows to collect statistics on all users listed in the internal database, which have not yet disabled this function in their settings. Access to user statistics can be gained either via the control socket or via the web interface.

Exporting Statistics

There exists a possibility to perform export of statistics not only via the **Agent**, but also by means of **Dr.Web MailD**, via storage type. Both this options can be enabled simultaneously.

Please note, that export of statistics via the **Agent** is disabled by default and can be used only for general statistics on all the Clients. Export via storage type must be adjusted for each Client separately and for a Super-Administrator as well.

When the first option is used, all statistics is transferred to the **Agent**, which either sends it to **Doctor Web** statistics server (**`StatisticServerHost`**, **`StatisticServerPort`** and **`UUID`** parameters in `[StandaloneMode]` section of **Agent**'s configuration file `%etc_dir/agent.conf`), or sends it to the central protection server (corresponding settings can be found in `[EnterpriseMode]` section of **Agent**'s configuration file `%etc_dir/agent.conf`).

With the second option statistics is exported by means of **Dr.Web MailD** to storage type objects. Storage syntax is similar to that of lookups with only few exceptions: prefixes are different and "$s" set of symbols is not used. The following types of prefixes are available:

- odbc – the syntax is the same as in requests to LDAP.

  In SQL-requests stored values can be specified in : name<type> format, where:

  - name - is the name of the saved object (for each parameter its own list of possible names is used);
  - type - is the type of the parameter to be used when storing a record to the database. For each object its default type is used, and it will be better not to change it.

  Default types:

  - varchar_long - similar to SQL_LONGVARCHAR in ODBC;
  - timestamp - similar to TIMESTAMP_STRUCT in ODBC;
  - int - 32-bit digital integer;
  - char(length) - string that ends with 0.
- oracle – the syntax is the same as in requests to ODBC.
- postgres, mysql, sqlite, firebird – the syntax is the same as in requests to ODBC, with one minor difference: char(length) type is not supported, and varchar_long type must be used for the line data.

Example:

```
ExportStatStorage = "odbc:insert into plugin_stat values(:plugin_name<varchar_long>,:size<int>,:num<int>)"
```

Please note, that in this request quotation marks are necessary, because commas are used in it.

To enable export of the statistics with `storage` type for all clients, first of all, set `Yes` as a value of **ExportStat** parameter in [Stat] section, then set the values for at least one of the following parameters in [Stat] section and set the commands for the statistics export:

- **ExportBlockObjectsStorage** – the list of objects for exporting statistics on the blocked messages;
- **ExportStatStorage** – the export of statistics on all messages, processed by **Dr.Web for UNIX mail servers**;
- **ExportPluginStatStorage** – export of the statistics on processed messages by each plug-in.

Detailed description of each parameter specified above can be found in chapter [Stat] Section.

To enable export of the statistics for each client separately, set **ExportStat**, **ExportBlockObjectsStorage**, **ExportStatStorage**, **ExportPluginStatStorage** parameters in the rules for each client separately similarly to the set parameters in [Stat] section.

Example:

```
[Rule:client1]
...
ExportStat = yes
      ExportBlockObjectsStorage = "odbc:insert
into client1_viruses values
       (:number<int>, :q_name<varchar_long>, :
virus_name<varchar_long>, \
                      :virus_code<int>,       :
plugin_name<varchar_long>,                    :
sender<varchar_long>,\
                 :client_ip<varchar_long>,    :
date<timestamp>)"
...
```

Quarantine

Mail messages are put to quarantine on request from any plug-in or from `drweb-maild` module itself. They are stored in `/quarantine/path/def/name/` directory, where `name` is the name of the module which has made a request.

When the message is put to quarantine, two files are created: one for the message body (its name is created according to the settings in **`FilenamesMode`** and **`FilenamesPrefix`** parameters, and all "_" symbols are replaced with "." symbols), one for the envelope.

Envelope is preserved in the following format:

- `int4_t` – length of sender address;
- `sN` – sender address;
- `int4_t` – number of recipients;
- `int4_t sN` – for each recipient, where `int4_t` is a 4-byte integer in network byte order.

If value of **`MoveAll`** parameter is set to `Yes`, then all mail processed by the **Dr.Web MailD** is saved to `/path/def/backup/` directory.

Besides saving the message body to the quarantine directory, the message itself is registered in the internal database, and the following information is saved to it: message envelope, saving time, reasons to save a message to quarantine, etc.

Quarantine can be effectively managed via the control socket. You can send, redirect, remove and perform search through messages in quarantine using the specific set of commands for control socket.

Maximum period of time for messages to be stored in quarantine can be set via the **`StoredTime`** parameter. Also you can limit the size of the quarantine (**`MaxSize`** parameter) and the number of messages in it (**`MaxNumber`** parameter).

If several restrictions are specified at once, all of them will be applied

simultaneously. **MaxSize** and **MaxNumber** restrictions are checked each time new message is saved to the quarantine. **StoredTime** restriction is checked periodically, and corresponding period is specified in **PulseTime** parameter.

drweb-qp utility deletes old messages from quarantine and moves them to the external DBI database. It works with Perl version 5.0 and later. Path to drweb-qp must be specified in **PathToDrwebQp** parameter. Initialization of drweb-qp is managed by **PulseTime** parameter. If value of **PulseTime** parameter is set to 0, then **StoredTime** restriction and drweb-qp utility will not be used.

Using DBI

It is possible to store quarantine messages not only in the file system, but also in DBI-storage. To use this feature you must have Perl version 5.0 or later, installed DBI and File::Temp modules and already configured DBI-storage. (The detailed description of setup and adjustment of DBI modules for operation with databases can be found in documentation on DBI.) To enable successful transfer of messages to the database, the database must be created with the use of SQL-ASCII symbol set.

Interaction with DBI is performed only via drweb-qp utility, path to which is specified in **PathToDrwebQp** parameter.

To use DBI you should do the following:

- set Yes as a value of **MoveToDBI** parameter and adjust **DBISettings**, **DBIUsername** and **DBIPassword** parameters correspondingly to enable access to the DBI-storage. These three parameters belong to the **DBI->connect** function. Detailed description of their syntax can be found in documentation on every DBI module (e.g. man DBD::mysql or man DBD::Pg);
- set up the following SQL commands:
    - **SQLInsertCommand** - this command adds a mail message to the DBI-storage.

- **SQLRemoveCommand** - this command removes a mail message from the DBI-storage. It is used when limitation is set on a storage time for messages in quarantine.
- **SQLSelectCommand** - this command provides access to the message kept in DBI-storage. It is used when message is requested (e.g. via control letter) from quarantine.

Possible problems:

If you have encountered an error of the following type:

```
maild ERROR Error in system call for [/opt/drweb/drweb-qp --Level debug --SyslogFacility Daemon --BaseDir /var/drweb/ --ProcessMail 1 --MoveToDBI 0 --StoredTime 86400 --SQLInsertCommand "" --MDClient "def" >/dev/null 2>&1 &]
```

try to increase the maximum available amount of memory for the `drweb-maild` process (e.g. using `ulimit -m` command).

Using Control Letters

Access to quarantine can be gained via special control letters. These letters in `Subject` field contain commands to be executed by `drweb-maild`. Letters must be sent to the address specified in **FilterMail** parameter from [`Notifier`] section of **Dr.Web MailD** configuration file or in local rules for this letter. ACL setup for control letters is performed by setting up value of **OnlyTrustedControlMails** parameter from [`Maild`] section of **Dr.Web MailD** configuration file.

A possibility to receive messages from quarantine is enabled by setting `Yes` as a value of **AccessByEmail** parameter from [`Quarantine`] section of **Dr.Web MailD** configuration file.

To receive a certain message from quarantine specify `q:relative_path_to_file` in the `Subject` field of the control

letter, where `relative_path_to_file` - is a relative path to file in quarantine (e.g. `/def/drweb/F/00014F8F.DW_SHOT_PRODUCT.QqFpdH`).

The message is requested from quarantine only if its sender or one of its recipients is the same as the sender of control letter.

The control letter can be automatically generated by MUA by following the corresponding link in report, sent by **Dr.Web MailD** notification service upon a transfer of message to quarantine.

Using drweb-qcontrol

To manage quarantine and to perform search through it, `drweb-qcontrol` utility is used. Its interface can be used both for messages kept in DBI-storage or in files on disc.

To get the list of messages to apply actions to, you may use unique identifiers: relative paths to files preserved in quarantine. In these identifiers special symbols can be used: "%" means zero or more arbitrary symbols, "_" means one arbitrary symbol. When identifiers are set, def/ must be specified in the beginning.

Available command line parameters:

- `-h [ --help ]` – use help.
- `--version` – view information on current program version.
- `-v [ --verbose ]` – information about all actions taken and results received is output to console.
- `-l [ --level ] arg (=error)` – log verbosity level used for logging of startup process. Possible values: `quiet`, `error`, `alert`, `info`, `debug`.
- `--syslogfacility arg (=mail)` – log type when syslogd system utility is used for logging. Possible values: `daemon`, `mail`, `local0...local7`.
- `-i [ --ipc-level ] arg (=error)` – IPC log verbosity level. Possible values: `quiet`, `error`, `alert`, `info`, `debug`.

- `--log-filename arg (=syslog)` – name of the log file or `syslog` value, if logging is performed by `syslogd` system utility.
- `--sendmail arg (=/opt/drweb/drweb-inject)` – path to the `drweb-inject` utility.
- `-s [ --socket ] arg (=local:/var/drweb/ipc/.ctl)` – path to **Dr.Web MailD** control socket.
- `--agent arg (=local:/var/drweb/ipc/.agent)` – use address of the **Dr.Web Control Agent** to receive configuration.
- `--timeout arg (=60)` – maximum period of time to wait for a receipt of configuration information from the **Dr.Web Control Agent**.

To get the list of messages to apply actions to, you may use unique identifiers: relative paths to files preserved in quarantine. In these identifiers special symbols can be used: "`%`" means zero or more arbitrary symbols, "_" means one arbitrary symbol. When identifiers are set, `def/` must be specified in the beginning.

Example:

`def/%00014F7F%` - selects from quarantine all messages with `00014F7F` number;

`def/drweb/%` - selects from quarantine all messages transferred by `drweb` plug-in.

Identifiers of files to be selected are received from search conditions or from the command line. Standard input stream (if no search criteria is specified or no identifier is set via command line) is also used for this purpose.

Possible actions:

- `--view` – view all messages with a certain identifier via program specified in `PAGER` environment variable. If no value is specified in `PAGER` variable, then cat program is used.
- `--send` – send all messages with a certain identifier to

original recipients. `drweb-inject` utility is used for dispatch.

- `--redirect [list_of_rcpts]` – send all messages with a certain identifier to the list of addressed. `drweb-inject` utility is used for dispatch.
- `--remove` – remove all messages with a certain identifier from quarantine.
- `--stat` – output statistics on messages with a certain identifier.

Example:

```
drweb-qcontrol --stat def/%
1. def/backup/B/00014F8B.DW_SHOT_PRODUCT.U0dshM
   from: a@1; to: a@fff; time: 2008-08-14
12:10:57
2. def/drweb/F/00014F8F.DW_SHOT_PRODUCT.QqFpdH
   from: a@4; to: a@fff; time: 2008-08-14
13:00:50
3. def/backup/C/00014F8C.DW_SHOT_PRODUCT.A39xp7
   from: a@2; to: a@fff; time: 2008-08-14
13:00:50
4. def/backup/F/00014F8F.DW_SHOT_PRODUCT.tMi6W2
   from: a@4; to: a@fff; time: 2008-08-14
13:00:50
5. def/drweb/3/00014F93.DW_SHOT_PRODUCT.n9xPjU
   from: a@3; to: a@fff; time: 2008-08-14
13:30:49
6. def/backup/3/00014F93.DW_SHOT_PRODUCT.ewYFVA
   from: a@3; to: a@fff; time: 2008-08-14
13:30:49
7. def/backup/4/00014F94.DW_SHOT_PRODUCT.JQ3sLH
   from: a@3; to: a@fff; time: 2008-08-14
13:30:49
```

Actions will be performed in the order they are specified (i.e. in one command several actions can be specified).

Example:

```
drweb-qcontrol --send --remove def/backup/F/00014F7F.DW_SHOT_PRODUCT.yv4ro9
```

This command will send a message with `def/backup/F/00014F7F.DW_SHOT_PRODUCT.yv4ro9` identifier to original recipients and then remove it from quarantine.

If **Dr.Web MailD** is set up to keep quarantined messages in DBI-storage then additional SQL command must be specified in command line:

- `--sql-remove-command` - allows to remove a message from quarantine using its file identifier (the latter is the only parameter in this query).

  Example:

  ```
  --sql-remove-command "DELETE FROM mail_export WHERE filename LIKE ?"
  ```

`drweb-qcontrol` utility supplies also a simple interface to perform search in stored messages.

Available search criteria:

- `--search-from {address}` - search is performed by sender address in the envelope.
- `--search-to {address}` - search is performed by recipient address in the envelope.
- `--search-headers {header_name[:value]}` - search is performed in headers. `header_name` is the name of the target header (full compliance is required). If `value` is not specified, then only header name is used. If `value` is specified, then it is looked up in the desctiption of the given header as a substring. Search of the header name and its value is case-insensitive.
- `--search-inbody {list of strings}` - search for given substrings is performed in the message body. The message body is treated as a single unit, and no MIME-decoding is performed. Search is case-insensitive.

Please note, that if special symbols `*`, `^`, `$` are used as arguments for **`--search-headers`** and **`--search-inbody`** parameters, then for correct processing of search commands backslash symbol (`\`) must be placed before them.

Example:

```
--search-inbody \* --stat
```

Every search criteria is checked independently (i.e. they are combined using `OR` principle).

Example:

```
--search-to addr1 --search-to addr2
```

It looks for messages with envelopes containing addresses of `addr1` or `addr2` receivers.

Example:

```
--search-from from@drweb.com --search-to
to@drweb.com –search-headers "Subject: [SPAM]"
--search-inbody "spam"
```

It finds all messages in quarantine, sent by from@drweb.com or sent to to@drweb.com, or with a [`SPAM`] word in their `Subject`, or with a spam word in the message body.

If any search criteria is specified in the command line with a list of files, then the search will be performed only in these files.

Example:

```
drweb-qcontrol --stat --search-from a@5 def/
backup/%
```

It outputs information to console about all archived messages, sent by a@5:

```
1. def/backup/5/00014F95.DW_SHOT_PRODUCT.1LXzg1
from: a@5; to: a@drweb.com time: 2008-8-14
15:1:46
```

Migration to New Quarantine Version

Beginning from the 6.0 version of the **Dr.Web MailD** quarantine organization has changed: message body is still saved to the file system, but message envelope and service information are now saved to the internal database.

`quarantine_migration.pl` script residing at `%bin_dir/maild/scripts/` directory is designed exclusively for quarantine migration. After startup it will detect all the necessary default settings and offer to perform migration to the new version of the quarantine. Migration will be performed automatically. After finishing the migration process it will output a report, containing time boundaries, number of processed and skipped messages and number of errors.

Interactive Management

During operation of **Dr.Web MailD**, interactive management for some of its modules can be performed. Interaction is organized via the control socket of the `drweb-maild` module (i.e. `drweb-maild` itself must be working, and control socket must be enabled).

To enable this possibility, do the following

1. Set `Yes` as a value of **`Control`** parameter in [`Maild`] section of **Dr.Web MailD** configuration file;
2. Connect to address set in **`ControlAddress`** parameter of the same section of **Dr.Web MailD** configuration file, and start entering commands in interactive mode. Please note, that only one interactive connection can be established.

Interaction is carried out in a row-wise mode: the user inputs some string and the `drweb-maild` outputs some string in response. So, it is impossible to use multiline commands, and complex rules must be entered line by line.

The end of output of information from the `drweb-maild` is

marked by the empty line.

Several interactive connections can be established simultaneously. Both IPv4 and IPv6 protocols are supported. **Dr.Web MailD** always opens listening socket at `/directory_specified_as_a_value_of_BaseDir_parameter/ipc/.ctl`, regardless of the value of **Control** parameter.

The following commands are available:

- **help** `[section|command]` - outputs complete list of commands from all the sections. After this command the name of the section can be specified, to receive information about all the commands from this section. Also you can specify the name of the specific command to get information about it. You can view the list of all commands with the **help all** command.
- **option** `[regex]` - outputs the list of parameters with their values, used by both `drweb-maild` and plug-ins. Names of parameters can be specified as a regular expression for the **option** command, which limits the list only to those parameters, which names match the regular expression. If regular expression is not specified, then all the parameters are output.
- **db-state** - outputs the current state of the internal database in the following format:

  ```
  Number: NC/NM
  Size: SC/SM
  ```

  where `NC` and `NM` are current and maximum amount of messages in database, and `SC` and `SM` are current and maximum size of the database in bytes. If `NM` or `SM` are equal to `0`, then maximum amount of messages and maximum database sizes are not limited.

- **queue-state** - outputs current status of messages in the internal queue. Total amount of messages and information about each message will be displayed. If total amount is great, it may indicate a lack of threads in the second pool of `drweb-maild` (regulated by **OutPoolOptions**

parameter).

- **send-stat -** forces transfer/export of the statistics (by analogy with action made on timeout set by **SendPeriod** parameter from [Stat] section of **Dr.Web MailD** configuration file). This command can be used if the value of **Send** parameter from [Stat] section of **Dr.Web MailD** configuration file is set to Yes**.**
- **send-report** [period] - forces the sending of the message with report on plug-ins operation (by analogy with action made on timeout set by **SendTimes** parameter in [Reports] section of **Dr.Web MailD** configuration file). This command can only be used when the value of **Send** parameter from [Stat] section of **Dr.Web MailD** configuration file is set to Yes. Period defines a period of time for which report is made (value is in {time} format). If this value is not set, report is sent for a 24-hour period.
- **backup** - forces back-up of internal database.
- **quarantine-pulse** - forces initialization of drweb-qp utility for quarantine processing (by analogy with action made on timeout set by **PulseTime** parameter from [Quarantine] section of **Dr.Web MailD** configuration file).
- **dump-cache-stat** - all cached statistics is moved from operating memory to internal database.
- **get** [(id1|-|id1-[id2]) [(plugin_name|-)]] - outputs information on messages stored in the internal database. idN stands for the number of requested messages, id1-id2 - outputs information on messages with numbers from the extent defined by these values, id1- - outputs information on all messages with numbers beginning with id1 (numbers must be specified in hexadecimal notation), plugin_name - name of the plug-in, which has moved the message to quarantine database. "-" symbol means that parameter is not specified. When no parameters are specified, the information on all messages from the database will be output.

Example:

get - drweb - outputs information about messages

moved to quarantine by `drweb` plug-in.

get – `outputs` – information on all messages stored in database.

- **send** `[(id1|-|id1-[id2]) [(plugin_name|-)] [force]]` - sends specified messages to recipients from the envelope. Only messages with **send**=`no` in **get** command output can be sent. Description of parameters is similar to that from **get** command. New parameter **force** initiates dispatch of messages with **send**=`yes`.
- **export** `[(id1|-|id1-[id2]) [(plugin_name|-)] [(dir_name|-)] [env]` - exports specified messages from database to separate files. Description of the parameters is similar to that from **get** command. Also there are two additional parameters:
  - **dir_name** - path to the directory where files are saved. If this path is not specified, then the value of **BaseDir** parameter from `[General]` section of **Dr.Web MailD** configuration file is used.
  - **env** - if it is specified, then the envelope is also exported to file in the following format: on the first line the sender's address is output, on the second line the receiver's addresses are output, delimited by commas.

  Name of the message file is created from the number of the message and `.eml` extension. Name of the envelope file is created from the number of the message and `.envelope` extension.

  Example:

```
export 00002D94 vaderetro /t env
Success export body to /t/00002D94.eml
and envelope to /t/00002D94.envelope
```

- **remove** `[(id1|-|id1-[id2]) [(plugin_name|-)]` - removes specified messages from database. Description of parameters is similar to that from **get** command.

Example:

```
remove 00002D93
Success remove record 00002D93
```

- **send_and_remove** [(id1|-|id1-[id2]) [(plugin_name|-)] [force_send] [ignore_send_error] - sends and removes specified messages from database. Value of the **force_send** parameter is similar to the value of **force** parameter of the **send** command. If the message was successfully sent by the **send_and_remove** command, or the dispatch of this message is not required (it was already sent some time before), then the message is deleted. If **ignore_send_error** parameter is specified, then the message is deleted no matter whether the dispatch was successful.
- **notify** - checks generation of notifications. The command is describes in `notify.*` files residing in the directory with documentation.
- **version** - outputs the current version of the product.
- **stop** - stops the product.
- **reload** - sends `SIGHUP` signal to `drweb-maild`.

User, Group and Alias Management

Specific settings for each recipient can be set and flexibly adjusted in rules. Also an arbitrary number of users can be combined in groups - and specific settings can be assigned to that groups (group settings are applied to every user in the corresponding group).

If there are too many users, it is not effective to specify settings for each of them in rules, because the more rules are specified, the slower is the search for the specific setting for a certain user. It is better to save rules for each user to the local database. In this case search will be performed faster and memory will be used more efficiently.

In addition to rules some extra information about the user can be

stored in the database. To view information about the certain user you should send **email-info** command to the control socket. Information will be output in the following format:

```
[client-id1/]email1 A=active1 S=stat1
name: name1
aliases: alias1 alias2 ..
groups: group1 group2
rules:
1: rule11
2: rule12
...
 custom:
tag1: info1..
tag2: info2..
...
```

Where:

- `client-id1` – identifier of the Client, to which the user belongs;
- `A=active1` – whether this user is active or not. If the user is not active, then all the rules with user settings are ignored;
- `S=stat1` – whether to collect separate statistics on this user or not. To enable collection of statistics on each user separately, set `high` log verbosity level for general statistics;
- `name: name1` – name of the user;
- `aliases:`, `groups:`, `rules:`, `custom:` – information about aliases, groups, group and personal settings, etc.

Users can be combined in groups. Each group has the same bundle of settings as the user: Client ID, name of the group, activity status, availability of separate statistics, list of the group members and additional service information. To view information about the certain group you should send **groups-info** command to the control socket. Information will be output in the following format:

```
[client-id1/]group1 A=active1 S=stat1
emails:
email1
email2
...
custom:
tag1: info1..
tag2: info2..
...
```

If some user has several e-mail addresses, they can be combined in the following way: one e-mail address is considered primary, and other addresses are set as aliases. As a result all these addresses will be treated as a single entity, and the same settings will be applied to them, and united statistics will be collected.

Value of the specific parameter is selected according to the following algorithm:

- Parameter value is looked up in rules for a certain user.
- If there is no such parameter in user's rules (or parameter value is not specified), search is performed through the rules for each group this user belongs to, from the last group in the list to the first one.
- If there is no such parameter in groups' settings (or parameter value is not specified), search is performed through Client's rules specified in configuration file.
- If the required parameter is still not found, search through global rules specified in configuration file is performed.
- If there is no such parameter in configuration file, then default hardcoded value will be used.

As you may notice, the order, in which groups are specified in the list for a certain user, is very important, because there are values which are found first that will be applied.

If more than one recipient is specified in a message, and for those recipients different values for the same parameter are found, then two possible ways to solve this problem exist:

1. For some parameters (described in [Riles] Section chapter) the message is duplicated, i.e. for each recipient its own copy of the message is created, and different settings are applied to different copies.
2. Parameters, for which the message cannot be duplicated, are ignored, and either Client's settings are applied, or global settings are used, or hardcoded default values are chosen.

Please note, that during check of settings user rules and groups rules are combined and processed as a single list (rules for the user go first). So, when these lists are worked through, a setting from the "user" part of the list may match the setting from the "group" part of the list - and the previously described algorithm will be invoked.

Management of users, groups and aliases can be performed either via the control socket or via the web interface.

User, group and alias management is performed with special commands. The following common terma sre used in commands:

- **client** – administrator of **Dr.Web for UNIX mail servers**. Empty identifier is assigned to it.
- **email** – user's email address (in compliance with RFC5322). It can be enclosed in angular brackets (<>) or quotes (''). Its length must be less then 1024 bytes.
- **client-email** – two values `[client-id/]email`, where `client-id` for **Dr.Web MailD** is always void.
- **emails-list** – `client-email` list, delimited with white spaces.
- **group** – name of the group, enclosed in single quotes. If the corresponding substring does not contain whitespaces, then quotes can be left out. If quotes are used, then before each single quote in the text an additional single quote must be specified. Length of the group name must be less then 1024 bytes.
- **client-group** – two values `[client-id/]group`, where `client-id` for **Dr.Web MailD** is always empty.
- **ext-client-group** = `[client-id/]group | client-id/` - similar to `client-group`, where `client-`

`id` for **Dr.Web MailD** is always empty.

- **`group-list`** – `client-group` list, delimited with white spaces.
- **`ext-group-list`** – `ext-client-group` list, delimited with white spaces.
- **`RULE`** – a rule from the list of **Dr.Web MailD** rules. If the string with this rule contain commas, and they are not enclosed in quotes, then before each comma a back slash `\` must be specified (for parameters with a single possible value) or several back slashes `\\\` (for parameters with several possible values, delimited by comma).

  Example:

  ```
  true cont headersfilter/RejectCondition = FileName = \"\.e\\\,e\"\, FileName = \"\.com\", headersfilter/RejectPartCondition = FileName = \"\.e\\\,e\"\, FileName = \"\.com\"
  true cont vaderetro/action = discard\, quarantine
  ```

- **`tag`** – an arbitrary string of symbols `[a-zA-Z0-9_-]` used as a search tag for information about some user or a group. For web interface value of this parameter is set to `web`.
- **`info`** – the whole line up to the line feed - i.e. it cannot contain line feeds or null symbols.
- **`settings`** – a number of settings for an object (user or group). They can be specified as `parameter_name=value` pairs. Parameters must be delimited by white spaces. The following parameters are available:
  - `A` (active) - can take on one of the following values: `0` (not activated) or `1` (activated). If the object is not activated, then all the rules connected with it are not used. By default any object is considered active (if the value of this parameter is not specified).
  - `S` (stat) - controls collection of statistics for the object. It can take on one of the following values:

`0` (not activated) or `1` (activated). Deactivation of this parameter affects only the process of statistics collection for the object, not the result - i.e. all the previously collected statistics is preserved. By default collection of statistics is enabled.

- `N` (name) - name of the user (for groups this parameter is ignored). It can be enclosed in single quotes in the same way as it is done for groups. If this parameter is not specified, then the name of the user is left blank. Maximum length of the user name is 1000 bytes.

  Examples:

  ```
  S=1 A=0 N='Some user'
  S=0
  ```

Please, note, that in order to preserve a sequencing of groups for a certain **`client-email`**, for each **`client-email`** set of groups is managed, but not a set of **`client-emails`** for a group.

Commands for User Management

When operating the control socket, *user* is every e-mail address, entered to system. The control of the addresses can be realized with the help of the following commands:

- **`email-set`** `client-email [settings]` – creation or update of `email` address for client, set in `client-email`. If address doesn't exist then it is created. If not all settings are specified in `settings`, then default values are set for missing settings.
- **`email-remove`** `client-email` – removal of `email` address for client, set in `client-email`. User is also deleted from all groups. If address doesn't exist or address is alias, then error is output.
- **`email-rename`** `client-email email` – change of the main user address, set in first parameter, to the address, set in second parameter. If address doesn't exist in first parameter, or address is alias, or address with the new address already exists, then error is output and renaming is not performed.

- **email-set-groups** client-email [list-of-groups] – set the list of groups, which contain address client-email address. The group order is important (group settings in the end of the list has more priority).

  If list-of-groups is empty, then the whole list of groups for client-email address is cleared. In the list list-of-groups the groups are delimited with white spaces. If client-email or some group doesn't exist in the list, then error is output and no action is taken. If same group is present twice in the list, the error is output. If client-email is alias, then original recipient is updated. If client-id is specified for the list-of-groups list, then it must match client-id from client-email address. Otherwise, error is output. If client-id is not specified in alias from list-of-groups, then it takes the value equal of client-id from client-email.

- **email-get-groups** emails-list – receipt of group list for all addresses from emails-list list. If some address from the list is missed, then error is output but execution of command continues. If client-email is alias, then information is output for the original recipient.

  Output format:

  ```
  client-id/email1: group1 group2 group3 ...
  client-id/email2: group21 group22 group23 ...
  ```

  where groupN can be enclosed to single quotes, if name of the group contain white spaces.

- **email-get-rules** emails-list – receipt of settings or rules for all addresses from emails-list list. If some address is not present in the list, then error is output but execution of the command continues. If alias is transferred, then settings for original recipient are output. Error is output for every non-existent address.

  Output format:

  ```
  [client-id1/]email1
  1: rule1
  ```

```
2: rule2
...
[client-id2/]email2
1: rule21
2: rule22
...
```

- **email-insert-rule** client-email index RULE – insertion of the new rule with sequence number index for email address and for client in client-email. If email or client do not exist, then error is output. Numeration (index) starts at 1. If index value is more than maximum number of rules for specified email, then new rule RULE is added to the end of rule list. Index index by order is assigned to new rule.

  Example: If two rules are specified for email, then when adding new rule with index, equal 10, rule adds to the end of the list with index, equal 3. If index ≤ 0, then error is output. If RULE is void (i.e. rule is not specified), then error is output.

  After successful modification the rules for the current group are output in output format for **email-get-rules**.

- **email-remove-rule** client-email index – removal of the rule with sequence number index for email address and for client in client-email. Numeration (index) starts at 1. If client-email doesn't exist, then error is output. If index value is more than maximum number of rules for specified email, or index ≤ 0, then error is output. If alias is transferred, then settings for the original address are updated.

  After successful modification rules are output in output format for **email-get-rules**.

- **email-get-custom** -tag emails-list – receipt of the information with tag tag, connected with each of the users in emails-list. If some address does not exist, then

error is output but execution of command continues. If information with `tag` doesn't exist, then void string is output. Information for each address is separated with new line. If symbol "-" is specified instead of `tag`, then information is output on all tags.

Output format:

```
[client-id1/]email1
tag: info..
[client-id2/]email2
tag2: info2..
```

- **`email-set-custom`** `tag client-email [info]` – setting `info` text, connected with `tag` for user of `client-email`. If user is not found then error is output. If `info` is not specified, then tag with all the information about user is deleted.
- **`email-info`** `emails-list` – receipt of complete information about all addresses from `emails-list`. If some address does not exist in the list, then error is output, but execution of the command continues. Address rules are output in compiled view for all groups and address settings. For alias, the information on groups and settings is taken from the original address. Rules' settings are output in the following order: first user settings, then group settings in the order reverse to the order of group movement. When compiling the rules activity control of the group and users is considered.

