Dr.WEB for UNIX Internet Gateways

Administrator Manual

Trademarks

Disclaimer

In no event shall Doctor Web and its resellers or distributors be liable for errors or omissions, or any loss of profit or any other damage caused or alleged to be caused directly or indirectly by this document, the use of or inability to use information contained in this document.

Dr.Web for UNIX Internet Gateways
Version 11.0
Administrator Manual
9/7/2018

Doctor Web Head Office

2-12A, 3rd str. Yamskogo polya

Moscow, Russia

125040

Website: https://www.drweb.com/

Phone: +7 (495) 789-45-87

Refer to the official website 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

Conventions and Abbreviations

The following symbols and text conventions are used in this guide:

Convention	Comment
!	Important note or instruction.
⚠	Warning about possible errors or important notes to which you should pay special attention.
Anti-virus network	A new term or an accent on a term in descriptions.
<IP-address>	Placeholders.
Save	Names of buttons, windows, menu items and other program interface elements.
CTRL	Keyboard keys names.
`/home/user`	Names of files and folders, code examples.
Appendix A	Cross-references on the document chapters or internal hyperlinks to web pages.

> Command-line commands, which are entered via a keyboard (in the terminal or terminal emulator), are marked with the command prompt character `$` or `#` in the current manual. The character indicates the privileges required for execution of the specified command. According to the standard convention for UNIX-based systems
>
> `$`—indicates that the command can be executed with user rights.
>
> `#`—indicates that the command can be executed with superuser (usually *root*) privileges. To elevate the privileges, use **su** and **sudo** commands.

List of abbreviations is in section Appendix G. List of Abbreviations.

Introduction

Thank you for purchasing Dr.Web for UNIX Internet Gateways. It offers reliable protection of your server and its users from distribution of various types of computer threats using the most advanced virus detection and neutralization technologies.

This manual is intended to help administrators of the servers that run an OS of the **GNU/Linux** family or other UNIX-like operating systems, such as **Solaris** and **FreeBSD**, to install and use Dr.Web for UNIX Internet Gateways 11.0..

Convention for Paths to Product Files

The product described in the present document is designed for operation in different **UNIX**-based operating system. Real paths to product files depend on the operating system installed on the user's computer. For notational convenience, the following conventions are used:

- *<opt_dir>*—directory where main product files are located (including executable files and libraries).
- *<etc_dir>*—directory where the configuration file and a key file are located.
- *<var_dir>*—directory where supporting and temporary product files are located.

Real paths corresponding to the conventions in different operating systems are given in the table below.

OS Type	**Convention**	**Real Path**
GNU/Linux, Solaris	*<opt_dir>*	`/opt/drweb.com`
	<etc_dir>	`/etc/opt/drweb.com`
	<var_dir>	`/var/opt/drweb.com`
FreeBSD	*<opt_dir>*	`/usr/local/libexec/drweb.com`
	<etc_dir>	`/usr/local/etc/drweb.com`
	<var_dir>	`/var/drweb.com`

For space considerations, examples use paths for **GNU/Linux** operating systems. In some places of the document, where it is possible, examples contain real paths for all of the operating systems.

About this Product

Dr.Web for UNIX Internet Gateways is designed to protect Internet-gateways running under UNIX (**GNU/Linux**, **Solaris** and **FreeBSD**) from viruses and other types of any malicious software, as well as to prevent distribution of these threats developed for various platforms.

Main components (anti-virus engine and virus databases) are not only highly effective and resource-sparing, but also cross-platform, which lets Doctor Web specialists create reliable anti-virus solutions protecting computers and mobile devices under popular operating systems from threats that target different platforms. Currently, along with Dr.Web for UNIX Internet Gateways, Doctor Web offers anti-virus solutions for both **UNIX**-based operating systems (such as **GNU/Linux**, **Solaris** and **FreeBSD**) and **IBM OS/2**, **Novell NetWare**, **macOS** and **Windows**. Moreover, other anti-virus products have been developed to deliver protection for devices that run **Android**, **Symbian**, **BlackBerry**, and **Windows Mobile**.

Components of the Dr.Web for UNIX Internet Gateways are constantly updated, and virus databases, databases of web resources categories and databases of rules for spam filtering of email messages are regularly supplemented with new signatures to ensure up-to-date protection of servers, workstations and mobile users and their programs and data. To provide additional protection against unknown viruses heuristic analysis methods are implemented in the anti-virus engine and to the Dr.Web Cloud service that stores information about the latest threats, signatures of which are absent in the database (this function is not available for all products).

Main Functions

Dr.Web for UNIX Internet Gateways main functions:

1. **Detection and neutralization of threats.** Searches for malicious programs (for example, viruses, including those that infect mail files and boot records, Trojans, mail worms) and unwanted software (for example, adware, joke programs, dialers). To find more information on computer threat types, refer to Appendix A. Types of Computer Threats.

   Threat detection methods:

   - *Signature analysis*, which allows detection of known threats
   - *Heuristic analysis* , which allows detection of threats that are not present in virus databases
   - *Dr.Web Cloud* service that collects up-to-date information about recent threats and sends it to Dr.Web products.

   Note that the heuristic analyzer may raise false positive detections. Thus, objects that contain threats detected by the analyzer are considered “suspicious”. It is recommended that you choose to quarantine such files and send them for analysis to Doctor Web anti-virus laboratory. For details on methods used to neutralize threats, refer to Appendix B. Neutralizing Computer Threats.

   When scanning the file system on the user's request, it is possible of either full scan of all the file system objects available to user, or selective scan of the specified objects only (separate directories or files that meet the specified criterias). In addition, it is possible to perform separate checks of boot records of volumes and executable files which support currently active processes in the system. In the latter case, when a threat is detected, it is not only neutralized the malicious executable file, but all processes running from it are forcibly terminated. In systems that implement a mandatory model of access to files with a set of different access levels, the scanning of files that are not available at the current access level can be done in special autonomous copy mode .

The Dr.Web Ctl command-line management tool included in the product allows to scan for threats file systems of remote network hosts, that provide remote terminal access via SSH.

> The remote scanning can be used only for detection of malicious and suspicious files on a remote host. To eliminate detected threats on the remote host, it is necessary to use administration tools provided directly by this host. For example, for routers and other “smart” devices, a mechanism for a firmware update can be used; for computing machines, it can be done via a connection to them (as an option, using a remote terminal mode) and respective operations in their file system (removal or moving of files, etc.), or via running an anti-virus software installed on them.

2. **Analyzing data transmitted to the Internet.** Not only user requests are monitored (i.e. attempts to connect to the web server and to transmit any file to it), but also data sent in response to users’ request. To analyze requests and sent data, the program connects via ICAP protocol as an external filter to the proxy server, processing HTTP connections of the local network users. Moreover, using the SpIDer Gate component, it is possible to perform barrier functions, which prevents receiving and transmitting infected files by the public server of the organization (*this option is available only for* **GNU/Linux**). To restrict access to unwanted websites, the product uses automatically updated databases of web resource categories, which are supplied together with Dr.Web for UNIX Internet Gateways; and white and black lists created by the system administrator manually. The product also refers to the Dr.Web Cloud service to check for the information whether the Internet resource is marked as malicious by other Dr.Web products.

> Not that the product *is not intended* for transit network traffic checks. It is intended for integration with locally installed HTTP proxy server (for example, **Squid**) or web server.

3. **Reliable isolation of infected or suspicious objects**. Such objects detected in the server's file system are moved to a special storage, quarantine, to prevent any harm to the system. When moved to quarantine, objects are renamed according to special rules and, if necessary, they can be restored to their original location only on demand.

   The threats detected by the Dr.Web ICAPD component in the HTTP protocol messages are not moved to Quarantine on the Internet gateway. Instead their load and transfer to a recipient are blocked, and the user is informed by a special HTML page with a message about blocking.

4. **Automatic update** of the anti-virus engine, virus databases, databases of web resource categories for the maintenance of the high level of protection against malware.

5. **Collection of statistics** on virus events, logging threat detection events. Notification on detected threats over SNMP to external monitoring systems and to the central protection server (if the product operates in central protection mode).
6. **Operation in central protection mode** (when connected to the central protection server, such as Dr.Web Enterprise Server or as a part of Dr.Web AV-Desk service). This mode allows implementation of a unified security policy on computers within the protected network. It can be a corporate network, a private network (VPN), or a network of a service provider (for example, a provider of Internet service).

Program Structure

Dr.Web for UNIX Internet Gateways is a product that consists of several components, each of which has its individual set of functions. The list of components included in Dr.Web for UNIX Internet Gateways are listed below.

Component	Description
Dr.Web ConfigD	Configuration daemon Dr.Web for UNIX Internet Gateways, which performs the following functions: • Starts and stops the product’s components depending on the settings. Automatically restarts components if a failure in their operation occurs. Starts components at the request of other components. Informs active components when another component starts or shuts down. • Stores information about present license keys and settings and provides this data to all components. Receives adjusted settings and license keys from the components of Dr.Web for UNIX Internet Gateways expected to provide such information. Notifies other components on changes in license keys and settings. Executable file: **drweb-configd** Internal name output to the log file: `ConfigD`
Dr.Web Virus-Finding Engine	Anti-virus engine. The main component of the anti-virus protection. Implements algorithms to detect viruses and malicious programs as well as algorithms to analyze suspicious behavior (by using signature and heuristic analysis). Used by all Dr.Web for UNIX Internet Gateways components via Dr.Web Scanning Engine. Executable file: **drweb32.dll** Internal name output to the log file: `CoreEngine`
Dr.Web Scanning Engine	Scanning engine. The component responsible for loading the anti-virus engine Dr.Web Virus-Finding Engine and virus databases. It transmits the contents of files and disk boot records to the anti-virus engine for scanning at the request of other components of Dr.Web for

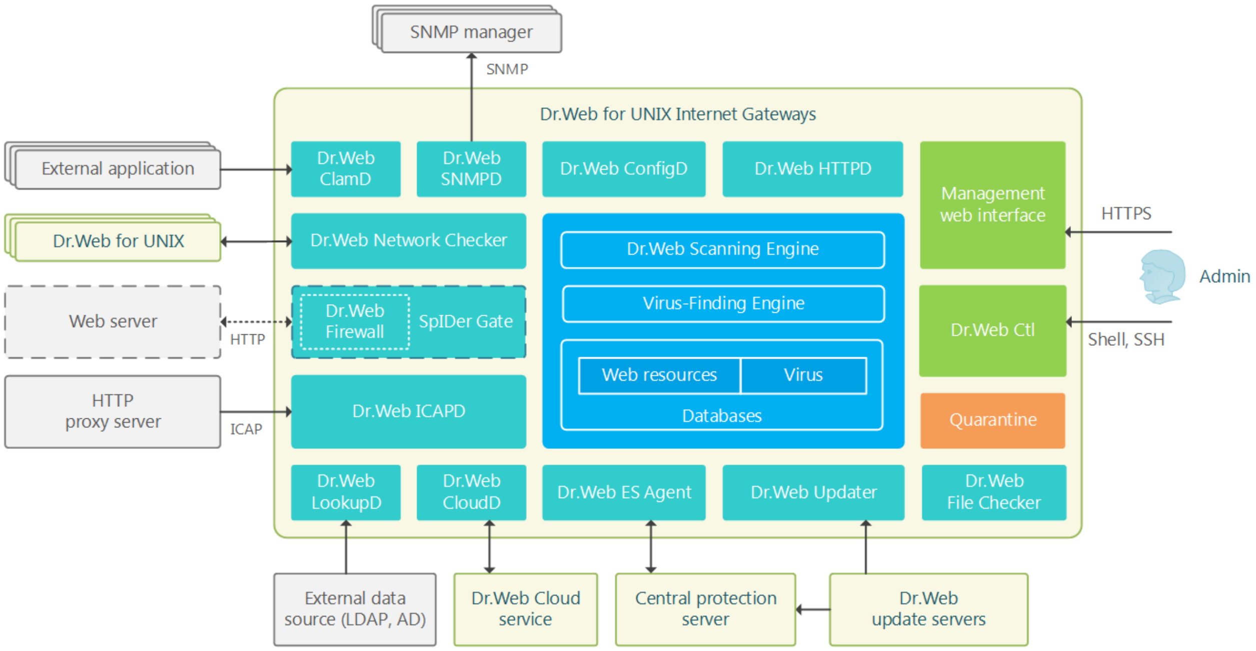
Component	Description
	UNIX Internet Gateways. It queues files that are waiting to be scanned. Cures the files that can be cured. From the point of view of other components of Dr.Web for UNIX Internet Gateways, this component provides the anti-virus scanning service. Can operate under the control of the Dr.Web ConfigD configuration daemon or in an autonomous mode (autonomously from other components). Used by all Dr.Web for UNIX Internet Gateways components for the anti-virus scanning. Executable file: **drweb-se** The internal name, displayed in log: `ScanEngine`
Dr.Web virus database	Automatically updated database of these virus signatures and other threats, also algorithms of detection and neutralization of malicious software. Used by the anti-virus engine Dr.Web Virus-Finding Engine and provided along with it.
Databases of web resource categories	Automatically updated database. The database contains information on web resources assigned to pre-defined categories. It is used for blocking access to web resources included to categories that are marked as unwanted. Used by components that scan network activity of users and applications, such as SpIDer Gate, Dr.Web ICAPD, Dr.Web MailD.
Dr.Web File Checker	The component which scans file system objects and manages quarantined files. It receives scanning tasks from other Dr.Web for UNIX Internet Gateways components and searches file system directories according to a received task, transmits files for scanning to Dr.Web Scanning Engine scanning engine and notifies components on scanning progress. It also removes infected files, moves them to quarantine, restores them from quarantine, and manages quarantine directories. The component creates and updates cache that stores information on scanned files to lessen the frequency of repeated file scanning. Used by components that scan file system objects. Executable file: **drweb-filecheck** The internal name, displayed in log: `FileCheck`
Dr.Web ICAPD	ICAP server analyzing requests and traffic which goes via HTTP proxy servers (such as **Squid**). It also prevents transmitting infected files and access to the network hosts belonging to the Internet resources categories and to black lists, created by the system administrator. If access to external servers must be forbidden, or transmitted data

Component	Description
	contains a threat, it instructs the proxy server to return to a user a special page informing that it is impossible to access the requested resource or that the transmitted file is infected. Executable file: **drweb-icapd** The internal name, displayed in log: `ICAPD`
Dr.Web ES Agent	Central protection agent. Makes it possible for the product to operate in centralized and mobile modes. Provides communication between the product and the central protection server, a license key file, updates to the virus databases and components. Sends to the server information on the components included in Dr.Web for UNIX Internet Gateways and their state as well as statistics of virus events. Executable file: **drweb-esagent** The internal name, displayed in log: `ESAgent`
SpIDer Gate	The component for monitoring network traffic and URLs. It is designed to check data downloaded from the network to the local host and transmitted from it to the external network for threats. The components also prevents connections with the network hosts, included not only to the unwanted categories of web resources, but also to black lists created by the system administrator. It is included only in the distributions for **GNU/Linux** OS. Executable file: **drweb-gated** The internal name, displayed in log: `GateD`
Dr.Web Firewall for Linux	Connection manager. Used by SpIDer Gate and provides connection routing for applications that operate on the server for scanning of the transferred traffic. It is included only in the distributions for **GNU/Linux** OS. Executable file: **drweb-firewall** The internal name, displayed in log: `LinuxFirewall`
Dr.Web Network Checker	An agent of the network data scanning. Used to send data to the scanning engine for actual scanning. The data is sent by components of the product via the network (such components as Dr.Web ClamD, SpIDer Gate, Dr.Web ICAPD).