  Output format:

```
[client-id1/]email1 A=active1 S=stat1
name: name1
aliases: alias1 alias2 ..
groups: group1 group2
rules:
1: rule11
2: rule12
...
 custom:
```

```
tag1: info1..
tag2: info2..
...
[client-id2/]email2 A=active2 S=stat2
name: name2
aliases: alias12 alias22 .. | alias for email2
groups: group3
rules:
1: rule21
2: rule22
...
custom:
tag21: info21..
tag22: info22..
...
```

groupN can be enclosed to single quotes, if name of the group contain white spaces.

Output format for alias:

```
[client-id1/]email1
aliases: alias for email
```

- **email-search** [range:START/NUMBER] [email:part-of-email] [name:'part-of-name'] [ignore:alias|nonalias] – address or part of the address search. It outputs the addresses, starting from START (numeration starts from 0), and the quantity of NUMBER elements. If START and NUMBER are not specified, then all found addresses are output. If START or NUMBER are negative, then error is output. If values START or NUMBER exceed the number of found addresses, then its values are not limited (accordingly, for "unlimited" START addresses are output from the first in the list, for "unlimited" NUMBER - all addresses in the list).

- part-of-email – substring in email address or alias for the search. If part-of-email is not specified, then all known addresses or aliases are output. Output format matches **email-info** output. User's unique identifier in part-of-email must be specified in full.
- part-of-name – substring in user's name (if the name contains single quote ', then another single quote ' should be used before it; if substring has no white spaces, then surround quotes can be skipped) - output contains only the users which names has the specified substring.
- ignore – defines which type of record should be ignored – aliases (search among usual addresses), nonalias – usual addresses (search only among aliases).

If email and name are specified simultaneously, then only users are output, which satisfy to both restrictions. As the user name is not saved for aliases, the simultaneous use in search of the substring for alias and user name is absurd.

- **email-count** [range:START/NUMBER] [email:part-of-email] [name:'part-of-name'] [ignore:alias|nonalias] –processing is realized similar to **email-search**, but the number of found addresses is output.

Commands for Alias Management

The control of the aliases can be realized with the help of the following commands:

- **aliases-get** emails-list – outputs list of aliases for all addresses from the emails-list list. If emails-list contains non-existing addresses or other aliases then error is output but the execution of the command continues. If the same address is found twice then the error is output.

  Output format:

  [client-id1/]email1: alias1 alias2 alias3

```
...
[client-id2/]email2:    alias21    alias22
alias23 ...
```

- **aliases-set** `client-email [emails-list]` – sets the list of aliases for `email` and client address, set in `client-email`. If `client-email` doesn't exist or appears to be alias, then error is output. If `emails-list` is not specified, then all aliases, linked to `client-email`, are deleted. If the list contains at least one address, which is registered or alias for a different address, then error is output and the execution of the command is cancelled. If `client-id` is specified for the address in `emails-list`, then it has to match `client-id` from `client-email` address. Otherwise the error is output. If `client-id` is not specified in `emails-list` in alias, then it is set equal to `client-id`, set in `client-email`.

Commands for Group Management

The control of the groups can be realized with the help of the following commands:

- **groups-set** `client-group [settings]` – creation or update of the group with the name `group` for client, set in `client-group`. If group doesn't exist, then it is created. If `settings` do not contain all settings, then default values are set for absent settings.
- **groups-remove** `client-group` – removal of the group with the name `group` for Client, set in `client-group`. If set group doesn't exist, then error is output. For each user in removable group, the group is deleted from the list of groups, to which the user is included.
- **groups-rename** `client-group group` – renaming of the group, set as a first parameter, by using the name, set as a second parameter. If specified group doesn't exist or the group with specified name already exists, then error is output and no actions are taken.
- **groups-get-rules** `[group-list]` – receipt of the

rules or settings for all groups from the list `group-list`. If some group from `group-list` doesn't exist, then error is output but execution of the command continues.

Output format:

```
[client-id1/]group1
1: rule1
2: rule2
...
[client-id2/]group2
1: rule21
2: rule22
...
```

- **`groups-insert-rule`** `client-group index RULE` – insertion of a new rule before the rule with the sequence number `index` for a group with the name `group` and for a client, set in `client-group`. If group with the set name doesn't exist, then error is output. Numeration (`index`) starts from `1`. If `index` value is more than maximum number of rules for the specified group, then new rule `RULE` is added to the end of the rule list. It is assigned with `index` in succession.

  Example:

  If for a group only two rules are set, then when trying to add a new rule with `index = 10`, then the rule is added to the end of the list with `index = 3`.

  If `index ≤ 0`, then error is output. If `RULE` is empty (i.e. rule is not specified), then error is output. After successful modification the rules for this group are output in output format **`groups-get-rules`**.

- **`groups-remove-rule`** `client-group index` – removal of the rule with `index` sequence number for `group` group and for `client`, set in`client-group`. Numeration (`index`) starts from `1`. If `group` or `Client` do not exist, then error is output. If `index` value is more than maximum number of rules for the specified group or `index ≤ 0`, then

error is output. After successful modification the rules for this group are output in output format **groups-get-rules**.

- **groups-info** [ext-group-list] – output of all users, which compose the group from ext-group-list list, and information about the activity and random information. If some group from ext-group-list doesn't exist, then error is output but command execution continues. If ext-group-list is not specified, then information is output on all existing groups for all Clients. If only Client identifiers occur, then information is output on all groups. Aliases in the address lists are not output.

  Output format:

  ```
  [client-id1/]group1 A=active1 S=stat1
  emails:
  email1
  email2
  ...
  custom:
  tag1: info1..
  tag2: info2..
  ...
  [client-id2/]group2 A=active2 S=stat2
  emails:
  email21
  email22
  ...
  custom:
  tag21: info21..
  tag22: info22..
  ...
  ```

- **groups-count** [ext-group-list] - command is executed similar to **groups-info**, but it only outputs the number of found groups.

- **groups-get-custom** -|tag group-list – receipt of the information with tag tag, connected with each of the group in group-list. If some group from group-list does not exist, then error is output but execution of command continues. If information with tag doesn't exist, then void string is output. Information for each group is separated with new line. If symbol "-" is specified instead of tag, then information is output on all tags.

  Output format:

  ```
  [client-id1/]group1
  tag: info..
  [client-id2/]group2
  tag2: info2..
  ```

- **groups-set-custom** tag client-group [info] – setting of the text info, connected with tag tag for client-group group. If group is not found, then error is output. If info is not set, then tag with all the information, linked to it, is deleted.

Working with Quarantine via Control Socket

Working with quarantine by means of control socket can be realized with the help of the following commands that use common notions:

- **Client** – administrator of **Dr.Web for UNIX mail servers**. It is assigned with empty identifier.
- **id** – path relative to the directory, set in **Path** parameter in [Quarantine] section, to the file with text body. For example, if the value of **Path** parameter in [Quarantine] section is /var/drweb/infected (default value), then identifier def/drweb/E/00020EBE.maild.xeAX4u links to the message, which body is located in the file /var/drweb/infected/def/drweb/E/00020EBE.maild.xeAX4u, where:
  - def – the word "def";
  - drweb – name of plug-in, that blocked the

message. If the message is blocked by `drweb-maild` component, then the value is set to `maild`. If the message is moved to archive, then value is set to `backup`.

- **`id-like`** – same as **`id`**, but special symbols can be used - "`%`" - zero or more random symbols, "`_`" - one random symbol - to set these identifiers.

  Example:

  `def/%00014F7F%` – all messages with `00014F7F` number, saved to quarantine;

  `def/drweb/%` – all messages, saved by `drweb` plug-in.

  Message body is saved to database in decoded mode (in UTF8 encryption), and all control symbols (ASCII 0..21 and 127), except for tabulation, are replaced with white spaces.

Result of the execution of each command is output with empty string at the end.

Commands for Quarantine Control

The control of quarantine can be realized with the help of the following commands:

- **`quarantine-search`** `[range:START/NUMBER] [sort:SORT_TYPE] [sender:EMAIL_SUBSTR] [rcpt:EMAIL_SUBSTR]* [period:DATE1[/DATE2]] [size:SIZE] [subject:'SUBJECT_SUBSTR'] [id:id-like] [order:ascent|descent]` – searches the messages in quarantine on specified criteria. Messages are output starting with `START` (numeration starts with `0`), and in the quantity of elements `NUMBER`. If `START` and `NUMBER` are not specified, then all found messages are output, which satisfy the other criteria. The value `NUMBER 0` signifies the output of all elements.

  Used parameters:

  - `SORT_TYPE` – type of the sorting. Possible values:

- ✓ `date` (by default) – sorting by delivery date of messages to quarantine;
- ✓ `size` – sorting by the message size;
- ✓ `sender` – sorting by the sender address;
- ✓ `subject` – sorting by the message subject.

- `EMAIL_SUBSTR` – substring for searching in the fields `rcpt` or `sender`.
- `period` – period, during which the messages are output. If it is not specified, then messages are output for the whole period.
- `DATE1` – output of the messages, which were moved to quarantine after specified time (inclusively).
- `DATE2` – upper time line of moving the message to quarantine (inclusively). Format `DATE` fits ISO format - `YYYYMMDDTHHMMSS`, where `T` - separator between time and date. The time is set and output as local time for a host with operating **Dr.Web MailD**.
- SIZE - maximum size in bytes and returns only the messages with the size exceeding the set value. Size is unlimited when the value is set to `0` .
- `SUBJECT_SUBSTR` – substring enclosed to quotes in the original subject of the message (i. e. before the message modification by the components of the product). If substring has no white spaces, then surrounding quotes can be skipped. If quotes are present in the name then the repeated symbol ' must be used before the name.
- `order` – order, that returns the result (`ascent` - ascending, `descent` - descending. Default value: `descent`.

If there was a mistake in some parameter then **quarantine-search** command won't run. If several templates of the recipients are specified, then only messages containing all templates are output (similar to the operation of logical operator AND). For all parameters, except rcpt, the last value specified in command line is used. For rcpt, the number of recipients increases with every new input value.

Output format:

```
N. id SENDER RCTPS
SIZE DATE SUBJECT
BLOCK_OBJECT1
BLOCK_OBJECT2
...
```

where:

- N – sequence number of the found message;
- SENDER – sender of the message from the envelope;
- RCPTS – recipients of the message from the envelope;
- SUBJECT – subject of the message (output in encryption UTF8);
- SIZE – size of the message in bytes;
- DATE – date of moving the message to quarantine;
- BLOCK_OBJECTN – blocking object for this message.

Examples:

```
# quarantine-search
```

Returns the list of all messages in quarantine, starting from the newest.

```
# quarantine-search range:45/15 id:def/drweb/%
```

Returns 15 newest messages in quarantine, missing first 45 for drweb plug-in.

```
# quarantine-search rcpt:vasya@pupkin.com
```

Returns all messages with recipient vasya@pupkin.com in quarantine, starting from the newest.

```
# quarantine-search sort:size sender:
period:20090101T100001/20090102T100000
size:5242880 id:def/vaderetro/%
```

Outputs the messages in descending order, received for `vaderetro` plug-in January 1, 2009 from 10 a.m. to 10 a.m. of the next morning, and size of which is more than 5 Mb.

Output example:

```
quarantine-search
0.        def/drweb/9/00021569.maild.BMED3y
<ai@drweb.com> <alias_ai81@drweb.com>
829 20091117T102126 [EICAR] test2
EICAR Test File (NOT a Virus!)
1.       def/backup/9/00021569.maild.3PLb8e
<ai@drweb.com> <alias_ai81@drweb.com>
828 20091117T100213 [EICAR] test
```

- **quarantine-count** `[range:START/NUMBER] [sort:SORT_TYPE] [sender:EMAIL_SUBSTR] [rcpt:EMAIL_SUBSTR]* [period:DATE1[/DATE2]] [size:SIZE] [subject:'SUBJECT_SUBSTR'] [id:id-like] [order:ascent|descent]` – operates similar to the **quarantine-search** command, but, instead of the messages, total number of found messages is output.

  Output examples:

  ```
  quarantine-count
  234
  ```

- **quarantine-remove** `id-like [part-of-email1, part-of-email2, ..]` – removes the specified recipients (searched as a substring) `part-of-email1, part-of-email2, ...` from the envelopes of the messages, identifiers of which match `id-like` (all specified recipients must be present in the envelope). If message has no recipients left or the list for their removal is not specified then message is removed entirely from quarantine.

Examples:

```
# quarantine-remove %/backup/% drweb.com>
```

All messages are deleted from quarantine and backup, the addresses of which are ended on `drweb.com`.

```
# quarantine-remove % <foo@dwreb.com> <foo2@dwreb.com>
```

All messages are deleted from quarantine and backup, recipients of which are simultaneously foo@dwreb.com and foo2@dwreb.com.

```
# quarantine-remove client2/drweb/E/00020EFE.maild.Q5FRbO
```

The message with the specified identifier is removed.

- **`quarantine-limits`** – outputs the current restrictions, set for quarantine.

  Output format:

  ```
  client-id1: NUMBER/MAX-NUMBER SIZE/MAX-SIZE
  ...
  total: NUMBER/MAX-NUMBER SIZE/MAX-SIZE
  ```

  where:

  - `NUMBER/MAX-NUMBER` – current/maximum number of messages. If maximum value is not set, then `0` is output.
  - `SIZE/MAX-SIZE` – current/maximum size of messages in quarantine (in bytes). If maximum value is not set, then `0` is output.
  - `client-id1` – client identifier, for which information is output.
  - `total` – information on all database.

- **`quarantine-send`** `id-like [email1 email2 ...]` – sends the messages from quarantine to the specified recipients (`email1 email2 ...`). If recipients are not specified, then messages are sent to original recipients from the envelope. Output format is defined on results of sending every message:

  ```
  RES in sending (to RCPTS_LIST): id
  ```

where:

- `RCPTS_LIST` – actual list of message's recipients.
- `RES` – `OK` or `ERROR`, depending on the sending result.
- `id` – file path with message body.

Output format:

```
OK in sending (to <ai@drweb.com> <as@sd>): def/backup/6/00004DD6.maild.VQ80Ro
OK in sending (to <ai@drweb.com> <as@sd>): def/backup/6/00004DC6.maild.PWfqe3
```

- **`quarantine-add`** `id from rcpt1 rcpt2...` – adds the specified file to quarantine, where `from` - message sender, `rcptN` - recipients. Addresses can be enclosed in angular brackets <>. If file with the specified `id` doesn't exist, then error is output.

Receiving Statistics via Control Socket

Statistics on operation of **Dr.Web for UNIX mail servers** for clients, users and groups can be received via command interface of control socket. Receipt of statistics can be realized with the help of special commands that use common notions:

- **`client`** – administrator of **Dr.Web for UNIX mail servers**. Empty identifier is assigned to it.
- **`email`** – user's email address (in compliance with RFC5322). It can be enclosed in angular brackets (<>) or quotes (`''`).
- **`client-email`** – `[client-id/]email`, where `client-id` – unique identifier of the `client`. It can be void for `client` by default. `client-id` – case-sensitive.
- **`group`** – name of the group in quotes (`''`). If substring contains no white spaces, then quotes can be omitted. If quotes are used, then use another `'` symbol before the name with `'` symbol.
- **`client-group`** = `[client-id/]group` - where `client-id` – unique identifier of `client`. It can be

empty for client by default, `client-id` – case-sensitive.

When working with statistics it is important to take into account that general statistics on checked messages for clients and users/groups is collected in different ways:

- statistics on clients is collected to internal cache, which flushed to internal database every 5 minutes (saving is made on running `dump-cache-stat` command, on receiving `HUP` signal, and on shutdown of **Dr.Web for UNIX mail servers**);
- statistics on users and groups is saved directly to the corresponding records of internal database. On trying to save information, if record is in database for more than 5 minutes, then new record is created, to which the further changes are saved.

As these commands work with internal database, then last record for users and groups contains statistics from the moment of its creation to the current moment. Statistics on clients is saved from time to time, thereby, delay in appearance of general statistics can be up to 5 minutes.

Commands for Working with Statistics

Commands for statistics have several general parameters:

- **`period`** = `period:DATE1[/DATE2]` – outputs statistics for selected period, including limits of time interval.
  where:
  - `DATE1` – lower limit of time interval. Output format and time format is described below.
  - `DATE2` – upper limit of time interval. If the parameter is not set, then the current time is taken. Time format is described below.

  If period is not set then all available statistics is output.
- **`ignore`** = `ignore:total|block` – filtration of output statistics.
- **`total`** – do not output general statistics on checked messages.
- **`block`** – do not output statistics on blocked messages. If

this parameter is not set, then all types of statistics are output.

- **plugin** = plugin:name – outputs information for specified plug-in, where name – plug-in name to output statistics on. If parameter **plugin** is not set, then information is output on all plug-ins. If non-existing plug-in is set, then error is output and command is cancelled. If * is specified, then general statistics is output.

If several similar parameters are specified then statistics is output on the last specified parameter.

Following commands are available:

- **stat-client** client-id|*|- [period] [ignore] [plugin] – statistical data production for set client client-id. If client with set client-id doesn't exist, then error is output and running of the command is cancelled. If instead of client-id "*" is specified, then statistics is output on all clients, if "-" is set (without quotes), then statistics is output for client by default (with empty identifier). Optional parameters can be set in random order.
- **stat-group** client-group [period] [ignore] [plugin] – statistical data production for group client-group. If group doesn't exist then error is output and running of the command is cancelled. Optional parameters can be set in random order.
- **stat-email** client-email [period] [ignore] [plugin] – statistical data production for current user client-email. If for specified address statistics doesn't exist (for example, address is specified incorrect), then empty string is output. If address is alias, then statistics is output on main address. Optional parameters can be set in random order.
- **stat-remove-client** client-id|*|- [period] [ignore] [plugin] – removal of statistics on set client client-id. If client with setclient-id doesn't exist, then error is output and running of the command is cancelled. If instead of client-id "*" is specified, then statistics is

output on all clients, if "-" is set (without quotes), then statistics is output for client by default (with empty identifier). As a result the number of deleted records is displayed.

- **stat-remove-group** client-group [period] [ignore] [plugin] – removal of statistics for group client-group. As a result the number of deleted records is displayed.
- **stat-remove-email** client-email [period] [ignore] [plugin] – removal of statistics for particular user client-email. As a result the number of deleted records is displayed.
- **remove-old-stat** [time] – removal of all statistics on all clients, groups and users, if it is older than time, set in time (type (time)). If the value is not set, then all statistics older than 24 hours is deleted.
- **dump-cache-stat** – flush the internal cache of general statistics on clients to internal database.This function is called from time to time by the product itself. It is also called on receipt of HUP signal or shutdown.

Statistics Output

Statistics on operation of each plug-in modules is output in defined format which consists of two parts:

1. general statistics on checked messages:
   PLUGIN DATE [P] [R] [D] [T] [Q] [RE] [N] [C] [S] [U] [F] [I] [DI] [DM] [DSV] [DC] [DD] [DSK] [DAR] [DE] [DTA] [DTD] [DTJ] [DTR] [DTH] [PS] [RS] [DS][TS] [QS] [RES] [NS] [CS] [SS] [US] [FS] [IS] [WT] ...
2. statistics on blocked messages:
   PLUGIN DATE FROM|- IP|- 'BLOCK1' TYPE1 'BLOCK2' TYPE2 ...

Where:

- PLUGIN – plug-in name, which statistics is based on. If * is specified, then general statics on all solution is displayed (for example, the messages, which were not transmitted by any plug-in, are included).

- DATE – time, when the record was created. For general statistics on checked messages, it means the beginning of time interval, during which statistics is saved. End of time interval is the beginning of the new record. If following record doesn't exist then 5 minutes are added to the beginning of the period. Format matches ISO format - YYYYMMDDTHHMMSS, where T – separator between time and date. Time is set and is output as local time for host with **Dr.Web for UNIX mail servers**.

The following values till WT inclusive are output in the following format:

NAME=VAL, where **NAME** – name of value (P, PS...), VAL – numerical value. If some of these values are not specified then it takes on a value equal to 0.

Possible values:

- P/PS – number/size in bytes of the messages, for which the action was implemented pass;
- R/RS – number/size in bytes of the messages, for which the action was implemented reject;
- D/DS – number/size in bytes of the messages, for which the action was implemented discard;
- T/TS – number/size in bytes of the messages, for which the action was implemented tempfail;
- Q/QS – number/size in bytes of the messages, for which the action was implemented quarantine;
- RE/RES – number/size in bytes of the messages, for which the action was implemented redirect;
- N/NS – number/size in bytes of the messages, for which the action was implemented notifiy;
- C/CS – number/size in bytes of clean messages;
- S/SS – number/size in bytes of the messages, marked as spam;
- U/US – number/size in bytes of the messages, marked as unconditional spam;

- F/FS – number/size in bytes of the messages, blocked by filter;
- I/IS – number/size in bytes of the messages with viruses;
- DI – number of infected attachments;
- DM – number of attachments, infected with modification of known virus;
- DSV – number of attachments, infected with unknown virus;
- DC – number of cured attachments;
- DD – number of deleted attachments;
- DSK – number of attachments that were passed without anti-virus check for various reasons;
- DAR – number of attachments that were passed without anti-virus check because of restrictions on archives;
- DE – number of attachments with processing error;
- DTA – number of attachments with adware;
- DTD – number of attachments with dialers;
- DTJ – number of attachments with jokes;
- DTR – number of attachments with riskware;
- DTH – number of attachments with hack tools;
- WT – time in milliseconds, that plug-in spent for processing the messages.

For blocked messages the list contains the following fields:

- BLOCK[12..] – name of blocking object (for example, virus). It is enclosed in quotes in same way as it is done for groups (see above).
- TYPE[12..] – type of blocked object. Name is taken from **NAME** (see above). Available values: DI–DTH, F, S, U.
- FROM – sender of the message from the envelope.
- IP – IP-address of message's sender.

Using drweb-inject Utility

`drweb-inject` utility is used to deliver mail via local **Sender** component. It receives message body via standart input stream and returns `0` as a response code at success, and `non-zero` - at failure.

The following command line parameters are available:

- `--help` - display help on command line parameters;
- `--version` - display current version;
- `--agent arg` - path to **Agent** to receive configuration (or an empty string for default parameters - in this case no requests to **Agent** will be made);
- `--timeout arg` - timeout for receiving configuration from **Agent**;
- `--id arg` - unique identifier of the **Sender** component to which a message will be dispatched;
- `-f [ --env-from ] arg` - insert address of a sender to the From field of the envelope;
- `-F [ --from ] arg` - if there is no `From` field in the dispatched message, then the `Full Name` from this argument will be used;
- `-i [ --ignore-dot ]` - do not interpret a string with a single "." (dot) symbol as the signal of completion of message body input;
- `-t [ --extract-recipients ]` - allows to add all the recipients from the "**`To:`**" message header to the envelope.

If `drweb-inject` utility is located not in the default directory, then path to it can be specified by `--sendmail` command line parameter at startup of `drweb-qcontrol`. You may also use **`-v`** to receive more detailed information.

In case no sender is specified, the name of the user, whose privileges program is operating with, is used. If such name is not found, operation of the program is terminated with a `non-zero`

error code.

Unified Score

Unified Score technology allows to detect unwanted mail messages with the help of the unified score assigned to the each message. Message score is a signed integer. The greater it is, the higher is the probability that the message is unwanted, and vise versa - the smaller it is, the lower is the probability that the message is unwanted. By default message is considered as clean if it's score less than `SpamThreshold` parameter value (i.e. `99` and less). If message score is greater than `SpamThreshold` value but less than `UnconditionalSpamThreshold` (i.e. from `100` to `999` by default), this message is considered as spam. If message score is greater than `UnconditionalSpamThreshold` value, the message is considered as unconditional spam (`1000` by default).

Message score can be modified in several ways:

- in **`Action`** type of variables optional `score (SCORE)` action can be used, where `SCORE` is an integer, which can be added to the current message score.
- vaderetro anti-spam plug-in assigns a score to the message, and this score is added to the total message score (and later - compared to values of spam thresholds).
- using `add_score` and `set_score` restrictions you can also modify message score (also this is possible with parameters of some other restrictions).
- using Reputation IP Filter scores of all messages from the current session can be modified.

Message score can be used in the following ways:

- in vaderetro plug-in it is compared to spam thresholds;
- in conditions of rules message scor can also be used (score prefix). For detailed information refer to the chapter [Rules] Section;
- in conditions of **Modifier** plug-in parameters the score can also be used and modified. For detailed information, refer to

the chapter Dr.Web Modifier Plug-in;

- if message score becomes greater than the value of the **MaxScore** parameter from the [MailD] section of the **Dr.Web MailD** configuration file, then action specified as a value of **MaxScoreAction** parameter is applied to the message;
- in some restrictions different actions can be applied to the message depending on its current score;
- whole sessions can be blocked in drweb-receiver, if total message score becomes greater then the value of **MaxSessionScore** parameter from the [Receiver] Section.
- using score_filter from Reputation IP Filter addresses with great total scores can be filtered out.

Simultaneous Use of Several Receiver/Sender Components

It is possible to connect several **Receiver** and/or **Sender** components to drweb-maild simultaneously.

This feature can be used for the following purposes:

- to enable concurrent interaction with several MTAs or SMTP-proxy;
- to enable differentiation of settings for each **Receiver**/**Sender** component (which allows to use different settings for monitored interfaces);
- to enable redirect of the messages from one MTA to another (i.e. for routing).

To make such simultaneous use possible

1. Assign unique identifier to each component from **Receiver** group and **Sender** group (ID of some **Receiver** may be the same as ID of some **Sender**, but there will be impossible to find a pair of **Receivers** with the same ID).
2. Inform each component how it can receive its configuration information.
3. Assign a component's unique identifier as a tag to each message sent by a certain **Receiver**.

4. After processing a message `drweb-maild` is looking for available **Sender** with the same ID as **Receiver**'s. If no available **Sender** with such ID is found, then message can be sent to the default **Sender** (with no unique ID specified), which must be always available.
5. The list of available **Senders** is generated at startup and is refreshed upon a receipt of `SIGHUP` signal.
6. Routing of messages generated by `drweb-notifier` is organized by **`MsgIdMap`** parameter from [`Notifier`] section of **Dr.Web MailD** configuration file. This parameter allows to define to which **Sender** reports should be sent in response to messages from certain **Receivers**.

Unique identifier for **Receiver**/**Sender** is set via `--unique-id` command line parameter. When components are started with this parameter, they create in `%var_dir/msgs/{in|out}` directory a number of subdirectories for their message queues, and in `%var_dir/ipc/` directory a special UNIX socket is created for **Sender**.

When the second copy of the component (e.g. `drweb-receiver`) is started, an additional adjustment must be performed: it is necessary to define how this second copy is going to receive configuration information.

There are two ways for a component to obtain configuration information:

- to create a new copy of `*.conf` file;
- to modify the existing copy of `*.conf` file (it is more easy, but less flexible).

To modify existing *.conf file, do the following

- create a new *.amc file and add information about a new copy of the component. The file name can be arbitrary;

  Example:

```
Application "MAILD"
```

```
id                      40
ConfFile                "/etc/drweb/
                        maild_smtp.conf"

Components
drweb-sender2           General, Logging,
                        Sender2
  drweb-receiver2        General, Logging, /
                         Maild/
                         ProtectedNetworks,
                         /Maild/
                         ProtectedDomains,\
                         /Maild/
                         IncludeSubdomains,
                         SASL, Receiver2
drweb-sender3           General, Logging,
                        Sender3
  drweb-receiver3        General,
                         Logging, /Maild/
                         ProtectedNetworks,
                         /Maild/
                         ProtectedDomains,\
                         /Maild/
                         IncludeSubdomains,
                         SASL, Receiver3
drweb-sender4           General, Logging,
                        Sender4
  drweb-receiver4        General, Logging, /
                         Maild/
                         ProtectedNetworks,
                         /Maild/
                         ProtectedDomains,\
                         /Maild/
                         IncludeSubdomains,
```

```
                          SASL, Receiver4
drweb-sender5           General, Logging,
                        Sender5
  drweb-receiver5        General, Logging, /
                         Maild/
                         ProtectedNetworks,
                         /Maild/
                         ProtectedDomains,\
                         /Maild/
                         IncludeSubdomains,
                         SASL, Receiver5
```

Where `drweb-receiver*` and `drweb-sender*` is the new names of the components to be used for interaction with **Dr.Web Agent**; `Receiver*` and `Sender*` is the new names of the corresponding section in configuration file.

Other parameters must be copied from the section with settings of the original component. Detailed description of syntax of `*.amc` files can be found below in documentation for **Dr.Web Agent**.

- copy the main section with component settings to `*.conf` file, rename this section (specifying the name set on the previous stage) and modify all other settings in the new section for the second component;
- start or restart **Dr.Web Agent** to make it receive new configuration information;
- start the new component with the following command line parameters specified:
  - `--unique-id id` - where `id` is the unique identifier of the component;
  - `--component name` - where `name` is the name used by the new component to interact with **Dr.Web Agent** (`drweb-receiver2` from the example above);

- `--section` - new name of the main section of the component (`Receiver2` from the example above).

Example:

```
%bin_path/drweb-receiver --unique-id id1 --component drweb-receiver2 --section Receiver2

%bin_path/drweb-sender --unique-id id1 --component drweb-sender2 --section Sender2
```

Creation of a new copy of `*.conf` file is more difficult, but allows to adjust all the parameters, not only from component's main section of configuration file.

To create a copy of *.conf file you should do the following:

- create a copy of original `*.conf` file and set up parameters in one's discretion (it is not necessary to rename its sections);
- create new `*.amc` file and include only information about the new component. You must also specify path to the new `*.conf` configuration file, created on a previous stage;
- start or restart **Dr.Web Agent** to make it receive new configuration information;
- start the new component with the following command line parameters specified:
  - `--unique-id id` - where `id` is the unique identifier of the component;
  - `--component name` - where `name` is the name used by the new component to interact with **Dr.Web Agent** (`drweb-receiver2` from the example above).

Example:

```
%bin_path/drweb-receiver --unique-id id2 --component drweb-receiver2
```

```
%bin_path/drweb-sender --unique-id id2 --
component drweb-sender2
```

Dr.Web Monitor can be set up to use new components for both ways of initialization. To make this possible add corresponding lines (about startup of new components) to the `*.mmc` file of **Dr.Web MailD**. Detailed description of syntax of `*.mmc` files can be found above, in documentation for **Dr.Web Monitor**.

Reputation IP Filter

Reputation IP Filter s a technology which allows to collect statistics on each IP address connected to **Dr.Web for UNIX mail servers**, and perform some actions to this IP according to collected data (e.g. temporary block this IP). This technology allows to successfully detect spammers and resist DHA attack

Reputation IP Filter module is enabled if only one filter is specified in **ReputationIPFilter** parameter of is value of **MaxConcurrentConnection** parameter is set to `0`. By default value of **ReputationIPFilter** parameter is set to `score_filter`, IP filter is enabled and IP addresses are filtered out according to the average score assigned to all the messages and sessions from these addresses.

All information on IP addresses is stored in the RAM memory and periodically is saved to files. Information is saved to files each time `drweb-receiver` process receives `SIGALRM` signal (it generates this signal by itself according to **StalledProcessingInterval** parameter setting), or when the `drweb-receiver` process stops. Files are read only when `drweb-receiver` is started.

Files are saved and loaded if only one filter is specified in **ReputationIPFilter** parameter. If there were no IP connections - no information is collected and saved. Information about IP connections is saved to `ipv4.bin` and `ipv6.bin` files (for IPv4 and IPv6 addresses correspondingly) from the directory specified in **BaseDir** parameter from the `[General]` section. If some error emerges during file saving or reading from file, this

information will be output to log. Data saved to these files is binary and OS-dependent - that is why it is not recommended to use these files on some other system.

IP check in **Reputation IP Filter** module is performed immediately after **SessionRestrictions** stage this IP is not marked as **Trusted IP** (to learn more about ***Restrictions** and **Trusted IP** refer to [Receiver] Section).

If it is necessary to protect some IP addresses from blocking by **Reputation IP Filter**, they should be marked as **Trusted IP**s in **SessionRestrictions** parameter. To unblock some IP address blocked by the **Reputation IP Filter** by mistake mark it as **Trusted IP** in **SessionRestrictions** parameter, and all the subsequent connections from this IP will be ignored by the **Reputation IP Filter**.

Reputation IP filter allows to assign a score to the IP address according to the collected statistics on connections and to block this IP address temporarily if its total score is greater than some threshold value.

The following filters are available: `anti_dha`, `errors_filter`, `score_filter`.

Reputation IP filter checks IP address immediately after it passes checks of the **SessionRestriction** parameter - if it is not marked as trusted (i.e. if it was marked as **Trusted IP** in **SessionRestrictions** parameter it will not be checked by the **Reputation IP filter**).

Filters are listed using comma as a delimiter, and are checked in order they were specified. For each filter its name is specified first, then its parameters are listed with white spaces used as delimiters (all these parameters are optional).

Parameters are set as **NAME**=VAL pairs (there must be no white spaces between value and equal sign).

General parameters for filters described below (`U` - is a positive integer, `I` - is an integer, `D` - is a positive floating-point number):

- **min_msgs**=U – minimum number of messages passed for check to the drweb-maild to activate a certain filter. If value is set to 0, then this parameter is ignored.
- **min_errors**=U – minimum number of errors registered on the stage of SMTP session to activate a certain filter. If value is set to 0, then this parameter is ignored.
- **min_wrong_rcpts**=U – minimum number of invalid recipients (which were declined after the RCPT TO command) transferred by the SMTP client, to activate a certain filter. If value is set to 0, then this parameter is ignored.
- **min_conn**=U – minimum number of connections from the IP address to activate a certain filter. If value is set to 0, then this parameter is ignored.
- **block_period**=T – sets a blocking period for IP address if it falls within restrictions of the certain filter. T has a {time} type. If value is set to 0, then blocking is not performed even if some IP falls within restrictions of this filter.
- **score**=I – a score to be assigned to all the messages in the current session. Also it will be added to the general score of the IP address. If score value is not equal to 0, then this parameter will be applied instead of block_period parameter, and a score will be assigned to the IP address instead of blocking.

For each of the filters there is a set of unique parameters and specific sets of default values for general parameters:

- anti_dha – resistance to DHA attacks (directory harvest attack). To use this filter you must specify the full range of protected addresses - as a value of **ProtectedEmails** parameter.

  Specific parameters:

  - **wrong_per_valid_rcpts**=D – a ratio between the number of invalid message recipients (which were declined after the RCPT TO command) and the number of valid recipients. It is the main parameter defining filter operation. If there were no valid recipoents,

then this value is considered to be equal to `1`. If the value of this parameter is set to `0`, the filter is totally ignored.

Default value: `10.0`

Default values for general parameters are:

- **min_msgs**=`0`
- **min_errors**=`0`
- **min_wrong_rcpts**=`20`
- **min_conn**=`0`
- **block_period**=`2h`
- **score**=`0`

- `errors_filter` – allows to filter out IP addresses according to the amount of errors emerged during SMTP session established from the certain IP address.

  Specific parameters:

  - **errors_per_msg**=`D` – a ratio between the number of errors emerged during the SMTP session and the number of messages passed to the `drweb-maild`. If no messages were passed to the `drweb-maild`, then this number is considered to be equal to `1`. If parameter value is set to `0`, then this check is ignored.

    Default value: `0`

  - **errors_per_conn**=`D` – a ratio between the number of errors emerged during the SMTP session and the number of connections from this IP address. The filter is applied only when the parameter value is other than `0`, and there was at least one connection from this IP.

    Default value: `2.0`

  If both parameters are specified, then **errors_per_msg** parameter is checked first, and **errors_per_conn** parameter is checked after it. If values of both parameters are set to `0`, the filter is ignored.

  Default values for general parameters:

  - **min_msgs**=`0`

- **min_errors**=100
- **min_wrong_rcpts**=0
- **min_conn**=50
- **block_period**=2h
- **score**=0

- score_filter – allows to filter out IP addresses according to average score assigned to all messages and sessions from this IP address. It is included into the general Unified Score system and, for example, allows to block spammers on the establishing SMTP-connection stage.

  Specific parameters:

  - **score_per_msg**=D – a ratio between general score for the certain IP address (a sum of all scores of messages sent from the given IP and scores of all sessions initiated from this IP) and the number of messages passed to the drweb-maild. If no messages were passed to the drweb-maild, then this number is considered to be equal to 1. If parameter value is set to 0, then this check is ignored.

    Default value: 0
  - **score_per_conn**=D – a ratio between general score for the certain IP address and the number of connections from this IP address. The filter is applied only when the parameter value is other than 0, and there was at least one connection from this IP.

    Default value: 100.0

  If both parameters are specified, then **score_per_msg** parameter is checked first and **score_per_conn** parameter is checked after it. If values of both parameters are set to 0, the filter is ignored.

  Default values for general parameters:

  - **min_msgs**=0
  - **min_errors**=0
  - **min_wrong_rcpts**=0

- **min_conn**=100
- **block_period**=2h
- **score**=0

Example:

```
ReputationIPFilter = errors_filter score=20, score_filter
```

The first filter will set a score equal to 20 to all messages in sessions established from those IP addresses, which generate too many errors during the SMTP session. The second filter blocks all IP addresses, which have too big average scores in comparison to the number of connections made from them.

Example:

```
ReputationIPFilter = errors_filter errors_per_msg=0.05 errors_per_conn=1 min_msgs=0 min_errors=10 min_wrong_rcpts=3 min_conn=50, score_filter score_per_msg=20 score_per_conn=30 min_wrong_rcpts=3, anti_dha wrong_per_valid_rcpts=0.02 min_wrong_rcpts=20
```

In this example, `errors_filter` filter will be triggered on one of the following conditions:

- ratio between the number of errors emerged during the SMTP session and the number of messages passed to the `drweb-maild` equal 0.05 (`errors_per_msg=0.05`);
- ratio between the number of errors emerged during the SMTP session and the number of connections from this IP address equal 1 (`errors_per_conn=1`);
- number of errors registered on the stage of SMTP session equal 10 (`min_errors=10`);
- number of invalid recipients (which were declined after the `RCPT TO` command) transferred by the SMTP client equal 50 (`min_wrong_rcpts=3`);
- minimum number of connections from the IP address equal 50 (`min_conn=50`).

`score_filter` filter will be triggered if:

- a ratio between general score for the certain IP address and the number of messages passed to the `drweb-maild` equal 20 `(score_per_msg=20)`;
- a ratio between general score for the certain IP address and the number of connections from this IP address equal 30 `(score_per_conn=30)`;
- number of invalid recipients (which were declined after the `RCPT TO` command) transferred by the SMTP client equal 3 `(min_wrong_rcpts=3)`.

`anti_dha` filter will be triggered if:

- a ratio between the number of invalid message recipients (which were declined after the `RCPT TO` command) and the number of valid recipients equal 0.02 `(wrong_per_valid_rcpts=0.02)`;
- If minimum number of invalid recipients (which were declined after the `RCPT TO` command) transferred by the SMTP client equal 20 `(min_wrong_rcpts=20)`.

Plug-ins

At the moment the following plug-ins of **Dr.Web for UNIX mail servers** solution are available:

- `drweb` anti-virus plug-in;
- `vaderetro` anti-spam plug-in;
- `headersfilter` plug-in, which filters mail messages by their headers;
- `modifier` plug-in, which allows to modify parts of the messages.

drweb anti-virus plug-in

drweb is an anti-virus plug-in for **Dr.Web for UNIX mail servers** solution. It performs anti-virus check of electronic mail.

For proper operation of drweb plug-in **Dr.Web anti-virus Engine** and drwebd module (**Dr.Web Daemon**) are required - they perform direct anti-virus check. drwebd module and anti-virus **Engine** are included into general distribution package of **Dr.Web for UNIX mail servers** solution and must be installed before drweb plug-in.

Messages are sent to drwebd module for scanning in segments. Therefore, MIME-processing support by the **Engine** as well as by drwebd module is not required. Once message analysis is complete, the plug-in sends scanning results to the drweb-maild module and (in case Yes is specified as a value of **AddXHeaders** parameter in the plug-in's configuration file) adds the following headers:

- X-Anti-virus: Name - where Name is the name and the version of anti-virus software;
- X-Anti-virus-Code - where Code is a return code of the drwebd module.

Settings of drweb plug-in can be found in plugin_drweb.conf configuration file.

Installing drweb plug-in

To connect drweb plug-in to **Dr.Web for UNIX mail servers** solution, you must add drweb to the list of plug-ins for message processing in **Dr.Web MailD** configuration file. In case you want to have a message processed by the drweb plug-in before it is moved to the database, you must add the name of this plug-in to the list of values of **BeforeQueueFilters** parameter from the [Filter] section of **Dr.Web MailD** configuration file.

Example:

```
BeforeQueueFilters = drweb, vaderetro
```

If you want to have a message processed by the `drweb` plug-in after it is moved to the database, you must add the name of this plug-in to the list of values of **`AfterQueueFilters`** parameter from the `[Filter]` section of the **Dr.Web MailD** configuration file.

Example:

```
AfterQueueFilters = drweb
```

Setup of drweb plug-in

All the main parameters regulating plug-in operation are set in `%etc_dir/plugin_drweb.conf` configuration file. Description of configuration file structure and parameter types can be found in Configuration Files. Parameters are described in the order they are presented in main configuration file.

In `[anti-virus]` section general settings for drweb plug-in are collected:

`[Antivirus]` section

`Address` = `{socket address}`	Socket for interaction between anti-virus plug-in and `drwebd`. It is possible to specify several sockets for interaction with **Daemons** on different servers. Address at the top of the list is considered to be the main one, and the rest is kept in reserve. Apart from standard address types, this parameter supports PID format. When it is used, real address of `daemon` is retrieved from its PID file. **Examples:** Specifying path to PID file: **`Address`** = `pid:%var_dir/run/drwebd.pid`

	Specifying several addresses: **Address** = pid:%var_dir/run/drwebd.pid, inet:3000@srv2.example.com Default value: **Address** = pid:%var_dir/run/drwebd.pid
Timeout = {time value}	Timeout for drwebd to execute a command. When parameter value is set to 0, time is not limited. Default value: **Timeout** = 30s
HeuristicAnalysis = {Yes | No}	Heuristic analyzer allows drwebd detect unknown viruses. When heuristic analyzer is disabled, only well-known viruses (information on which is stored in virus databases) will be detected. Enabling of heuristic analyzer can result in emergence of false alarms due to similarity in legitimate program activity and virus activity. Usage of heuristic analyzer can also slightly extend scan time. Default value: **HeuristicAnalysis** = Yes
TCP_NODELAY = {Yes | No}	Yes value makes socket operate with **TCP_NODELAY** parameter enabled. Do not change default No value of this parameter if you don't have network problems. Default value: **TCP_NODELAY** = No

ReportMaxSize = {size}	Maximum size of drwebd log file. When **ReportMaxSize** = 0, log file size is not limited. It is not recommended to set parameter value to null, otherwise size of log files can exceed several Mbytes after detection of malware or mail bombs in messages. Default value: **ReportMaxSize** = 50k
AddXHeaders = {Yes | No}	If Yes value is specified, X-Anti-Virus and X-Anti-Virus-Code headers are added to scanned messages. Default value: **AddXHeaders** = Yes
Paranoid = {Yes | No}	If Yes value is specified, messages will be scanned in paranoid mode. With this mode enabled, messages are sent to the **Daemon** segment by segment as well as all-in-one-piece. Such strategy allows to increase possibility of virus detection, but extends scan time. Please note, that if the message contains some object to which action pass is applied, then duplication of statistical information about this object may occur (if some virus is found when the attachment is processed, and when the message itself is processed). Also some additional actions (notify, redirect) may be applied twice. Default value: **Paranoid** = No

RegexsForCheckedFilename = {list of regular expressions}	List of regular expressions, used by anti-virus plug-in to check file names in report, provided by drwebd after message scan. Names of archived files will begin with ">" symbol (number of ">" symbols depends on archive nesting level). If any part of the file name matches some regular expression from the list, action specified in **BlockByFilename** parameter settings is applied. This check is performed only to files, where no viruses have been found. **Example**: **RegexsForCheckedFilename** = "^>.*?\\s{5\\,}" Action, specified in **BlockByFilename** parameter settings, is applied to all messages, containing archives or files with names, including five or more white spaces. Default value: **RegexsForCheckedFilename** =
LicenseLimit = {actions}	Action to be applied to messages which have not been scanned due to license expiration. Mandatory values are: pass, tempfail, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Default value: **LicenseLimit** = pass
Infected = {actions}	Action to be applied to messages, infected with known virus. Mandatory values are: cure, remove, discard, reject. Optional values are: quarantine, redirect, notify.

	Please note that several values may be specified at one time. Default value: **Infected** = cure, quarantine
Suspicious = {actions}	Action to be applied to messages which could be infected with unknown virus. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Default value: **Suspicious** = reject, quarantine, notify
Incurable = {actions}	Action to be applied to incurable messages. Mandatory values are: remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note that several values may be specified at one time. Default value: **Incurable** = reject, quarantine, notify
Adware = {actions}	Action to be applied to messages containing adware. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Default value: **Adware** = reject, quarantine, notify

Dialers = {actions}	Action to be applied to messages containing dialers. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note that several values may be specified at one time. Default value: **Dialers** = reject, quarantine, notify
Jokes = {actions}	Action to be applied to messages containing jokes, which can scare or annoy user. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Default value: **Jokes** = reject, quarantine, notify
Riskware = {actions}	Action to be applied to messages containing riskware. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note that several values may be specified at one time. Default value: **Riskware** = reject, quarantine, notify

Hacktools = {actions}	Action to be applied to messages containing programs used to gain unauthorized access to computer systems. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note that several values may be specified at one time. Default value: **Hacktools** = reject, quarantine, notify
SkipObject = {actions}	Action to be applied to messages containing objects, which cannot be scanned by daemon due to the following reasons: • Password protected or broken archive, symbolic link or non regular file is attached to message. • Message scan is aborted due to timeout. (To find more information refer to the description of **SocketTimeout** and **FileTimeout** parameters in main configuration file drweb32.ini). Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Default value: **SkipObject** = pass
ArchiveRestriction = {actions}	Action to be applied to messages containing archives, which cannot be scanned by daemon due to the excess of limits set for archives in main configuration file drweb32.ini:

	• Archive compression ratio exceeds **MaxCompressionRatio** parameter value in drweb32.ini configuration file. • Size of packed object exceeds **MaxFileSizeToExtract** parameter value in drweb32.ini configuration file. Archive nesting level exceeds **MaxArchiveLevel** parameter value in drweb32.ini configuration file. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time.
	Default value: **ArchiveRestriction** = reject, quarantine, notify
ScanningErrors = {actions}	Action to be applied to messages causing drwebd errors during scan (e.g. **Daemon** has run short of memory or does not have proper privileges for further processing). Mandatory values are: pass, remove, reject, tempfail. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time.
	Default value: **ScanningErrors** = reject, quarantine

ProcessingErrors {actions}	=	Action to be applied to messages causing plug-in errors during scan (e.g. anti-virus plug-in has run short of memory or cannot connect to daemon). Mandatory values are: pass, discard, reject, tempfail. Optional values are: quarantine, redirect, notify. Please note that several values may be specified at one time. Default value: **ProcessingErrors** = reject
BlockByFilename {actions}	=	Action to be applied when one of regular expressions from **RegexsForCheckedFilename** parameter matches any file name in drwebd report. Mandatory values are: pass, discard, reject, tempfail. Optional values are: quarantine, redirect, notify. Please note, that several values may be specified at one time. Please note, that when communication with the drwebd is performed via the TCP socket, a different format of file names will be used in reports. **Example**: 127.0.0.1 [17078] >/var/drweb/msgs/db/6/00007976/.msg/1.part - Ok i.e. they will begin not with the ">" symbol, but with an IP address and the number of the scanning process. So, regular expressions in the value of **RegexsForCheckedFilename** parameter should be created with consideration for this difference. Default value: **BlockByFilename** = reject, quarantine, notify

When message is blocked by the anti-virus plug-in, SMTP-response from the **Dr.Web MailD** has 550 5.7.0 error code and a text message which is determined by values of parameters described below. Values of these parameters must be enclosed in quotation marks.

UseCustomReply {Yes | No}	=	Reply strings to be used as SMTP reply if messages have been rejected. Default value: **UseCustomReply** = No
ReplyInfected {text value}	=	Reply string to be used as SMTP reply when **Infected** = reject or **Incurable** = reject actions are applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:** 550 5.7.0 "Text part of reply" Default value: **ReplyInfected** = "DrWEB anti-virus: Message is rejected because it contains a virus."
ReplyMalware {text value}	=	Reply string to be used as SMTP reply when **Adware**, **Dialers**, **Jokes**, **Riskware**, **Hacktools** = reject actions are applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:** 550 5.7.0 "Text part of reply" Default value: **ReplyMalware** = "DrWEB anti-

	virus: Message is rejected because it contains a malware."
ReplySuspicious = {text value}	Reply string to be used as SMTP reply when **Suspicious** = reject action is applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example**: 550 5.7.0 "Text part of reply" Default value: **ReplySuspicious** = "DrWEB anti-virus: Message is rejected because it contains suspicious content."
ReplySkipObject = {text value}	Reply string to be used as SMTP reply when **SkipObject** = reject action is applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:** 550 5.7.0 "Text part of reply" Default value: **ReplySkipObject** = "DrWEB anti-virus: Message is rejected because it cannot be checked."

ReplyArchiveRestriction = {text value}	Reply string to be used as SMTP reply when **ArchiveRestriction** = reject action is applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:** `550 5.7.0 "Text part of reply"` Default value: **ReplyArchiveRestriction** = "DrWEB anti-virus: Message is rejected because it contains archive which violates restrictions."
ReplyError = {text value}	Reply string to be used as SMTP reply when **ScanningErrors**, **ProcessingErrors**, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:** `550 5.7.0 "Text part of reply"` Default value: **ReplyError** = "DrWEB anti-virus: Message is rejected due to software error."
ReplyBlockByFilename = {text value}	Reply string to be used as SMTP reply when **BlockByFilename** = reject action is applied, and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be quoted if it contains white spaces. **Example:**

	550 5.7.0 "Text part of reply" Default value: **ReplyBlockByFilename** = "DrWEB MailD: Message is rejected due to filename pattern"

headersfilter plug-in

Headersfilter plug-in filters messages according to their headers. When filtration rules are set, regular expressions (Perl syntax) can be used.

Installing headersfilter plug-in

To connect `headersfilter` plug-in to **Dr.Web for UNIX mail servers** solution, you must add `headersfilter` to the list of plug-ins for message processing in **Dr.Web MailD** configuration file. In case you want to have a message processed by the `headersfilter` plug-in before it is moved to the database, you must add the name of this plug-in to the list of values of **BeforeQueueFilters** parameter from the `[Filter]` section of **Dr.Web MailD** configuration file.

Example:

BeforeQueueFilters = drweb, headersfilter

If you want to have a message processed by the `headersfilter` plug-in after it is moved to the database, you must add the name of this plug-in to the list of values of **AfterQueueFilters** parameter from the `[Filter]` section of the **Dr.Web MailD** configuration file.

Example:

AfterQueueFilters = headersfilter

Setup of headersfilter plug-in

All the main parameters regulating plug-in operation are set in `%etc_dir/plugin_headersfilter.conf` configuration file. Description of configuration file structure and parameter types can be found in Configuration_Files. Parameters are described in the order they are presented in main configuration file.

In `[Headersfilter]` section general settings for `headersfilter` plug-in are collected.

Filtration parameters are defined by the rules which are described below. Rules are analyzed in the same order as they are listed in the section, i.e. a rule which is set first in the list is analyzed first. Rules are analyzed until a suitable rule is found and the plug-in executes the action which is set for this rule.

If **`Reject*`** rule is applied to a message, the message is not processed further. If **`Accept*`** rule is applied to a message, other rules are ignored and the message is processed by other plug-ins of the **Dr.Web MailD**.

`[headersfilter]` section

`ScanEncodedHeaders` `= {Yes | No}`	Headers scan before decoding. For example, `Yes` value for **`ScanEncodedHeaders`** parameter and condition **`RejectCondition`** `Subject = "iso-8859-5"` allow to filter out messages, `Subject` field of which is encoded with iso-8859-5. Please note, that with `Yes` value specified all encoded headers will be scanned twice: before and after decoding.
	Default value: **`ScanEncodedHeaders`** `= Yes`

RejectCondition = {set of conditions}	Message filtering rules. If message header matches any condition specified, it is filtered out. Action to be applied for filtered messages can be specified in **Action** parameter of this section. Conditions can be specified for any header. They usually consist of header name and regular expression: `HEADER = regular_expression` You can combine several conditions using brackets or logical operators `OR` and `AND`. "`!=`" (not equal) operator can also be used. Expressions containing white spaces must be enclosed in quotation marks. **Example**: **RejectCondition** `Subject = "money" AND Content-Type = "text/html"` Moreover, there are two additional types of filtration: • `No HEADER` - conditions, suitable for messages without a certain header. **Example:** **RejectCondition** `No From` - allows filtering out all messages without `From` field. • `HEADER = "8bit"` - allows filtering out all messages with headers containing 8-bit symbols.
	Default value: **RejectCondition =**

AcceptCondition = {set of conditions}	Rules for accepting messages. If message header matches any condition specified, scan stops, and message is immediately sent to other plug-ins for further processing. Acceptance conditions can be specified for any header. For more detailed description of **AcceptCondition** parameter refer to description of **RejectCondition** parameter, provided above. Default value: **AcceptCondition =**
FilterParts = {Yes | No}	Processing of rules, specified by **RejectPartCondition** and **AcceptPartCondition** parameters. Default value: **FilterParts** = Yes
RejectPartCondition = {set of conditions} **AcceptPartCondition** = {set of conditions}	Rules are similar to those from **RejectCondition** and **AcceptCondition** parameters, but they affect only headers of attached objects. Also the following condition can be used: **FileName** = mask, where mask is a regular expression, which complies with POSIX 1003.2. Filtration of messages according to these rules is possible if Yes value is specified for **FilterParts** parameter. Default value: **RejectPartCondition =** **AcceptPartCondition** =
MissingHeader = {text value}	Set of missing headers to be used as condition for filtration. **Example:**

	MissingHeader = "To", "From" Default value: **MissingHeader** =
Action = {actions}	Action to be applied to filtered messages. Mandatory values are: pass, tempfail, discard, reject. Optional values are: quarantine, redirect, notify, add-header. Please note, that several values may be specified in one string with comma used as delimiter. Default value: **Action** = reject, notify

When message is blocked by plug-in, SMTP-response from the **Dr.Web MailD** has 550 5.7.0 error code and a text message which is determined by values of parameters described below. Values of these parameters must be enclosed in quotation marks.

UseCustomReply = {Yes | No}	Reply strings to be used as SMTP-reply when messages have been rejected. Default value: **UseCustomReply** = No
ReplyRuleFilter = {text value}	Reply string to be used as SMTP reply when **Action** = reject is applied and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be enclosed in quotation marks if it contains white spaces. **Example:** 550 5.7.0 "Text part of reply" Default value: **ReplyRuleFilter** = "DrWEB HeadersFilter plugin: Message is rejected by headers rule

```
                    filter."
```

vaderetro anti-spam plug-in

Vaderetro is a plug-in used in **Dr.Web for UNIX mail servers** solution. It filters out spam using **VadeRetro** library, designed by Vade Retro Technology company (a division of GoTo Software company).

VadeRetro library analyzes mail in the fully autonomous mode without requesting external sources for additional information on spam. Moreover, the library assures for high processing speed and constantly increasing level of message analysis, which is possible due to the fact, that the library's code is being dynamically updated.

Depending on the results of the analysis, each message processed by the **VadeRetro** library recieves a score - an integer in the range from -10000 to +10000. The less is the value, the higher is the probability that the message is not spam. Threshold is defined by the **SpamThreshold** parameter of the `vaderetro` configuration file. If the evaluation score given to a message is higher than or equal to the value of the **SpamThreshold** parameter, the message is classified as spam.

At the final stage of analysis **VadeRetro** library can add to the message the following headers:

- `X-Drweb-SpamScore: n` , where `n` is the score that VadeRetro assignes to a message.
- `X-Drweb-SpamState: b` , where `b` is **Yes** for spam and infected messages and **No** for not-spam message and mail bounce notifications.
- `X-Drweb-SpamState-Num: s` , where `s` is a message classification result. `s` take the following values: 0, 1, 2 and 3.
  - `s = 0` – this message is not spam;
  - `s = 1` – this message is spam;
  - `s = 2` – this message contains a virus;
  - `s = 3` – this message is a mail bounce notification.

This header is added only in case if **Yes** is specified for the **AddXDrwebSpamStateNumHeader** parameter of the vaderetro configuration file.

- X-Drweb-SpamVersion: version , where version is the version of **VadeRetro** library. This header is added only if **Yes** is specified for the **AddVersionHeader** parameter of the vaderetro configuration file.
- X-Spam-Level: z , where z is a set of "*" (each of them is equal to 10 score points, assigned to a message). This header is added only if **Yes** is specified for the **AddXSpamLevel** parameter of the vaderetro configuration file.
- X-DrWeb-SpamReason: some_text, where some_text is some encoded diagnostic message from the anti-spam module. It is necessary for improvement of the quality of spam detection. This header is added only when **AddXHeaders** parameter for this message is set to yes.

Moreover, vaderetro plug-in can add value of **SubjectPrefix** parameter from the vaderetro configuration file to the subject of the message, classified by the **VadeRetro** library as infected or spam. This works only when any value is specified for the **SubjectPrefix** parameter. For notifications a value of **NotifySubjectPrefix** parameter can be added to the beginning of the **Subject** field and for messages marked as unconditional spam using **UnconditionalSpamThreshold** parameter, a value of **UnconditionalSubjectPrefix** is added to the beginning of the **Subject** field.

Messages, which were marked as spam by mistake must be sent to vrnonspam@drweb.com. Spam messages, accidentally passed by the spam filter, must be sent to vrspam@drweb.com.

Installing vaderetro plug-in

To connect vaderetro plug-in to **Dr.Web for UNIX mail servers** solution, you must add vaderetro to the list of plug-ins for message processing in **Dr.Web MailD** configuration file.

On startup, **Dr.Web MailD** temporarily renames libvaderetro.so to libvaderetro.so.cache to avoid conflict on updating.

In case you want to have a message processed by the `vaderetro` plug-in before it is moved to the database, you must add the name of this plug-in to the list of values of **`BeforeQueueFilters`** parameter from the `[Filter]` section of **Dr.Web MailD** configuration file.

Example :

```
BeforeQueueFilters = drweb, vaderetro
```

If you want to have the message processed by the `vaderetro` plug-in after it is moved to the database, you must add the name of this plug-in to the list of values of **`AfterQueueFilters`** parameter from the `[Filter]` section of the **Dr.Web MailD** configuration file.

Example:

```
AfterQueueFilters = vaderetro
```

Setup of vaderetro plug-in

All the main parameters regulating plug-in operation are set in `%etc_dir/plugin_vaderetro.conf` configuration file. Description of configuration file structure and parameter types can be found in Configuration Files. Parameters are described in the order they are presented in main configuration file.