Component	Description
	Besides, it allows Dr.Web for UNIX Internet Gateways to arrange a distributed scanning of files: to receive/transmit files for scanning from/to remote hosts. For that purpose, remote hosts must feature an installed and running Dr.Web for UNIX-based operating systems. In the distributed scanning mode. it allows automatic distribution of scanning load among remote hosts by reducing load on hosts with a large number of scanning tasks (for example, on mail servers, file servers, Internet gateways). For security reasons, files are transmitted over SSL. Executable file: **drweb-netcheck** The internal name, displayed in log: `NetCheck`
Dr.Web HTTPD	Web interface for managing Dr.Web for UNIX Internet Gateways components. It consists of management web interface (it should be installed separately) and service interface for operation of Dr.Web Link Checker(can be installed additionally) browser extension. You can access the interface via any browser on a local or remote host. In-built web interface enables the product to use neither third-party web servers (such as **Apache HTTP Server**) nor remote administration tools, such as **Webmin**. For security reasons, web interface interacts with user over HTTPS. Executable file: **drweb-httpd** The internal name, displayed in log: `HTTPD`
Dr.Web Ctl	Tool for managing Dr.Web for UNIX Internet Gateways from the command line. Allows the user to start file scanning, to view quarantined objects, to start a virus database update procedure, to connect the product to or to disconnect it from the central protection server, to view and to configure parameters. Executable file: **drweb-ctl** The internal name, displayed in log: `Ctl`
Dr.Web Updater	An update component. Downloads from Doctor Web servers updates of the virus databases and databases of web resource categories, anti-virus engine. The updates can be downloaded automatically, according to a schedule, and on user's demand (via Dr.Web Ctl or management web interface). Executable file: **drweb-update** The internal name, displayed in log: `Update`

Component	Description
Dr.Web SNMPD	An SNMP agent. Designed for integration of Dr.Web for UNIX Internet Gateways with external monitoring systems over SNMP. Such integration allows you to monitor the state of the product's components and to collect statistics on threat detection and neutralization. Supports SNMP v2c and v3. Executable file: **drweb-snmpd** The internal name, displayed in log: `SNMPD`
Dr.Web ClamD	Component emulating interface of the anti-virus daemon **clamd,** which is a component of **ClamAV®** anti-virus. Allows all applications that support **ClamAV®** to transparently use Dr.Web for UNIX Internet Gateways for anti-virus scanning. Executable file: **drweb-clamd** The internal name, displayed in log: `ClamD`
Dr.Web CloudD	The component that sends the following information to the Dr.Web Cloud service: visited URLs and information about the scanned files, to check them for threats not yet described in virus databases. Executable file: **drweb-cloudd** The internal name, displayed in log: `CloudD`
Dr.Web LookupD	Component retrieving data from external data sources (directory services, such as **Active Directory**) using LDAP protocol. The data are used in rules of traffic monitoring. Executable file: **drweb-lookupd** The internal name, displayed in log: `LookupD`

The figure below shows the structure of Dr.Web for UNIX Internet Gateways and its operation with external applications.



Quarantine Directories

Quarantine directories of Dr.Web for UNIX Internet Gateways 11.0 serve for isolation of files that pose a threat to system security and cannot be currently cured. Such threats are those that are unknown to Dr.Web for UNIX Internet Gateways (that is, a virus is detected by the heuristic analyzer but the virus signature and method to cure are absent in the databases) or those that caused an error during curing. Moreover, a file can be quarantined at user request if the user selected this action in the list of detected threats or specified this action in settings as reaction to this threat type.

When a file is quarantined, it is renamed according to special rules. Renaming of isolated files prevents their identification by users or applications and complicates access to them in case of attempt to bypass quarantine management tools implemented in Dr.Web for UNIX Internet Gateways. Moreover, when a file is moved to quarantine, the execution bit is reset to prevent an attempt to run this file.

Quarantine directories are located in

- *user home directory* (if multiple user accounts exist on the computer, a separate quarantine directory can be created for each of the users)
- *root directory of each logical volume* mounted to the file system

Dr.Web quarantine directories are always named as `.com.drweb.quarantine` and are not created until the Quarantine action is applied. At that, only a directory required for isolation of a concrete object is created. When selecting a directory, the file owner name is used: search is performed upwards from the location where the malicious object resides and if the owner home directory is reached, the quarantine storage created in this directory is selected. Otherwise, the file is isolated in the quarantine created in the root directory of the volume (which is not always the same as the file system root directory). Thus, any infected file moved to quarantine is always located on the volume, which provides for correct operation of quarantine in case several removable data storages and other volumes are mounted to different locations in the system.

A user can manage quarantine contents from the command line using the utility Dr.Web Ctl, or via the management web interface (if it is installed). Every action is applied to the consolidated quarantine; that is, changes affect all quarantine directories available at the moment.

> Operation with quarantined objects is allowed even if no active license is found. However, isolated objects cannot be cured in this case.
>
> Not all anti-virus components of Dr.Web for UNIX Internet Gateways can use Quarantine for threat isolation. For example, it is not used by the Dr.Web ClamD, as well as by Dr.Web ICAPD and Dr.Web MailD components (may not be included in the your product).

File Permissions and Privileges

To scan objects of the file system and neutralize threats, Dr.Web for UNIX Internet Gateways (or rather the user under whom it runs) requires the following permissions:

Action	Required rights
Listing all detected threats	Unrestricted. No special permission required.
List archive contents (display only corrupted or malicious elements)	Unrestricted. No special permission required.
Moving to quarantine	Unrestricted. The user can quarantine all infected files regardless of read or write permissions on them.
Deleting threats	The user needs to have write permissions for the file that is being deleted. If threat is detected in a file located in a container (an archive, email message, etc.), its removal is replaced with moving of a container to quarantine.
Curing	Unrestricted. The access permissions and owner of a cured file remain the same after curing.

Action	Required rights
	The file can be removed if deletion can cure the detected threat.
Restoring a file from quarantine	The user should have permissions to read the file and to write to the restore directory.
Deleting a file from quarantine	The user must possess write permissions to the file that was moved to quarantine.

To enable operation of the command-line management Dr.Web Ctl tool with superuser (*root*) privileges, you can use the **su** command, which allows to change the user, or the **sudo** command, which allows you to execute a command as another user.

Note that Dr.Web Scanning Engine scanning engine cannot check file which size exceeds 4 Gbytes (on attempt to scan such files, the following error message displays: *"File is too large"*).

Operation Modes

Dr.Web for UNIX Internet Gateways can operate both in standalone mode and as a part of an *anti-virus network* managed by a *central protection server*. Operation in *central protection mode* does not require installation of additional software or Dr.Web for UNIX Internet Gateways re-installation or removal.

- *In Standalone mode*, the protected computer is not connected to an anti-virus network and its operation is managed locally. In this mode, configuration and license key files are located on local disks and Dr.Web for UNIX Internet Gateways is fully controlled from the protected computer. Updates to virus databases are received from Doctor Web update servers.
- *In Central protection mode (Enterprise mode)*, protection of the computer is managed by the central protection server. In this mode, some functions and settings of Dr.Web for UNIX Internet Gateways can be adjusted in accordance with the general (corporate) anti-virus protection policy implemented on the anti-virus network. The license key file used for operating in enterprise mode is received from the central protection server. The key file stored on the local computer, if any, is not used. Statistics on virus events is sent to the central protection server. Updates to virus databases are also received from the central protection server.
- *In Mobile mode*, Dr.Web for UNIX Internet Gateways receives updates from Doctor Web update servers, but operation of the product is managed with the local settings. The used key file is received from the central protection server. You can switch to mobile mode only if it is allowed in the central protection server settings.

Central Protection Concept

Doctor Web solutions for central protection use client-server model (see the figure below).

Workstations and servers are protected by *local anti-virus components* (herein, Dr.Web for UNIX Internet Gateways) installed on them, which provides for anti-virus protection of remote computers and allows connection between the workstations and the central protection server.

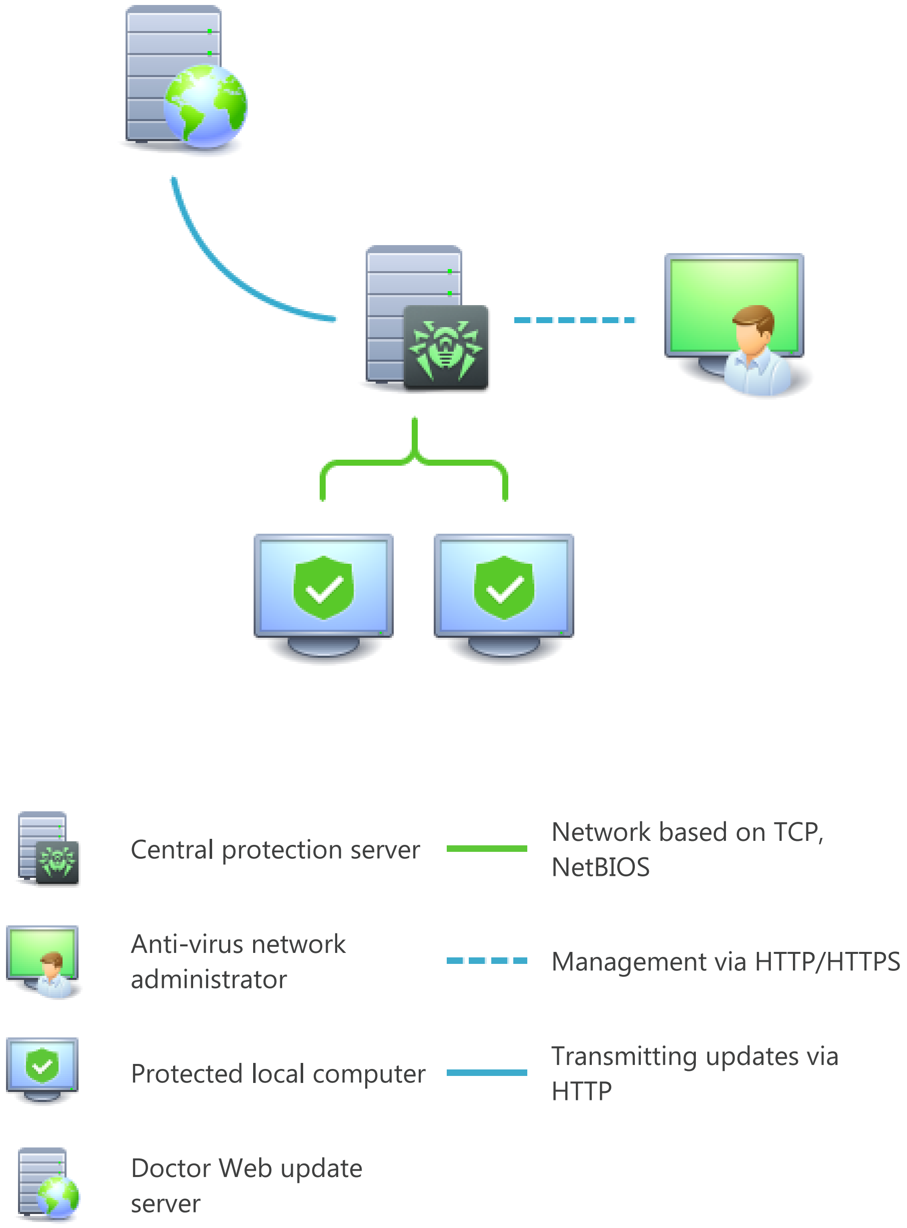

Central protection server
Network based on TCP, NetBIOS
Anti-virus network administrator
Management via HTTP/HTTPS
Protected local computer
Transmitting updates via HTTP
Doctor Web update server

Figure 1. Logical structure of the Anti-virus Network

Local computers are updated and configured from the *central protection server*. The stream of instructions, data and statistics in the anti-virus network goes also through the central protection server. The volume of traffic between protected computers and the central server can be quite sizeable, therefore solutions provide options for traffic compression. To prevent leak of sensitive data or substitution of software downloaded onto protected computers, encryption is also supported.

All necessary updates are downloaded to the central protection server from Doctor Web update servers.

Local anti-virus components are configured and managed from the central protection server according to commands received from anti-virus network administrators. Administrators manage central protection servers and topology of anti-virus networks (for example, validate connections to the central protection server from remote computers) and configure operation of local anti-virus components when necessary.

> ⚠ Local anti-virus components are not compatible with anti-virus products of other companies or anti-virus solutions of Dr.Web if the latter do not support operation in central protection mode (for example, Dr.Web Anti-virus, version 5.0). Installation of two anti-virus programs on the same computer can cause a system crash and loss of important data.
>
> ---
>
> Note that the current version of Dr.Web for UNIX Internet Gateways *does not fully* implement the central protection mode: the central protection server cannot manage operation settings of the program components and cannot send scan tasks for the program.

Connecting to an Anti-Virus Network

Dr.Web for UNIX Internet Gateways can be connected to the central protection server of an anti-virus network using the `esconnect` command of the Dr.Web Ctl command-line-based management tool.

> Verification of central protection server requires use of public encryption keys, that is, each server is supplied with a unique public key. By default, the central protection agent Dr.Web ES Agent does not allow connection to the server unless you provide a file containing a public key for authentication of the used server. Such public key file should be obtained from the administrator of your anti-virus network serviced by the server to which you want to connect Dr.Web for UNIX Internet Gateways.

If Dr.Web for UNIX Internet Gateways is connected to the central protection server, you can switch the product into the Mobile mode or switch it back into the Central protection mode. Switching the Mobile mode on or off is accomplished with the help of the **`MobileMode`** configuration parameter of theDr.Web ES Agent component.

Operation can switch to Mobile mode only if it is allowed in the settings on the central protection server.

Disconnecting from an Anti-Virus Network

Dr.Web for UNIX Internet Gateways can be disconnected from the central protection server of an anti-virus network using the `esdisconnect` command of the Dr.Web Ctl command-line-based management tool.

System Requirements

You can use Dr.Web for UNIX Internet Gateways on a computer that meets the following requirements:

Component	**Requirement**
Platform	CPU with the **Intel/AMD** architecture and command system are supported: 32-bit (*IA-32, x86*); 64-bit (*x86_64, x64, amd64*).
Hard disk space	At least 1 GB of free disk space on a volume where the Dr.Web for UNIX Internet Gateways directories are located.
Operating System	**GNU/Linux** (kernel 2.6.37 or newer and library **glibc** 2.13 or newer), **FreeBSD** or **Solaris** for **Intel x86/amd64** platforms. ⚠ For systems operating on 64-bit platforms, support of 32-bit applications *must* be enabled (probably, additional libraries must be installed for this, see below). Operation system must support the **PAM** authentication mechanism. For the correct operation of the Dr.Web Firewall for Linux component, OS kernel must be built with inclusion of the following options: • *CONFIG_NETLINK_DIAG, CONFIG_INET_TCP_DIAG;* • *CONFIG_NF_CONNTRACK_IPV4, CONFIG_NF_CONNTRACK_IPV6, CONFIG_NF_CONNTRACK_EVENTS;* • *CONFIG_NETFILTER_NETLINK_QUEUE, CONFIG_NETFILTER_NETLINK_QUEUE_CT, CONFIG_NETFILTER_XT_MARK.* The set of required options from the specified list can depend on the used OS distribution kit. Tested operating system distributions are listed below.
Other	The following valid network connections: • Valid Internet connection to enable updates for virus databases and Dr.Web for UNIX Internet Gateways components. • When operating in central protection mode, connection to the server on the local network is enough; connection to the Internet is not required.

Tested Operating System Distributions

The product was tested on the following distributions:

- **GNU/Linux**:

Linux distribution name	Versions	Platforms
Astra Linux Special Edition (Smolensk)	1.5	x86_64
CentOS	6.9, 7.4	x86, x86_64
Debian	7.11, 8.10, 9.3	x86_64
Fedora	27	x86, x86_64
Red Hat Enterprise Linux	7.4	x86_64
SUSE Linux Enterprise Server	11 SP4, 12 SP3	x86_64
Ubuntu	14.04, 16.04	x86_64

Other **GNU/Linux** distributions that meet the above-mentioned requirements have not been tested for compatibility with Dr.Web for UNIX Internet Gateways but may be supported. If a compatibility issue occurs, contact technical support on the official website at https://support.drweb.com/request/.

- **FreeBSD**:

Versions	Platforms
10.3, 11.1	x86, x86_64

- **Solaris**:

Versions	Platforms
10 u11	x86, x86_64

⚠ For **FreeBSD** and **Solaris**, the product can be installed only from the universal package.

Additional Packages

- For **CentOS**, **Debian**, **Fedora**, **Red Hat Enterprise Linux**, **Ubuntu** on the platform *x86_64*, the package that enables support for 32-bit applications (**libc6-i386** or **glibc.i686**, depending on OS).

For convenient work with Dr.Web for UNIX Internet Gateways in the command line, you can enable command auto-completion in the used command shell (if disabled).

If you encounter any problem with installation of additional packages and components, refer to manuals for the used distribution of the operating system.

Disclaimer

- SpIDer Gate *can have conflicts* with other firewalls installed in your operating system (such as **Shorewall** and **SuseFirewall2** in the **SUSE Linux Enterprise Server** OS and **FirewallD** in the **Fedora** OS, **CentOS**, **RedHat Enterprise Linux**). The sign of conflict is message about the error of SpIDer Gate with a code x109 or message about the error of Dr.Web Firewall for Linux with a code x102. Methods to resolve a conflict are described in the section “Known Errors” for errors x109 and x102 respectively.
- In case if the used OS includes the version of **NetFilter** less than 1.4.15, SpIDer Gate may operate incorrectly. This problem is related to the internal error of **NetFilter**, and looks like as follows: after disabling SpIDer Gate, the network connections are broken and cannot be re-established. If you face this problem, it is recommended that you upgrade your OS to a version that includes **NetFilter** 1.4.15 or above. The ways to resolve the problem are described in the section “Description of known errors”.

Supported HTTP Proxy Servers

For integration with HTTP proxy server, the installed and configured HTTP proxy server **Squid** 3.0 and newer is required. **Squid** should be built with the support of ICAP (compiled with the `--enable-icap-client` option).

In the mode of Internet barrier, there are no requirements for web servers and HTTP proxy servers.

Internet barrier and transparent proxy modes run only on **GNU/Linux**.

Compatibility with Security Subsystems

By default, Dr.Web for UNIX Internet Gateways does not support **SELinux**. In addition, Dr.Web for UNIX Internet Gateways operates in reduced functionality mode in the **GNU/Linux** systems that use mandatory access models (for example, in systems supplied with the **PARSEC** mandatory access subsystem that appends different privilege levels to users and files).

If installation of Dr.Web for UNIX Internet Gateways is required for systems with **SELinux** (as well as for systems that use mandatory access models). It is necessary to execute additional settings of a security subsystem so that Dr.Web for UNIX Internet Gateways operates in full functionality mode. For details, refer to the section Configuring Security Subsystems.

Licensing

Permissions to use Dr.Web for UNIX Internet Gateways are granted by the *license* purchased from Doctor Web company or from its partners. License parameters determining user rights are set in accordance with the License agreement (see https://license.drweb.com/agreement/), which the user accepts during product installation. The license contains information on the user and the vendor as well as usage parameters of the purchased product, including:

- List of components licensed to the user
- License period
- Other restrictions (for example, number of computers on which the purchased product is allowed for use).

For evaluation purposes users may also activate *demo period*. After successful activation, demo period provides users with full functionality of Dr.Web for UNIX Internet Gateways for the whole activated period.

Each Doctor Web product license has a unique serial number associated with a special file stored on the user computer. This file regulates operation of product components in accordance with the license parameters and is called a *license* key file. Upon activation of a demo period, a special key file, named a *demo* key file, is automatically generated.

If a license or a demo period are not activated on the computer, Dr.Web for UNIX Internet Gateways components are blocked. Moreover, updates for virus databases and components cannot be downloaded from Doctor Web update servers. But you can activate the product by connecting it to the central protection server as a part of the anti-virus network administered by the enterprise or Internet service provider. In this case, operation of Anti-virus and updating are managed by the central protection server.

Note that the current version of Dr.Web for UNIX Internet Gateways *does not* fully implement the central protection mode: the central protection server cannot manage operation settings of the program components.

Installing and Removing the Product

This section describes how to install and uninstall the Dr.Web for UNIX Internet Gateways version 11.0. In this section, you can also find information on how to obtain current updates and a procedure of upgrading to a new version, if the previous version of Dr.Web for UNIX Internet Gateways is already installed on your computer.

Besides, this section describes the procedure of custom installation and uninstallation of the product components (for example, to resolve errors that occurred during the course of theDr.Web for UNIX Internet Gateways operation or to get an installation with a limited function set) and configuration of advanced security subsystems (such as **SELinux**) that could be necessary for installation and operation of the product.

To perform these procedures, root permissions are required (i.e. privileges of the *root* user). To elevate privileges when installing or uninstalling the product, use the **su** command for changing the current user or the **sudo** command to execute the specified command with the privileges of another user.

> ⚠ Compatibility *is not guaranteed* for Dr.Web for UNIX Internet Gateways and anti-virus products of other developers. Due to the fact that installation of two anti-viruses on one machine can lead to *errors in the operation system and loss of important data*, before the installation of Dr.Web for UNIX Internet Gateways, *it is strongly recommended* that you delete anti-virus products of other developers from the computer.
>
> If your computer *already has* other Dr.Web anti-virus product installed from the universal package (`.run`), and you want to install one more Dr.Web anti-virus product (for example, you have Dr.Web for Linux product from the universal package installed, and in addition you want to install Dr.Web for UNIX Internet Gateways), it is necessary to make sure that the version of the installed product *is the same* as the version of the product you want to install. If the product version that you plan on installing is newer that the installed product version, *before* installation, it is necessary to upgrade the installed product to the version of the product you want to install additionally.
>
> For **FreeBSD** and **Solaris**, the product can be installed only from the universal package.

Installing the Product

To install Dr.Web for UNIX Internet Gateways, do one of the following:

1. From the Doctor Web's official website, download the installation file that contains a universal package for UNIX systems. The package includes an installer (due to the fact that the installation program is developed for the command line mode, for its operation in the mode of the graphical desktop, you will need to have a terminal emulator available).
2. Install the product in the form of a set of native packages (to do this, you will need to connect to the corresponding package repository of Doctor Web).

⚠ For **FreeBSD** and **Solaris**, the product can be installed only from the universal package.

Regardless of the selected way to install Dr.Web for UNIX Internet Gateways, after the installation completes, you need to activate the license and to install the received key file. Moreover, you can connect the product to a central protection server. For details, refer to Licensing.

Otherwise, *anti-virus protection remains disabled*.

After you installed the product by any of the mentioned means, you can uninstall or update it if there are fixes for its components available or if a new product versions is released. If required, you can also configure security subsystems of **GNU/Linux** for correct operation of the installed product. If there is a problem with functioning of any individual components, you can perform their custom installation and uninstallation, without uninstalling the entire installed product.

Installing the Universal Package

Dr.Web for UNIX Internet Gateways is distributed as an installation file named `drweb-<`*version*`>-av-igw-<`*OS*`>-<`*platform*`>.run`, where <*OS*> is a type of **UNIX-based** operation system, <*Platform*> is the platform for which the product is intended (`x86` for 32-bit platforms and `amd64` for 64-bit platforms). For example:

```
drweb-11.0.7-av-igw-linux-x86.run
```

Note that the installation file name corresponding to the above-mentioned format is referred to as <*file_name*>`.run` below in this section.

To install Dr.Web for UNIX Internet Gateways components:

1. If you do not have the installation file containing the universal package, download it from the Doctor Web's official website: https://download.drweb.com/.
2. Save the installation file to the hard disk drive of your computer.
3. Allow the archive to be executed, for example, by using the following command:

```
# chmod +x <file_name>.run
```

4. Execute the archive using the following command:

```
# ./<file_name>.run
```

   or use the standard file manager of the graphical shell for both changing the file properties (permissions) and running the file.

   This will run an integrity check of the archive, after which the archived files are unpacked to a temporary directory and an installation program is started. If the user does not have root privileges, the installation program attempts to elevate its privileges asking you for the root password (**sudo** is used). If the attempt fails, the installation process aborts.

If the path to the temporary directory in the file system has not enough free space for the unpacked files, the installation process is aborted and an appropriate message is displayed. In this case, change the value of the `TMPDIR` system environment variable so that it points to a directory with enough free space and repeat the installation. You can also use the `--target` option.

After that the installer for the command-line mode is automatically started (to run it in a graphical desktop environment, you need any terminal emulator).

5. Follow the installer’s instructions.

6. You can also start the installation program in a silent mode by executing the following command:

```
# ./<file_name>.run -- --non-interactive
```

In this case the installation program is started in the silent mode and will operate without a user interface (this means it also will not have any dialogs that are normally displayed in the command-line mode).

Note that

- Using this option means that you *accept* the terms of the Dr.Web License Agreement. The License Agreement’s text is located in the `/opt/drweb.com/share/doc/LICENSE` file. The file extension indicates the language of the License Agreement. If the `LICENSE` file does not have any extension, the Dr.Web License Agreement is written in English. If you *do not accept* the terms of the License Agreement, you must uninstall the product after its installation.
- Administrative (root) privileges are required to start the uninstall program in silent mode. To elevate the privileges, you can use the **su** and **sudo** commands.

If the used **GNU/Linux** distribution features **SELinux**, the installation process can be interrupted by the security subsystem. If such situation occurs, set **SELinux** to the *Permissive* mode. To do this, enter the following command:

```
# setenforce 0
```

And restart the installer. After the installation completes, configure **SELinux** security policies to enable correct operation of the product components.

For details on conventions for <*opt_dir*>, <*etc_dir*>, and <*var_dir*>, refer to the Introduction.

All unpacked installation files are deleted once the installation process completes.

It is recommended that you save file <*file_name*>`.run`, from which the installation was performed, for the possibility of reinstallation of the product or its components without the need to update the product version.

Installing from Command Line

Once you start the program for the command-line-based installation, a message will be displayed inviting you to install the product.

1. To start the installation process, enter *Yes* or *Y* in response to the “Do you want to continue?” question. If you choose not to install the Anti-virus on your computer, enter *No* or *N*. In this case, the installation will be canceled.
2. After that, you need to view the terms of Dr.Web License Agreement which is displayed on the screen. Press ENTER to scroll the text down line by line or SPACEBAR to scroll it down one screenful at a time. Note that options to scroll the License agreement up are not provided.
3. After you read the License agreement text, you are prompted to accept the terms. Type *Yes* or *Y* if you accept the License agreement. If you refuse to accept it, type *No* or *N*. In the latter case, the installer exits.
4. Once you accept the terms of the License Agreement, installation starts automatically. During the procedure, information about the installation process, including the list of installed components, will be displayed on the screen.
5. Once the installation successfully completes, then—in case an automated configuration procedure is available in the product—an interactive setup script for the product is automatically started. After it finishes its operation, an appropriate message will be displayed on the screen, informing you on how to manage the operation of the product.

If an error occurs, a message describing the error is displayed on the screen and then the installer exits. When the installation process fails due to an error, remove the problems that caused this error and start the installation again.

Interactive Setup Script

The Interactive Setup Script allows you to install the product’s license key file that you have

1. If you enter *n* or *no*, execution of the script will end. If you wish to configure such an integration, enter *y* or *yes* as the answer to the question “Do you want to continue?”.
2. If a valid key file is not available on your computer (in the product’s standard directory for keeping the key file), the script will offer you to specify the path to a valid key file. Otherwise (i.e. if a valid key file has been found), this step will be automatically skipped.

   To skip this step, enter *0*. Later, you can install a key file manually. If a valid key file is already available on your computer, specify the path to it and press ENTER. The file will be copied to the product’s standard directory for keeping the key file.
3. After you finish adjusting the settings, press ENTER to end the execution of the script.

Installing from Repository

Dr.Web for UNIX Internet Gateways’s native packages are stored in the Dr.Web official repository at https://repo.drweb.com/. Once you have added the Dr.Web repository to the list of those used by your operating system’s package manager, you can install the product from native packages as you install any other programs from the operating system’s repositories. Required dependencies are automatically resolved.

All the commands mentioned below—the commands used to add repositories, to import digital signature keys, to install and remove packages—must be performed with superuser (**root**) privileges. To elevate the privileges, use the **su** command (to change the current user) or the **sudo** command (to execute the specified command with another user’s privileges).

Note that for the **FreeBSD** and **Solaris** operating systems, the product can be installed only from the universal package.

Debian, Mint, Ubuntu (apt)

The Dr.Web for UNIX Internet Gateways anti-virus engine uses a 32-bit architecture *x86*; in 64-bit systems **Debian**, **Mint**, **Ubuntu** (for platforms *x86-64*, *x64*, *amd64*), a permission could be required for installation of packages for the platform *x86*. It could be obtained via the following command:

```
# dpkg --add-architecture i386
```

1. The repository for these operating systems is digitally signed by Doctor Web. To access the repository, import and add to the package manager storage the digital signature key via execution of the following command:

```
# apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys 10100609
```

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

```
deb http://repo.drweb.com/drweb/debian 11.0 non-free
```

Besides, you can execute items 1 and 2 by downloading from the repository and installing a special DEB package https://repo.drweb.com/drweb-repo11.deb.

3. To install Dr.Web for UNIX Internet Gateways from the repository, use the following commands:

```
# apt-get update
# apt-get install drweb-internet-gateways
```

You can also use alternative package managers (for example, **Synaptic** or **aptitude**) to install the product. Moreover, it is recommended to use alternative managers, such as **aptitude**, to solve a package conflict if it occurs.

ALT Linux, PCLinuxOS (apt-rpm)

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

```
rpm http://repo.drweb.com/drweb/altlinux 11.0/<arch> drweb
```

where <*arch*>—representation of the used packet architecture:

- For the **32-bit** version: `i386`
- For **64-bit** version: `x86_64`

2. To install Dr.Web for UNIX Internet Gateways from the repository, use the following commands:

```
# apt-get update
# apt-get install drweb-internet-gateways
```

You can also use alternative package managers (for example, **Synaptic** or **aptitude**) to install the product.

Mageia, OpenMandriva Lx (urpmi)

1. Connect the repository using the following command:

```
# urpmi.addmedia drweb https://repo.drweb.com/drweb/mandriva/11.0/<arch>/
```

where <*arch*>—representation of the used packet architecture:

- For the **32-bit** version: `i386`
- For **64-bit** version: `x86_64`

3. To install Dr.Web for UNIX Internet Gateways from the repository, use the following command:

```
# urpmi drweb-internet-gateways
```

You can also use alternative package managers (for example, **rpmdrake**) to install the product.

Red Hat Enterprise Linux, Fedora, CentOS (yum, dnf)

1. Add a file `drweb.repo` with the contents described below to the `/etc/yum.repos.d` directory:

```
[drweb]
name=DrWeb - 11.0
baseurl=https://repo.drweb.com/drweb/el5/11.0/$basearch/
gpgcheck=1
enabled=1
gpgkey=https://repo.drweb.com/drweb/drweb.key
```

> If you plan on logging the indicated above contents to a file using such commands as **echo** with redirecting of an output, a symbol `$` must be escaped: `\$`.
>
> Besides, you can execute item 1 by downloading from the repository and installing a special RPM package https://repo.drweb.com/drweb-repo11.rpm.

2. To install Dr.Web for UNIX Internet Gateways from the repository, use the following command:

```
# yum install drweb-internet-gateways
```

In the **Fedora** operating system, starting from version 22, it is recommended that instead of manager **yum** the manager **dnf** is used, for example:

```
# dnf install drweb-internet-gateways
```

You can also use alternative package managers (for example, **PackageKit** or **Yumex**) to install the product.

SUSE Linux (zypper)

1. To add the repository, use the following command:

```
# zypper ar -t YUM 'https://repo.drweb.com/drweb/el5/11.0/$basearch/' drweb
```

2. To install Dr.Web for UNIX Internet Gateways from the repository, use the following commands:

```
# zypper refresh
# zypper install drweb-internet-gateways
```

You can also use alternative package managers (for example, **YaST**) to install the product.

Upgrading the Product

There are two modes for updating Dr.Web for UNIX Internet Gateways:

1. Getting updates of packages and components released in the course of operation of the current product version (usually such updates contain error fixing and minor improvements in component functioning);
2. Upgrading to a newer version. This upgrading option is used if Doctor Web released a new version of the product you use, and it has new features.

Getting Current Upgrades

After installation of the product using any method described in the corresponding section, the package manager automatically connects to the Dr.Web package repository:

- If installation was performed from the universal package (file `.run`), and the system uses DEB packages (for example, such operating systems as **Debian**, **Mint**, **Ubuntu**), there is no package manager in an operating system (**FreeBSD**, **Solaris**), for operation with Dr.Web packages, an individual version of package managers **zypper** is used. It is automatically installed during the product installation.

  To get and install the updated Dr.Web packages with this manager, go to the <*opt_dir*>`/bin` directory (for **GNU/Linux**—`/opt/drweb.com/bin`), and execute the following commands:

  ```
  # ./zypper refresh
  # ./zypper update
  ```

- In all other cases use commands for updating of the package manager used in your OS, for example:
  - For **Red Hat Enterprise Linux** and **CentOS**, use the command **yum**
  - For **Fedora**, use the command **yum** or **dnf**
  - For **SUSE Linux**, use the command **zypper**
  - For **Mageia**, **OpenMandriva Lx**, use the command **urpmi**
  - For **Alt Linux**, **PCLinuxOS**, **Debian**, **Mint**, **Ubuntu**, use the command **apt-get**.

You can also use alternate package managers developed for your operating system. If necessary, refer to the instruction manual for the package manager you use.

If a new product version is released, packages with its components are put into the section of the Dr.Web repository corresponding to the new product version. In this case, an update requires switching of the package manager to a new Dr.Web repository section (refer to Upgrading to a Newer Version).

Upgrading to a Newer Version

Introductory Remarks

Please note that your version of Dr.Web for UNIX Internet Gateways must be upgraded in the same way that was used to install the product:

- If the current version was installed from the repository, an upgrade requires updating program packages from the repository.
- If the current version was installed from the universal package, then to upgrade the product, you need to install another universal package that contains a newer version of the product.

To identify how the product version, which requires an update, has been installed, check whether the directory with the product's executables contains a program removal script `uninst.sh`. If it does contain this script, then the current version was installed from the universal package; otherwise, it was installed from the repository.

Note that for the **FreeBSD** and **Solaris** operating systems, the product can be installed only from the universal package.