In `[VadeRetro]` section general settings for `vaderetro` plug-in are collected:

`[Vaderetro]` section

FullCheck = {Yes | No}	Full spam check for each message is performed. After passing this check each message will get a certain score within the range from -10000 to +10000. The less score received, the better chance message is not spam. If the score is greater than or equal to threshold value set by **SpamThreshold** parameter in plug-in configuration file, message is classified as unconditional spam. Please note, that full check can slow down total operating speed. Default value: **FullCheck** = Yes
NoHamFrom = {Yes | No}	When Yes value of this parameter is specified, check of messages sent to embedded addresses of **Dr.Web MailD** (e. g. nospam@domain.ru) is not performed. Default value: **NoHamFrom** = Yes
AddXHeaders = {Yes | No}	X-Drweb-SpamState and X-Drweb-SpamScore headers are added to message. The first one contains information whether or not message is spam. The second one contains grand total of score points upon full check. Default value: **AddXHeaders** = Yes
AddVersionHeader = {Yes | No}	X-Drweb-SpamVersion header with information on VadeRetro version is added to message. Default value: **AddVersionHeader** = No

AddXDrwebSpamStateNumHeader = {Yes | No}	X-Drweb-SpamState-Num header added to message. It includes numerical value, assigned by **VadeRetro** library upon classification: • 0 - this message is not spam; • 1 - this message is spam; • 2 - this message is infected with virus; • 3 - this is mail bounce notification. Default value: **AddXDrwebSpamStateNumHeader** = No
AddXSpamLevel = {Yes | No}	X-Spam-Level header is added to message. It consists of * symbols. Each * symbol costs 10 score points. For example, message with 110 score points will get X-Spam-Level: ********** header. Default value: **AddXSpamLevel** = No
CheckForViruses = {Yes | No}	Enables heuristic check for viruses in spam messages. Default value: **CheckForViruses** = Yes
CheckDelivery = {Yes | No}	Enables check for SMTP non-delivery notifications. Default value: **CheckDelivery** = No
AllowRussian = {Yes | No}	Determine whether to add extra score points to messages with Cyrillic text or not. Default value: **AllowRussian** = Yes

AllowCJK = {Yes | No}	Determine whether to add extra score points to messages with Chinese, Japanese or Korean text or not. Default value: **AllowCJK** = Yes
WhiteList = {lookups}	Index of files containing white lists. In these files approved email addresses are specified, one address per line. Wildcards are allowed. For example, to add email addresses belonging to a certain domain, use "*" symbol in place of the user name: *@mycompany.com. **Example:** hello@myneighbourhood.co.uk *@mycompany.com Default value: **WhiteList** =
BlackList = {lookups}	Index of files containing black lists. In these files banned email addresses are specified, one address per line. Wildcards are allowed. For example, to add email addresses belonging to a certain domain, use "*" symbol in place of the user name: *@hiscompany.com. Default value: **BlackList** =
SubjectPrefix = {text value}	Prefix added to the beginning of **Subject** string, if message is spam. It is added, when message score is greater than **SpamThreshold** value and message is classified as spam. Default value: **SubjectPrefix** =

UnconditionalSubjectPrefix = {text value}	Prefix added to the beginning of **Subject** string, if message is spam. It is added, when message score is greater than **UnconditionalSpamThreshold** value and message is classified as unconditional spam. Default value: **UnconditionalSubjectPrefix** =
NotifySubjectPrefix = {text value}	Prefix added to the beginning of **Subject** string, if message is the mail bounce notification (and rated "3" by **VadeRetro** library). Default value: **NotifySubjectPrefix** =
PathToVadeRetro = {path to file}	Path to **VadeRetro** anti-spam library. Dynamic update is possible via `update.pl` script. It will download new version, replace the old library and send `SIGHUP` signal to `drweb-maild`. Default value: **PathToVadeRetro** = `%var_dir/lib/libvaderetro.so`
UnconditionalSpamThreshold = {numerical value}	If message score is greater than or equal to this parameter value, message is classified as unconditional spam. In this case action specified in **UnconditionalAction** parameter is applied to message. Value specified in **UnconditionalSpamThreshold** parameter must be greater than or equal to the value of **SpamThreshold** parameter. Default value: **UnconditionalSpamThreshold** = 1000

SpamThreshold = {numerical value}	If message score is greater than or equal to this parameter value, message is regarded as spam. In this case action specified in **Action** parameter is applied to message. This check is performed only if message score is less than value of **UnconditionalSpamThreshold** parameter. Value specified in **SpamThreshold** parameter must be lesser than or equal to the value of **UnconditionalSpamThreshold** parameter. Default value: **SpamThreshold** = 100
UnconditionalAction = {actions}	Actions to be applied to unconditional spam. Mandatory values are: pass, remove, discard, reject. Optional values are: quarantine, redirect. Default value: **UnconditionalAction** = pass
Action = {actions}	Actions to be applied to spam messages. Mandatory values are: pass, reject, discard, tempfail. Additional values are: quarantine, redirect. Default value: **Action** = pass
NotifyAction = {actions}	Action applied to mail bounce notifications (which ate rated "3" by **VadeRetro** library). Mandatory values are: pass, reject, discard, tempfail. Additional values are: quarantine, redirect. Default value: **NotifyAction** = pass

UseCustomReply = {Yes | No}	Reply strings to be used as SMTP-reply when messages are rejected. Default value: **UseCustomReply** = No
SpamCustomReply = {text value}	Reply string to be used as SMTP-reply when **Action**, **UnconditionalAction**, **NotifyAction** = reject actions are applied and also when **UseCustomReply** = yes. You can specify only text part of the reply string. Text must be enclosed in quotation marks if it contains white spaces. **Example:** 550 5.7.0 "Text part of reply" Default value: **SpamCustomReply** = "Dr.Web vaderetro plugin: this is spam!"
FromProtectedNetworkScoreAdd = {numerical value}	Adds specified value to the current message score, if the address of a message sender belongs to the **ProtectedNetwork** list. Value may be negative. If you want to disable this function, specify 0 as a value of this parameter. Default value: **FromProtectedNetworkScoreAdd** =
UseReplyCache = {Yes | No}	Enables and disables **ProtectedNetworkReplyCacheLifeTime** and **ReplyToProtectedNetworkScoreAdd** parameters. If these parameters are disabled, then reply_cache storage is not used. Default value:

	UseReplyCache =
ProtectedNetworkReplyCacheLifeTime = {time}	Storage time for entries in reply_cache. If message sender address is listed in **ProtectedNetwork**, then addresses of all the recipients of this message are added to the special storage - reply_cache - for a period of time specified by this parameter. If address of a recipient is already in reply_cache storage, this entry is renewed. For reply messages which senders are in the reply_cache, it is possible to modify their scores using **ReplyToProtectedNetworkScoreAdd** parameter.
	Default value: **ProtectedNetworkReplyCacheLifeTime** =
ReplyToProtectedNetworkScoreAdd = {numerical value}	Value added to scores of messages which senders are listed in reply_cache.
	Default value: **ReplyToProtectedNetworkScoreAdd** =

Modifier plug-in

Dr.Web Modifier plug-in is used for:

- content analysis – search for objects with particular MIME-types (graphics, executable files, media files) and MIME-objects satisfying certain conditions in the bodies of processed messages;
- letter bodies modification – removal of MIME-objects that satisfy certain conditions, modification of headers of chosen MIME-objects and their content;
- used to block, quarantine, redirect, add headers and scores depending on the detected MIME-objects in the bodies of

processed messages.

The following types of regular expressions are supported by the **Dr.Web Modifier**: basic regular expressions, extended regular expressions and Perl-compatible regular expressions.

There are four types of rules, that can be specified in configuration file of **Modifier** plug-in.

Rules of the first type are applied to the whole message:

- `pass`, `accept` - this message is to be passed. When message is processed with global rules, no more checks will be performed for it after receipt of any of the previous commands. When message is processed with local rules, receipt of accept command tells **Dr.Web Modifier** to start processing it with global rules;
- `reject` - reject message;
- `discard` - reject message without any notifications;
- `notify` - notify administrator. After this command, name of the report template to be used in generation of notification must be specified. Otherwise errors will emerge during message processing. Templates reside in the directory, path to which is specified in the value of **`TemplatesBaseDir`** parameter of **Dr.Web MailD** configuration file.

  Example:

  `GlobalRules` = `select message, notify rule`

  Necessary prefix `admin_` and `.msg` extension are inserted by the **Dr.Web Notifier** automatically;

- `tempfail` - report a temporary server failure to sender;
- `redirect` - redirect message to the specified address;
- `quarantine` - send message to quarantine.

`stop` command aborts rule processing. Actions are applied to a message according to already processed rules: `pass`, `accept`, `reject`, etc - and depend on the last executed command.

`accept` rule is almost identical to `pass` + `stop`, with the minor

difference: `accept` aborts processing with local rules. For global rules `accept` is equal to `pass`.

`reject`, `discard` and `tempfail` are ultimate action, which stops message processing and prevents all the following actions from being executed.

Each of these commands must be supplemented with an additional text field to insert notifications.

Example:

```
GlobalRules = select mime.headers Subject
"word1|word2|wordN", if found, reject,
notify rule, quarantine, endif
```

In given example above `notify` and `quarantine` commands will not be executed, since letter handling stops on `reject`.

Example:

```
GlobalRules = select mime.headers Subject
"word1|word2|wordN", if found, notify
rule, quarantine, reject, endif
```

In this example letter will be copied into quarantine, then notification to administrator will be sent and the letter will be rejected.

Example:

```
GlobalRules = select mime.headers Subject
"word1|word2|wordN", if found, tempfail,
endif, select mime.headers Subject
"word1|word2|wordN", if found, pass,
endif
```

In given example in presence words "word1", "word2" or "wordN" letter will be declined and sender will receive server failure notification. Next to the `tempfail` action part of the rule will not be processed.

Each mail message consists of the number of elements: MIME-objects, their headers and content, attached MIME-objects in multi-part messages. With all these objects different actions can be performed: deletion, signature adding, text replacement or modification, etc. All other types of rules are applied to separate elements or sets of elements.

Before each command one of the following instructions must be specified: `select`, `or`, `and`, `nand`, `nor`. After each command parameters of selection are specified.

Rules of the second type are applied to separate elements:

- `select message`

  This command selects root MIME element of a mail message;

- `select mime(headers), select mime.headers`

  `select mime(prologue), select mime.prologue`

  `select mime(body), select mime.body`

  `select mime(epilogue), select mime.epilogue`

  These commands select various MIME-objects. The difference between command with brackets and command with dots is that the first one selects MIME-objects with specified element, and the last one selects the element itself.

  Example:

  ```
  select mime(headers) Content-type "x-video"
  remove
  ```

  This set of commands removes all video elements from the message.

  ```
  select mime.headers Content-type "x-video"
  ```

remove

This set of commands remove information about data type from all video elements. Complex MIME-objects can be selected only if they are messages themselves.

- select mime(headers) *header_name regular_expression_to_match_header_body*
  select mime(prologue) *regular_expression*
  select mime(body) *regular_expression*
  select mime(epilogue) *regular_expression*

  These commands select elements with text matching the specified template.

- select sender <*regular_expression*>
  select recipient <*regular_expression*>

  These commands select entries with information about recipients and sender. This information is taken from the envelope. When the required symbol sequence is found, select sender and select receiver commands are processed as select message command.

  Example:

  You can add a greeting to the end of the message for administrator with the following commands:

  ```
  select recipient "root@localhost",
  append_text "hello, root"
  ```

Sometimes it turns out to be necessary to select elements according to several criteria. To make such operation possible, you may combine rules with corresponding logical operators:

- and - leave only those items in the selection, which are compliant to the specified rule;
- nand - leave only those items in the selection, which are NOT compliant to the specified rule;
- or - add only those items to the selection, which are compliant to the specified rule;

- `nor` - add only those items to the selection, which are NOT compliant to the specified rule.

Please note, that these operators can be used only with selections of multi-part MIME-objects, not with separate parts of such objects.

Example:

Select elements written in `html` with "`<script`" part in the text:

```
select mime(headers) Content-type html
and mime(body) "\<script"
```

These are two separate rules, and they are applied sequentially. The first one selects all elements with `html` in `Content-type` header. The second one chooses from the previously selected elements those with "`<script`" set of symbols (case is not important).

Example:

```
select mime(headers) Content-type html
nand mime(body) "\<script"
```

According to the first criteria all elements with `html` in `Content-type` header will be selected. According to the second criteria elements with "`<script`" part in the text will be excluded from this selection.

Example:

```
select mime(headers) Content-type html
or mime(body) "\<script"
```

According to the first criteria all elements with `html` in `Content-type` header will be selected. According to the second criteria elements with "`<script`" part in the text will be added to this selection.

Example:

```
select mime(headers) Content-type html
```

```
nor mime(body) "\<script"
```

According to the first criteria all elements with `html` in `Content-type` header will be selected. According to the second criteria elements without "`<script`" part in the text will be added to this selection.

If you specify `select` before the sequent rule, previous selection will be undone.

Example:

```
select mime(headers) Content-type html
select mime(body) "\<script"
```

According to the first criteria all elements with `html` in `Content-type` header will be selected. According to the second criteria previous selection will be discarded, and only elements containing "`<script`" part in the text will be selected.

If no operators are specified, all sequent rules are ignored and selection stays unmodified.

Example:

```
select mime(headers) Content-type html
mime(body) "\<script"
```

The selection will be made according to the first criteria only - i.e. only elements with `html` in `Content-type` header will be selected.

To make **Modifier** plug-in compatible with `vaderetro` plug-in comparison instructions ">n" and "<n" can be used in search through message headers. Modifier rules can be applied to a certain header if it contains an integer (e.g. `X-Drweb-SpamScore "30"`) and matches some rule (e.g. `select mime(headers) X-Drweb-SpamScore "<50"`)

In this case back slash before the "<" symbol is not required, because otherwise - with `select mime(headers) X-Drweb-`

`SpamScore "\<50"` rule - elements with `X-Drweb-SpamScore "<50"` header will be selected.

`select_mimes` command allows to select MIME-objects according to their headers. It accelerates plug-in operation when it is necessary to select headers and objects according to the same criteria. If you want to select the whole object, you may select only one element from this object.

Rules of the third type are used for modification of selected elements.

They are applied only to the content of MIME-objects.

- `replace` *expression_for_replacement regular_expression_to_be_replaced*

  `replace_all` *new_text*

  These commands replace one text with another.

  Example:

  Renaming of executable files in attachments:

```
select mime.headers Content-disposition "filename=.*\\.exe",\
or mime.headers Content-type "name=.*\\.exe",\
replace "\\.ex_" "\\.exe",\
pass
```

  These commands don't work for multi-part message parts, i. e. for message, which consists of multi-part MIME-object with two sub-objects, the following commands:

```
select message
replace_all «text»
```

  will not make any effect, because multi-part objects themselves don't contain data, but serve as containers for other objects only.

For `replace` and `replace_all` commands function calls can be used as *expression_for_replacement* and *new_text*. You may specify them as `${func_name}`. Argument for these functions is a current *regular_expression_to_be_replaced*.

The following functions are supported:

- `urlencode` - encode argument to a string, which can be used as URL;
- `self` - return the regular expression unmodified.

Example:

```
select mime.headers "Subject" "^.*$", replace_all "old:${self} new:${lc}"
```

`Subject` header of the message complied with the specified pattern (e.g. "`This is Subj`") will be replaced by: "`old: This is Subj new:this is subj`".

Example:

```
select mime.body ".*", replace "Upper:${uc}" "http://\\S+"
```

Some text from the message body, complied with the specified pattern (e.g. "`Text1 http://vasya.pup.kin Text2`") will be replaced with: "`Text1 Upper:HTTP://VASYA.PUP.KIN Text2`".

Example:

```
select mime.body ".*", replace "http://check-url.com?url=${urlencode}" "http://\\S+"
```

Some text from the message body, complied with the specified pattern (e.g. "`Visit http://vasya.com?id=3`") will be replaced with: "`Visit http://check-`

```
url.com?url=http%3A%2F%2Fvasya%2Ecom%3Fid%
3D3".
```

- `remove`

  This command removes all types of selected objects except for root MIME-object.

  Example:

  `remove` command cannot be used in rules like the following:

  ```
  GlobalRules = select mime(body) "text",
  remove, pass
  GlobalRules = select mime(body) "script",
  remove, pass
  ```

- `prepend_text`
  `append_text`
  `prepend_html`
  `append_html`

  These commands add plain text or html fragments to selected MIME objects (e.g. signature to the message).

  Example:

  ```
  select message
  append_html "<h1>checked by anti-spam</
  h1>" [[7b:]encoding]
  ```

  These commands append a signature to the message in encoding set by optional parameter `encoding` and with 7-bit `context transfer` encoding, set by prefix `"7b:"`.

  Language files (with `.lng` extension) can be used as a source, if it is necessary to insert text in specific encoding. To select the required string from `.lng` file use `$1, $2 ... $n` parameters, where `n` - is the number of a string in `.lng` file.

Example:

If `.lng` file looks as follows:

```
1 = string1
2 = some other string
...
```

then `append_text $2` is equal to `append_text "some other string"` command.

It is also possible to use lookups via `LookupsLite` value type, where only values themselves or "file" type of lookups can be used.

Example:

```
append_text "lookup:file:path_to_file"
```

A header may be added to the message with the following commands:

```
select message, addheader "foo:bar"
```

These set of commands add a header with `foo` name and `bar` value to the selected message element. Name and value of a header are delimited by colon.

Rules of the fourth type are used for creation of if/else structures:

- `goto` - unconditional transfer;
- `goto(y)` - conditional transfer if at least one element was selected;
- `goto(n)` - conditional transfer if no elements were selected.

A positive integer can be used as parameter value, specifying how many rules must be skipped in one hop.

Example:

If you want to reject all messages with executable files in attachments, you may use the following set of commands:

```
mime(header) Content-type "executable"
goto(n) 1
reject
```

The code above is executed as follows:

```
selection=find(mimes with content type "*executable*")
if(selection){
    reject mail;
}
```

It is also possible to use `if [not] found .... else .... endif` commands.

Example:

```
select mime.headers "X-DrWeb-SpamState" "yes",\
if found,\
select mime(headers) Content-type "image",\
remove,\
endif,\
```

The above set of commands allows to remove all images from the message marked as spam by **Vaderetro** plug-in.

Please note, that when quotation marks are used in regular expressions, it may become necessary to escape them with several "\" symbols for correct parsing or rules. In current version of the program for escaping of the quotation mark six "\" symbols are required.

Example:

```
GlobalRules = select mime.headers Subject ".*\\
\\\\"", if found, reject, endif
```

Also there exists a possibility to check score of each message is added. Initial score assigned to a message in the beginning of processing is equal to 0. During processing plug-ins can modify message score. Is is possible to check and modify message score using if score, add_score and set_score commands. "if score" works similar to "if found" command, but checks only message scores (i.e. it ignores results of all previous "select" commands).

Please note that due to processing configuration file with multiple parsers, for escaping of the backslash seven "\" symbols are required.

Example:

```
GlobalRules = select mime.headers "Subject" "^\
\\\\\\\$", if found, reject, endif
```

Reject letters with the solely "\" symbol in the subject.

Example:

```
....
if found,\
   set_score 10,\
endif,\
```

Sets 10 as a new score of a message, if it is complied with certain conditions.

Example:

```
....
add_score 11,\
```

Increases message score by 11.

Example:

```
....
if score >100,\
```

```
   reject,\
else,\
   add_score -5,\
endif
```

If message score is greater than 100, then this message is rejected. Otherwise its score is reduced by 5.

`if score` argument must be specified as a single string, without white spaces - i.e. `<100` but not `< 100` - and must contain comparison operation symbol and an integer argument.

The following comparison operations may be used with `if score`:

- `if score <2` - if score is less than 2
- `if score >5` - if score is greater than 5
- `if score =8` - if score is equal 8

Integer argument - is a 32-bit integer within the range from -2bln to +2bln. The score can be overflown during message processing - and this overflow can cause incorrect operation of other modules. So, it is strongly NOT recommended to use unreasonably big score values in rules (e.g. to specify 2000000000 for the `add_score` parameter).

After you append some text to selected MIME-objects, selection will be discarded.

	remove	replace	replace all	append text	prepend text	append html	prepend html	add header	add score	set score	accept	discard	reject	tempfail	notify	redirect	quarantine
mime.header	+	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-
mime.prologue	+	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-
mime.epilogue	+	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-
mime.body	+	+	+	-	-	-	-	-	-	-	-	-	-	-	-	-	-
mime(headers)	+	*	*	-	-	-	-	+	-	-	-	-	-	-	-	-	-
mime(prologue)	+	*	*	-	-	-	-	+	-	-	-	-	-	-	-	-	-
mime(epilogue)	+	*	*	-	-	-	-	+	-	-	-	-	-	-	-	-	-
mime(body)	+	*	*	-	-	-	-	+	-	-	-	-	-	-	-	-	-
sender	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
recipient	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
message	-	-	-	-	-	-	-	-	+	+	+	+	+	+	+	+	+

Table 1. Influence of various rules on different types of objects.

- ***** the same as for the mime.body
- **+** applied
- **-** ignored

Examples:

- select elements according to the complex criteria:

```
GlobalRules = select mime(headers) Content-type
"text", and mime(body) "typical spam",\
```

- if such elements are found, discard the message:

```
goto(n) 1,\
discard,\
```

- otherwise select all executable files and remove them:

```
select mime(headers) Content-disposition ".
exe",\
remove,\
```

- add signature to the message body:

```
select message, append_text "checked!"
```

Please note, that there must be no white spaces after `goto(n) 1,\` and `discard,\`. Otherwise the rule will not work.

Adding html file to the message:

```
GlobalRules = select message, append_html "lookup:file:/maild-files/somehtml.html"
```

Removing messages from selected users:

```
GlobalRules = select mime(headers) From "weirdohacker@server.net", if found, reject, endif
```

Redirecting messages:

```
GlobalRules = select mime.headers To "someaddress@my-net.com", replace_all "anotheraddress@my-net.com"
```

In this example original message will be delivered to the `someaddress@my-net.com` and its copy will be sent to the `anotheraddress@my-net.com`.

If you do not want the message to be delivered to the original recipient, then you may use the following rule:

- select messages according to the specified criteria:

```
GlobalRules = select mime.headers Subject "Help",\
if found,\
select mime.headers To "someaddress@my-net.com",\
if found,\
```

- redirecting selected messages to the specified address:

```
redirect "anotheraddress@my-net.com",\
```

- removing selected messages to prevent them being delivered to original recipients:

```
discard,\
endif,\
stop,\
endif,\
```

Redirecting messages according to their subject:

```
GlobalRules = \
```

- checking messages for support department:

```
select mime.headers Subject "support|bugreport
[s]|help",\
if found,\
```

- the following commands will be passed, if the template was not found:

```
select mime.headers To "@company.com", \
if found,\
redirect "support@company.com",\
endif,\
pass, \
endif,\
```

- the following commands will be executed if the client want to place an order:

```
select mime.headers Subject "price|buy|order",\
if found,\
select mime.headers To "@company.com", \
if found,\
redirect "sell@company.com",\
endif,\
pass, \
```

```
endif,\
```

- all other topics:

```
select mime.headers To "@company.com", \
redirect "inbox@company.com",\
pass
```

Search for executable files in attachments and rename them:

```
select mime.headers Content-disposition "filename=.*\\.exe", or mime.headers Content-type "name=.*\\.exe",\
replace "\\.ex_" "\\.exe",\
pass
```

Installing Modifier plug-in

To connect **Modifier** plug-in to **Dr.Web for UNIX mail servers** solution, you must add modifier to the list of plug-ins for message processing in **Dr.Web MailD** configuration file. In case you want to have a message processed by the **Modifier** plug-in before it is moved to the database, you must add the name of this plug-in to the list of values of **BeforeQueueFilters** parameter from [Filter] section of **Dr.Web MailD** configuration file.

Example:

```
BeforeQueueFilters = modifier
```

If you want to have a message processed by the **Modifier** plug-in after it is moved to the database, you must add the name of this plug-in to the list of values of **AfterQueueFilters** parameter from the [Filter] section of the **Dr.Web MailD** configuration file.

Example:

```
AfterQueueFilters = modifier
```

Setup of Modifier plug-in

All the main parameters regulating plug-in operation are set in `%etc_dir/plugin_modifier.conf` configuration file. Description of configuration file structure and parameter types can be found in Configuration Files. Parameters are described in the order they are presented in main configuration file.

In `[Modifier]` section general settings for **Modifier** plug-in are collected:

`[Modifier]` section

`GlobalRules` = `{list of rules}`	List of general rules for message processing. More detailed description and examples you can find in readme file. Rules are set in **`GlobalRules`** parameter, with comma used as delimiter. There are four groups of rules:
	• commands, which affect e-mail message in whole. Possible actions:
	  ○ `pass` - this message is to be passed, no more checks will be performed;
	  ○ `reject` - reject email and notify sender;
	  ○ `discard` - reject email without any notifications;
	  ○ `notify` - send a notification;
	  ○ `tempfail` - report a temporary server failure to sender.
	• commands, which affect selected parts of e-mail messages.
	• commands, used to modify selected elements.
	• commands, used to create `if/else` condition sets.
	Example:

	The rule below adds to the message a file in html format: **GlobalRules** = select message, append_html "lookup:file:/maild-files/somehtml.html" The rule below deletes messages from specified users: **GlobalRules** = select mime (headers) From "weirdohacker@server.net", if found, reject, endif
	Default value: **GlobalRules** = rule1, rule2, ...
Encoding = {text value}	Encoding specified by plug-in for text, inserted with commands append_text and prepend_text directly from rules.
	Default value: **Encoding** = koi8-r
UseCustomReply = {Yes | No}	Send SMTP reply specified in **ReplyRuleFilter** parameter when incoming message is rejected by modifier plugin.
	Default value: **UseCustomReply** =
ReplyRuleFilter = {text value}	Reply string to be used as SMTP reply when message is rejected by modifier plugin.
	Default value: **ReplyRuleFilter** =

Integration with Mail Transfer Systems

The subject of this chapter is integration of **Dr.Web for UNIX mail servers** solution with various mail transfer systems. To simplify integration process configuration scripts for specific MTAs are included into distribution package.

`configure_mta.sh` script is responsible for setting up interaction between **Dr.Web for UNIX mail servers** solution and currently used mail system. After startup it checks whether the required mail system is installed. If it appears to be missing, the script finish its operation. If the required mail system is installed, the script asks the user several questions about some essential settings necessary for basic setup. Setup can be performed manually as well (please, refer to corresponding chapters of this manual for additional information).

`configure_mta.sh` script configures MTA as follows:

- Connection using special transport is performed for Exim;
- After-queue Mode configuration is performed for Postfix;
- Zmailer is configured to be used in context filter mode at the stage of SMTP-session.

Thus, for example, to configure Postfix to operate using Milter protocol, you should configure MTA according to the steps described in the corresponding section instead of launching `configure_mta.sh` script.

Integration with SMTP-proxy Mode

Dr.Web MailD can operate as a proxy server for mail protocols, which allows to use it with a great number of mail systems. In this mode `drweb-receiver` module works as SMTP/LMTP server and `drweb-sender` module works as SMTP/LMTP client. Moreover, `drweb-sender` module can transfer messages directly to the local mail system.

drweb-receiver contains a high-performance SMTP-server, implemented using modern multiplexors (like epoll, kevent and /dev/poll). This SMTP-server is multithreaded. It supports several connections per each thread, IPv6 protocol and the number of SMTP-extensions:

- PIPELINING (RFC2920)
- 8BITMIME (RFC1652)
- ENHANCEDSTATUSCODES (RFC3463)
- SIZE (RFC1870)
- AUTH (RFC4954)

When mail is received directly from Internet, several new technologies recently implemented in drweb-receiver will make mail filtering easier and more efficient. Restrictions and Reputation IP Filter allow to filter mail on the stage of SMTP-session (and prevent DHA-attacks, for example).

All settings for drweb-receiver and drweb-sender modules are collected in [Receiver] and [Sender] sections of **Dr.Web MailD** configuration file and are described in Receiver and Sender of the current Manual.

Integration with CommuniGate Pro

Setup of CommuniGate Pro

To make CommuniGate Pro (CGP hereafter) send and receive messages from **Dr.Web MailD**, do the following:

1. Connect to CGP using `WebAdmin` program for remote administration;
2. Go to `Settings -> General -> Helpers` menu;
3. Add a new `content-filter` with the following parameters:
   `Use Filter:` `DrWeb Maild`
   `Log: Problems`
   `Path:` `%bin_dir/drweb-cgp-receiver`
   `Time-Out:` `2 minutes`
   `Auto-Restart:` `15 seconds`
4. Check whether privileges with which CGP is executed are appropriate for startup of `drweb-cgp-receiver`;
5. Go to `Settings -> Queue -> Rules` menu;
6. Create a new rule like "`check messages of less than N bytes`".

To create a new rule, do the following:

1. Select the name of the rule (for example, `drweb-filter`) and click **Create New** button;
2. Click **Edit** button and specify `External Filter` value in the **Action** field;
3. In **Parameters** field enter the same value as in the **Filter** field from `Settings -> General -> Helpers` menu.

To avoid a repeated check of messages received from `GROUP`, `LIST` or `RULES` (http://www.communigate.com/

CommuniGatePro/Transfer.html), you can add the following setting to the rule:

```
"Submit Address", "not in", "GROUP*,LIST*, RULES*"
```

When message is uploaded through `PIPE`, authenticated flag may be lost. Then if there were some plug-in in **`AfterQueueFilters`** list, you must add the following line to the rule:

```
Any Recipient not in alldomains@main.domain, all@*
```

where `main.domain` is the main domain of CGP server.

For information on advanced settings (i.e. enabling or disabling filtering for each user) refer to the documentation distributed with CGP.

Setup of Dr.Web MailD

In interaction with CGP, `drweb-cgp-sender` module of **Dr.Web MailD** acts as a **Sender** component. The module is started with privileges of the `mail` group, which makes it possible to write to `cgp` directory. `drweb-cgp-receiver` module of **Dr.Web MailD** at the same time acts as a **Receiver** component. The module is started by CGP mail system itself with `root` privileges.

To assure proper operation of **Dr.Web MailD** in such a configuration you must explicitly specify the name of the user with which privileges other **Dr.Web MailD** modules are started. This name may be set in the **`ChownToUser`** parameter of the `[CgpReceiver]` settings section in the **Dr.Web MailD** configuration file, or you may leave this parameter without a value and run the whole complex with root privileges.

As `drweb-cgp-sender` transfers new messages to CGP via `PIPE` driver, then to avoid cycling of messages you must add a special header to them. This header is set by **`UseSecureHash`** and **`SecureHash`** parameters of the `[CgpSender]` section of the **Dr.**

Web MailD configuration file.

In this case `drweb-cgp-receiver` module will pass messages with this header without check. You can also disable usage of this header by setting `No` as a value of the **`UseSecureHash`** parameter in the [`CgpSender`] section of the **Dr.Web MailD** configuration file. It will instruct `drweb-cgp-receiver` module to pass without check all the messages received from `PIPE` driver.

All settings providing proper operation of **Dr.Web MailD** with CGP are collected in `[CgpReceiver]` and [`CgpSender`] sections of **Dr.Web MailD** configuration file and are described in CgpReceiver and CgpSender chapters of the current Manual.

When **Dr.Web MailD** work with CGP the following processes should be running:

- `drweb-notifier`
- `drweb-cgp-sender`
- `drweb-maild`

Operation Principles

Dr.Web MailD interacts with CGP in the following way:

- A letter comes to CGP.
- After checking settings CGP sends message to check to `drweb-cgp-receiver` (which acts as a helper) if necessary.
- On receive the letter `drweb-cgp-receiver` looks for `SecureHash` header:
  - if header is founded, `drweb-cgp-receiver` returns `OK` status and passes the letter to CGP for further processing;
  - otherwise, the message passes for check to `drweb-maild`;
- `drweb-maild` applies to letter plug-ins that can modify it (for example, add headers);

- if no viruses found and letter has not been changed, OK status returns to CGP;
- if during the processing letter was changed, `DISCARD` status returns to CGP and further processing of the letter is carried by `drweb-maild` due to the fact that helper protocol doesn't allow changed letter to be returned.

- The letter passes to **Sender** and, after adding `SecureHash` header (if `UseSecureHash = yes`), moves to submit directory `/var/CommuniGate/Submitted/` that is periodically checked by CGP.

Value of **SubmitDir** parameter of **Dr.Web MailD** configuration file has to be equal `/var/CommuniGate/Submitted`. Otherwise letters checked by **Dr.Web MailD** will not reach reach recipients.

- After checking the `/var/CommuniGate/Submitted/` directory and reception of the letter, CGP moves to step 2:
  - in case of correct settings, the letter will not be checked again;
  - in case of inaccuracies in settings, the letter will be passed back to CGP after checking `SecureHash` header;
  - in case of incorrect settings, an infinite checking loop is possible (or until CGP will not detect it).