If you cannot update the product the way you installed it initially, uninstall your current version of Dr.Web for UNIX Internet Gateways, and then install a new version using any convenient method. Installation and uninstallation procedures for previous Dr.Web for UNIX Internet Gateways versions are the same as installation and uninstallation described in the current manual for version 11.0. For additional information, see Administrator manual for your current version of Dr.Web for UNIX Internet Gateways.

If the current version of the product is operating in the central protection mode, it is recommended that you record the address of the used central protection server. For example, to determine the address to which Dr.Web for UNIX Internet Gateways of the version higher than 6.0.2, you can use the following command:

```
$ drweb-ctl appinfo
```

In the output provided by this command, from the line that looks like:

```
ESAgent; <PID>; RUNNING 1; Connected <address>, on-line
```

save the <*address*> part (which can look like `tcp://`<*IP address*>:<*port*>, for example: `tcp://10.20.30.40:1234`). In addition, it is recommended that you save the server public key file.

In case there are any problems with finding out the parameters of the connection that you are currently using, refer to the Administrator's Manual for the product version that you are currently using and to the administrator of your anti-virus network.

Installing Universal Package for an Upgrade

Install Dr.Web for UNIX Internet Gateways 11.0 from the universal package. If an automatic update of the installed product is impossible, during the installation of the new version, you will get an offer to automatically remove the components of the older version of the product installed on your computer.

If during the update process you need to remove the installed product version, and there are multiple Dr.Web's server products are installed *together* on your server (for example, products for file servers, for mail servers, and for Internet gateways), you need to select *only the packages listed below for removal, in order to keep other server products—that will not be upgraded—fully functional (i.e. to keep the products* for file servers and for mail servers intact):

- `drweb-internet-gateways-doc`
- `drweb-icapd-web`
- `drweb-icapd`

Upgrading from the Repository

Note that you *cannot* upgrade Dr.Web for UNIX Internet Gateways 6.0.2 to version 11.0 from the repository if several Dr.Web's version 6.0.2 server products are installed *together* on your server (for example, if the product for file servers, the product for mail servers, and the product for Internet gateways are installed). In this case, install the new version of Dr.Web for UNIX Internet Gateways on a separate machine.

To upgrade your current version of Dr.Web for UNIX Internet Gateways that was installed from the Doctor Web's repository, do one of the following, depending on the required type of packages:

- **RPM packages (yum, dnf)**.

1. Change the used repository (from the package repository of your current version to the package repository 11.0).

You can find the name of the repository that stores 11.0 packets in the Installing from the Repository section. For details on how to change repositories, refer to help guides of the used operating system distribution.

2. Install the new version using the following command:

```
# yum update
```

or, if the manager **dnf** is used (similar to the **Fedora** OS of the version 22 and earlier):

```
# dnf update
```

If during the update of packages there is an error, uninstall and repeat the installation of the product. If necessary, see sections Uninstalling the Product Installed from the Repository and Installing from the Repository (items for the OS and the package manager that you are using).

- **DEB packages (apt-get)**.

1. Change the used repository (from the package repository of your current version to the package repository 11.0).
2. Update the product using the following commands:

```
# apt-get update
# apt-get dist-upgrade
```

> Please note that for the **Ubuntu** 14.04 (64-bit version) OS, the **apt-get** `dist-upgrade` command may fail. In this case use the **aptitude** package manager (to upgrade the product, issue the **aptitude** `dist-upgrade` command).

Key File Transfer

Regardless of the selected method to upgrade the product, the license key file which you already have (if you have one) will be automatically transferred and installed to the correct location required for the new version of the product.

> If any problem occurs during the automatic installation of the key file, you can install it manually.

If a valid license key file was lost, contact the technical support.

Restoring Connection to the Central Protection Server

If it is possible, your connection to the central protection server will be restored automatically after the upgrade (if the product had been connected to a central protection server before the upgrade). In case the connection has not been automatically restored, then to reestablish the connection of the upgraded Dr.Web for UNIX Internet Gateways to the anti-virus network, execute the following command:

```
$ drweb-ctl esconnect <address> --Key <path to a file of the server public key>
```

In case there are any problems with the connection process, contact the administrator of your anti-virus network.

Removing the Product

Depending on the method that you used to install Dr.Web for UNIX Internet Gateways, you can uninstall the product in one of the following ways:

1. Starting the uninstaller to uninstall the universal package.

2. Uninstalling the packages installed from the Doctor Web’s repository with the help of the system’s package manager.

Uninstalling the Universal Package

Dr.Web for UNIX Internet Gateways that was installed from the universal package for UNIX systems can be uninstalled via the command line (if you are using a graphical desktop environment, you will need a terminal emulator for this option).

> Note that the uninstallation tool uninstalls not only Dr.Web for UNIX Internet Gateways, but also *all the other* Dr.Web products installed on your computer.
>
> If any other Dr.Web products are installed on your computer, besides Dr.Web for UNIX Internet Gateways, then, to delete only Dr.Web for UNIX Internet Gateways, use the custom components installation/removal procedure, instead of running the automatic removal tool.

Uninstalling the Product via the Command Line

The uninstallation tool is started by the `uninst.sh` script, which is located in the <*opt_dir*>`/bin` directory (in **GNU/Linux** this is `/opt/drweb.com/bin`). Uninstallation procedure of Dr.Web for UNIX Internet Gateways is described in section Uninstalling from the Command Line.

You can also start the uninstallation tool in silent mode by executing the command

```
# env DRWEB_NON_INTERACTIVE=yes /opt/drweb.com/bin/uninst.sh
```

In this case, the uninstallation tool is run in silent mode and operates without the user interface (including program dialogs for command-line mode). Note that root privileges are required to start the uninstallation tool in silent mode. To elevate the privileges, you can use the **su** and **sudo** commands.

Uninstalling from Command Line

Once the command-line-based uninstallation program starts, an offer to remove the product is displayed in the command line.

1. To start the uninstalling, enter *Yes* or *Y* in response to the “Do you want to continue?” question. To exit the removal program, type *No* or *N*. In this case, removal will be canceled.
2. An automatic uninstallation procedure will be launched. During this procedure messages about the removal process will be displayed on the screen and logged into an uninstallation log.
3. Once the process is completed, the uninstallation program will automatically terminate.

Uninstalling the Product Installed from the Repository

All commands mentioned below for package uninstallation require superuser (**root**) privileges. To elevate the privileges, use the **su** command (to change the current user) or the **sudo** command (to execute the specified command with other user's privileges).

Debian, Mint, Ubuntu (apt)

To uninstall the root meta-package of Dr.Web for UNIX Internet Gateways, enter the following command:

```
# apt-get remove drweb-internet-gateways
```

To uninstall all the installed Dr.Web packages, enter the following command (in certain operating systems, the '`*`' character must be escaped: '`\*`'):

```
# apt-get remove drweb*
```

To automatically uninstall all packages that are no longer used, enter also the following command:

```
# apt-get autoremove
```

Please, note that uninstallation with the help of the **apt-get** command has the following special aspects:

1. The first mentioned variant of the command uninstalls only the `drweb-internet-gateways` package; any other packages that could have been automatically installed to resolve the dependencies of this package will remain in the system.
2. The second mentioned variant of the command uninstalls all the packages whose name starts with "`drweb`" (the standard name prefix for Dr.Web's products). Note that this command uninstalls all packages with this prefix, not only those of Dr.Web for UNIX Internet Gateways.
3. The third mentioned variant of the command uninstalls all the packages that have been automatically installed to resolve dependencies of other packages and are no longer necessary (e.g., due to the uninstallation of the dependent packages). Note that this command uninstalls all packages that are not used, not only those of Dr.Web for UNIX Internet Gateways.

You can also use alternative managers (for example, **Synaptic** or **aptitude**) to uninstall packages.

ALT Linux, PCLinuxOS (apt-rpm)

In this case, uninstalling of Dr.Web for UNIX Internet Gateways is the same as on **Debian** and **Ubuntu** operating systems (see above).

You can also use alternative managers (for example, **Synaptic** or **aptitude**) to uninstall packages.

Mageia, OpenMandriva Lx (urpme)

To uninstall Dr.Web for UNIX Internet Gateways, enter the following command:

```
# urpme drweb-internet-gateways
```

To automatically uninstall all packages that are no longer used, enter the following command:

```
# urpme --auto-orphans drweb-internet-gateways
```

⚠ Please, note that uninstallation with the help of the **urpme** command has the following special aspects:

1. The first mentioned variant of the command uninstalls only the `drweb-internet-gateways` package; any other packages that could have been automatically installed to resolve the dependencies of this package will remain in the system.
2. The second mentioned variant of the command uninstalls the `drweb-internet-gateways` package as well as all the packages that have been automatically installed to resolve dependencies of other packages and are no longer necessary (e.g., due to the uninstallation of the dependent packages). Note that this command uninstalls all packages that are not used, not only those of Dr.Web for UNIX Internet Gateways.

You can also use alternative managers (for example, **rpmdrake**) to uninstall packages.

Red Hat Enterprise Linux, Fedora, CentOS (yum, dnf)

To uninstall all the installed Dr.Web packages, enter the following command (in certain operating systems, the '`*`' character must be escaped: '`\*`'):

```
# yum remove drweb*
```

In the **Fedora** operating system, starting from version 22, it is recommended that instead of manager **yum** the manager **dnf** is used, for example:

```
# dnf remove drweb*
```

⚠ Please, note that uninstallation with the help of the **yum** (**dnf**) command has the following special aspects:

This variant of the command uninstalls all the packages whose name starts with "`drweb`" (the standard name prefix for Dr.Web’s products). Note that this command uninstalls all packages with this prefix, not only those of Dr.Web for UNIX Internet Gateways.

You can also use alternative managers (for example, **PackageKit** or **Yumex**) to uninstall packages.

SUSE Linux (zypper)

To uninstall Dr.Web for UNIX Internet Gateways, enter the following command:

```
# zypper remove drweb-internet-gateways
```

To uninstall all the installed Dr.Web packages, enter the following command (in certain operating systems, the '`*`' character must be escaped: '`\*`'):

```
# zypper remove drweb*
```

⚠ Please, note that uninstallation with the help of the **zypper** command has the following special aspects:

1. The first mentioned variant of the command uninstalls only the `drweb-internet-gateways` package; any other packages that could have been automatically installed to resolve the dependencies of this package will remain in the system.
2. The second mentioned variant of the command uninstalls all the packages whose name starts with "`drweb`" (the standard name prefix for Dr.Web's products). Note that this command uninstalls all packages with this prefix, not only those of Dr.Web for UNIX Internet Gateways.

You can also use alternative managers (for example, **YaST**) to uninstall packages.

Additional Information

Product Packages and Files

Packages

Dr.Web for UNIX Internet Gateways consists of the following packages:

Package	**Contents**
`drweb-bases`	Files of virus databases and of the anti-virus engine (Dr.Web Virus-Finding Engine)
`drweb-boost`	**Boost libraries**
`drweb-clamd`	Files of the Dr.Web ClamD component.
`drweb-cloudd`	Files of the Dr.Web CloudD component

Package	Contents
drweb-common	The main configuration file—drweb.ini, main libraries, documentation, and a hierarchy of the product's directories. During the installation of this package, a user named drweb and a group named drweb are created.
drweb-configd	Files of the Dr.Web ConfigD and the Dr.Web Ctl components
drweb-dws	Files of a database of web resource categories
drweb-esagent	Files of the Dr.Web ES Agent component.
drweb-filecheck	Files of the Dr.Web File Checker component.
drweb-internet-gateways-doc	PDF documentation for the product
drweb-internet-gateways	The root meta-package of the product
drweb-gated	Files of the SpIDer Gate component.
drweb-firewall	Files of the Dr.Web Firewall for Linux component.
drweb-httpd	Files of the Dr.Web HTTPD component and of the management web interface (a meta-package).
drweb-httpd-bin	Files of the Dr.Web HTTPD component.
drweb-httpd-webconsole	Files of the management web interface.
drweb-icu	Libraries for Unicode support and internationalization
drweb-icapd	Files of the Dr.Web ICAPD component.
drweb-libs	Main libraries of the product *)
drweb-lookupd	Files of the Dr.Web LookupD component
drweb-netcheck	Files of the Dr.Web Network Checker component.
drweb-openssl	**OpenSSL** libraries
drweb-protobuf	**Protobuf** libraries
drweb-se	Files of the Dr.Web Scanning Engine component.
drweb-snmpd	Files of the Dr.Web SNMPD component.
drweb-update	Files of the Dr.Web Updater component

*) The product’s versions intended for 64-bit systems include two packages: drweb-libs and drweb-libs32 that contain libraries for 64-bit and 32-bit components respectively.

In the section Custom Component Installation and Uninstallation there are typical component sets for a custom installation that provide solutions for typical tasks of the product.

Files

After the installation of Dr.Web for UNIX Internet Gateways, its files are located in the /opt, /etc, and /var directories of the file system.

Structure of the used directories

Directory	Contents
<etc_dir>/	The integrated configuration file and the product’s license-key file.
/etc/init.d/	Managing startup script for configuration daemon Dr.Web ConfigD.
<opt_dir>/	Main directory of the product.
bin/	Executable files of all the product’s components (except for Dr.Web Virus-Finding Engine).
include/	Header files of the used libraries.
lib/ lib64/	The libraries used for 32- and 64-bit platforms.
man/	System help files: **man**.
share/	Auxiliary product files.
doc/	Product documentation (readme files and the text of the license agreement).
drweb-bases/	Files of Dr.Web’s virus databases (source files supplied during installation).
scripts/	Auxiliary script files.
<var_dir>/	Auxiliary and temporary files of the product.
bases/	Files of Dr.Web virus databases (the updated version).
cache/	Cache of updates.
drl/	Lists of servers that are used to get updates.

Directory	Contents
`lib/`	Anti-virus engine—Dr.Web Virus-Finding Engine—as a dynamic-link library (`drweb32.dll`) and the settings for working in the central protection mode.
`update/`	Directory for a temporary storage of updates during their download.

For details on conventions used for directories, refer to the Introduction.

Custom Component Installation and Uninstallation

If necessary, you can choose to install or uninstall only certain product components by installing or uninstalling the respective packages. Custom component installation or uninstallation should be performed the same way the product was installed.

To reinstall a component, you can uninstall it first and then install again.

Typical Component Kits for a Custom Installation

If it is required to install the product with the limited functionality, instead of installation of the product’s root meta-package from the repository or from the universal package, you can install only component packages that provide the required functionality. The packages required to resolve dependencies will be automatically installed. The table below displays component sets designed to resolve typical product tasks. In the column **Package for Installation**, there is a list of packages required for installation to obtain the specified component suite.

Custom Component Kit	Package for Installation	Will be Installed
Minimum kit for console scanning	`drweb-filecheck`	• Dr.Web Ctl • Dr.Web ConfigD • Dr.Web Scanning Engine • Dr.Web File Checker • Dr.Web Updater • Virus database
Suite for the emulation **ClamAV** (**clamd**)	`drweb-clamd`	• Dr.Web Ctl • Dr.Web ConfigD • Dr.Web Scanning Engine • Dr.Web File Checker • Dr.Web Network Checker • Dr.Web Updater • Dr.Web ClamD • Virus database

Custom Component Kit	Package for Installation	Will be Installed
Suite for checking the access to websites using a proxy server via the ICAP protocol (without the anti-virus traffic scanning)	`drweb-icapd`	• Dr.Web Ctl • Dr.Web ConfigD • Dr.Web ICAPD • Dr.Web Updater • Database of web resource categories

Suite for checking the access to websites using a proxy server via the ICAP protocol (with the anti-virus traffic scanning).
Note: The package `drweb-se` *could be skipped for installation if the anti-virus scanning is performed on another server, which receives data for scanning via Dr.Web Network Checker.*
*Mark *) labels components that will not be installed if the package* `drweb-se` *is not installed.*
	`drweb-icapd`
	`drweb-netcheck`
	`drweb-se *`
		• Dr.Web Ctl
		• Dr.Web ConfigD
		• Dr.Web ICAPD
		• Dr.Web Network Checker
		• Dr.Web Scanning Engine *)
		• Dr.Web Updater
		• Virus database *)
		• Database of web resource categories

Suite for a local scanning of HTTP connections
Note: If the anti-virus scanning of connections is not required, packages `drweb-netcheck` *and* `drweb-se` *are not required for installation. The package* `drweb-se` *could be skipped for installation if the anti-virus scanning is performed on another server, which receives data for scanning via Dr.Web Network Checker. The package* `drweb-dws` *could be skipped for installation if there is no requirement for the URL to be included in the categories of unwanted web resources.*
*Mark *) labels components that will not be installed if the package* `drweb-se` *is not installed. Mark **) labels components that will not be installed if the package*`drweb-dws` *is not installed. The component Dr.Web Updater (marked with ***)*
	`drweb-gated`
	`drweb-firewall`
	`drweb-netcheck *`
	`drweb-se *`
	`drweb-dws **`
		• Dr.Web Ctl
		• Dr.Web ConfigD
		• SpIDer Gate
		• Dr.Web Firewall for Linux
		• Dr.Web Network Checker
		• Dr.Web Scanning Engine *)
		• Dr.Web Updater ***)
		• Virus database *)
		• Database of web resource categories **)

Custom Component Kit	Package for Installation	Will be Installed
will be installed only if virus databases or database of web resource categories are installed.		