Known Problems

In OS Linux after the alteration and update of the command line via `Helpers` setting the previous process of the filter remains in the `zombie` condition until restart of the CGP.

Description:

When `drweb-cgp-receiver` is started, the following messages are displayed:

```
/usr/libexec/ld-elf.so.1: Shared object "libstdc++.so.6" not found, required by
```

"libboost_thread.so"

Solution:

System cannot find necessary libraries located in the %bin_dir/lib/ directory. It is necessary to copy libraries libstdc++.so.6 and libgcc_s.so.1 (or make symbolic links to them) from %bin_dir/lib/ to the system directory with libraries.

Integration with Sendmail MTA

To coordinate operation of Sendmail system with **Dr.Web MailD**, Sendmail system requires Milter API support. If your Sendmail system does not support Milter API library, you must rebuild your SendMail system to add this library to the supported libraries. For more information refer to the corresponding documentation on your Sendmail system.

Be sure that **SecureHash** parameter value in the [Sender] section of the **Dr.Web MailD** configuration file is specified (arbitrary string of symbols can be set as parameter value, recommended length is more than or equal to 10 symbols). Also Yes value must be specified for **UseSecureHash** parameter in the same section.

MailD is fully compatible with Sendmail versions 8.12.3 and above. When working with earlier version some issues may occur (see Known Problems section). Detailed instructions for integration in this documentation is written for Sendmail version 8.14.0 or above.

Interaction between Sendmail MTA and **Dr.Web MailD** is performed via Milter API (drweb-milter module is used as a **Receiver** component) and is implemented as follows:

- Through the transport connection defined by `drweb-milter` transport address `__ADDRESS__`, Sendmail system receives internal commands from `Milter API` and the message itself. The message is transferred in segments depending on the stage of the mail session (`helo`, `mail from:`, `rcpt to:`, etc.). Therefore, the message is saved by `drweb-milter` module to the temporary directory. Through `Milter API`, `drweb-milter` transmits instructions about the message to the Sendmail system.

  `Milter API` is a multithreaded library - several mail sessions can be processed simultaneously. In the interaction scheme given above Sendmail system is a client and `drweb-milter` is a server, therefore in the `sendmail.cf` configuration file `drweb-milter` address must be specified, and Sendmail system chooses the appropriate client address for this connection;

- Through another transport connection `drweb-milter` module transfers commands to `drweb-maild` module and waits for response.

In the scheme given above `drweb-milter` module works as agent between the Sendmail system interface and `drweb-maild` module. Sendmail and `drweb-milter` module can operate on different computers; `drweb-milter` and `drweb-maild` modules must operate on the same computer.

Setup of Sendmail

To set up interaction between Sendmail and **Dr.Web MailD** you may need to make changes to `sendmail.mc` and `sendmail.cf` configuration files.

If you do not want to recompile the `sendmail.cf` configuration file, you can just insert or add there the following lines (if correspondent definitions are already present in the file):

For versions 8.14.0 and later:

```
-------------------- cut ------------------
```

```
############################
# Input mail filters
############################
O InputMailFilters=drweb-filter
O Milter.LogLevel=6
############################
#      Xfilters
############################
Xdrweb-filter,  S=__ADDRESS__,
F=T, T=C:1m;S:5m;R:5m;E:1h
------------------- cut ---------------------
```

To check locally sent messages (with `mail` or `sendmail` system call), all changes made to `sendmail.cf` configuration file must be copied to `submit.cf` and `submit.mc` files.

Please note, that `submit.cf` and `submit.mc` files are read-only by default, so you must change access permissions (providing write access) before making any changes to these files. Moreover, you must add `nobodyreturn` value to the **`O PrivacyOptions`** parameter.

Example:

```
-------------------- cut -----------------
# privacy flags
O PrivacyOptions=goaway,noetrn,nobodyreturn
-------------------- cut -----------------
```

Or in `{sendmail_src}/cf/cf/feature/msp.m4`:

```
-------------------- cut -----------------
define(`confPRIVACY_FLAGS'
`goaway,noetrn,nobodyreturn,restrictqrun')
-------------------- cut -----------------
```

If filter is not available, you can enable the following flags (`F=`):

- `R` – fail to deliver;
- `T` – delay delivery.

If neither `F=R`, nor `F=T` is specified, the message is passed without check.

You may also add to `sendmail.mc` the following lines:

For versions 8.14.0 and later:

```
------------------- cut ---------------------
INPUT_MAIL_FILTER(`drweb-filter',
`S=__ADDRESS__,
F=T, T=C:1m;S:5m;R:5m;E:1h')
define(`confMILTER_LOG_LEVEL',`6')
------------------- cut ---------------------
```

Timeout must be set according to values of timeouts set for Sendmail:

```
O Timeout.datablock=XX
```

(the default value is 1 hour, `XX=>1h`).

After making changes to `sendmail.cf` configuration file you must recompile it.

`__ADDRESS__` string specifies the address of transport used to connect to `drweb-milter`. Its format and value are the same as those used in **`Address`** parameter from [`Milter`] section of **Dr.Web MailID** configuration file.

For TCP-sockets address must specified in the following format:

```
inet:__PORT__@__HOST__
```

where `__PORT__` and `__HOST__` must have definite values (e. g. `inet:3001@localhost`).

For UNIX sockets address must specified in the following format:

```
local:__SOCKPATH__
```

where `__SOCKPATH__` string must define the path accessible with privileges the filter is started (e.g. `local:/var/run/drweb-milter.sock`).

Additional information about filter configuration can be found in documentation for Sendmail system. You must restart Sendmail after specifying values for all necessary parameters.

To enable logging of identifiers of Sendmail messages in `drweb-maild` module (`sendmails message ID`) as well as sending to `drweb-maild` information about successful authorization, the following line must be included in `sendmail.cf`:

```
------------------- cut ---------------------
O Milter.macros.envfrom=i,{auth_type}, ...
------------------- cut ---------------------
```

(suspension points are for any other parameters).

To allow **Dr.Web MailD** define IP-address and host name of the sender as well as to transfer interface address of the recipient to `drweb-maild` module, include in `sendmail.cf` configuration file the following line:

```
------------------- cut ---------------------
O Milter.macros.connect=_,{if_addr}, ...
------------------- cut ---------------------
```

(suspension points are for any other parameters).

To disable output to `syslog` the following messages:

```
------------------- cut ---------------------
X-Authentication-Warning:      some.domain.com:
drweb set sender to DrWeb-DAEMON@some.domain.
com using -f
```

```
------------------- cut --------------------
```

you must include user, with which privileges `drweb-milter` operates (`drweb` user by default), to the `trusted-users` list in `submit.cf`. This can be done by adding a user to the list directly in `submit.cf` and `sendmail.cf` configuration files:

```
------------------- cut --------------------
####################
#   Trusted users   #
####################
Tdrweb
------------------- cut --------------------
```

Or by adding the following line to the `submit.mc` file:

```
------------------- cut --------------------
define(`confTRUSTED_USERS', `drweb')
------------------- cut --------------------
```

Setup of Dr.Web MailD

All settings providing proper operation of `drweb-milter` with **Sender** component are collected in [`Sender`] and [`Milter`] sections of **Dr.Web MailD** configuration file and are described in Sender and Milter chapters of the current Manual.

When **Dr.Web MailD** work with mail system Sendmail the following processes should be running:

- `drweb-notifier`
- `drweb-sender`
- `drweb-maild`
- `drweb-milter`

Known Problems

Description:

When UNIX socket is used for communication between the filter and Sendmail, `Milter API` library (distributed with Sendmail) did not remove (prior to version 8.12.2) the file used for socket.

Solution:

For versions 8.12.x the following patch is available - `listener-8.12.0-1.patch`. For versions 8.11.x this file must be removed manually or via script which controls the filter. The issue is resolved in Sendmail version 8.12.2

Description:

When demo key is used in local scan mode, `size` value of the message being transmitted to the next server is doubled on passing through the filter (the message itself either remains unchanged or a tiny "banner" message is added to it).

Solution:

Current issue is resolved in Sendmail version 8.12.3 and later.

Description:

When filter is active on heavily loaded computers the following entries can be found in the log:
`"... Milter (drweb-filter): select(read): interrupted system call"`

Solution:

Current issue is resolved in Sendmail version 8.12.3 and later.

Description:

When filter is active on heavily loaded computers the following entries can be found in the log:

```
"... Milter (drweb-filter): select(read): timeout before data write"
"... Milter (drweb-filter): to error state"
```

Solution:

The problem is that Sendmail cannot establish connection with the filter within the specified timeout. In versions 8.11 and later the timeout is set to 5 seconds and cannot be changed. In versions 8.12 and later this timeout changes in the description of the filter (value for C):

```
Xdrweb-filter, S=__ADDRESS__, F=T, T=C:1m; S:5m;R:5m;E:1h
```

Integration with Mail Postfix

Main Operation Principles

Dr.Web MailD can be connected to Postfix in three different ways:

- In `after-queue` mode (http://www.postfix.org/FILTER_README.html#advanced_filter);
- In `before-queue` mode (http://www.postfix.org/SMTPD_PROXY_README.html);
- Using `milter` protocol (http://www.postfix.org/MILTER_README.html).

Only Postfix versions 2.3.3 and above can be used with milter protocol.

Operation in After-queue and Before-queue Modes

In `after-queue` mode **Dr.Web MailD** interacts with Postfix in the following way:

`drweb-receiver` acting as SMTP/LMTP server receives a new message from Postfix SMTP-module and redirects it to `drweb-maild` module for analysis. According to the results of analysis message is either sent to the mail system (may be as a modified copy) or blocked (in this case additional reports can be sent to the mail system). Redirecting messages to Postfix mail system is performed via `drweb-sender` acting as SMTP/LMTP client, which dispatches messages to `smptd` daemon.

For more detailed information on configuring filters for Postfix refer to the Postfix documentation, which can be found, for example, at http://www.postfix.org/FILTER_README.html.

Dr.Web MailD can also interact with Postfix server in the `before-queue` mode as well (but we do not recommend to use this mode if system load is great). For more information on configuration for operation in `before-queue` mode refer to the http://www.postfix.org/SMTPD_PROXY_README.html.

Operation using Milter Protocol

Interaction with Postfix system via milter protocol is ogranized in the following way:

- Through the transport connection defined by the transport address of `drweb-milter` (which is acting as **Receiver** component) Postfix system receives internal commands from `Milter API` and the mail message itself. The message is transferred in segments depending on the stage of the mail

session (`helo`, `mail from:`, `rcpt to:`, etc.). These segments are saved by `drweb-milter` module to the temporary directory. Through `Milter API`, `drweb-milter` transmits instructions to the Postfix system about actions to be applied to the message.

`Milter API` is a multithreaded library, which allows to process several mail sessions simultaneously. In the interaction scheme described above Postfix system is a client and `drweb-milter` is a server, therefore in the `mail.cf` configuration file of Postfix system address of `drweb-milter` module must be specified, and Postfix system chooses the appropriate client address for this connection.

- Through another transport connection `drweb-milter` module transfers `drweb-maild` module commands and waits for response.

In the scheme given above `drweb-milter` module works as agent between the Postfix system interface and `drweb-maild` module. Postfix and `drweb-milter` module can operate on different computers; `drweb-milter` and `drweb-maild` modules must operate on the same computer.

Mail Postfix Setup

Setup of Operation in After-queue Mode

To set up interaction between **Dr.Web MailD** and Postfix in `after-queue` mode add the following lines to the `main.cf` configuration file of the Postfix system:

```
content_filter = scan:_ADDR_REC_
receive_override_options = no_address_mappings
```

where `_ADDR_REC_` is the address of the listening `drweb-receiver` module (**`Address`** parameter of the `[Receiver]`

section of **Dr.Web MailD** configuration file) - `127.0.0.1:8025`, for example.

In the `master.cf` configuration file of Postfix system the following lines must be added:

```
scan unix  -  - n - NN smtp
  -o smtp_send_xforward_command = yes
_ADDR_SEN_ inet n - n - NN smtpd
  -o content_filter =
  -o receive_override_options = no_unknown_recipient_checks,
       no_header_body_checks
  -o smtpd_helo_restrictions =
  -o smtpd_client_restrictions =
  -o smtpd_sender_restrictions =
  -o smtpd_recipient_restrictions = permit_mynetworks, reject
  -o mynetworks = 127.0.0.0/8
  -o smtpd_authorized_xforward_hosts = 127.0.0.0/8
```

where `_ADDR_SEN_` is the address to which `drweb-sender` module is connected to send messages (**`Address`** parameter of the [`Sender`] section of **Dr.Web MailD** configuration file) - `127.0.0.1:8026`, for example.

It is recommended, that `NN` number (maximum number of processes, executed by Postfix server) will be the same as the number of threads in pools of `drweb-receiver` and `drweb-sender` modules (**`PoolOptions`** parameter in [`Receiver`] section and **`OutPoolOptions`** in [`Sender`] section of **Dr.Web MailD** configuration file). To remove this limitation, specify "-" (minus symbol) in place of `NN` number.

When installation of the **Dr.Web for UNIX mail servers** is performed, all the described changesare made to Postfix configuration files automatically by the `configure_mta.sh` script. So, by default **Dr.Web for UNIX mail servers** and Postfix will be set up for operation in the `after-queue` mode.

After applying the changes to configuration files, restart Postfix.

Setup of Operation using Milter Protocol

To operate in this mode version 2.3.3 or later of Postfix system is required.

By default **Dr.Web for UNIX mail servers** ant Postfix are set up to interact in `after-queue` mode. To start using `milter` protocol you must edit Postfix configuration files one more time: change **`content_filter`** parameter to **`smtpd_milters`** parameter and remove all the changes made to `master.cf` file. Necessary restrictions can be specified directly in Postfix configuration files.

Address of the transport connection through which Postfix interacts with `drweb-milter` module can be specified as a TCP-socket or as a UNIX socket.

Address is specified in the **`smtpd_milters`** parameter of the Postfix configuration file `main.cf`. In case the connection is established through a TCP-socket, the parameter value is set in the following format: `inet:host@port` (for example, **`smtpd_milters`**`=inet:127.0.0.1:3001`). In case the connection is established through the UNIX socket, the address is set in the following format: `unix:pathname`, where `pathname` is an absolute path to the UNIX socket.

If UNIX socket is used, Postfix must have privileges for writing to the socket file.

Address of the transport connection between Postfix system and `drweb-milter` module must be also specified in **Address** parameter of the [`Milter`] section of **Dr.Web MailD** configuration file. Format and value of this parameter must be identical to the format and value of the `smtpd_milters` parameter of `main.cf` file.

Apart from transport address, the following parameters must be specified in the `main.cf` configuration file:

- **`milter_content_timeout`** = `300s` - this timeout of Postfix system is very important. It defines a maximum period of time for **Dr.Web MailD** to check a message in the **BeforeQueueFilters** mode. It is recommended to keep the value of this parameter greater than the value of the **ProcessingTimeout** parameter of the [`Milter`] section of **Dr.Web MailD** configuration file;
- **`milter_default_action`** = `tempfail` - this parameter defines action of the Postfix if any errors emerge during interaction with `drweb-milter` module;
- **`milter_protocol`** = `6` - the required version of `milter` protocol;
- **`milter_mail_macros`** = `_` - this parameter allows **Dr.Web MailD** to retrieve IP-address and host name of the sender;
- **`milter_end_of_data_macros`** = `i auth_type` - this parameter allows to retrieve information about authorization and a message ID to add information on the message to `drweb-milter` log.

Setup of Dr.Web MailD

You must also set up operation of **Dr.Web MailD**. In case **Dr.Web** system is started using **Dr. Web Monitor**, `drweb-milter` module must be started as a **Receiver** component. To make this possible you must uncomment the line from `%etc_dir/monitor/maild_postfix.mmc` which is responsible for the initialization of `drweb-milter` module. It is also recommended to

comment out the line, responsible for the initialization of `drweb-receiver` module. As a result, `drweb_postfix.mmc` contains similar lines:

```
# drweb-receiver local:%var_dir/ipc/.agent 15 30 MAIL drweb:drweb
drweb-milter local:%var_dir/ipc/.agent 15 30 MAIL drweb:drweb
```

In is also necessary to configure operation of `drweb-sender` module. Specify the following parameters in [`Sender`] section of **Dr.Web MailD** configuration file:

```
Address = /usr/local/sbin/sendmail
Method = pipe
MailerName = postfix
```

In **`Address`** parameter the path to the `sendmail` program from Postfix package is set.

You must also specify the value of the **`SecureHash`** parameter of the [`Sender`] section in **Dr.Web MailD** configuration file (a string of arbitrary symbols can be set as the value of this parameter, and the recommended length is no less than 10 symbols). Also a `Yes` value must be specified for **`UseSecureHash`** parameter in this section.

Once all the required parameters are specified, you must (re)start **Dr.Web MailD** and after that – Postfix.

All settings providing proper operation of `drweb-milter` with **Sender** and **Receiver** components are collected in [`Receiver`], [`Sender`] and [`Milter`] sections of **Dr.Web MailD** configuration file and are described in Receiver, Sender and Milter chapters of the current Manual.

When **Dr.Web MailD** work with mail system Postfix the following processes should be running:

- `drweb-notifier`

- drweb-sender
- drweb-maild
- drweb-receiver

Integration with Exim MTA

Integration instructions in this documentation are written for Exim version 4.xx. If you want to use earlier versions of Exim (3.xx), please refer to its documentation (i.e, http://www.exim.org/index.html).

When **Dr.Web MailD** interacts with Exim mail system, drweb-receiver module acts as **Receiver** component, and drweb-sender acts as **Sender** component. There are two possible ways to connect Exim mail system to **Dr.Web MailD**:

- Connection by means of special transport.

  Advantages: recompilation of Exim is not required, and system can operate with relatively old versions of Exim.

  Disadvantages: system performance is reduced.

- Connection by means of Exim local_scan function. In this case, **Receiver**, unlike other components, receives configuration data not from **Dr.Web Agent**, but from the configuration file of Exim mail system.

  Advantages: system performance is increased.

  Disadvantages: recompilation of Exim is required, and Exim version must be 4.50 or later.

Setup of Exim MTA

Initial configuration is identical for both ways of connection:

First, it is necessary to add `drweb` user to the list of trusted users in the `MAIN CONFIGURATION SETTINGS` section of the Exim configuration file:

```
------------------ cut -----------------------
##############################################
#           MAIN CONFIGURATION SETTINGS      #
##############################################

trusted_users = drweb
------------------ cut -----------------------
```

Please note, that if Exim performs mail delivery immediately after receipt of messages from `drweb-sender`, and serious delays occur in the process of this delivery (e.g. when SMTP-protocol is used), then timeout from **`PipeTimeout`** parameter of [`Sender`] section can be applied, because Exim does not return the code of successful receipt to `drweb-sender` until the delivery is finished. To avoid this problem, you may configure Exim to send all messages to the queue first, and only after that - to perform delivery.

Add the new `acl` to the Exim configuration file:

```
acl_check_drweb_scanned:
warn
condition = ${if and {{def:received_protocol}
{eq {${received_protocol}}\
{drweb-scanned}}} {yes}{no}}
control = queue_only
accept
```

and then - enable it:

```
acl_not_smtp = acl_check_drweb_scanned
```

Connecting to Exim using Special Transport

The description below is valid only for Exim 4.xx. For information on adjusting the settings for earlier versions of Exim (3.xx), refer to the corresponding documentation (for example, at http://www.exim.org/index.html).

In Exim settings you must add a special transport and a router. Find `Routers Configuration` section in configuration file of the mail system. It begins with the following header:

```
------------------ cut ----------------------
################################################
#       ROUTERS CONFIGURATION                  #
# Specifies how remote addresses are handled #
################################################
#         ORDER DOES MATTER                    #
#  A remote address is passed to each in       #
#      turn until it is accepted.              #
################################################
```

And right after the following line

```
begin routers
```

add the following description:

```
drweb_router:
  driver = accept
   condition = "${if eq {$received_protocol}
{drweb-scanned}{0}{1}}"
# check_local_user
  retry_use_local_part
  transport = drweb_transport
```

If check of the recipients is necessary, uncomment `check_local_user` parameter.

In the Exim configuration file find the section in which transport is described. It begins with the following header:

```
------------------ cut ----------------------
################################################
#     TRANSPORTS CONFIGURATION                 #
################################################
#         ORDER DOES NOT MATTER                #
#  Only one appropriate transport is called    #
#          for each delivery.                  #
################################################
------------------ cut ----------------------
```

You must add a description of the required transport to this section:

```
drweb_transport:
  driver = lmtp
  socket = __ADDRESS__
  batch_max = 100
  timeout = 5m
  user = drweb
# headers_add = "X-Maild-Checked: DrWEB for Exim"
```

Where _ADDRESS_ is the address of `drweb-receiver` listening module (**Address** parameter of [Receiver] section in **Dr.Web MailD** configuration file) - for example, a UNIX socket `%var_dir/ipc/.drweb_maild`.

Then you must specify the path to Exim mail system in the **Address** parameter of [Sender] section in **Dr.Web MailD** configuration file (for example, `/usr/exim/bin/exim/`), and specify `Exim` as a value of **MailerName** parameter from the same [Sender] section.

Restart **Dr.Web MailD** and Exim mail system after all changes are made.

Connecting to Exim using Local_scan Function

Working with **Dr.Web MailD** in this mode requires Exim mail system version 4.50 or later.

Preparation of the system has several stages. First, you must recompile Exim with support of `local_scan` function:

- Copy `%bin_dir/doc/maild/local_scan/local_scan.c` to `exim*/Local/` directory.
- To `Makefile` of Exim system, which is located in `exim*/Local/` directory, add parameters specified in `%bin_dir/doc/maild/local_scan/Makefile.sample`. If corresponding parameters are already specified in `Makefile`, you may uncomment and edit them.
- In `Makefile` of Exim system you must also specify the name of a user which privileges are used to start Exim (the name must be the same as specified for **Dr.Web MailD**). User name is defined by the **`EXIM_USER`** parameter. By default, **`EXIM_USER`** = `drweb`.
- Compile and install Exim system. If the execution of `make` or `make install` commands is interrupted with error messages like:
  `/libexec/ld-elf.so.1: Shared object "libgcc_s.so.1" not found, required by "libboost_thread.so"`

  then there are two possibilities to fix it:
  - You can copy the libraries `libstdc++.so.6` and `libgcc_s.so.1` (or make links to them) from `%bin_dir/lib/` to the system directory with libraries.

- You can execute the following command from the console:

  ```
  $ export LD_LIBRARY_PATH=%bin_dir/lib/:$LD_LIBRARY_PATH
  ```

  and then compile and install Exim once again from the console.

Then you must configure Exim system. For quick configuration you may use values of parameters from `%bin_dir/doc/maild/local_scan/configure.sample` file. Just copy the necessary lines to the `local_scan` section of the Exim configuration file.

To retrieve information on the settings of **Receiver** component, execute the following command from console:

```
$ PATH_TO_BIN_DIR/exim -bP local_scan
```

where `PATH_TO_BIN_DIR` is the path to Exim binaries.

In the Exim configure file the following additional parameters can be set:

DrwebTimeout = {time in seconds}	Period during which SendMail is waiting for drweb-maild to scan a message. It is recommended to set a greater value than for the SendTimeout parameter in the MailBase options section. Default value: **DrwebTimeout** = 60 s
DrwebBaseDir = {Yes | No}	MailD base directory where sockets, database etc. are stored Default value: **DrwebBaseDir** = %var_dir/

DrwebProcessingError = { pass | discard |reject | tempfail}	This parameter defines what action should be applied to messages which generate errors at scanning for example, antivirus plugin runs short of memory or cannot connect to drweb-maild. If parameter value is not set, or several values are specified (for example, discard and pass) – tempfail value is used by default. Default value: **DrwebProcessingError** = tempfail
DrwebLogLevel = { Quiet | Error | Alert | Info | Debug }	Logging detail level. Default value: **DrwebLogLevel** = Debug
DrwebIpcLevel = { Quiet | Error | Alert | Info | Debug }	Ipc library logging detail level. Default value: **DrwebLogLevel** = Debug
DrwebSyslogFacility = { Daemon | Mail | Local0 .. Local7 }	Type of facility, which generates a notification message on event when using syslogd system service. Default value: **DrwebSyslogFacility** = Daemon
DrwebMaxSize = { Daemon | Mail | Local0 .. Local7 }	Max size of checked massage. 0 - no size limitation Default value: **DrwebMaxSize** = 200 k

Setup of Dr.Web MailD

To configure **Dr.Web MailD** for interaction with Exim specify the path to the Exim mail system in **Address** parameter from the [

Sender] section of **Dr.Web MailD** configuration file (for example, /usr/exim/bin/exim/) and set Exim as a value of **MailerName** parameter from the same section.

There is no need to start drweb-receiver module separately, because when operation through local_scan function is performed, **Receiver** module is embedded into Exim. If **Dr.Web MailD** is started using **Dr.Web Monitor**, comment out the line in % etc_dir/monitor/maild_exim.mmc, which is responsible for the startup of drweb-receiver:

```
#drweb-receiver local:%var_dir/ipc/.agent 15 30 MAIL drweb:drweb
```

Restart **Dr.Web MailD** and Exim mail system after all changes are made.

All settings providing proper operation of **Dr.Web MailD** with Exim are collected in [Receiver] and [Sender] sections of **Dr.Web MailD** configuration file and are described in Receiver and Sender chapters of the current Manual.

When **Dr.Web MailD** work with mail system Postfix the following processes should be running:

- drweb-notifier
- drweb-sender
- drweb-maild
- drweb-receiver

Known Problems

If Exim mail system at restart generates error like:

```
transport drweb_transport: cannot find transport driver "lmtp"
```

it means, that it has been configured without LMTP transport support. You can either switch to SMTP transport (for more information refer to documentation on Exim MTA, for example at

http://www.exim.org/), or recompile Exim with LMTP transport support. If the latter variant is used, you must add or uncomment the following line in `/Local/Makefile` file of the Exim system: **`TRANSPORT_LMTP`** `= yes`.

Integration with Qmail MTA

Principle of operation of Qmail is based on overriding the mail system (proxying). Via interface set for `qmail-queue` module, the filter receives a message, checks it and if it is not infected, the filter moves it to `qmail-queue`.

Operation in this mode has the following limitation: UNIX-sockets, which `drweb-qmail` listens for scan requests (are set in **`ListenUNIXSockets`** parameter from the [`Qmail`] section of the **Dr.Web MailD** configuration file) must be located within the range of certain paths. To display a list of paths, run `qmail-queue` with `--help` command line parameter).

For integration with **Dr.Web MailD** Qmail version not earlier than 1.03 is required. To avoid possible loss of incoming mail, filter must be installed only when Qmail is stopped.

Setup of Qmail

To connect **Dr.Web MailD** to Qmail, do the following:

- Save original `qmail-queue` file and remember its location. You will need it later.
- Copy `qmail-queue` from `%bin_dir` directory to `/qmail/bin/`. Do not forget to set appropriate privileges for a new `qmail-queue` (which is filter of **Dr.Web MailD**) as well as for the `qmail-queue.original` you have copied.

It is recommended to use the configuration in which **Dr.Web MailD** and `qmail-queue` work with privileges of `drweb` user. To assure proper operation of this configuration, set the following privileges for `qmail-queue`:

```
-rws--x--x X drweb qmail SIZE DATE qmail-queue
-rws--x--x X qmailq qmail SIZE DATE qmail-queue.original
```

Use commands given below:

```
$ chown drweb:qmail qmail-queue
$ chmod 4711 qmail-queue
$ chown qmailq:qmail qmail-queue.original
$ chmod 4711 qmail-queue.original
```

Setup of Dr.Web MailD

All settings providing proper operation of **Dr.Web MailD** with Qmail are collected in [`Sender`] and [`Qmail`] sections of **Dr.Web MailD** configuration file and are described in Sender and Qmail chapters of the current Manual.

When **Dr.Web MailD** work with mail system Qmail the following processes should be running:

- `drweb-notifier`
- `drweb-sender`
- `drweb-maild`
- `drweb-qmail`

Known Problems

Description:

At start, Qmail returns one of the following errors:

1. `terminate called after throwing an instance of 'St9bad_alloc'`
   `what(): St9bad_alloc`
2. `bash: xmalloc: cannot allocate 2 bytes (0 bytes allocated)`
3. `qmail-queue.real: error while loading shared libraries: libc.so.6: failed to map segment from shared object: Cannot allocate memory`
4. `/var/qmail/bin/qmail-smtpd:`
   `error while loading shared libraries:`
   `libc.so.6: failed to map segment from shared object:`
   `Cannot allocate memory`

Solution:

The problem arises because limitation for used memory in the initialization script is too high. For example, if Dave Sill scripts are used, the value indicated in the instruction `softlimit -m 2000000` should be increased by adding for instance a zero to the right (to `20000000`).

Description:

In reply for all messages received via SMTP-protocol Qmail returns a string like this:

```
451 qq trouble making network connection (#4.3.0)
```

Solution

`qmail-queue` may have not enough privileges to connect to the UNIX-socket, created by `drweb-qmail` (which operates as **Receiver** component of **Dr.Web MailD**), or paths specified by defalt for `qmail-queue` do not lead to this socket. Check the

privileges and make sure that the value of **ListenUNIXSocket** parameter from the [Qmail] section of **Dr.Web MailD** configuration file matches default paths (a list of these paths can be obtained by running qmail-queue with --help command line parameter).

Description:

For each message received via SMTP-protocol Qmail returns to the console upon receipt of a message body a string like follows:

```
qmail-inject: fatal: qq temporary problem (#4.3.0)
/usr/libexec/ld-elf.so.1: Shared object "libstdc++.so.6" not found,
required by "libboost_program_options.so"
```

Solution:

The system cannot find necessary libraries which are located in %bin_dir/lib/. It is necessary to copy (or create a symbolic link) libstdc++.so.6 and libgcc_s.so.1 from %bin_dir/lib/ to the system directory with libraries.

Integration with ZMailer MTA

drweb-zmailer module is compatible only with ZMailer v. 2.99.55 or later.

Dr.Web MailD can interact with ZMailer in two modes:

- As a context filter at the stage of SMTP-connection.

  Advantages: it is possible to block the message at the stage of SMTP-connection.

Disadvantages: decreased performance when system load is high as only SMTP-traffic is checked.

- As a context filter at the routing stage.

  Advantages: stable performance when system load is high; all mail coming through ZMailer is checked (including local mail and mail transferred via `UUCP` protocol).

  Disadvantages: message cannot be blocked when it is received (i.e. `reject` and `tempfail` actions are similar to `discard`); usage of **`SecureHash`** is necessary to increase performance and avoid cycling of messages.

`drweb-zmailer` module is used as **Receiver** component of **Dr.Web MailD** in interactions with ZMailer.

To assure proper operation of `drweb-zmailer` and filters it is recommended to install patches (if possible).