1. Installation and Uninstallation of Product Components Installed from Repository

If your product is installed from repository, for custom component installation or uninstallation use the respective command of the package manager, used in your OS. For example:

1. To uninstall Dr.Web ClamD (package `drweb-clamd`) from the product installed on OS **CentOS**, use the command:

```
# yum remove drweb-clamd
```

2. To additionally install Dr.Web ClamD (package `drweb-clamd`) to the product installed on OS **Ubuntu Linux**, use the command:

```
# apt-get install drweb-clamd
```

If necessary, use a help file of the package manager used in your OS.

⚠ The Dr.Web for UNIX Internet Gateways anti-virus engine uses a 32-bit architecture *x86*; in 64-bit systems **Debian**, **Mint**, **Ubuntu** (for platforms *x86-64*, *x64*, *amd64*), a permission could be required for installation of packages for the platform *x86*. It could be obtained via the following command:

```
# dpkg --add-architecture i386
```

2. Installation and Uninstallation of Product Components Installed from the Universal Package

If the product is installed from the universal package and you want to additionally install or reinstall a package of a component, you will need an installation file (with the `.run` extension), from which the product was installed. In case you did not save this file, download it from the Doctor Web’s official website.

Unpacking the Installation File

When you launch the .run file, you can also specify the following command-line parameters:

`--noexec`—unpack the product’s installation files instead of starting the installation process. The files will be placed to the directory that is specified in the `TMPDIR` environment variable (usually, `/tmp`).

`--keep`—do not delete the product installation files and the installation log automatically after the installation completes.

`--target` <*directory*>—unpack the product’s installation files to the specified <*directory*>.

For a full list of command-line parameters that can be specified for the launching of the .run file, enter the following command:

```
$ ./<file_name>.run --help
```

For a custom installation, you need to use the unpacked installation files. If there is no directory containing these files, you should first unpack them. To do that, enter the following command:

```
$ ./<file_name>.run --noexec --target <directory>
```

After the command is executed, a nested directory named <*file_name*> will appear in the directory <*directory*>.

Custom Installation of the Components

Installation RUN file contains packages of all components of Dr.Web for UNIX Internet Gateways (in the RPM format) and supporting files. Package files of each component have the following structure:

```
<component_name>_<version>~linux_<platform>.rpm
```

where <*version*> is a string that contains the version and time of the product’s release, and <*platfrom*> is a platform for which the product is intended. Names of all the packages containing the components of Dr.Web for UNIX Internet Gateways start with the "drweb" prefix.

Package manager is enabled for the installation of packages to the installation kit. For the custom installation, you should use a service script `installpkg.sh`. To do that, first, you need to unpack the contents of the installation package to a directory.

To install packages, superuser permissions are required (i.e. privileges of the *root* user). To elevate your privileges, use the **su** command for changing the current user or the **sudo** command to execute the specified command with the privileges of another user.

To start installation or reinstallation of a component package, go to the directory which contains the unpacked installation kit, and execute the following command via the console (or via a console emulator—terminal for the graphical mode):

```
# ./scripts/installpkg.sh <package_name>
```

For example:

```
# ./scripts/installpkg.sh drweb-clamd
```

If it is necessary to start the full product installation, launch the automatic installation script. To do that, use the following command:

```
$ ./install.sh
```

Besides that, you can install all product packages (to install the missing or accidentally deleted components as well) by launching the installation of the root meta-package of the product:

```
# ./scripts/installpkg.sh drweb-internet-gateways
```

Custom Uninstallation of the Components

For the custom uninstallation of a component, use the appropriate uninstallation command of the package manager of your OS if your OS uses the RPM format of packages:

- In **Red Hat Enterprise Linux** and **CentOS**, use the command **yum** `remove` *<package_name>*
- In **Fedora**, use the command **yum** `remove` *<package_name>* or **dnf** `remove` *<package_name>*
- In **SUSE Linux**, use the command **zypper** `remove` *<package_name>*
- In **Mageia**, **OpenMandriva Lx**, use the command **urpme** *<package_name>*
- In **Alt Linux** and **PCLinuxOS**, use the command **apt-get** `remove` *<package_name>*.

For example (for **Red Hat Enterprise Linux**):

```
# yum remove drweb-clamd
```

If your OS uses DEB packages (also if you use **MSVS** 3.0 OS), or if there is no package manager in your system (**FreeBSD**, **Solaris**), for the custom uninstallation, you should use the package manager **zypper**, which is automatically installed within the product installation. To do that, go to the directory *<opt_dir>*`/bin` (for **GNU/Linux**—`/opt/drweb.com/bin`) and execute the following command:

```
# ./zypper remove <package_name>
```

For example:

```
# ./zypper remove drweb-clamd
```

If it is necessary to start the full product uninstalling, launch the automatic uninstallation script. To do that, use the following command:

```
# ./uninst.sh
```

To reinstall a component, you can uninstall it first and then install by launching the custom or full installation from the installation kit.

Configuring Security Subsystems

Presence of the **SELinux** enhanced security subsystem in the OS (as well as the use of mandatory access control systems, such as **PARSEC** (as opposed to the classical discretionary model used by UNIX) causes problems in the work of Dr.Web for UNIX Internet Gateways when its default settings are used. To ensure correct operation of Dr.Web for UNIX Internet Gateways in this case, it is necessary to make additional changes to the settings of the security subsystem and/or to the settings of Dr.Web for UNIX Internet Gateways.

This section discusses the settings that ensure correct operation of Dr.Web for UNIX Internet Gateways in the following cases:

- Configuring **SELinux** Security Policies.
- Setting up the permissions of the **PARSEC** mandatory access control system (the **Astra Linux** OS)

⚠ Configuring the permissions of the **PARSEC** mandatory access control system for Dr.Web for UNIX Internet Gateways will allow the components of Dr.Web for UNIX Internet Gateways to bypass the restrictions of the set security policies and to get access to the files that belong to different privilege levels.

Note that even if you have not configured the permissions of the **PARSEC** mandatory access control system for Dr.Web for UNIX Internet Gateways, you still will be able to launch file scanning directly from the command line. To do this, use the **drweb-ctl**command in the autonomous mode, by specifying the `--Autonomous` option in the command call. When scanning is launched this way, it is possible to scan only those files that can be accessed with the privileges not exceeding those of the user who launched the scanning. This mode has several features:

- To launch the autonomous copy you need the valid key file, the work with central protection server is not supported (it is possible to install the key file, exported from central protection server). Herewith, even if Dr.Web for UNIX Internet Gateways is connected to central protection server, the autonomous copy do net send to it any notifications on threats, detected during the work in autonomous mode.

- All additional components that support the functioning of the autonomous copy, will be launched under the current user and will work with specially generated configuration file.
- All the used temporary files and UNIX sockets are created only in the directory with an unique name, which is created when the autonomous copy is launched. The unique temporary directory is created in the system directory for temporary files (path to this directory is available in the `TMPDIR` environment variable).
- All the required paths to virus databases, anti-virus engine and executable files used during scanning are defined by default or retrieved from the special environment variables.
- The number of the autonomous copies working simultaneously is not limited.
- When the autonomous copy is terminated, the set of supporting components also terminates.

Configuring SELinux Security Policies

If your **GNU/Linux** distribution includes **SELinux** (*Security-Enhanced Linux*), you may need to configure **SELinux's security policies to get the servicing components of the Dr.Web product (such as the** scanning engine) to operate correctly after the installation of the Dr.Web product.

1. Universal Package Installation Issues

If **SELinux** is enabled, installation from the installation file (`.run`) can fail because an attempt to create the *drweb* user, under which Dr.Web for UNIX Internet Gateways components operate, can be blocked.

In case of failure, check the **SELinux** operation mode with the **getenforce** command. The command outputs one of the following:

- `Permissive`—protection is active but a permissive strategy is used: actions that violate the security policy are not denied but information on the actions is logged.
- `Enforced`—protection is active and restrictive strategy is used: actions that violate security policies are blocked and information on the actions is logged.
- `Disabled`—**SELinux** is installed but not active.

If **SELinux** is operating in *Enforced* mode, change it to *Permissive*. For that purpose, use the following command:

```
# setenforce 0
```

This command (until the next reboot) enables *Permissive* mode for **SELinux**.

> Note that regardless of the operation mode enabled with the **setenforce** command, after the restart of the operating system, **SELinux** returns to the safe operation mode specified in the its settings (file with **SELinux** settings usually resides in the `/etc/selinux` directory).

After the successful product installation, enable *Enforced* mode again before starting the product. For that, use the following command:

```
# setenforce 1
```

2. Problems with the Product’s Operation

In some cases, when **SELinux** is enabled, some Dr.Web for UNIX Internet Gateways’s components (for example, **drweb-se** and **drweb-filecheck**) cannot start. If so, object scanning and file system monitoring become unavailable. In this case errors *119* and *120* can appear in the system log **syslog** (normally located in the `/var/log/` directory).

> Messages on *119* and *120* errors can also indicate an attempt to start Dr.Web for UNIX Internet Gateways on 64-bit version of the operating system if the 32-bit application support library is missing (see System Requirements).

When the **SELinux** security system denies access, such an event is logged. In general, when the **audit** daemon is used on the system, the log of the audit is stored in the `/var/log/audit/audit.log` file. Otherwise, messages about blocked operations are saved to the general log file (`/var/log/messages` or `/var/log/syslog`).

If the scanning components of the product do not function because they are blocked by **SELinux**, you will need to compile special *security policies* for them.

> Note that certain **Linux** distributions do not feature the utilities mentioned below. If so, you may need to install additional packages with the utilities.

Configuring SELinux Security Policies:

1. Create a new file with the **SELinux** policy source code (a `.te` file). This file defines restrictions related to the described policy module. The policy’s source code can be created in one of the following ways:

    1) Using the **audit2allow** utility, which is the simplest method. The utility generates permissive rules from messages on access denial in system log files. You can set to search messages automatically or specify a path to the log file manually.

       Note that you can use this method only if Dr.Web for UNIX Internet Gateways’s components have violated **SELinux** security policies and these events are registered in the audit log file. If not, wait for such an incident to occur or force-create permissive policies by using the **policygentool** utility (see below).

The **audit2allow** utility resides either in the `policycoreutils-python` package or in the `policycoreutils-devel` package (for **RedHat Enterprise Linux**, **CentOS**, **Fedora** operating systems, depending on the version) or in the `python-sepolgen` package (for **Debian** and **Ubuntu** operating systems).

Example of using **audit2allow**:

```
# grep drweb-se.real /var/log/audit/audit.log | audit2allow -M drweb-se
```

In the given example, the **audit2allow** utility performs a search in the `/var/log/audit/audit.log` file to find access denial messages for the **drweb-se** component.

The following two files are created: policy source file `drweb-se.te` and the `drweb-se.pp` policy module ready to install.

If no security violation incidents are found in the system audit log, the utility returns an error message.

In most cases, you do not need to modify the policy file created by the **audit2allow** utility. Thus, it is recommended to go to step 4 for installation of the `drweb-se.pp` policy module. Note that the **audit2allow** utility outputs invocation of the **semodule** command. By copying the output to the command line and executing it, you complete step 4. Go to step 2 only if you want to modify security policies which were automatically generated for Dr.Web for UNIX Internet Gateways components.

2) Using the **policygentool** utility. For that purpose, specify the name of the component that you want to be treated differently and the full path to its executable file.

Note that the **policygentool** utility, included in the `selinux-policy` package for **RedHat Enterprise Linux** and **CentOS Linux** OS, may not function correctly. If so, use the **audit2allow** utility.

Example of policy creation using **policygentool**:

- For the **drweb-se** component:

```
# policygentool drweb-se /opt/drweb.com/bin/drweb-se.real
```

- For the **drweb-filecheck** component:

```
# policygentool drweb-filecheck /opt/drweb.com/bin/drweb-filecheck.real
```

You will be prompted to specify several general properties for created the domain. After that, three files that determine the policy will be created (for each of the components):

<module_name>.te, *<module_name>.fc* and *<module_name>.if*.

2. If required, edit the generated policy source file *<module_name>.te* and then use the **checkmodule** utility to create a binary representation (a `.mod` file) of this source file of the local policy.

> ! Note that to ensure successful execution of the command, the `checkpolicy` package must be installed in the system.

Example usage

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

3. Create a policy module for installation (a `.pp` file) with the help of the **semodule_package** utility.

   Example:

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

4. To install the created policy module, use the **semodule** utility.

   Example:

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

For details on **SELinux** operation and configuration, refer to documentation for the used **Linux distribution.**

Configuring the Permissions of PARSEC (Astra Linux)

In operating systems equipped with the **PARSEC** security subsystem (mandate access control system), due to the variation in privilege levels required to access different files, the if the user works at any privilege level other than the zeroth, the command-line-based management tool Dr.Web Ctl for Dr.Web for UNIX Internet Gateways cannot interact with the Dr.Web ConfigD configuration daemon, if they work at a different privilege level; access to the consolidated quarantine may also become unavailable.

To configure permissions, superuser permissions are required (i.e. privileges of the *root* user). To elevate your privileges, use the **su** command for changing the current user or the **sudo** command to execute the specified command with the privileges of another user.

Configuring the Correct Launch of Dr.Web for UNIX Internet Gateways at Any Privilege Level

In order for all the components of Dr.Web for UNIX Internet Gateways to be able to correctly interact with each other when they are launched with different privilege levels, modify the script that launches the Dr.Web ConfigD configuration daemon (**drweb-configd**):

1. Log into the system using the privilege level zero
2. Open the `/etc/init.d/drweb-configd` script file in any text editor (root privileges are required).
3. In this file find the definition of the `start_daemon` function and replace the line:

```
"$DAEMON" -d -p "$PIDFILE" >/dev/null 2>&1
```

with the line:

```
execaps -c 0x100 -- "$DAEMON" -d -p "$PIDFILE" >/dev/null 2>&1
```

4. In some OSes, (for example, **Astra Linux SE** 1.3), an additional indication of component launch dependence from the **PARSEC** subsystem could be required. In this case, it is also necessary to modify a string in the file:

```
# Required-Start: $local_fs $network
```

Change this string in the following way:

```
# Required-Start: $local_fs $network parsec
```

5. Save the file and reboot the operating system.

Getting Started

1. To start using the installed Dr.Web for UNIX Internet Gateways, you need to activate it by obtaining and installing a key file.
2. Further scanning of the operability of the product is recommended.
3. Integrate Dr.Web for UNIX Internet Gateways with an HTTP proxy server that you are using (please see the provided instructions regarding the integration with a **Squid** proxy server).
4. To protect a local web server from threats coming from the external network, change the settings of the SpIDer Gate monitor.
5. Check what components are running and enable additional components, which are disabled by default, if you need them for the protection of your server (for example, the Dr.Web ClamD or Dr.Web SNMPD component, depending on the distribution). Note that you may also need to perform other actions apart from enabling the additional components, for example, you may need to adjust their default configuration. To view the list of installed and running components and their settings, use one of the following:
   - The command-line-based management tool—Dr.Web Ctl (use the **drweb-ctl** `appinfo`, **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands).
   - The management web interface of Dr.Web for UNIX Internet Gateways (by default, you can access it via a web browser at `https://127.0.0.1:4443/`).

Registration and Activation of the Product

Purchasing and Registering License

After a license is purchased, updates to product components and virus databases are regularly downloaded from Doctor Web update servers. Moreover, if the customer encountered any issue when installing or using the purchased product, they can take advantage of technical support service provided by Doctor Web or its partners.

You can purchase any Dr.Web product as well as obtain a product serial number either on the online store or from our partners. For details on license periods and license types, visit the Doctor Web official website at https://www.drweb.com/.

License registration is required to prove that you are a legal user of Dr.Web for UNIX Internet Gateways and to activate the functions of the anti-virus, including the regular updates of virus databases. It is recommended that you register the product and activate the license once the installation completes. A purchased license can be activated on the Doctor Web’s official website at https://products.drweb.com/register/.

During activation, it is required to enter the serial number of the purchased license. The serial number is supplied with the product or via email when purchasing or renewing the license online.

If you have used the product in the past, you may be eligible for a 150-day extension to your new license. To enable the bonus, enter your registered serial number or provide the license key file.

If you have several licenses for using Dr.Web for UNIX Internet Gateways on several servers, but choose to use the product only on one server, you can specify this and, hence, license validity period will be automatically extended.

Obtaining Demo License

A demo period for your copy of the product can be obtained on the Doctor Web official website at https://download.drweb.com/demoreq/biz/. After you select the product and fill the registration form, you will receive an email with a serial number or key file for Dr.Web for UNIX Internet Gateways activation.

Another demo period for the same computer can be obtained after a certain time period.

You can use the license command of the Dr.Web Ctl (**drweb-ctl**) command-line tool, which allows to get a demo key file or a licensed key file for a serial number of a registered license automatically.

Key File Installation

The key file is a special file stored on the local computer. It corresponds to the purchased license or activated demo period for Dr.Web for UNIX Internet Gateways. The file contains information on the provided license or demo period and regulates usage rights in accordance with it.

During Dr.Web for UNIX Internet Gateways operation, the key file must be located in the default <*etc_dir*> directory (`/etc/opt/drweb.com` for **Linux**) under the name `drweb32.key`.

Components of the product regularly check whether the key file is available and valid. The key file is digitally signed to prevent its editing. So, the edited key file becomes invalid. It is recommended that you do not open your key file in text editors in order to avoid its accidental invalidation.

If no valid key file (license or demo) is found, or if the license is expired, operation of the anti-virus components is blocked until a valid key file is installed.

It is recommended that you keep the license key file until it expires, and use it to reinstall the product or install it on a different computer. In this case, you must use the same product serial number and customer data that you provided during the registration.

Dr.Web key files are usually packed in a ZIP archive if sent via email. The archive with a key file is named `agent.zip` (note that if there is *several* archives in an email message, you should use only `agent.zip`). Before installing a key file, unpack it using any suitable tool and extract a key file to any directory (for example, to your home directory or to a USB flash drive).

If you have a key file corresponding to the valid license for the product (for example, if you obtained the key file by email or if you want to use Dr.Web for UNIX Internet Gateways on another server), you can activate the product by specifying the path to the key file. For that, do the following:

1. Unpack the key file if archived
2. Do one of the following:
   - Copy the key file to the <*etc_dir*> directory and rename the file to `drweb32.key` if necessary.
   - In the Dr.Web for UNIX Internet Gateways configuration file specify the key file path as the **`KeyPath`** parameter value.
3. Uninstall Dr.Web for UNIX Internet Gateways by entering the following command:

   ```
   # drweb-ctl reload
   ```

   to apply all changes.

You can also use the following command:

```
# drweb-ctl cfset Root.KeyPath <path to the key file>
```

In this case, restart of Dr.Web for UNIX Internet Gateways is not required. The key file will not be copied to the <*etc_dir*> directory and will remain in its original location.

For details on conventions for <*opt_dir*>, <*etc_dir*>, and <*var_dir*>, refer to the Introduction.

If the key file is not copied to the <*etc_dir*> directory, the user becomes responsible for ensuring that the file is protected from corruption or deletion. This installation method is not recommended as the key file can be accidentally deleted from the system (for example, if the directory, where the key file resides, is periodically cleaned up). Remember that if a key file is lost, you can request the support for a new one, but the number of such requests is limited.

Subsequent Registration

If a key file is lost but the existing license is not expired, you must register again by providing the personal data you specified during the previous registration. You may use a different email address. In this case, the license key file will be sent to the newly specified address.

The number of times you can request a key file is limited. One serial number can be registered *no more than 25 times*. If requests in excess of that number are sent, no key file will be delivered. To receive a lost key file, contact Doctor Web technical support, describe your problem in detail, and state personal data you entered upon serial number registration. The license key file will be sent by email.

After the key file is sent to you by email, you need to install it manually.

Testing the Operation of the Product

The *EICAR (European Institute for Computer Anti-Virus Research*) Test helps testing performance of anti-virus programs that detect viruses using signatures. This test was designed specially so that users could test reaction of newly-installed anti-virus tools to detection of viruses without compromising security of their computers.

Although the *EICAR* test is not actually a virus, it is treated by the majority of anti-viruses as if it were a virus. On detection of this “virus”, Dr.Web anti-virus products report the following: **EICAR Test File (NOT a Virus!)**. Other anti-virus tools alert users in a similar way. The **EICAR** test file is a 68-byte COM-file for **MS DOS**/**MS Windows** that outputs the following line on the console when executed:

```
EICAR-STANDARD-ANTIVIRUS-TEST-FILE!
```

The EICAR test contains the following character string only:

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

To create your own test file with the “virus”, you may create a new file with the line mentioned above.

If Dr.Web for UNIX Internet Gateways operates correctly, the test file is detected during a file system scan regardless of the scan type, and the user is notified on the detected threat: **EICAR Test File (NOT a Virus!)**.

An example of a command that checks operation of the program by means of **EICAR** test from the command line:

```
$ tail <opt_dir>/share/doc/drweb-common/readme.eicar | grep X5O > testfile &&
drweb-ctl rawscan testfile && rm testfile
```

From the file <*opt_dir*>`/share/doc/drweb-common/readme.eicar` (supplied with the product), this command retrieves a string that represent a body of the **EICAR** test file, then writes it to the file `testfile` located in the current catalog, checks the received file, and removes the created file.

> The above-mentioned test requires write access to the current catalog. In addition, make sure that it does not contain a file named `testfile` (if necessary, change the file name in the command).
>
> For details on conventions for <*opt_dir*>, <*etc_dir*>, and <*var_dir*>, refer to the Introduction.

If a test virus is detected, the following message is displayed:

```
<path to the current directory>/testfile - infected with EICAR Test File (NOT a
Virus!)
```

If an error occurs during the test, refer to the description of known errors (see Appendix F. Known Errors).

> If SpIDer Guard is enabled, a malicious file can be immediately removed or quarantined (depending on the configuration of the component). In this case, the command **rm** will inform that the file is missing, which implies that the monitor operates in normal mode.

Integration with Squid Proxy Server

1) Configuring Dr.Web ICAPD

To integrate Dr.Web ICAPD with a **Squid** HTTP proxy server, you will need to review the current values of parameters in the Dr.Web ICAPD's settings section (the `[ICAPD]` section) and change them if necessary:

- In the **`ListenAddress`** parameter, specify the address of the network socket (<*IP address*>:<*port*>) which will be listened to by Dr.Web ICAPD waiting for connections from an HTTP proxy server (by default, the `127.0.0.1:1344` socket is used).
- In **`Block*`** parameters, enable or disable blocking of the respective website categories and threat types by Dr.Web ICAPD.
- If required, you can use the **`WhiteList`** and **`BlackList`** parameters to define the websites that must not be blocked and the websites that must be blocked. Note that the **`BlackList`** parameter has higher priority than the **`WhiteList`** parameter, that is, if the same website is included in the values of both parameters, access to this website will be blocked.
- To configure access to websites in a more fine-grained way (on the basis of various conditions), you can also edit the scanning rules.

> The default values of the **`UsePreview`**, **`Use204`** and **`AllowEarlyResponse`** parameters in the Dr.Web ICAPD's section of the settings allow the component to use the corresponding features of the Internet Content Adaptation Protocol (ICAP) (i.e. allow it to use the *ICAP preview* mode, to return the 204 status code not only in the *ICAP preview* mode, and to start sending an "early" response before the entire request has been received from the proxy server). It is recommended that you do not change the default values if no problems with HTTP request processing occur.

After all settings are adjusted, restart Dr.Web for UNIX Internet Gateways (use the command **drweb-ctl** `reload`). You can also restart the configuration daemon Dr.Web ConfigD (use the **service** `drweb-configd restart` command).

2) Configuring Squid

To enable interaction between **Squid** and Dr.Web ICAPD, edit the `squid.conf` configuration file (usually located in `/etc/squid3/`) to allow using ICAP. To configure **Squid**, set the following parameters:

1. Enabling **Squid** to use the ICAP.
2. Registering Dr.Web ICAPD as the ICAP service used by **Squid**.
3. Enabling the use of the *ICAP preview* mode (optionally).
4. Allowing to transfer clients' data (i.e. the IP address and the user name of a user who has passed authentication at the proxy server) to use it inside the rules of Dr.Web ICAPD (optionally).
5. Enabling the support of constant connections between Dr.Web ICAPD and **Squid** (optional; using constant connections is not obligatory, but this increases the performance of the simultaneous use of **Squid** + Dr.Web ICAPD).

When configuring **Squid**, remember the following:

- To make **Squid** check HTTP requests (*REQMOD*) and HTTP responses (*RESPMOD*) via the ICAP, add two ICAP services of the corresponding types.
- To make **Squid** use Dr.Web ICAPD as an ICAP service, the address and port specified in `icap_service` should match the address and port specified in the **`ListenAddress`** parameter in the Dr.Web ICAPD's settings.
- Dr.Web ICAPD will not work with **Squid**, if the **`icap_preview_size`** parameter value is not `0`.
- **Squid** forms the "Client's IP address" and "Username" values automatically and redirects them to Dr.Web ICAPD as headers of its ICAP request. The correctness and availability of this data is not guaranteed. Dr.Web ICAPD assumes that the user name and the user's IP address are transferred by the proxy server in the `X-Client-Username` and `X-Client-IP` headers; and assumes that only those value encoding methods are used that are defined by default in **Squid**'s settings. For this reason, when configuring **Squid**, it is recommended that you do not change the parameter values that influence the method of transferring this data (like `icap_client_username_encode` and `icap_client_username_header`).

The used **Squid** version should be built with the support of ICAP (that is, compiled with the `--enable-icap-client` option). Otherwise, it is not possible to establish connection between **Squid** and Dr.Web ICAPD.

The list of parameters that can be configured depends on the version of the **Squid** server that you are using (below you can find the description of configuring the following **Squid** versions: 3.2 (and later), 3.1, and 3.0). If the strings mentioned bellow are already in the configuration file, their values should be changed to the specified ones. If the mentioned parameters are already in the file, but they are commented out, uncomment them. If there are no required parameters in the **Squid** configuration file, add them to the file, for example, to the end.

Only the `#1` and `#2` steps are obligatory for configuring interaction between Dr.Web ICAPD and **Squid**. If other settings, out of those which are mentioned below, are not required, do not add them to the **Squid** configuration file.

For Squid 3.2 and later versions

```
#1
icap_enable on

#2
icap_service i_req reqmod_precache bypass=0 icap://127.0.0.1:1344/reqmod
icap_service i_res respmod_precache bypass=0 icap://127.0.0.1:1344/respmod

adaptation_access i_req allow all
adaptation_access i_res allow all

#3
icap_preview_enable on
icap_preview_size 0

#4 (In Squid 3.2, the icap_send_client_ip and
#icap_send_client_username parameters have been renamed)
adaptation_send_client_ip on
adaptation_send_username on

#5
icap_persistent_connections on
```

For Squid 3.1

```
#1
icap_enable on

#2 (In Squid 3.1, the format used to configure a service has been changed
#and the icap_access parameter has been renamed)
icap_service i_req reqmod_precache bypass=0 icap://127.0.0.1:1344/reqmod
icap_service i_res respmod_precache bypass=0 icap://127.0.0.1:1344/respmod

adaptation_access i_req allow all
adaptation_access i_res allow all

#3
icap_preview_enable on
icap_preview_size 0

#4
icap_send_client_ip on
icap_send_client_username on

#5
icap_persistent_connections on
```

For Squid 3.0

```
#1
icap_enable on

#2
icap_service i_req reqmod_precache 0 icap://127.0.0.1:1344/reqmod
icap_service i_res respmod_precache 0 icap://127.0.0.1:1344/respmod

icap_class icapd_class_req i_req
icap_class icapd_class_resp i_res

icap_access icapd_class_req allow all
icap_access icapd_class_resp allow all

#3
icap_preview_enable on
icap_preview_size 0

#4
icap_send_client_ip on
icap_send_client_username on

#5
icap_persistent_connections on
```

After changing **Squid**'s settings, restart it.

Additional Information

If necessary, you can limit the size of data that **Squid** will send for scanning via the ICAP protocol. For this purpose, the configuration file must be added with a condition that must satisfy (or not satisfy) the content of the header `Content-Length`, for example:

```
acl <name> rep_header Content-Length ^[0-9]{7,}$
```

(condition <*name*> will be true, if the header `Content-Length` in the server response contains a number larger than 999999).

Then the added condition should be used to `allow` or `deny` scanning of the server response via the ICAP protocol (the word `all` must be replaced in the connection parameters of **Squid** to the external ICAP server with the condition name <*name*>). Due to the fact that the example indicated above could be true when the header `Content-Length` has a number larger that 999999, we will use it to *deny* the scanning of responses, whose condition <*name*> is true:

```
#Squid 3.1 and later versions
adaptation_access i_res deny <name>

#Squid 3.0 and earlier versions
icap_access icapd_class_resp deny <name>
```

Presence of the header `Content-Length` is not guaranteed in the webserver response. If it is not available, the indicated method for size restriction of data, that is sent by **Squid** for scanning to the ICAP server, will not work.

After changing **Squid**'s settings, restart it.

For details on configuration of **Squid** in a more fine-grained way to restrict scanning of web traffic, see documentation of **Squid**. See, for example, http://www.squid-cache.org/Doc/.

Protecting a Local Web Server

This option is available only in the product distributions for **GNU/Linux** OSes.

To protect a web server that is running on the same host on which Dr.Web for UNIX Internet Gateways is installed, you need to configure the Dr.Web Firewall for Linux component in such a way that traffic coming to the web server will be checked by the SpIDer Gate monitor.

To configure protection for a web sever, change the value of the **`InputDivert`** parameter, which is located in the configuration file, in the section with the settings of Dr.Web Firewall for Linux (the `[LinuxFirewall]` section):

```
InputDivert = Auto(interface:<network interface> protected:<list of ports>)
```

where

- <*network interface*>—is the name of a network interface (`eth0`, `wlan` etc.) through which inbound connections that must be checked access the web server.
- <*list of ports*>—a list of ports serviced by the web server (80, 8080 etc.).

To view and to change the settings of Dr.Web Firewall for Linux and SpIDer Gate you can use the following means:

- The command-line-based management tool—Dr.Web Ctl (use the **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands).

- The management web interface of Dr.Web for UNIX Internet Gateways (by default, you can access it via a web browser at https://127.0.0.1:4443/).

For example, the following command:

```
# drweb-ctl cfset LinuxFirewall.InputDivert 'Auto(interface:eth0
protected:80,8080)'
```

will configure the Dr.Web Firewall for Linux in such a way that the data received via the eth0 network interface and directed to the local ports 80 or 8080 will be checked by the SpIDer Gate monitor.

Brief Instructions

How to Connect Dr.Web for UNIX Internet Gateways to Squid

Follow the instructions provided in the Integration with Squid Proxy Server section.

How to Protect a Web Server

Follow the instructions provided in the Protecting a Local Web Server section.

How to Restart Dr.Web for UNIX Internet Gateways

To restart the product when it is already running, you can also use the script that controls the Dr.Web ConfigD configuration daemon. Startup, stop, or restart of the daemon cause respectively the startup, stop or restart of Dr.Web for UNIX Internet Gateways.

The default directory of the shell script that controls the operation of Dr.Web ConfigD is `/etc/init.d`. The name of the script is `drweb-configd`. It has the following parameters:

Parameter	Description
`start`	Instructs to start Dr.Web ConfigD if it is not running. When Dr.Web ConfigD starts, Dr.Web ConfigD launches all the required modules of Dr.Web for UNIX Internet Gateways.
`stop`	Instructs to shut down Dr.Web ConfigD if it is running. When Dr.Web ConfigD is shutting down, Dr.Web ConfigD also shuts down all the components of Dr.Web for UNIX Internet Gateways.
`restart`	Instructs to restart (shut down and then start) Dr.Web ConfigD. Dr.Web ConfigD shuts down and then starts all the modules of Dr.Web for UNIX Internet Gateways. If Dr.Web ConfigD is not running, the parameter has the same effect as start.
`condrestart`	Instructs to restart Dr.Web ConfigD only if it is running.
`reload`	Instructs to send a HUP signal to Dr.Web ConfigD if the component is running. Dr.Web ConfigD forwards this signal to all the components of Dr.Web for UNIX Internet Gateways. The parameter is used to make all components reread their configuration.
`status`	Instructs to output the current state of Dr.Web ConfigD to the console.

To restart Dr.Web for UNIX Internet Gateways (or start it, if it is not running), use the following command:

```
# /etc/init.d/drweb-configd restart
```

How to Connect to the Central Protection Server

1. Obtain the address of the central protection server and the file of its public key from your anti-virus network administrator. You may also need additional parameters, such as an identifier and password for your workstation or identifiers of the main group and tariff group.
2. Use the `esconnect` command of the Dr.Web Ctl command-line tool provided in Dr.Web for UNIX Internet Gateways.