To install patches do the following:

- Open the `$(ZMAILER_SRCHOME)/smtpserver` directory, where `ZMAILER_SRCHOME` is the path to the directory with ZMailer binaries.
- Run the following command:

  ```
  $ patch < smtpdata.c.XXX.patch
  ```

  where `XXX` stands for the version of Zmailer to be patched.

When **Dr.Web MailD** work with mail system Zmailer the following processes should be running:

- `drweb-notifier`
- `drweb-sender`
- `drweb-maild`

Context Filter Mode at the Stage of SMTP-session

To enable **Dr.Web MailD** support in ZMailer, do the following:

- copy (or make a symlink) `drweb-zmailer.sh` to `$MAILBIN` directory (path to the file is specified in `zmailer.conf`);
- edit `smtpserver.conf` file by adding the following line: `PARAM contentfilter $MAILBIN/drweb-zmailer.sh`.

As command line parameters cannot be specified in `contentfilter`, you must define them in `drweb-zmailer.sh` script.

Context Filter Mode at the Routing Stage

All messages processed by the mail server pass the routing stage. Therefore the end of the routing stage is a most suitable time for the filter to connect. To make such connection possible, edit `$MAILBIN/cf/process.cf` as described below:

Find the following lines:

```
LOGMSG=()     # This is a LIST of files where to log..
#|     The LOGMSG variable is used by the intercept facility (in  crossbar.cf)
#|   to make sure only a single copy of a message is saved when required.
#|   Each sender - recipient address pair can cause an intercept which can
#|   specify a file to save the message to.  This variable is appended to
#|  elsewhere, and processed at the end of this function.
```

Add the following:

```
###-> Dr.Web MailD support
ch='"DEFAULT_BIN_PATH/drweb-zmailer.sh"  --hash __EDIT_THIS__ --file $POSTOFFICE/router/$file'
```

```
        case  "$ch" in
              -1*) #reject or disacrd
                    /bin/rm -f "$file"
                    return
                    ;;
              1*) #tempfail
                    /bin/rm -f "$file"
                    return
                    ;;
              *);;
        esac
###-> end of Dr.Web MailD support
```

Replace `__EDIT_THIS__` (value of the `--hash` parameter) with the value equal to the value of the **`SecureHash`** parameter in the [`Sender`] section in **Dr.Web MailD** configuration file and specify `Yes` for **`UseSecureHash`** parameter in this section.

Additional Settings

A simple way to disable receipt of messages with empty SMTP envelope of a sender (usually error messages or messages saying that delivery has failed are sent with empty SMTP envelope; also such messages are sent by spammers) is to install `policytest.c.XXX.patch`. Installation procedure is similar to installing `smtpdata.c.XXX.patch`.

As ZMailer starts `drweb-zmailer` module each time a new message is processed, for optimized system performance all `drweb-zmailer` settings must be specified in command line (they can be defined for example in `drweb-zmailer.sh` script).

In the command line of `drweb-zmailer` the following parameters can be specified:

- `-h [ --help ]` – display help and exit;

- `-v [ --version ]` – display version and exit;
- `-u [ --user ] arg (=drweb)` – name of the user with which privileges `drweb-maild` is started.

  As ZMailer starts `drweb-zmailer` with `root` privileges by default, then the whole **Dr.Web MailD** complex must be started with `root` privileges (and an empty line must be set as a value for this parameter `-u ""`), or a necessary value for this parameter must be specified;

- `-l [ --level ] arg (=info)` – log verbosity level. Possible values: `Quiet`, `Error`, `Alert`, `Info`, `Debug`;
- `-i [ --ipclevel ] arg (=info)` – IPC log verbosity level. Possible values: `Quiet`, `Error`, `Alert`, `Info`, `Debug`;
- `-f [ --facility ] arg (=mail)` – log type when syslogd system utility is used for logging. Possible values: `Daemon`, `Mail`, `Local0-7`;
- `-b [ --basedir ] arg (=%var_dir)` – base directory of **Dr.Web MailD**. Value of this parameter is similar to that of **`BaseDir`** parameter from [`General`] section of **Dr.Web MailD** configuration file;
- `-t [ --timeout ] arg (=30)` – maximum time to process one message;
- `--file arg` – path to a file to be processed. It must be specified only in the context filter mode at the routing stage;
- `--hash arg` – value of the **`SecureHash`** parameter from the [`Sender`] section of **Dr.Web MailD** configuration file. It must be specified only in the context filter mode at the routing stage only;
- `--interface arg (=1)` – `smtpserver` version. It must be specified only in the context filter mode at the SMTP-session: 0 - is for version 2.99.55 and earlier; 1 - is for version 2.99.56 and later;
- `-e [ --error-action ] arg (=reject)` – action applied to a message when internal error occurs during operation of the filter. Possible values: `pass`, `reject`, `discard`, `tempfail`.

Integration with Courier

Setup of Courier

To connect **Dr.Web MailD** to Courier mail system, do the following:

1. Set up privileges for the `drweb-courier` module by performing the following commands:

   ```
   $ chown COURIER_USER:drweb "DEFAULT_BIN_PATH/drweb-courier"
   $ chmod 6771 "DEFAULT_BIN_PATH/drweb-courier"
   ```

   where `COURIER_USER` is a user with which privileges Courier is started. Also make sure that the read, write and execution permissions are set for all directories and subdirectories in `%var_dir` directory for the `drweb` group.

2. Copy `drweb-courier` module (or create a symlink) to Courier filters directory (by default it is `/usr/local/libexec/filters/`).
3. Register drweb-courier module in the Courier mail system as global:

   ```
   $ /usr/local/sbin/filterctl start drweb-courier
   ```

   Later, to disable the filter, you must execute the following command:

   ```
   $ /usr/lib/courier/sbin/filterctl stop drweb-courier
   ```

4. Create (or edit) `enablefiltering` file to set services to perform check (`esmtp` or `uucp` - if more than one is specified, they are separated with white spaces).
5. Make sure that **`BaseDir`** and **`SocketDirs`** parameters in the [`Courier`] section of **Dr.Web MailD** configuration file

correspond to the configuration of your Courier mail system. For more information perform the command: `man courierfilter`.

6. Enable filtering in Courier system:

```
$ /usr/lib/courier/sbin/courierfilter start
```

`drweb` user with which privileges **Dr.Web Daemon** operates, must be included in `courier` group to gain read access to files created in spool by Courier mail system.

Transmission of processed messages to Courier MTA

Settings for transmission of processed messages to MTA are determined in `[Sender]` of configuration file. Following parameters must be set:

```
MailerName = Courier
```

```
Method = pipe
```

`Address` = path to system for sending email messages (by default: `/usr/lib/courier/bin/sendmail`)

Setup of Dr.Web MailD

All settings providing proper operation of **Dr.Web MailD** with Courier are collected in [`Sender`] and [`Courier`] sections of **Dr.Web MailD** configuration file and are described in Sender and Courier chapters of the current Manual.

When **Dr.Web MailD** work with mail system Courier the following processes should be running:

- `drweb-notifier`
- `drweb-sender`
- `drweb-maild`
- `drweb-courier`

Using Proxy

Proxy included in **Dr.Web for UNIX mail servers** allows to achieve several goals in management of computing resources:

1. **Dr.Web for UNIX mail servers** efficiency may improve greatly, when **Receiver** and **Sender** modules are working separately from `drweb-maild` module, so that mail processing and mail checking operations are performed on different hosts.
2. Computing resources in a network can be managed flexibly, using load balancing scheme `N:M`, where `N` is the number of hosts processing mail traffic, and `M` is the number of hosts checking mail for viruses and spam.

Please note, that `drweb-maild` components do not support cluster implementation and cannot share internal data (statistics, quarantine, database settings, etc.) with each other. As a result each `drweb-maild` component will have its own statistics, quarantine and configuration.

Proxy consists of the following components: `drweb-proxy-client` and `drweb-proxy-server`.

- `drweb-proxy-client` – works on a computer, where **Receiver** and **Sender** components are operating. It is started instead of `drweb-maild` and plays its part in interactions with other components.
- `drweb-proxy-server` – works on a computer, where the `drweb-maild` module is operating, and plays the role of **Receiver** and **Sender** components.

Both `drweb-proxy-client` and `drweb-proxy-server` components interact with each other, enabling transfer of original mail messages and their modifications to other components of **Dr.Web for UNIX mail servers** on different hosts for further processing.

`drweb-notifier`, `drweb-monitor` and `drweb-agent` components are working on each host.

General operating scheme with a proxy looks like the following:

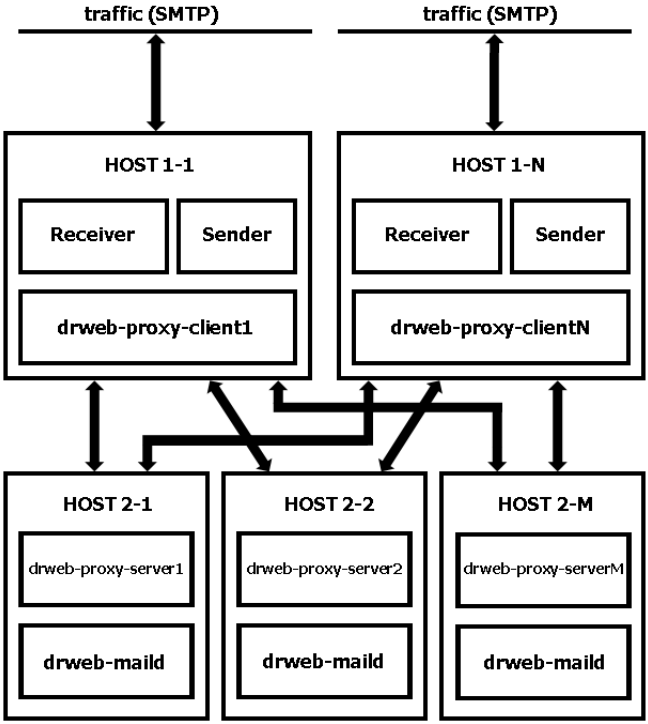


Figure 16. Diagram of operation via proxy

As it appears from the scheme, both `drweb-proxy-client` and `drweb-proxy-server` can interact with an arbitrary number of complementary components residing on different hosts. This is implemented using a special balance system.

A certain weight is assigned to the each socket address specified in a value of **`ProxyServersAddresses`** or **`ProxyClientsAddresses`** parameters (from `[ProxyClient]`

and [ProxyServer] sections correspondingly). So, addresses are specified in the following format:

```
ADDRESS1 [WEIGHT1], ADDRESS2 [WEIGHT2] ..
```

where ADDRESS has a basic address type, and WEIGHT is an optional numeric value from a range of 0 to 100, defining a weight of this address. This WEIGHT determines a relative work load on a certain host in the network. The greater is the value the greater is the load of a certain server.

ProxyServersAddresses parameter from the [ProxyClient] section specifies HOST2-* addresses (ref. to the scheme above), which are used by drweb-proxy-server* components to receive requests.

ProxyClientsAddresses parameter from the [ProxyServer] section specifies HOST1-* addresses (ref. to the scheme above), which are used by drweb-proxy-client* components to receive requests.

Example:

```
ProxyServersAddresses = inet:8066@10.3.0.73 10,
inet:8066@10.3.0.72 5
```

In this case 10.3.0.73 host will receive twice as much mail messages as 10.3.0.72 host does. If the WEIGHT is not specified, it is considered to be equal to 1 by default. If several addresses have the same WEGHT, they are considered equivalent and receive the same amount of requests.

If the WEIGHT is set to 0, then such addresses are considered backup-addresses. Requests are sent to them only if no other available addresses with WEIGHTs equal to or greater than 1 are left. General address selection algorithm looks like the following:

1. An attempt to send a message to the address with the greatest WEIGHT value is made. If an error emerges, then the next address with lesser WEIGHT is selected (this WEIGHT must be equal to or greater than 1). If no available

addresses with WEIGHTs ≥ 1 are left, then backup addresses are used (p. 3 of the procedure).

2. An attempt to send the message to the selected address is made. If an error emerges, this address is marked as unavailable, and the procedure starts from the beginning (p.1 of the procedure).
3. If all the addresses which WEIGHT is greater than 0 turned out to be unavailable, then an attempt to send the message to the backup address is made. Backup addresses are checked in the order they were listed. If backup addresses are also not available, then an error is returned.

WEIGHT values should be selected and assigned according to available resources on each server.

When messages are passed for check to the drweb-maild module and processed by plug-ins from the **BeforeQueue**, then all processed mail is returned to the client that has sent it for processing.

If messages are processed by plug-ins from the **AfterQueue**, then address of the client to receive already processed mail will be selected according to weights of clients addresses in **ProxyClientsAddresses**.

Duplicated messages (ref. to Rules description) and messages generated by drweb-maild (reports and notifications) will also be sent to the client selected from the **ProxyClientsAddresses** list - regardless of the any queue in which plug-ins reside.

For messages sent to the client selected from the **ProxyClientsAddresses** list, settings specified in Rules (if there are any) will be applied (e.g. value of **SenderAddress** parameter).

Please note, that when proxy interacts with Milter, Qmail or Courier MTAs (and drweb-milter module correspondingly), it is better not to put plug-ins to the **AfterQueue**. Right now proxy does not support backdoor-connection to the **Receiver** component. So, when the response from drweb-maild is not returned to the

Receiver immediately (e.g. when plug-in is in the **AfterQueue**), then drweb-milter finishes SMTP-session only after expiration of **ProcessingTimeout** period.

Optimal (but not the only one possible) connection procedure for M=N=1 is described below. It allows to avoid the majority of possible mistakes in setup and adjustment.

Use following procedures to set up proxy if M=N=1 (proposed way is not the only possible but is preferable to avoid any errors in configuration):

1. Set up and adjust **Dr.Web MailD** on HOST1-1 (i.e. on the host to be used to process mail traffic, and where drweb-proxy-client component resides). Check validity of configuration with the following command:
   - /etc/init.d/drweb-monitor check – for Linux and Solaris
   - /usr/local/etc/rc.d/00.drweb-monitor.sh check – for FreeBSD
2. Run **Dr.Web MailD** on HOST1-1 and check if the mail is processed correctly.
3. Setup **Dr.Web MailD** on HOST2-1 (i.e. on the host to be used to check mail messages, and where drweb-proxy-server component resides). During setup you may skip adjustment of **Receiver** and **Sender** components - they are not necessary on this host.
4. Adjust configuration on HOST2-1 similarly to HOST1-1.
5. Disable startup of **Receiver** and **Sender** components by commenting-out corresponding lines in mmc file from the %etc_dir/monitor directory on the HOST2-1. Also the startup of drweb-proxy-server component must be enabled.

   Please note, that at attempt to start simultaneously on the same host drweb-proxy-server and **Receiver**/**Sender** components **Dr.Web Monitor** will finish its operation, and no components will be initialized. Information about this error will be output to log.

6. Specify IP address of the `HOST1-1` as a value of the **`ProxyClientsAddresses`** parameter from the `[ProxyServer]` section in the **Dr.Web MailD** configuration file on the `HOST2-1`. This address must be the same that is specified as a value of **`Address`** parameter from the `[ProxyClient]` section. Mail will be sent to it.
7. Check validity of configuration on the `HOST2-1` with the following command:
   - `/etc/init.d/drweb-monitor check` – for Linux and Solaris
   - `/usr/local/etc/rc.d/00.drweb-monitor.sh check` – for FreeBSD

   If everything is configured correctly, you may start **Dr.Web MailD** on the `HOST2-1`.
8. Specify IP address of the `HOST2-1` as a value of the **`ProxyServersAddresses`** parameter from the `[ProxyClient]` section in the **Dr.Web MailD** configuration file on the `HOST1-1`. This address must be the same that is specified as a value of **`Address`** parameter from the `[ProxyServer]` section. Requests for message checks will be sent to in.
9. Disable startup of `drweb-maild` component by commenting-out corresponding line in `mmc` file from the `%etc_dir/monitor` directory on the `HOST1-1`. Also the startup of `drweb-proxy-client` component must be enabled.

   Please note, that at attempt to start simultaneously on the same host `drweb-proxy-client` and `drweb-maild` components **Dr.Web Monitor** will finish its operation, and no components will be initialized. Information about this error will be output to log.

10. Check validity of configuration on the `HOST1-1` with the following command:
    - `/etc/init.d/drweb-monitor check` – for Linux and Solaris
    - `/usr/local/etc/rc.d/00.drweb-`

`monitor.sh check` – for FreeBSD

If everything is configured correctly, you may restart **Dr.Web MailD** - and all the mail will be transferred to the `HOST2-1` for scan from now on.

11. You may also disable **Dr.Web Daemon** and **Dr.Web Updater** on the `HOST1-1` (if there are no more **Dr.Web** products on the system). These modules are no longer necessary.

You can also apply this algorithm for situations, when `M` and/or `N` are greater than `1`: just connect additional hosts as it was described above, end edit values of corresponding parameters (**`ProxyClientsAddresses`** from the `[ProxyServer]` section and **`ProxyServersAddresses`** from the `[ProxyClient]` section) in configuration files on those hosts.

`WEIGHT` values must be set according to the amount of available resources on each host.

Dr.Web console for UNIX mail servers

Setup and configuration of **Dr.Web for UNIX mail servers** can be performed via separate web interface **Dr.Web console for UNIX mail servers**. It is implemented as a plug-in to Webmin (detailed information about Webmin interface is available on its official website at http://www.webmin.com/).

To achieve optimal performance of **Dr.Web console for UNIX mail servers** web interface, make sure that the following Perl modules are installed to your system:

- `XML::Parser` - Perl module for parsing XML documents;
- `JSON::XS` - module for data interchange using JSON;
- `XML::XPath` - set of modules for parsing and evaluating XPath statements;
- `Text::Iconv` - Perl interface to iconv() codeset conversion function;
- `perl-devel` (or `libperl-dev`, depending on the UNIX distribution) - a package to build `Text::Iconv`;
- `Date::Parse` - Perl module to convert date to UNIX-format;
- `CGI` - Perl module enabling operation with Common Gateway Interface;
- `CGI::Carp` - Perl module for creation of HTTPD report about errors;
- `MIME::Words` - Perl module enabling operation with RFC 2047 encoding.
- `MIME::Base64` - module for operation with Base64 encoding;
- `MIME::QuotedPrint` - module for operation with the quoted-printable encoding;
- `MIME::Entity` - Perl module for decoding and parsing MIME-messages;

- `MIME::Parser` - Perl module for parsing MIME-threads;
- `MIME::Head` - Perl module for parsing headers of MIME-messages;
- Storable - module for storing data structures on disk.
- `POSIX` - module used for access to POSIX system commands.
- `Digest::MD5` - module for using MD5 encryption algorithm.
- `Encode` - module used for encoding of strings.
- `Encode::Byte` - module used for various single byte character encoding.
- `Encode::JP` - module used for Japanese characters encoding.
- `File::Stat` - Perl module with interface to embedded stat () functions.
- `File::Find` - Perl module with interface to perform search through directory tree.
- `Encode::CN` - module used for Chinese characters encoding.
- `Encode::HanExtr` - extra sets of Chinese encodings.

If some modules are missing, it is recommended to install them from console. Names of the modules may vary, but usually they are included into the following packages: `perl-Convert-BinHex`, `perl-IO-stringy`, `perl-MIME-tools`, `perl-XML-Parser`, `perl-XML-XPath`. For installation in rpm-systems it is recommended to choose `noarch.rpm` packages.

To install these modules you may use one the following commands (depending on the OS you are using).

For Debian/Ubuntu:

```
apt-get install libperl libjson-perl libjson-xs-perl libxml-parser-perl libxml-xpath-perl libtimedate-perl libmime-tools-perl
```

Root privileges are required.

For Red Hat/Fedora/CentOS 5:

```
yum install perl-JSON perl-JSON-XS perl-XML-
Parser perl-XML-XPath perl-TimeDate perl-MIME-
tools
```

Root privileges are required.

For other OS:

```
cpan JSON JSON::XS XML::Parser XML::XPath
Date::Parse MIME::Words MIME::Entity MIME::
Parser MIME::Head
```

Root privileges are required.

Web interface layout and appearance may differ depending on Webmin version and browser used.

Due to peculiarities of Webmin implementation, **Dr.Web console for UNIX mail servers** web interface can not be correctly displayed in Internet Explorer 7. In case of problems with displaying of web-pages, try to use Internet Explorer 8 or 9 (and later) or use another browser.

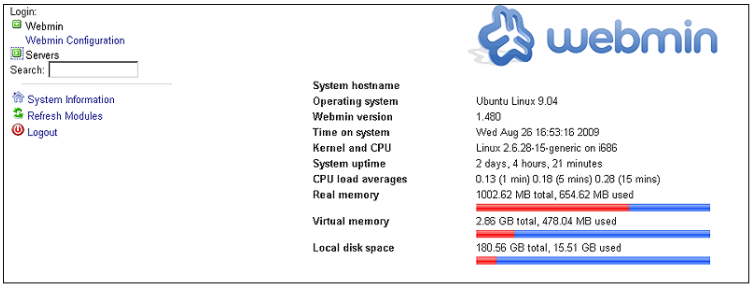
Installation

To start working with **Dr.Web console for UNIX Mail Servers**, do the following:

- install Webmin;
- install Webmin plug-in module **Dr.Web console for UNIX Mail Servers** located in `%bin_dir/web/`.

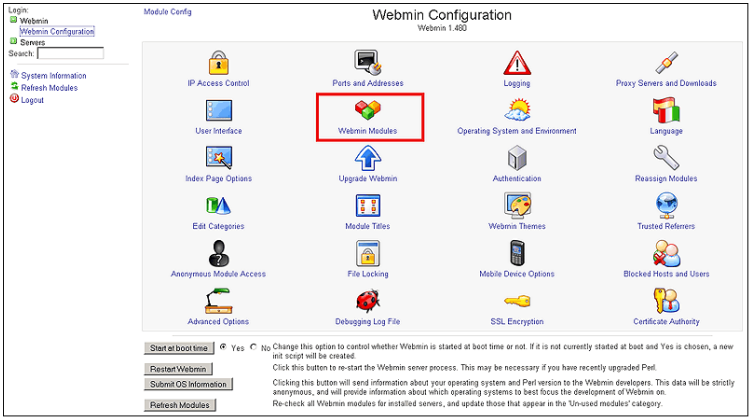
Webmin configuration and installation of modules is done using Webmin web interface.

Figure 17. Main page of Webmin Web interface



Installation of the new modules can be performed on **Webmin Configuration** page of the **Webmin** section of main menu, in **Webmin Modules** subsection.

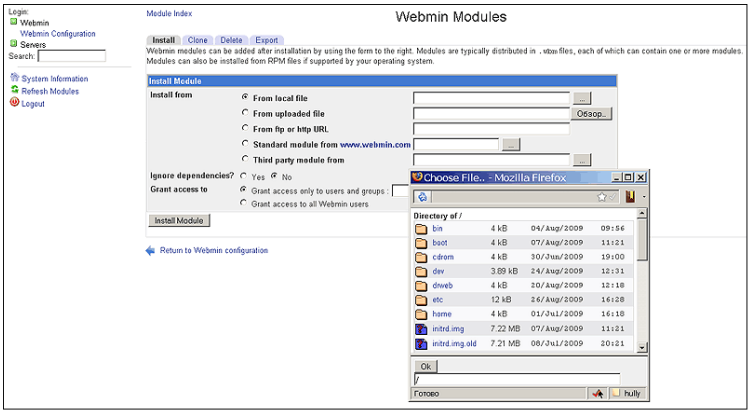
Figure 18. Webmin Configuration



To install necessary modules:

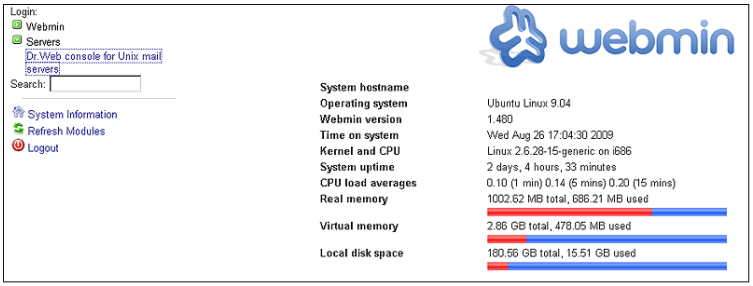
1. Click the **Browse** button near the **From local file** text field on the **Webmin Modules** page. A separate browser window will be opened to provide navigation through folders and files.
2. Choose the corresponding installation package from the list (`%bin_dir/web/drweb-maild-web.wbm.gz` by default).

Figure 19. Webmin modules



3. One click on any item from the list selects it to the field below. With the second click on previously selected folder, it opens. With the second click on previously selected file, navigation window closes, and full path to selected file appears in From local file text field. You may also click **OK** button when you are finished with selection of required file.
4. After finishing with selection of the installation package file, click the **Install Module** button.
5. After installation is finished, in **Servers** section of main menu a link to the new **Dr.Web console for UNIX Mail Servers** module will appear.

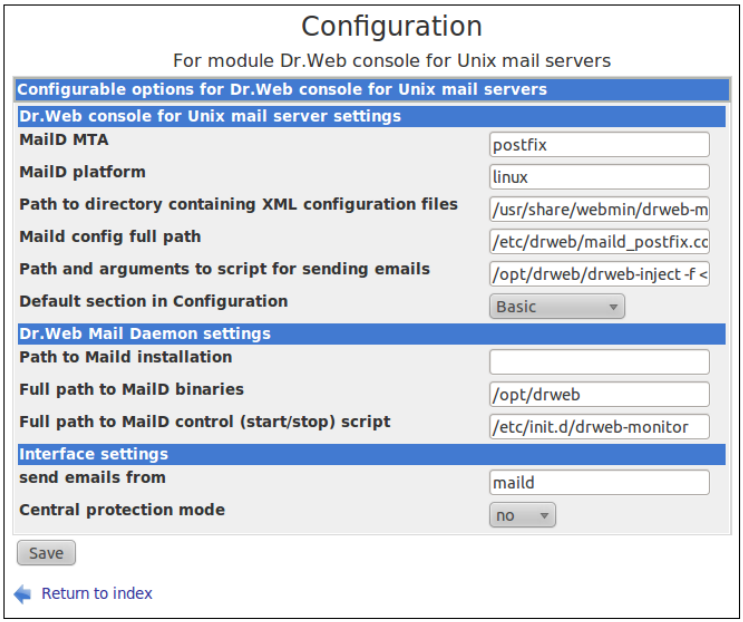
Figure 20. Dr.Web console for UNIX Mail Servers module



Basic configuration

At the very top of **Dr.Web console for UNIX Mail Servers** pages **Module config** link is located, under which basic module settings are gathered. There you can specify path to configuration file and to init script, path to mail sending script, mail address inserted to **From** field of the notification letters and operation mode.

Figure 21. Module configuration

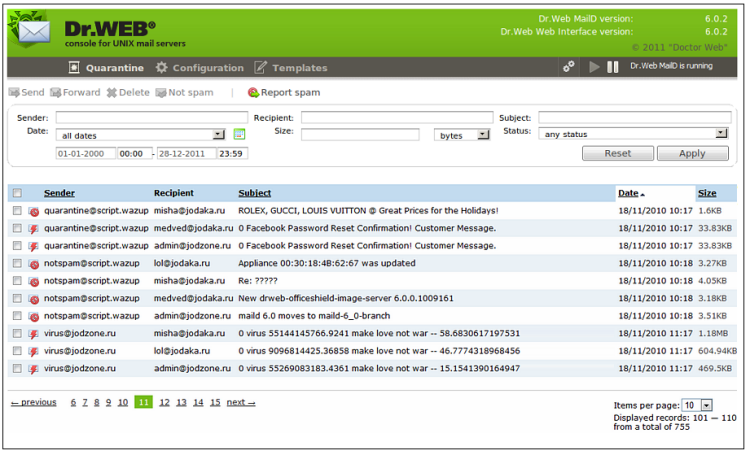


Please, don't forget to change **send emails from** field default value. Otherwise, messages sent from quarantine (for example, in case of filter false positive) and messages with notifications may not reach their destination.

User Interface

You will not be able to use standard browser **Back** function navigating through the **Dr.Web console for UNIX Mail Servers** chapter. If you click **Back** button or corresponding key combination, you will get straight to the previous chapter from main menu.

Figure 22. Dr.Web console for UNIX Mail Servers



On the right side of the module header information about current versions of **Dr.Web MailD** and **Dr.Web for UNIX mail servers** Web interface is shown.

Under the module header there are three sections: **Quarantine**, **Configuration** and **Templates**. By default mail page of **Quarantine** section is opened.

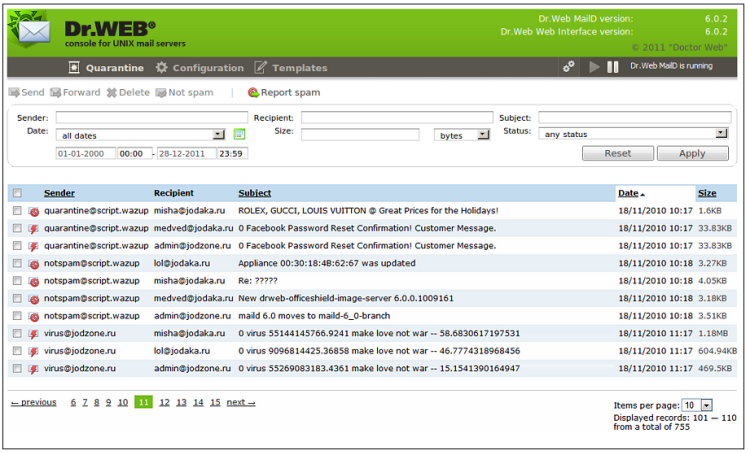
Near section headers there are three buttons: **Interface configuration**, **Start Dr.Web MailD** and **Stop Dr.Web MailD** - with current status information beside. When operating in central protection mode, **Stop Dr.Web MailD** button will stop all local **Dr.Web** services running in central protection mode.

If **Dr.Web MailD** is operating in central protection mode, it will require to manually reload the page with web interface after changing access permissions for settings in **Dr.Web ESS Control Center**.

Quarantine

Messages filtered out by anti-virus or anti-spam module of **Dr.Web for UNIX mail servers**, may be moved to quarantine. On the **Quarantine** tab of the **Dr.Web for UNIX mail servers** web interface all the necessary tools for successful managing of quarantined messages are collected.

Figure 23. "Quarantine" tab



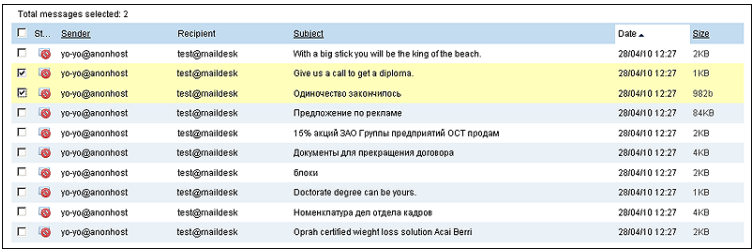
The **Quarantine** tab contains the following elements:

- a toolbar
- a filters palette
- a table with the list of quarantined messages
- additional navigation facilities and a drop-down menu for selection of the number of messages per page

Toolbar

Toolbar buttons (except for the **Report spam** button) become active when some quarantined message is selected from the list.

Figure 24. Toolbar



Using the toolbar you can:

- send quarantined messages to their original recipients. Select corresponding messages from the list and press **Send**.
- forward quarantined messages to some address. Select corresponding messages from the list and press **Forward**. As a result a separate window opens with the following fields to enter information to: **Recipient** field (e-mail address of the recipient), **Subject** field (any text for the subject), **Message** field (any text for the description of the forwarded message), **Attachments** (forwarded messages as attachments).
- delete quarantined messages. Select corresponding messages from the list and press **Delete** or press the DEL key on the keyboard.
- remove a "`spam`" status from messages, which were filtered out by anti-spam module and put to quarantine by mistake. Select messages with "`spam`" status from the list and press **Not Spam**. After that all corresponding messages will be automatically sent to recipients and deleted from quarantine.
- report spam.

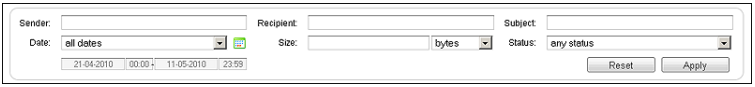
This option cannot be applied to messages from the list. If the user has considered some message to be spam, he should save it to the file system and after that send it to the **Doctor Web** laboratory using the interface provided by **Dr.Web for UNIX mail servers**.

After you press **Report spam** button, a separate window opens, enabling you to upload the file which contains the suspicious message.

Filters Palette

Filters palette simplifies the processing of quarantined messages.

Figure 25. Filters palette



Using the filter system you may select messages upon the following criteria:

- **Sender** - e-mail address of a message sender. In this field a full e-mail address or some part of it may be entered.
- **Recepient** - e-mail address of a message recipient. In this field a full e-mail address or some part of it may be entered.
- **Subject** - any text to be looked for in the corresponding field of quarantined messages. As a result only those messages will be displayed, which Subject field contains the specified text string (both complete and partial match are allowed).
- **Date** - the date when message was put to quarantine. A specific time period can be selected from the drop-down list, or set up using the calendar available upon pressing the button. The following date options are available:
  - ✓ **all dates** - select all the messages stored in quarantine.
  - ✓ **today** - select messages which were put to quarantine between 12:00 a.m. today and the present moment.

- ✓ **yesterday** - select messages which were put to quarantine between 12:00 a.m. yesterday and 12:00 a.m. today.
- ✓ **this week** - select all messages which were put to quarantine from the beginning of the week until the present moment.
- ✓ **this month** - select all messages which were put to quarantine from the beginning of the month until the present moment.
- ✓ **custom period** - select all messages for an arbitrary period of time. You can use the calendar to specify boundaries of the required period.

Figure 26. Calendar

Calendar window opens automatically after you select a **custom period** option from the drop-down menu, or when you press the button. Specify the boundaries of the required time period and press **OK**. After calendar window closes, selected boundary values will appear in the corresponding fields. You can also specify the exact time value or time interval.

If you specify the exact time interval, then only those messages will be selected, which were put to quarantine during that specific period of time (including the boundaries). So if you want to select messages received at the certain minute of the certain hour, you may specify the same time values as period boundaries in the corresponding fields.

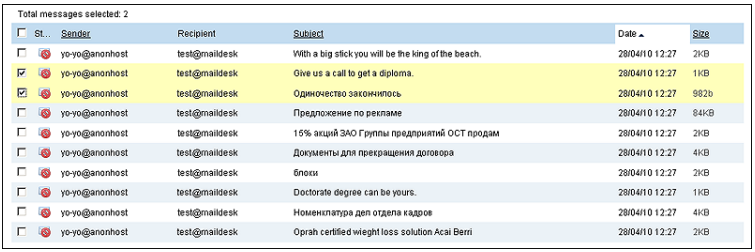
- **Size** - numeric value. By default it is treated as the message size in bytes, but you can set other dimensions (Kbytes or Mbytes) using the corresponding drop-down list. When you use this criteria, then only those messages will be selected, which size is greater then or equal to the specified value. If the **Size** value is set to `0`, then this criteria is not considered in the search.
- **Status** - the reason why the message has been moved to quarantine. The following reasons are considered during message processing and can be used for filtering:
  - ✓ **virus** - the message was considered infected by the virus and sent to quarantine by the anti-virus module of **Dr.Web for UNIX mail servers**
  - ✓ **spam** - the message was considered spam and sent to quarantine by the anti-spam module of **Dr.Web for UNIX mail servers**
  - ✓ **rule** - the message was sent to quarantine according to internal message processing rules
  - ✓ **processing error** - an error emerged during processing of this message, so it was also sent to quarantine

After all selection criteria are specified, press **Apply** button. To return to the defaults press **Reset** button.

List of Messages

When there are some messages in quarantine, on the **Quarantine** tab the list of these messages is displayed in tabular form.

Figure 27. List of messages



Administrator has access to all quarantined messages which were sent to recipients from groups controlled and managed by this Administrator.

Tabular data is organized in the following way:

- **Status** column - contains information about message statuses (reasons why they were moved to quarantine). All statuses have unique icons: - message contains virus, - message is marked as spam, - message is quarantined according to internal filtering rules, - message has evoked an error during processing. When you rest the mouse pointer on the status icon a tooltip will appear with the detailed description of the reason for quarantine.
- **Sender** column - contains information about sender e-mail address. It is possible to sort messages by subject in direct and inverted alphabetic order.
- **Recipient** column - contains information about recipient e-mail address. It is possible to sort messages by subject in direct and inverted alphabetic order.
- **Subject** column - contains information about message subject. It is possible to sort messages by subject in direct and inverted alphabetic order.
- **Date** column - contains information about the date the message was put to quarantine. For messages quarantined in the last twenty four hours only time is specified. It is possible to sort messages by date in ascending and descending order.
- **Size** column - contains information about message size. It is possible to sort messages by size in ascending and descending

order.

To choose some message from the list select the corresponding check box. To choose all messages in the list select the check box in the left corner of the table header

Values of **Recipient**, **Subject** and **Date** fields are links to the corresponding message: click any of them and you will switch to the message screen.

Figure 28. Quarantined message

On this screen you can view message content, source (using the **View source** link), headers (using the **Headers** link) and attachments (if there are any).

Press the Back to quarantine button to return to the main **Quarantine** screen.

Navigation Palette

Figure 29. Navigation palette

Additional navigation facilities include:

- a page navigator for jumping to the previous or the next page of the table in the form of **previous** and **next** links (you may use hotkeys: press CTRL together with the right or left arrow to jump to the next or the previous page correspondingly);

- a page navigator for browsing through all pages of the table in the form of page number links. Link to the current page is not active and marked by the green color.
- a line with the total number of messages in the list and with the range of index numbers of currently displayed messages.

If you want **Dr.Web for UNIX mail servers** to output more messages per page in **Quarantine** section, you may use a drop-down list in the bottom-right corner of the screen. Possible values are: `10`, `20`, `50`, `100`. After you make a selection, the table will be automatically refreshed.

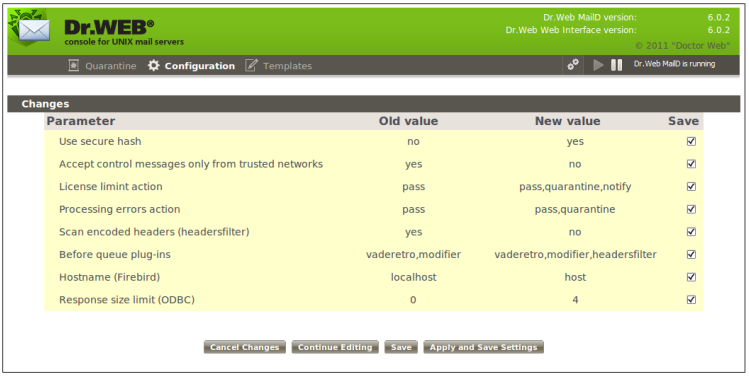
When the table is refreshed or its content is somehow sorted all previous selections are removed.

Configuration

Parameters values can be selected from drop-down lists, by pressing  buttons and specified manually in corresponding text fields. Detailed description of almost each parameter can be found in corresponding reference beside, under the **more** link. After changing any parameter value you can immediately undo the change  or restore default value  only with one click on the corresponding icon appeared beside.

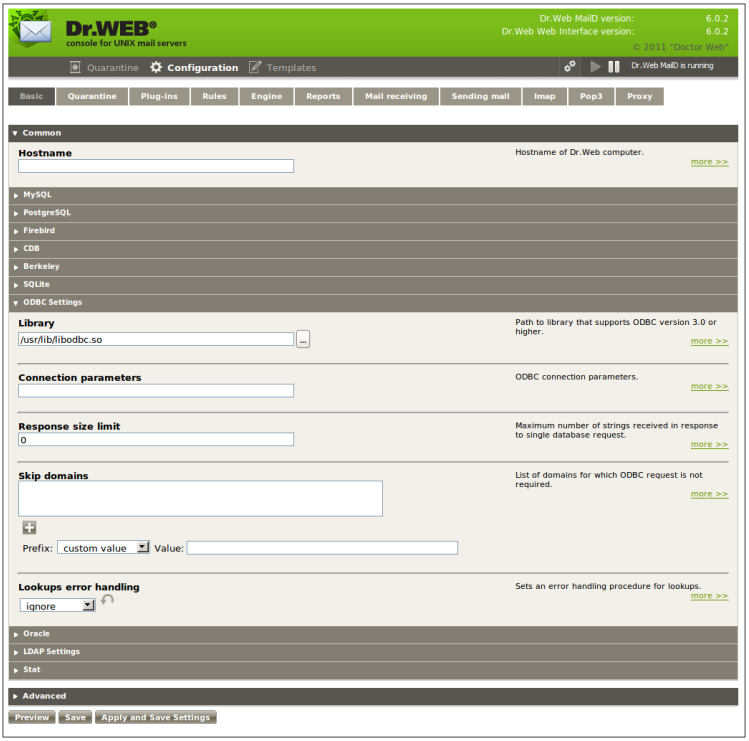
To revise all changes, click **Preview**. On the preview page you can choose whether to save or not all changes or some of them (by unchecking the box near each changed value). If you want to make additional changes, click **Continue Editing** to return to the previous page. If you want to discard the changes, click **Cancel**. Click **Save** to save the changes or **Apply and Save** to save and apply them immediately.

Figure 30. Preview screen



"Basic settings" tab

Figure 31. Basic settings



On this tab you can set up export of statistics and **Dr.Web MailD** interaction with various databases. Parameter values on this tab can be selected from drop-down menus or specified manually in corresponding text fields. Queries to the LDAP server must begin with double or triple slash.

Example

```
//127.0.0.1/dc=origin?description?sub?(cn=$u)
```

Double slash is used when you need to specify the address of the

LDAp-server.

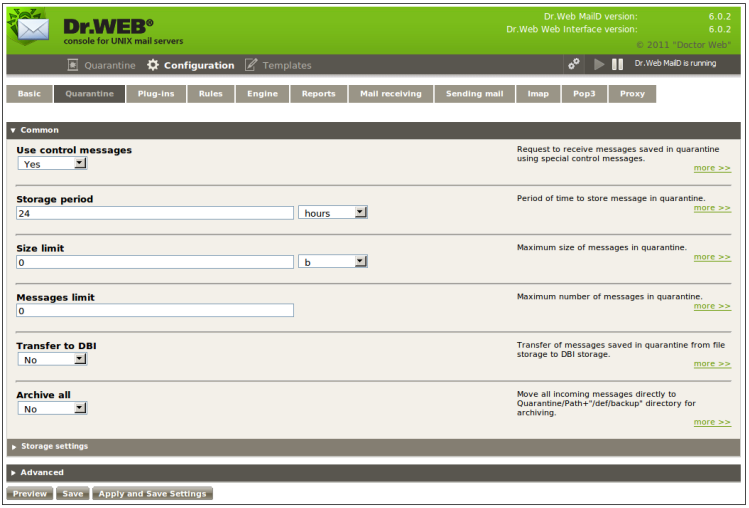
Example

```
///?description?sub?(cn=$u)
```

When using a query using a tiple slash, server specified in the value of **Hostname** parameter from `[LDAP]` section of **Dr.Web MailD** configuration file is used.

"Quarantine" tab

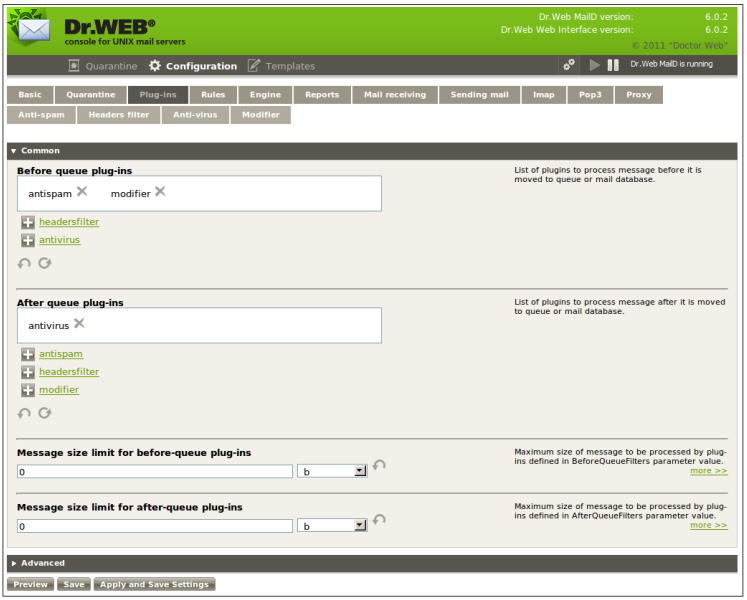
Figure 32. Quarantine settings



On this tab you can set up basic functionality of **Quarantine** section: define period of time to store messages in quarantine, set appropriate privileges for quarantined messages, set renaming options, set up interaction with DBI storage.

"Plug-ins" tab

Figure 33. Settings of plug-ins



This tab contains general settings for all mail filtering plug-ins included in **Dr.Web for UNIX mail servers** solution. Specific settings for each plug-in can be found on corresponding tabs.

Values of additional actions such as **`redirect`**, **`add-header`** and **`add-score`** are not separated by parentheses "(" and ")" and entered "as is". I.e. additional actions are set as immediate values:

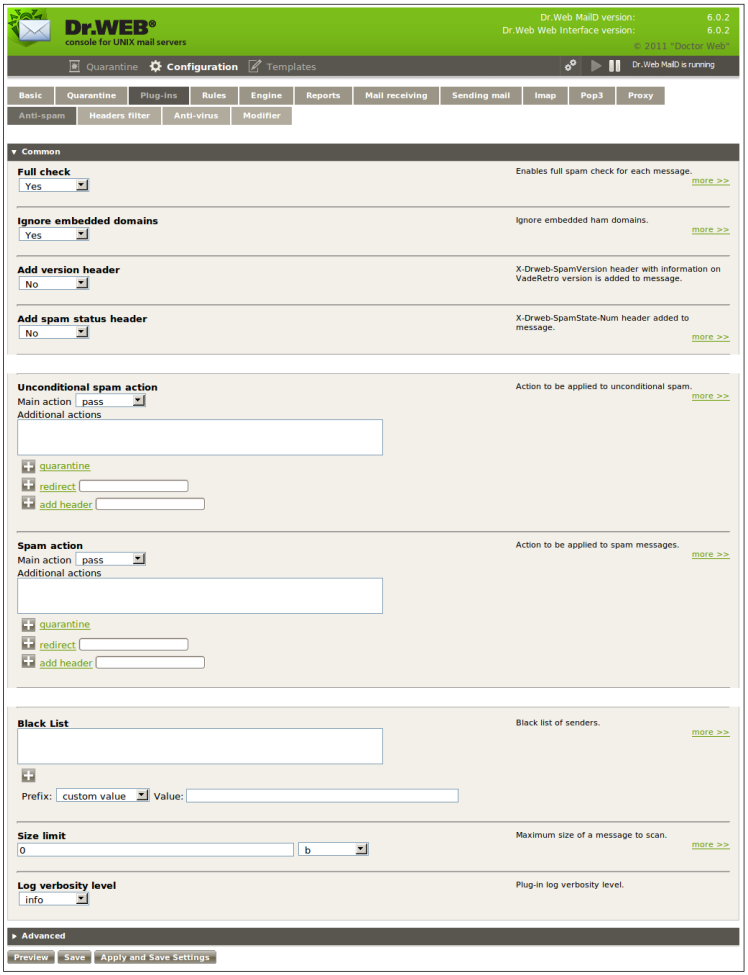
- **`redirect`** action is specified as address list with "|" separator is set: `address1@domain | address2@domain | address3@domain;`
- **`add score`** action only score value is specified ;
- **`add header`** is specified as `[NAME:]BODY`, where `NAME` is header name (`X-DrWeb-MailD` by default), and `BODY` is header value.

Value of **add header** additional action escapes by double quotation marks when added to configuration file (see Configuration File).

As example, after addition of the word "Infected" in title place for infected files in "Antivirus" tab, the following string would be added to plugin configuration file:

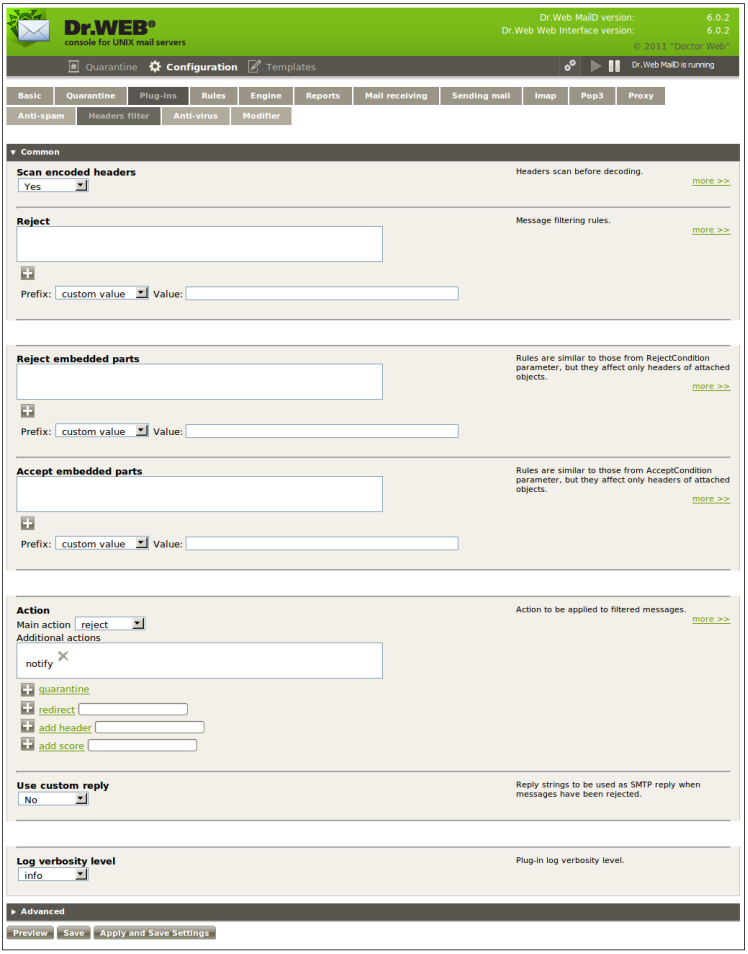
```
Infected = cure,quarantine,notify,"add-header (infected!)"
```

Figure 34. Anti-spam settings



This tab contains general settings for VadeRetro anti-spam plug-in included in **Dr.Web for UNIX mail servers** solution.

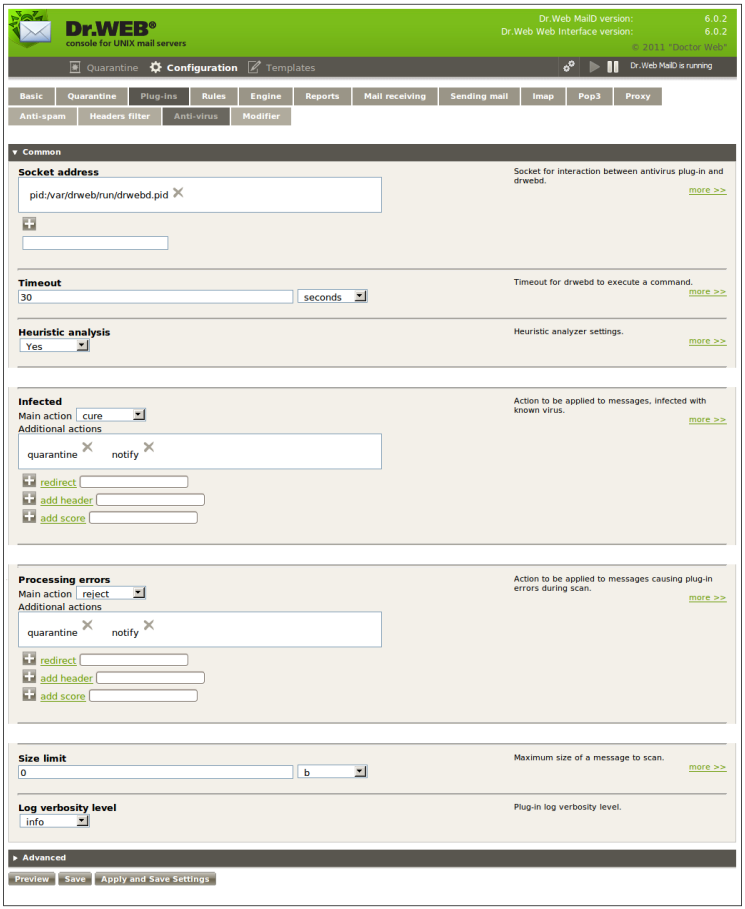
Figure 35. Headers Filter settings



This tab contains general settings for headersfilter plug-in which allows to filter messages according to their headers. The whole string "`HEADER = regular_expression`" must be specified in "value" field for all **`~Condition`** parameters. In text field near

redirect value of **Action** parameter you can specify any e-mail address to be used for redirecting filtered messages.

Figure 36. Anti-virus settings

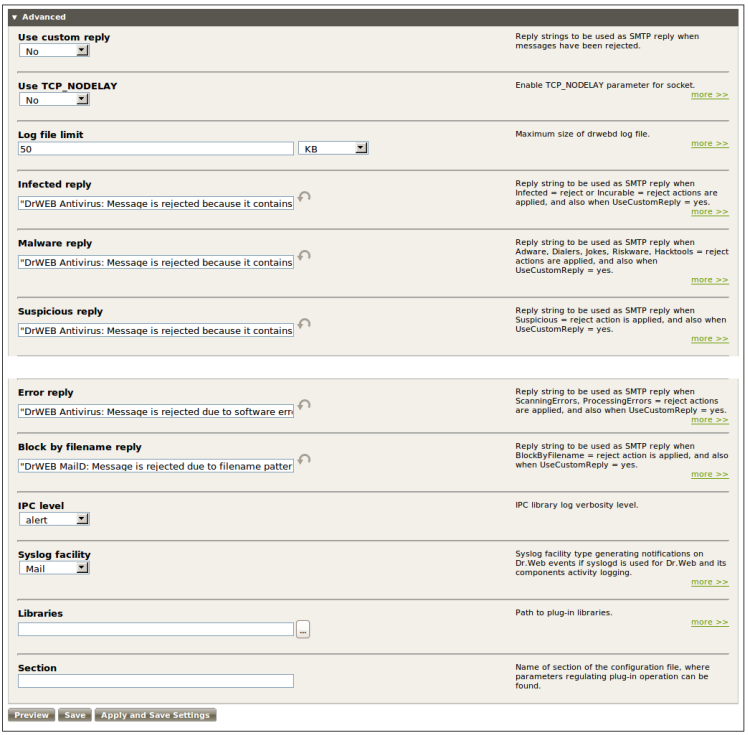


This tab contains general settings for drweb anti-virus plug-in included in **Dr.Web for UNIX mail servers** solution.

In text field near **redirect** value of any parameter related to actions section you can specify any e-mail address to be used for redirecting filtered messages (by default e-mail address specified in **RedirectMail** parameter value on **Engine** tab is used).

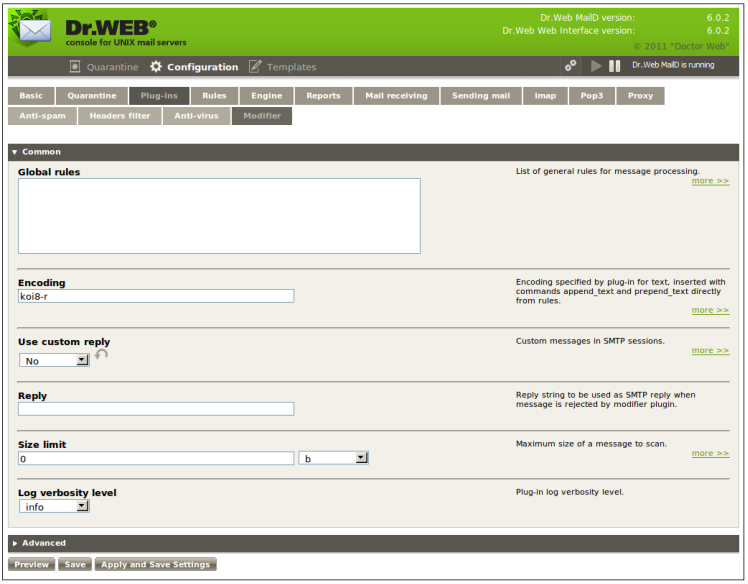
Advanced settings allow to set up custom replies about blocked messages to be sent.

Figure 37. Anti-virus advanced settings



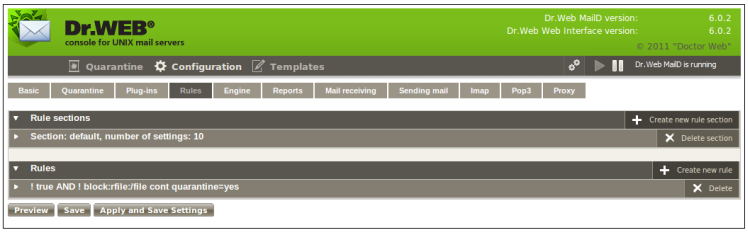
This tab contains general settings for Modifier plug-in included in **Dr.Web for UNIX mail servers** solution.

Figure 38. Modifier settings



"Rules" tab

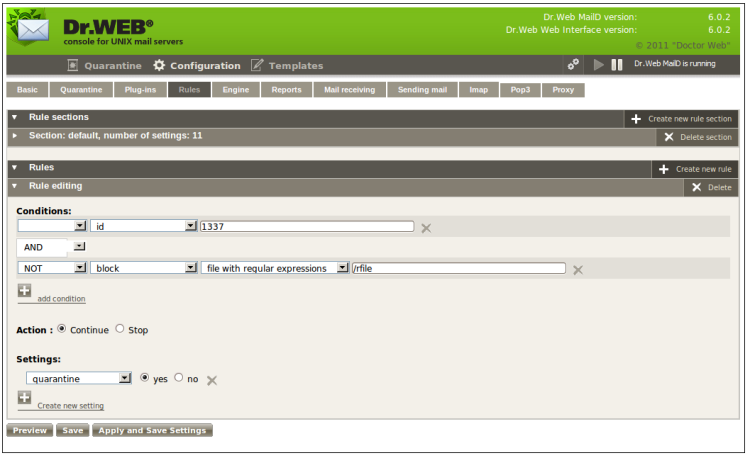
Figure 39. Rules



This tab contains settings for `[Rules]` section of **Dr.Web MailD** configuration file. Both separate rules and bundles of settings can be specified here. Separate rules are created by pressing **Create new rule button**. To edit any rule (either newly created or the

old one) click on it.

Figure 40. Rule editing

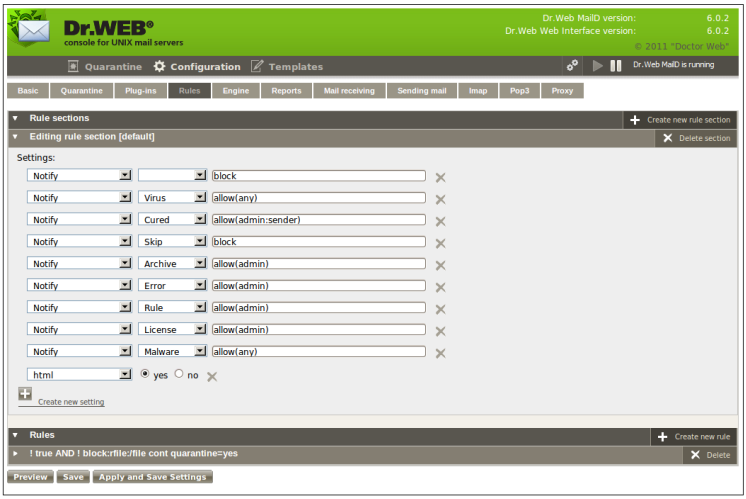


In rule editing menu **Conditions**, **Actions** and **Settings** for each rule can be specified. Conditions can be made up of logical operators.

Current version of **Dr.Web console for UNIX mail servers** doesn't support complex rules that contain associations. Working with these rules is possible only through configuration file.

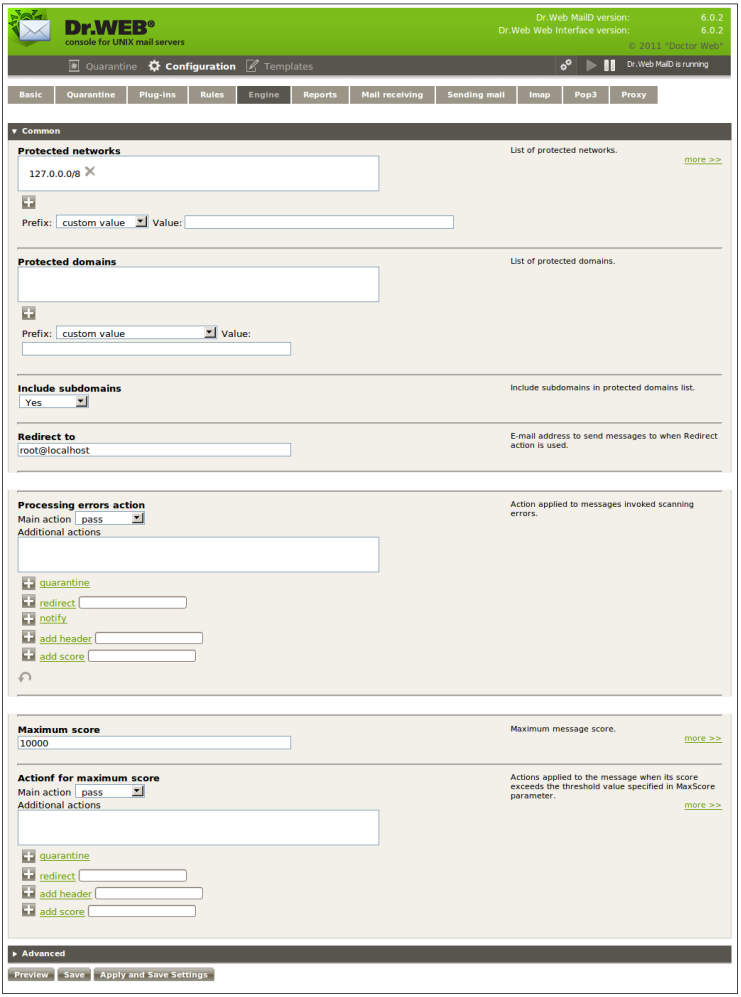
Separate bundles of settings are created by pressing **Create new rules section** button. To edit any rules section (either newly created or the old one) click on it.

Figure 41. Rules section editing



"Engine" tab

Figure 42. General Engine settings

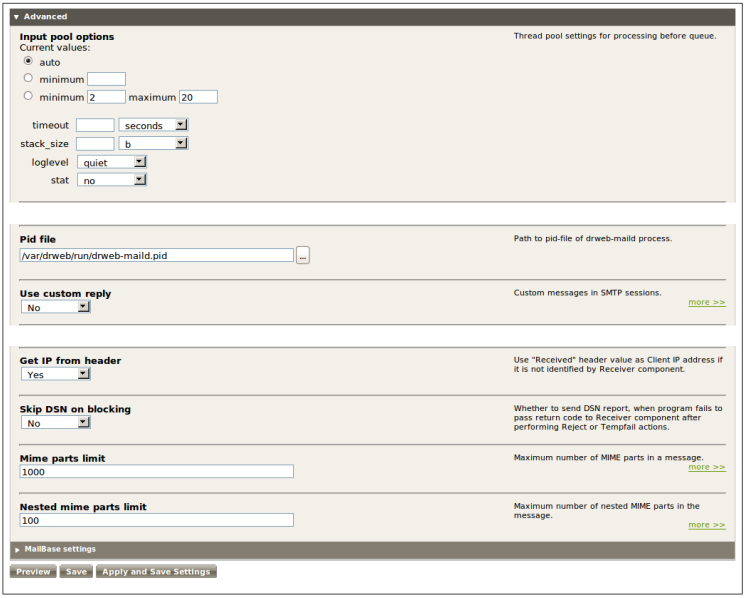


On this tab you can specify e-mail address to be used by default for

redirect action set as value of certain parameters and enable interactive management of `drweb-maild` by control letters.

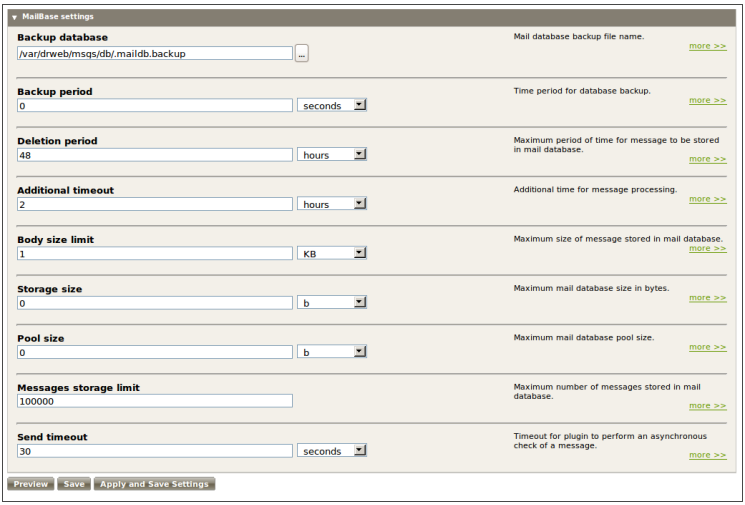
Pool options and custom replies can be specified by choosing appropriate values in corresponding fields.

Figure 43. Advanced Engine settings



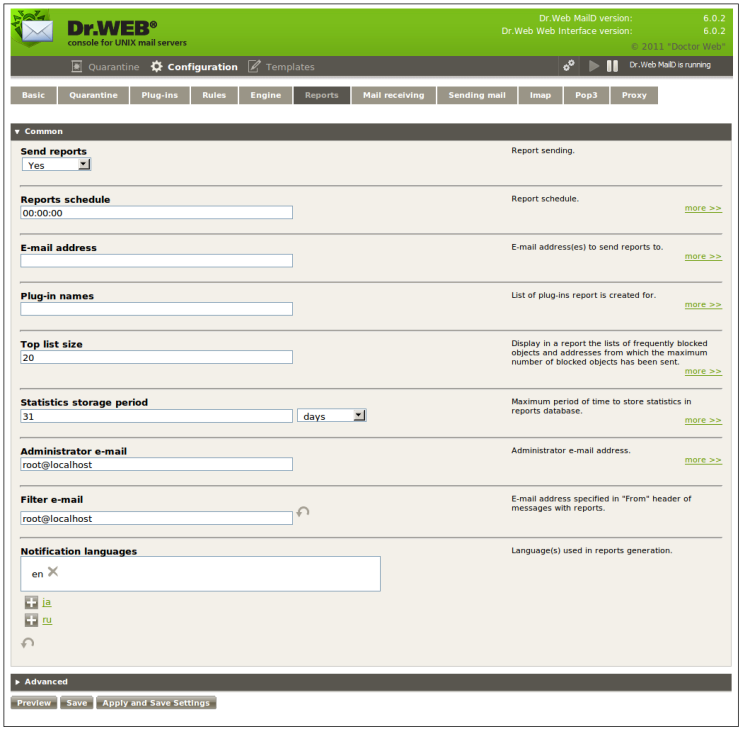
In MailBase section you can manage mail database settings.

Figure 44. MailBase settings



"Reports" tab

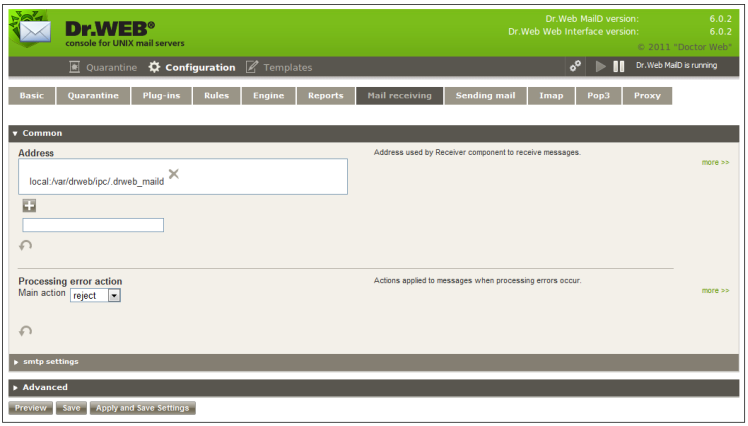
Figure 45. Reports settings



On this tab you can specify various options concerning statistic reports to be sent to Administrator and to be stored in database.

"Mail receiving" tab

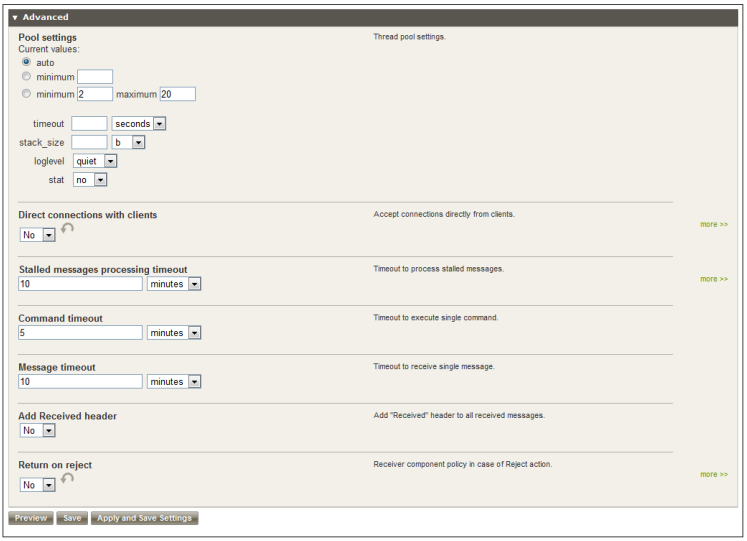
Figure 46. General Mail Receiving settings



On this tab you may specify one or several addresses for receiving SMTP/LMTP-requests and actions applied to messages which envoked processing errors.

On the tab with advanced settings values of parameters responsible for proper interaction between **Dr.Web MailD** and installed MTA can be set.

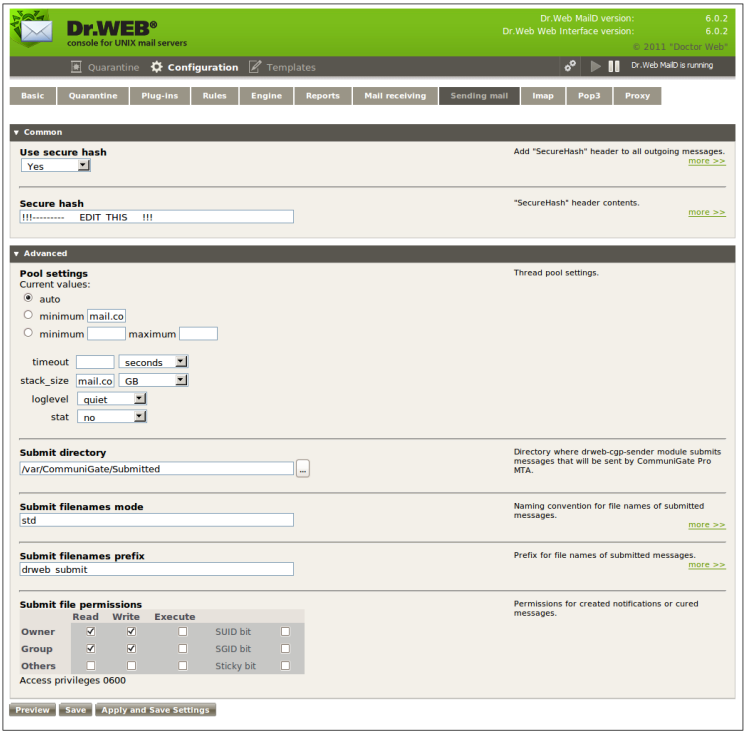
Figure 47. Advanced Mail Receiving settings



Current version of **Dr.Web for UNIX mail servers** doesn't support configuration of simultaneous usage of several Receiver/Sender components via the Web Interface.

"Sending mail" tab

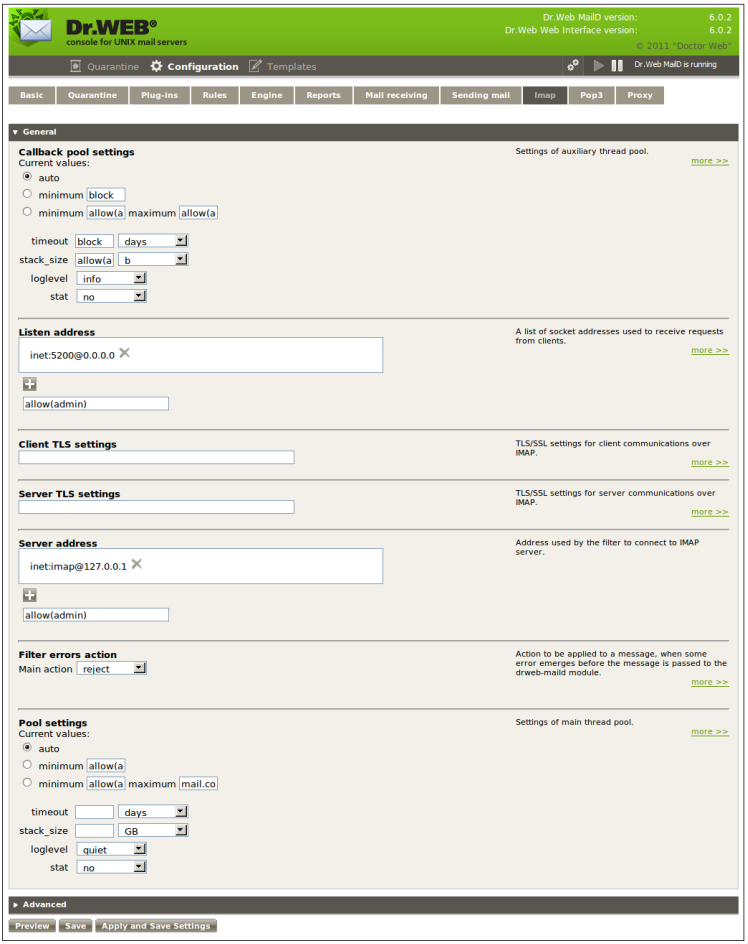
Figure 48. Mail Sending settings



On this tab you can specify actions to be applied to outgoing messages and set timeouts for execution of commands and message processing for **Daemon** and plug-ins.

"IMAP" tab

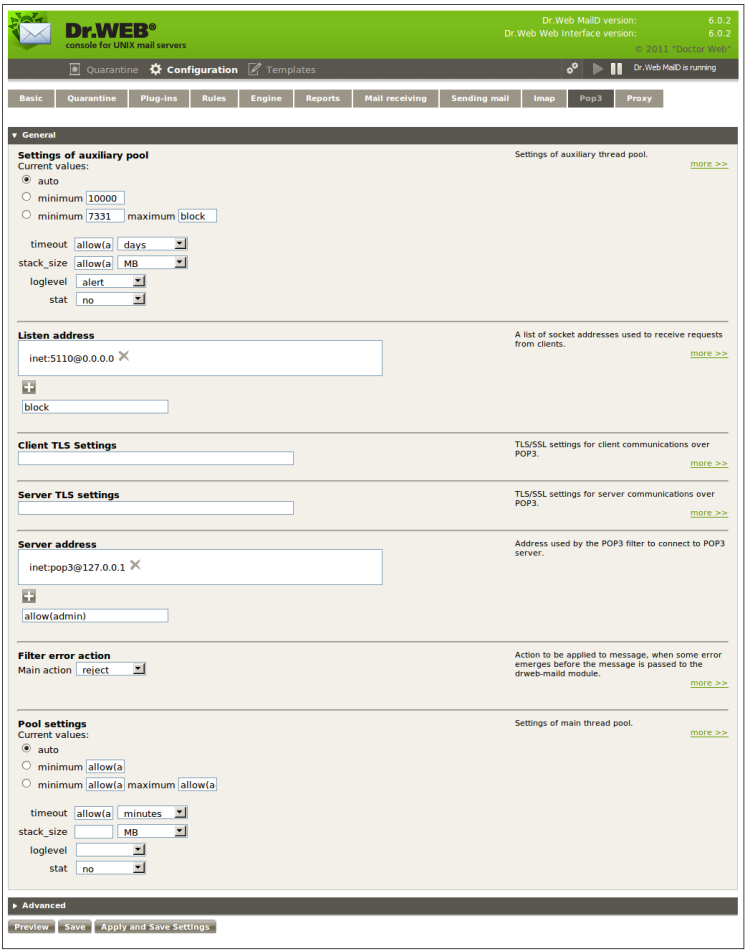
Figure 49. IMAP settings



On this tab you can specify IMAP-filter settings to check mail received via IMAP protocol.

"POP3" tab

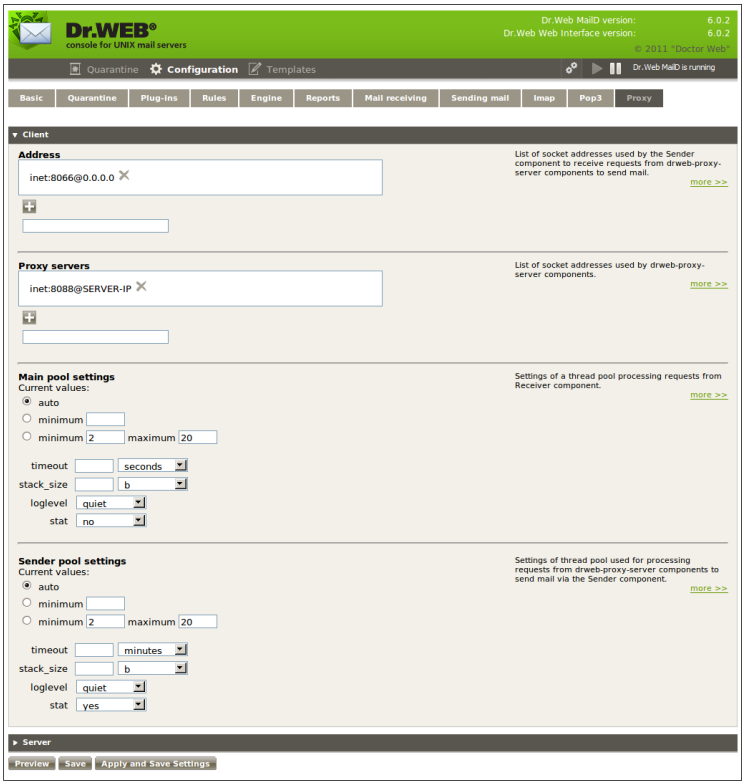
Figure 50. POP3 settings



On this tab you can specify POP3-filter settings to check mail received via POP3 protocol.

"Proxy" tab

Figure 51. Proxy settings



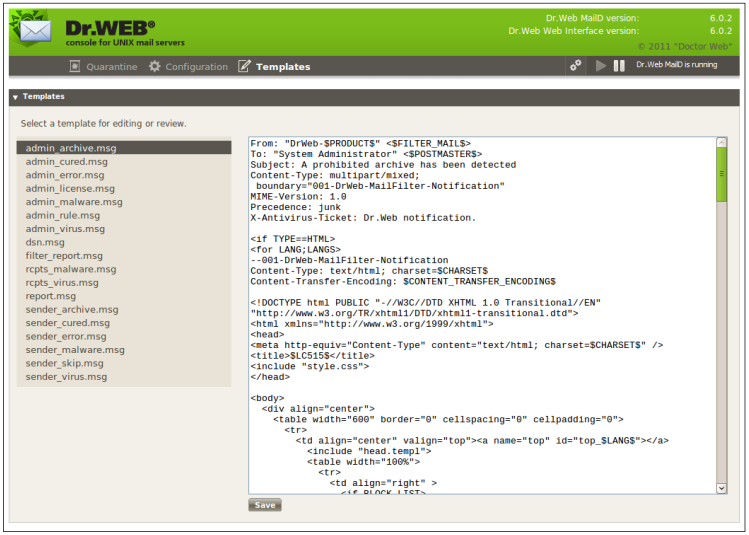
On this tab you can set up operation of proxy, which enables different **Dr.Web for UNIX mail servers** components residing on different hosts interact with each other.

Templates

This section contains templates of reports to be sent to various

recipients upon detection of malicious objects in mail messages and occurrence of errors during **Daemon** or plug-in operation.

Figure 52. Templates



ADMIN_ARCHIVE.msg – template for report generated on detection of archives which cannot be scanned by **Daemon** due to excess of limits set for archives in main configuration file drweb32.ini. It is forwarded to Administrator.

ADMIN_CURED.msg – template for report generated on cure of infected message. It is forwarded to Administrator.

ADMIN_ERROR.msg – template for report generated on **Daemon** or plug-in errors. It is forwarded to Administrator.

ADMIN_LICENSE.msg – template for report generated when message cannot be checked due to license limitations. It is forwarded Administrator.

ADMIN_MALWARE.msg – template for report generated on

detection of malware in mail message. It is forwarded to Administrator.

`ADMIN_RULE.msg` – template for report generated on rejection of message by rule. It is forwarded to Administrator.

`ADMIN_VIRUS.msg` – template for report generated on detection of virus in mail message. It is forwarded to Administrator.

`DSN.msg` – template for delivery status notification.

`RCPTS_MALWARE.msg` – template for report generated on detection of malware in mail message. It is forwarded to recipient.

`RCPTS_VIRUS.msg` – template for report generated on detection of virus in mail message. It is forwarded to recipient.

`REPORT.msg` – template for regular **Daemon** report.

`SENDER_ARCHIVE.msg` – template for report generated on detection of archives which cannot be scanned by **Daemon** due to excess of limits set for archives in main configuration file `drweb32.ini`. It is forwarded to initial message sender.

`SENDER_CURED.msg` – template for report generated on cure of infected message. It is forwarded to initial message sender.

`SENDER_ERROR.msg` – template for report generated on **Daemon** or plug-in errors. It is forwarded to initial message sender.

`SENDER_MALWARE.msg` – template for report generated on detection of malware in mail message. It is forwarded to initial message sender.

`SENDER_SKIP.msg` – template for report generated on message scan failure. It may happen when password protected or broken archive, or non regular file is attached to message, or when message scan is aborted due to timeout. Report is forwarded to initial message sender.

`SENDER_VIRUS.msg` – template for report generated on detection of virus in mail message. It is forwarded to initial message sender.

Run in Enterprise Mode

To startup **Console** in central protection mode, it is necessary to configure **Agent** as described in corresponding section. After making the necessary changes, click  in the navigational menu on the top of the page. In the Basic configuration window, set `Central Protection Mode` parameter value to `Yes` or `Auto`.

`Central Protection Mode` parameter can take three values:

- `No` – in this mode **Console** interacts with local configuration file and doesn't have access to configuration received by the **Agent** from **Enterprise Server**. Configuration changes made in this mode takes effect only when **Agent** is set to `Standalone` mode.
- `Yes` – **Console** receive configuration data from the **Agent** 's socket. If **Agent** work in `Stanalone` mode, the following warning is output on the **Console**:
  `Unable to connect to Dr.Web Agent at local: %var_dir/ipc/.agent`
- `Auto` – **Console** mode is set according to **Agent**'s mode.

If **Agent** have problems connecting to the server, the following behaviours are possible:

- If **Dr.Web Enterprise Server** is unavailable or authorization process fails during first time connection, **Agent** terminates. Check your settings and try to restart **Agent** and **Console**.
- If **Agent** had worked previously with this server, but it's temporary unavailable (for example, in the event of connection problems), **Agent** use backup copies of configuration files received from the server earlier. These files are encrypted and not intended for editing by users. Edited files becomes invalid.

When Console enters `Enterprise` mode, **(CPM)** (Central Protection Mode) inscription appears in the top navigation menu of **Console**.

Рис. 53. Operation mode of Dr.Web console for UNIX mail servers

Configuration of User Permissions

When **Agent** run in `Enterprise` mode, **Dr.Web Control Center** administrator can partially or completely block user's permission to configure **Dr.Web** components installed on workstation.

To set permissions of workstation user:

- Enter the **Dr.Web Control Center**.
- In the main menu, select **Network**, then click the name of a workstation in the hierarchical list. In the control menu (left pane), select **Permissions.** This opens the permissions configuration window.

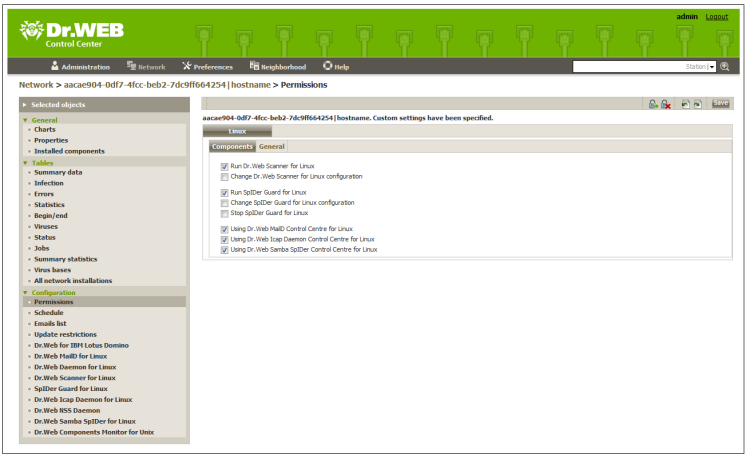


Figure 54. User permissions configuration

- In **Components** choose components aviable to change by workstation user. For example, to allow workstation user edit **Dr.Web for UNIX mail servers** configuration, select the **Using Dr.Web MailD Control Centre for Linux** checkbox and push **Save.**
- To disable workstation user edit **Dr.Web for UNIX mail servers** configuration, clear the **Using Dr.Web MailD Control Centre for Linux** checkbox and click **Save.** In this mode **Console** display corresponding warning and block **Apply and Save Settings, Preview** and **Save** buttons.

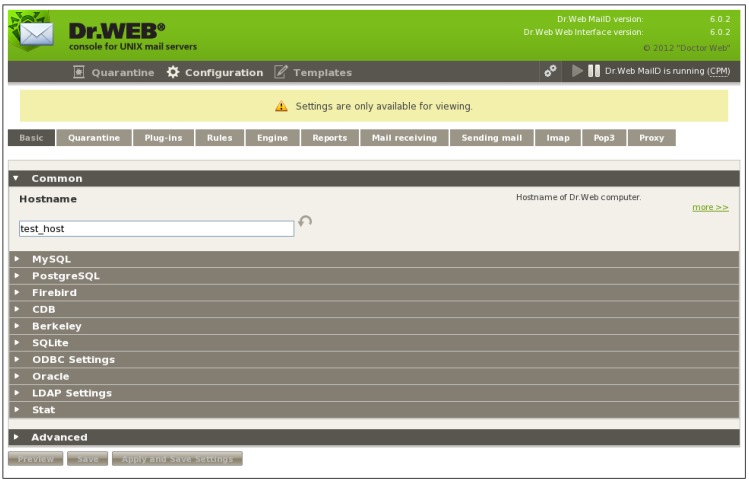


Figure 55. Read-only user permissions

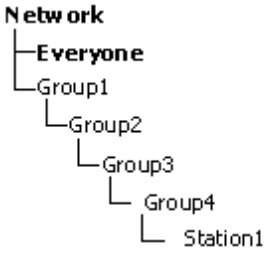
Configuration of Workstation

When a new workstation is created, its configuration settings are inherited from one of the groups it belongs to. That group called the *primary group*. If the settings of the primary group are modified, these changes are inherited by all workstations included into the group, unless the workstations have been customized. When creating a workstation, you can specify what group will be regarded as primary. By default, this is the **Everyone** group.

Inheritance in nested groups depends on group hierarchy. If a station have no personal settings, it inherits the configuration from parental group, and this process repeats recursively. Therefore the search for group configuration is performed upwards through the hierarchical tree of nested groups, starting from the station primary group and stopping at the root group. If no personal settings are selected for all the nesting groups, then the **Everyone** group settings are inherited.

Example

The structure of the hierarchical list is as follows:



The `Group4` is the primary group for the `Station1`. To determine which settings to inherit for the `Station1`, the search is carried out in the following order: `Station1` → `Group4` → `Group3` → `Group2` → `Group1` → `Everyone`.

You can edit configuration inherited from the primary group in two ways:

- Using **Control Center** interface. To do so, select **Network** in the main menu, then click the name of a workstation in the hierarchical list. In the control menu (left pane), select the component you want to configure. To perform this operation you must have corresponding rights. Configuration process is similar to configuring via the Console. When necessary changes are made, click **Save** to save settings.

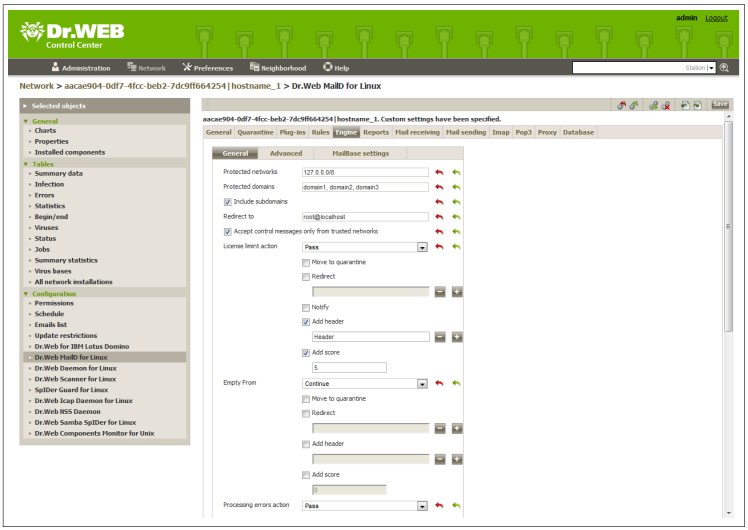


Figure 56. Configuration of Dr.Web MailD for Linux via Dr. Web Control Center interface

With appropriate permission, parameter can be reconfigured via **Console**. Configuration process is similar to configuring in Standalone mode. If workstation user has insufficient rights access is granted in read-only mode.

Types of Administrator Accounts

There are four types of administrator accounts:

- *Administrators with full rights* have exclusive rights to the administration of Enterprise Server and of the whole network. They can view and edit the configuration of the anti-virus network and create new administrator accounts of both types. An administrator with full rights can configure the anti-virus software of a workstation, limit and disable user intervention into the administration of the anti-virus software on the workstation (see p. Setting Users' Permissions).

  Full-rights Administrator can view and edit the list of current administrator accounts.

- *Administrators with read-only rights* can only view the settings of the anti-virus network and its separate elements, but cannot modify them.
- *Group Administrators* have access to all system group and those custom groups which they are allowed to manage (including nested groups). *Group Administrator* accounts could be created for custom groups only (see System and User Groups). Only those groups which such administrators are allowed to access are displayed for them in the hierarchical tree.
  The list of current administrator accounts is not available for *Group Administrators*.
- You can grant *Group Administrators with full-rights* to manage their groups as well as read-only rights.
- After Server is installed, the **admin** account for administrator with full rights is created automatically. Access password for this account is specified during the **Enterprise Server** installation.

Thus, *Administrators with full rights* can:

- Add new and delete already existing administrators accounts.
- Edit settings for all administrators of anti-virus network.

Group administrators and *administrators with read-only rights* can:

- Edit some of settings of their account only.

Contacts

Dr.Web for UNIX mail servers solution is improved constantly. You can find the news and latest information on available updates on the website:

http://www.drweb.com/

Sales department:

http://buy.drweb.com/

Technical support:

http://support.drweb.com/

Please include the following information in the problem report:

- full name and version of your operating system;
- versions of **Dr.Web for UNIX mail servers** modules;
- configuration files for all modules;
- log files for all modules.