   For connection it is required to use the option `--Key`, by specifying the path to the public key file of the server. You can additionally enter the identifier of your host (the ID of your “workstation”, if we use the terminology used by the central protection server) and a password for authentication on the central protection server by using the `--Login` and `--Password` parameters. In this case, connection to the server will be established only if you specify a correct identifier-password pair. If the parameters are not specified, connection to the server will be established only if it is approved on the server (automatically or by the administrator of the anti-virus network, depending on the server’s settings).

   Moreover, you can use the `--Newbie` option (connect as a new user). If this mode is allowed on the server, then after this connection is approved, the server automatically generates a unique identifier/password pair, which will be further used for connection of this agent to the server. Note that in this mode the central protection server generates a new account for the host even if this host already has another account on the server.

   A standard example of the command instructing Dr.Web for UNIX Internet Gateways to connect to the central protection server:

   ```
   # drweb-ctl esconnect <server address> --Key <path to the server’s public key file>
   ```

After establishing a connection to the central protection server, the product will operate in central protection mode or in mobile mode, depending on the permissions set on the server and the value of the configuration parameter **`MobileMode`** of the Dr.Web ES Agent component. To allow unconditional use of mobile mode, set the parameter’s value to `On`. For operation in central protection mode, set the parameter’s value to `Off`.

A standard example of the command instructing Dr.Web for UNIX Internet Gateways that is connected to a central protection server to switch into mobile mode is as follows:

```
# drweb-ctl cfset ESAgent.MobileMode On
```

> If the used central protection server does not support or does not allow mobile mode, adjusting the **`MobileMode`** parameter cannot switch operation of Dr.Web for UNIX Internet Gateways to mobile mode.

How to Disconnect From the Central Protection Server

To disconnect the product from the central protection server and switch its operation into standalone mode, use the `esdisconnect` command of the Dr.Web Ctl command-line tool provided in Dr.Web for UNIX Internet Gateways:

```
# drweb-ctl esdisconnect
```

To use the product in standalone mode, a valid license key file is required. Otherwise, anti-virus functions of the product will be *blocked* after the operation is switched to standalone mode.

How to Activate the Product

1. Register on the official website of Doctor Web at https://products.drweb.com/register/.
2. At the email address that you specified during the registration you will receive an archive containing a valid license key file (you can also download this archive directly from the website after you have finished the registration).
3. Carry out the key file installation procedure.

How to Upgrade the Product

Update component versions or upgrade to a new version.

Note that during the upgrade you can be asked to remove the current product version.

How To Add or Remove Component of the Product

Follow the Custom Component Installation and Uninstallation procedure.

Note that when installing and uninstalling the component, other product components could be additionally installed or uninstalled to resolve dependencies.

How to Manage Components Operation

To view the status of the product's components and to manage their operation, you can use:

- The command-line-based management tool Dr.Web Ctl (use the **drweb-ctl** `appinfo`, **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands. To view the list of available management commands, use the command **drweb-ctl** `--help`).
- The management web interface of Dr.Web for UNIX Internet Gateways (by default, you can access it via a web browser at `https://127.0.0.1:4443/`).

How to View Log of the Product

According to default settings the general log of all product’s components is displayed in **syslog** file (the file for logging messages by the system component **syslog** depends on the system and is located in the directory `/var/log`). General log settings are defined in the configuration file in the section `[Root]` (parameters **`Log`** and **`DefaultLogLevel`**). For each component in their settings section, parameters **`Log`** and **`LogLevel`** are available. They set the log storage location and the logging level of messages that the component outputs in the log.

To change the logging settings, use the Dr.Web Ctl command line management tool and the Dr.Web for UNIX Internet Gateways management web interface (if it is installed).

- To identify errors, we recommend you to configure output of the general log of all components to a separate file and enable output of extended debug information to the log. For that, execute the following commands:

```
# drweb-ctl cfset Root.Log <path to log file>
# drweb-ctl cfset Root.DefaultLogLevel DEBUG
```

- To return to the default logging method and verbosity level for all components, execute the following commands:

```
# drweb-ctl cfset Root.Log -r
# drweb-ctl cfset Root.DefaultLogLevel -r
```

Components of the Product

This section contains a description of the components of Dr.Web for UNIX Internet Gateways. For each of them, you can find information about its functions, operation principles, and parameters stored in the configuration file of the product.

Dr.Web ConfigD

The configuration daemon—Dr.Web ConfigD—is the core component of Dr.Web for UNIX Internet Gateways. It provides central storage of configuration information for all program components, manages operation of all components, and organizes trusted data exchange between them.

Operating Principles

Main Functions

1. Starts and stops the product’s components depending on the settings. Automatically restarts components if a failure in their operation occurs. Starts components at the request of other components. Informs active components when another component starts or shuts down.
2. Provides for a centralized access of all components to configuration settings. Provides special components with interface for centralized management of configuration parameters. Notifies all required components about changes in configuration.
3. Provides components with information from the used license key file. Receives new license information from special components. Notifies running components on changes in license data or in configuration parameters.

The configuration daemon Dr.Web ConfigD is always started with *root* privileges. It launches other components of Dr.Web for UNIX Internet Gateways and communicates with them via a preliminarily open socket. The configuration daemon receives connections from other components via an information socket (publicly available) and a management socket (available only for components with superuser privileges). Loads configuration parameters and license data from files or delivers them from the used central protection server via the Dr.Web ES Agent, as well as substitutes default correct values for configuration parameters. Thus, by the moment when any component starts or `SIGHUP` signal is sent, the configuration daemon has an integral and consistent set of parameters for Dr.Web for UNIX Internet Gateways.

Upon receipt of `SIGHUP` signal, the configuration management daemon rereads configuration parameters and license data. If required, the daemon sends all components notifications instructing them to reread their configuration. Upon receipt of `SIGTERM` signal, the daemon shuts all components down and only after that finishes its own operation. The daemon also removes all temporary files of components after they are shut down.

Communication Principles

1. All components use only configuration parameters and license data received from the configuration daemon Dr.Web ConfigD on their startup.
2. The daemon collects messages from all the controlled components into an integrated log. All information output by a component to the error stream *stderr* is collected by the daemon and written to the integrated log of the product with a mark indicating what component has output this.
3. When shutting down, the controlled components return an exit code. If the code differs from 101, 102, or 103, the configuration daemon restarts this component. Thus, abnormal termination of a component triggers its restart and registration of an error message from *stderr* in the product's log.
   - If a component exits with code 101, the component will be started again only after license parameters are changed. Thus, if a component cannot operate because of license restriction, it terminates its operation and outputs code 101 to *stderr*.
   - If a component exits with code 102, the component will be started again only after configuration parameters change. Thus, if a component cannot operate because of its configuration, it terminates its operation and outputs code 102 to *stderr*. The configuration daemon will attempt to start the component again only after any parameters are changed.
   - Components started by the configuration daemon at request can terminate their operation when idle and output code 103. It is such components as Dr.Web Scanning Engine and Dr.Web File Checker.
   - If new parameter values received by the component from the configuration daemon cannot be applied "on the fly", that is, if the restart is required, the component exits with code 0. If so, Dr.Web ConfigD restarts the component.
   - If a component cannot connect to the configuration daemon or a communication protocol error occurs, the component outputs an appropriate message to *stderr* and exits with code 1.
4. Signal exchange:
   - The configuration daemon sends the component `SIGHUP` signal, which instructs to change parameters of configuration.
   - The configuration daemon sends the component `SIGTERM` signal, which instructs the component to terminate operation in 30 seconds.
   - `SIGKILL signal is sent by the`configuration daemon to trigger force termination of components which failed to shut down within 30 seconds after they received a `SIGTERM signal.`

Command-Line Arguments

To run the configuration daemon Dr.Web ConfigD, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-configd [<parameters>]
```

The configuration daemon Dr.Web ConfigD can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.
`--config`	**Description:** Instructs to use the specified configuration file for further operation. **Short form:** `-c` **Arguments:** <*path to the file*>—the path to the configuration file that you want to use.
`--daemonize`	**Description:** Instructs to run the component as a daemon; that is, without access to the terminal. **Short form:** `-d` **Arguments:** None.
`--pid-file`	**Description:** Instructs to use the specified PID file for further operation. **Short form:** `-p` **Arguments:** <*path to the file*>—the path to a file into which you would like to the process ID (PID) to be Stored.

Example:

```
$ /opt/drweb.com/bin/drweb-configd -d -c /etc/opt/drweb.com/drweb.ini
```

The command runs Dr.Web ConfigD as a daemon which uses the following configuration file: `/etc/opt/drweb.com/drweb.ini`.

Startup Notes

To enable the operation of the product, Dr.Web ConfigD must be running as a daemon. During standard booting, Dr.Web ConfigD is automatically launched when the operating system starts; for this purpose Dr.Web ConfigD comes together with a standard management script located in /etc/init.d. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

> To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-configd`

Configuration Parameters

The daemon Dr.Web ConfigD uses configuration parameters which are specified in the `[Root]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`DefaultLogLevel` *{logging level}*	Defines default logging level of event logging *for all* Dr.Web for UNIX Internet Gateways components. *The value of this parameter is used for all the components in the product which do not have their own different logging levels set up in their configuration.* **Default value:** `Notice`
`LogLevel` *{logging level}*	Logging level of event logging for Dr.Web ConfigD. **Default value:** `Notice`
`Log` *{log type}*	Logging method of the configuration daemon and logging method of those components for which another value of this parameter is not specified. Note that upon its initial startup, before the configuration file is read, the configuration daemon uses the following values of the parameter: • As a daemon (if run with the `-d` option)—`SYSLOG:Daemon` • Otherwise—`Stderr` If a component is working in a background mode (was launched with the `-d` option from the command line), then the `Stderr` value *cannot be used* for this parameter. **Default value:** `On`
`PublicSocketPath` *{path to file}*	Path to the socket used for interaction between all Dr.Web for UNIX Internet Gateways components. **Default value:** `/var/run/.com.drweb.public`
`AdminSocketPath` *{path to file}*	Path to the socket used for interaction between Dr.Web for UNIX Internet Gateways components with elevated (administrative) privileges.

	Default value: `/var/run/.com.drweb.admin`
`CoreEnginePath` *{path to file}*	Path to the dynamic library of the anti-virus engine Dr.Web Virus-Finding Engine. **Default value:** <*var_dir*>`/lib/drweb32.dll` • For **Linux**, **Solaris**: `/var/opt/drweb.com/lib/drweb32.dll` • For **FreeBSD**: `/var/drweb.com/lib/drweb32.dll`
`VirusBaseDir` *{path to directory}*	Path to the directory with virus database files. **Default value:** <*var_dir*>`/bases` • For **Linux**, **Solaris**: `/var/opt/drweb.com/bases` • For **FreeBSD**: `/var/drweb.com/bases`
`KeyPath` *{path to file}*	Path to the product key file (license or demo). **Default value:** <*etc_dir*>`/drweb32.key` • For **Linux**, **Solaris**: `/etc/opt/drweb.com/drweb32.key` • For **FreeBSD**: `/usr/local/etc/drweb.com/drweb32.key`
`CacheDir` *{path to directory}*	Path to the cache directory (used to hold cache for updates as well as cache for information about checked files). **Default value:** <*var_dir*>`/cache` • For **Linux**, **Solaris**: `/var/opt/drweb.com/cache` • For **FreeBSD**: `/var/drweb.com/cache`
`TempDir` *{path to directory}*	Path to the directory with temporary files. **Default value:** *Path copied from the system environment variable* `TMPDIR`, `TMP`, `TEMP` *or* `TEMPDIR` *(the environment variables are searched in this particular order). Otherwise* `/tmp`, *if there are no these environment variables.*
`RunDir` *{path to directory}*	Path to the directory with all PID files of running components and sockets used for interaction between the product's components. **Default value:** `/var/run`
`VarLibDir` *{path to directory}*	Path to the directory with libraries used by product components. **Default value:** <*var_dir*>`/lib` • For **Linux**, **Solaris**: `/var/opt/drweb.com/lib` • For **FreeBSD**: `/var/drweb.com/lib`
`VersionDir` *{path to directory}*	The path to a directory, where the information on Dr.Web for UNIX Internet Gateways components current versions is stored. **Default value:** <*var_dir*>`/version` • For **Linux**, **Solaris**: `/var/opt/drweb.com/version` • For **FreeBSD**: `/var/drweb.com/version`

`DwsDir` *{path to directory}*	Path to the directory that contains files of an automatically updated database of Internet resource categories . **Default value:** <*var_dir*>`/dws` • For **Linux**, **Solaris**: `/var/opt/drweb.com/dws` • For **FreeBSD**: `/var/drweb.com/dws`
`AdminGroup` *{group name | GID}*	Group of users with administrative privileges for Dr.Web for UNIX Internet Gateways management. These users, in addition to the *root* superuser, are allowed to elevate privileges of Dr.Web for UNIX Internet Gateways components to superuser privileges. **Default value:** *Is determined during the installation of the product.*
`TrustedGroup` *{group name | GID}*	Group of trusted users. The parameter is used in the work of the network traffic monitor component—SpIDer Gate. Network traffic of these users is skipped by SpIDer Gate without being scanned. *Note that you cannot specify a non-existent group here, as in this case SpIDer Gate will fail to start.* *If the parameter value is missing, you cannot specify the* `Auto` *value for the* **`OutputDivert`** *parameter in SpIDer Gate settings.* **Default value:** `drweb`
`DebugIpc` *{Boolean}*	Indicates whether detailed IPC messages are included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). IPC messages show the interaction between the configuration daemon and other components. **Default value:** `No`
`UseCloud` *{Boolean}*	Indicates whether to refer to Dr.Web Cloud service to receive information about malicious files and URLs. **Default value:** `No`
`AntispamCorePath` *{path to file}*	The parameter is not used. **Default value:** <*var_dir*>`/lib/vaderetro.so` • For **Linux**, **Solaris**: `/var/opt/drweb.com/lib/vaderetro.so` • For **FreeBSD**: `/var/drweb.com/lib/vaderetro.so`
`VersionNotification` *{Boolean}*	Notify a user on availability of updates to update the currently installed product version. **Default value:** `Yes`

Dr.Web Ctl

You can manage operation of Dr.Web for UNIX Internet Gateways from the command line with the help of a special command-line tool—Dr.Web Ctl (**drweb-ctl**).

You can do the following actions from the command line:

- Start scanning file system objects including boot records
- Launch of scanning of files on remote network hosts (see note below).
- Start updating anti-virus components (virus databases, anti-virus engine, etc. depending on the distribution).
- View and change parameters of Dr.Web for UNIX Internet Gateways configuration
- View the status of the product's components and statistics on detected threats
- View quarantine and manage quarantined objects (via the Dr.Web Ctlcomponent).
- Connect to the central protection server or disconnect from it.

Commands entered by the user to control the product can have an effect only if the Dr.Web ConfigD configuration daemon is running (by default, it is automatically launched at the operating system's startup).

> Note that some control commands require superuser privileges.
>
> To elevate privileges, use the **su** command (change the current user) or the **sudo** command (execute the specified command with other user privileges).

The Dr.Web Ctl tool supports auto-completion of commands for managing Anti-virus operation if this option is enabled in the used command shell. If the command shell does not allow auto-completion, you can configure this option. For that purpose, refer to the instruction manual for the used OS distribution.

> When shutting down, the tool returns the exit code according to convention for the POSIX compliant systems: 0 (zero)—if an operation is successfully completed, non-zero—if otherwise.
>
> Note that the tool returns a non-null exit code only in case of internal error (for example, the tool could not connect to a component, a requested operation could not be executed, etc.). If the tool detects (and possibly) neutralizes a threat, it returns the null exit code, because the requested operation (such as `scan`, etc.) is successfully competed. If it is necessary to define the list of detected threats and applied actions, analyze the messages displayed on the console.
>
> Codes of all errors are listed in the Appendix F. Known Errors section.

Remote host scanning

Dr.Web for UNIX Internet Gateways allows to perform scanning for threats of files located on remote network hosts. Such hosts can be not only full computing machines (workstations and servers) but also routers, set-top boxes and other “smart” devices that form the so-called Internet of things. To perform the remote scanning, it is necessary for the remote host to provide a remote terminal access via SSH (Secure Shell). Besides, it is required to know an IP address and a domain name of the remote host, name and password of the user, who could remotely access the system via SSH. The indicated user must have access rights to the scanned files (at least the reading rights).

This function can be used only for detection of malicious and suspicious files on a remote host. Elimination of threats (i.e. isolation in the quarantine, removal and curing of malicious objects) using means of the remote scanning is impossible. To eliminate detected threats on the remote host, it is necessary to use administration tools provided directly by this host. For example, for routers and other “smart” devices, a mechanism for a firmware update can be used; for computing machines, it can be done via a connection to them (as an option, using a remote terminal mode) and respective operations in their file system (removal or moving of files, etc.), or via running an anti-virus software installed on them.

Remote scanning is performed only via the command-line tool Dr.Web Ctl (the remotescan command is used).

Command-Line Call Format

1. Command Format for Calling the Command-Line Utility to Manage the Product

The call format for the command-line tool which manages Dr.Web for UNIX Internet Gateways operation is as follows:

```
$ drweb-ctl [<general options> | <command> [<argument>] [<command options>]]
```

Where:

- *<general options>*—options that can be applied on startup when the command is not specified or can be applied for any command. Not mandatory for startup.
- *<command>*—command to be performed by Dr.Web for UNIX Internet Gateways (for example, start scanning, output the list of quarantined objects, and other commands).
- *<argument>*—command argument. Depends on the specified command. It can be missing for certain commands.
- *<command options>*—options for managing the operation of the specified command. They can be omitted for some commands.

2. General Options

The following general options are available:

Option	Description
`-h, --help`	Show general help information and exit. To display the help information on any command, use the following call: `$ drweb-ctl <command> -h`
`-v, --version`	Show information on the module version and exit
`-d, --debug`	Instructs to show debug information upon execution of the specified command. It cannot be executed if a command is not specified. Use the call `$ drweb-ctl <command> -d`

3. Commands

Commands to manage Dr.Web for UNIX Internet Gateways can be divided into the following groups:

- Anti-virus scanning commands.
- Commands to manage updates and operation in central protection mode.
- Configuration management commands.
- Commands to manage detected threats and quarantine.
- Information commands.

> To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-ctl`

3.1. Anti-virus Scanning Commands

The following commands to manage anti-virus scanning are available:

Command	Description
`scan <path>`	**Purpose:** Start checking the specified file or directory via the Dr.Web File Checker component.

Command	Description
	Arguments: *<path>*—path to the file or directory which is selected for scanning. *This argument may be omitted, if you use the* `--stdin` *or the* `--stdin0` *option. To specify several files that satisfy a certain criterion, use the* **find** *utility (see the* Usage Examples*) and the* `--stdin` *or* `--stdin0` *option.* **Options:** `-a [--Autonomous]`—run a separate instance of Dr.Web Scanning Engine and Dr.Web File Checker to perform specified checks and terminate their operation after the scanning task is completed. Note that threats detected during stand-alone scanning are not added in the common threat list that is displayed using the `threats` command (see below). `--stdin`—get the list of paths to scan from the standard input string (*stdin*). Paths in the list need to be separated by the next line character ('`\n`'). `--stdin0`—get the list of paths to scan from the standard input string (*stdin*). Paths in the list need to be separated by the zero character NUL ('`\0`'). When using `--stdin` and `--stdin0` options, the paths in the list should not contain patterns or regular expressions for a search. Recommended usage of the `--stdin` and `--stdin0` options is processing a path list (generated by an external utility, for example, **find**) in the `scan` command (see Usage Examples). `--Report <BRIEF|DEBUG>`—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout <number>`—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0* `--PackerMaxLevel <number>`—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ArchiveMaxLevel <number>`—set the maximum nesting level when scanning archives (zip, rar, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8*

Command	Description
	`--MailMaxLevel` <*number*>—set the maximum nesting level when scanning email messages (pst, tbb, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ContainerMaxLevel` <*number*>—set the maximum nesting level when scanning other containers (HTML and so on). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MaxCompressionRatio` <*ratio*>—set the maximum compression ratio of scanned objects. The ratio must be at least equal to *2*. **Default value**: *3000* `--HeuristicAnalysis` <*On|Off*>—enable or disable heuristic analysis during the scanning. **Default value:** *On* `--OnKnownVirus` <*action*>—action applied to a threat detected by using signature-based analysis. **Allowed values:** *REPORT, CURE, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnIncurable` <*action*>—action applied on failure to cure a detected threat or if a threat is incurable. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnSuspicious` <*action*>—action applied to a suspicious object detected by heuristic analysis. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnAdware` <*action*>—action applied to detected adware programs. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnDialers` <*action*>—action applied to dialers. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnJokes` <*action*>—action applied to joke programs. **Allowed values**: *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnRiskware` <*action*>—action applied to potentially dangerous programs (riskware). **Allowed values:** *REPORT, QUARANTINE, DELETE*.

Command	Description
	Default value: *REPORT* `--OnHacktools` *<action>*—action applied to hacktools. **Allowed values:** *REPORT, QUARANTINE, DELETE.* **Default value:** *REPORT* If threat is detected in a file located in a container (an archive, email message, etc.), its removal (*DELETE*) is replaced with moving of a container to quarantine (*QUARANTINE*).
`bootscan` *<disk drive>* `| ALL`	**Purpose:** Start checking boot records on the specified disks via the Dr.Web File Checker component. Both MBR and VBR records are scanned. **Arguments:** *<disk drive>*—path to the block file of a disk device whose boot record you want to scan. You can specify several disk devices separated by spaces. The argument is mandatory. If `ALL` is specified instead of the device file, all boot records on all available disk devices will be checked. **Options:** `-a [--Autonomous]`—run a separate instance of Dr.Web Scanning Engine and Dr.Web File Checker to perform specified checks and terminate their operation after the scanning task is completed. Note that threats detected during stand-alone scanning are not added in the common threat list that is displayed using the `threats` command (see below). `--Report` *<BRIEF|DEBUG>*—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` *<number>*—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0* `--HeuristicAnalysis` *<On|Off>*—enable or disable heuristic analysis during the scanning. **Default value:** *On* `--Cure` *<Yes|No>*—enable or disable attempts to cure detected threats. If the value is set to *No*, only a notification about a detected threat is displayed. **Default value**: *No* `--ShellTrace`—enable display of additional debug information when scanning a boot record.

Command	Description
`procscan`	**Purpose:** Start checking executable files containing code of currently running processes with the Dr.Web File Checker. If a malicious executable file is detected, it is neutralized, and all processes run by this file are forced to terminate. **Arguments:** None. **Options:** `-a [--Autonomous]`—run a separate instance of Dr.Web Scanning Engine and Dr.Web File Checker to perform specified checks and terminate their operation after the scanning task is completed. Note that threats detected during stand-alone scanning are not added in the common threat list that is displayed using the `threats` command (see below). `--Report` *<BRIEF|DEBUG>*—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` *<number>*—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0* `--HeuristicAnalysis` *<On|Off>*—enable or disable heuristic analysis during the scanning. **Default value:** *On* `--PackerMaxLevel` *<number>*—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--OnKnownVirus` *<action>*—action applied to a threat detected by using signature-based analysis. **Allowed values:** *REPORT, CURE, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnIncurable` *<action>*—action applied on failure to cure a detected threat or if a threat is incurable. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnSuspicious` *<action>*—action applied to a suspicious object detected by heuristic analysis. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT*

Command	Description
	`--OnAdware` <*action*>—action applied to detected adware programs. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnDialers` <*action*>—action applied to dialers. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnJokes` <*action*>—action applied to joke programs. **Allowed values**: *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnRiskware` <*action*>—action applied to potentially dangerous programs (riskware). **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnHacktools` <*action*>—action applied to hacktools. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* *Note that if a threat is detected in an executable file, Dr.Web for UNIX Internet Gateways terminates all processes started from the file.*
`netscan` <*path*>	**Purpose:** Start distributed scanning of the specified file or directory via the Dr.Web Network Checker agent for network data scanning. If there are no configured connections to other hosts that are running Dr.Web for UNIX, then the scanning will be done only via the locally-available scanning engine (similar to the `scan` command). **Arguments:** <*path*>—path to the file or directory which is selected to be scanned. **Options:** `--Report` <*BRIEF|DEBUG*>—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` <*number*>—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0* `--HeuristicAnalysis` <*On|Off*>—enable or disable heuristic analysis during the scanning.

Command	Description
	Default value: *On* `--PackerMaxLevel` *<number>*—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ArchiveMaxLevel` *<number>*—set the maximum nesting level when scanning archives (zip, rar, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MailMaxLevel` *<number>*—set the maximum nesting level when scanning email messages (pst, tbb, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ContainerMaxLevel` *<number>*—set the maximum nesting level when scanning other containers (HTML and so on). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MaxCompressionRatio` *<ratio>*—set the maximum compression ratio of scanned objects. The ratio must be at least equal to *2*. **Default value**: *3000* `--Cure` *<Yes|No>*—enable or disable attempts to cure detected threats. If the value is set to *No*, only a notification about a detected threat is displayed. **Default value**: *No*
`flowscan` *<path>*	**Purpose:** to start scanning the specified file or directory via Dr.Web File Checker using the “flow” method. ⚠ For on-demand scanning of files and directories, it is recommended that you use the `scan` command. **Arguments:** *<path>*—path to the file or directory which is selected to be scanned. **Options:** `--ScanTimeout` *<number>*—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0*

<table>
<tr><th>Command</th><th>Description</th></tr>
<tr><td></td><td>`--HeuristicAnalysis` *<On|Off>*—enable or disable heuristic analysis during the scanning.
Default value: *On*
`--PackerMaxLevel` *<number>*—set the maximum nesting level when scanning packed objects.
If the value is set to *0*, nested objects will be skipped during scanning.
Default value: *8*
`--ArchiveMaxLevel` *<number>*—set the maximum nesting level when scanning archives (zip, rar, etc.).
If the value is set to *0*, nested objects will be skipped during scanning.
Default value: *8*
`--MailMaxLevel` *<number>*—set the maximum nesting level when scanning email messages (pst, tbb, etc.).
If the value is set to *0*, nested objects will be skipped during scanning.
Default value: *8*
`--ContainerMaxLevel` *<number>*—set the maximum nesting level when scanning other containers (HTML and so on).
If the value is set to *0*, nested objects will be skipped during scanning.
Default value: *8*
`--MaxCompressionRatio` *<ratio>*—set the maximum compression ratio of scanned objects.
The ratio must be at least equal to *2*.
Default value: *3000*
`--OnKnownVirus` *<action>*—action applied to a threat detected by using signature-based analysis.
Allowed values: *REPORT, CURE, QUARANTINE, DELETE*.
Default value: *REPORT*
`--OnIncurable` *<action>*—action applied on failure to cure a detected threat or if a threat is incurable.
Allowed values: *REPORT, QUARANTINE, DELETE*.
Default value: *REPORT*
`--OnSuspicious` *<action>*—action applied to a suspicious object detected by heuristic analysis.
Allowed values: *REPORT, QUARANTINE, DELETE*.
Default value: *REPORT*
`--OnAdware` *<action>*—action applied to detected adware programs.
Allowed values: *REPORT, QUARANTINE, DELETE*.
Default value: *REPORT*
`--OnDialers` *<action>*—action applied to dialers.</td></tr>
</table>

Command	Description
	Allowed values: *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnJokes` <*action*>—action applied to joke programs. **Allowed values**: *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnRiskware` <*action*>—action applied to potentially dangerous programs (riskware). **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* `--OnHacktools` <*action*>—action applied to hacktools. **Allowed values:** *REPORT, QUARANTINE, DELETE*. **Default value:** *REPORT* If threat is detected in a file located in a container (an archive, email message, etc.), its removal (*DELETE*) is replaced with moving of a container to quarantine (*QUARANTINE*).
`proxyscan` <*path*>	**Purpose:** Start scanning the specified file or directory via Dr.Web File Checker using the “flow” method (normally this method is used internally by the Dr.Web ClamD component). Note that threats detected by this scanning method are not included into the list of detected threats that is displayed by the `threats` command (see below). For on-demand scanning of files and directories, it is recommended that you use the `scan` command. **Arguments:** <*path*>—path to the file or directory which is selected to be scanned. **Options:** `--Report` <*BRIEF*|*DEBUG*>—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` <*number*>—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited.

Command	Description
	Default value: *0* `--HeuristicAnalysis` <*On|Off*>—enable or disable heuristic analysis during the scanning. **Default value:** *On* `--PackerMaxLevel` <*number*>—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ArchiveMaxLevel` <*number*>—set the maximum nesting level when scanning archives (zip, rar, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MailMaxLevel` <*number*>—set the maximum nesting level when scanning email messages (pst, tbb, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ContainerMaxLevel` <*number*>—set the maximum nesting level when scanning other containers (HTML and so on). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MaxCompressionRatio` <*ratio*>—set the maximum compression ratio of scanned objects. The ratio must be at least equal to *2*. **Default value**: *3000*
`rawscan` <*path*>	**Purpose:** to start “raw” scanning of the specified file or directory by Dr.Web Scanning Engine directly, without the use of Dr.Web File Checker. ⚠ Note that threats detected by “raw” scanning are not included into the list of detected threats that is displayed by the `threats` command (see below). It is recommended that you use this command only to debug the functioning of Dr.Web Scanning Engine. Note that the command outputs the “cured” status, if at least *one* threat is neutralized of those threats that are detected in a file (not *all* threats might be neutralized). Thus, it is *not recommended* to use this command if you need thorough file scanning. In the latter case it is recommended to use the `scan` command.

Command	Description
	Arguments: `<path>`—path to the file or directory which is selected to be scanned. **Options:** `--ScanEngine` *<path>*—path to the UNIX socket of the Dr.Web Scanning Engine. If not specified, an autonomous instance of the scanning engine is started (which will be shut down once the scanning is completed). `--Report` *<BRIEF|DEBUG>*—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` *<number>*—specify timeout to scan one file, in ms. If the value is set to *0*, time on scanning is not limited. **Default value**: *0* `--PackerMaxLevel` *<number>*—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ArchiveMaxLevel` *<number>*—set the maximum nesting level when scanning archives (zip, rar, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MailMaxLevel` *<number>*—set the maximum nesting level when scanning email messages (pst, tbb, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ContainerMaxLevel` *<number>*—set the maximum nesting level when scanning other containers (HTML and so on). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: 8 `--MaxCompressionRatio` *<ratio>*—set the maximum compression ratio of scanned objects. The ratio must be at least equal to *2*. **Default value**: *3000* `--HeuristicAnalysis` *<On|Off>*—enable or disable heuristic analysis during the scanning. **Default value:** *On*

Command	Description
	`--Cure` <*Yes|No*>—enable or disable attempts to cure detected threats. If the value is set to *No*, only a notification about a detected threat is displayed. **Default value**: *No* `--ListCleanItem`—enable outputting the list of clean (non-infected) files found inside a container that was scanned. `--ShellTrace`—enable display of additional debug information when scanning a file.
`remotescan` <*host*> <*path*>	**Purpose:** Connect to the specified remote host and start scanning the specified file or directory using SSH. Note that threats detected by remote scanning will not be neutralized and also will not be included into the list of detected threats that is displayed by the `threats` command (see below). This function can be used only for detection of malicious and suspicious files on a remote host. To eliminate detected threats on the remote host, it is necessary to use administration tools provided directly by this host. For example, for routers and other “smart” devices, a mechanism for a firmware update can be used; for computing machines, it can be done via a connection to them (as an option, using a remote terminal mode) and respective operations in their file system (removal or moving of files, etc.), or via running an anti-virus software installed on them. **Arguments:** <*host*>—IP address or a domain name of the remote host. <*path*>—path to the file or directory which is selected to be scanned. **Options:** `-l [--Login]` <*name*>—login (user name) used for authorization on the remote host via SSH. *If a user name is not specified, there will be an attempt to connect to a remote host on behalf of the user who has launched the command.* `-i [--Identity]` <*path to file*>—path to the file containing a private key used for authentication of the specified user via SSH. `-p [--Port]` <*number*>—number of the port on the remote host for connecting via SSH. **Default value**: *22*

Command	Description
	`--Password` <*password*>—password used for authentication of a user via SSH. *Please note that the password is transferred as a plain text.* `--Report` <*BRIEF*|*DEBUG*>—specify the type of the report with scanning results. **Allowed values:** • *BRIEF*—brief report. • *DEBUG*—detailed report. **Default value**: *BRIEF* `--ScanTimeout` <*number*>—specify timeout to scan one file, in ms. If the value is set to 0, time on scanning is not limited. **Default value**: 0 `--PackerMaxLevel` <*number*>—set the maximum nesting level when scanning packed objects. If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ArchiveMaxLevel` <*number*>—set the maximum nesting level when scanning archives (zip, rar, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MailMaxLevel` <*number*>—set the maximum nesting level when scanning email messages (pst, tbb, etc.). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--ContainerMaxLevel` <*number*>—set the maximum nesting level when scanning other containers (HTML and so on). If the value is set to *0*, nested objects will be skipped during scanning. **Default value**: *8* `--MaxCompressionRatio` <*ratio*>—set the maximum compression ratio of scanned objects. The ratio must be at least equal to *2*. **Default value**: *3000* `--HeuristicAnalysis` <*On*|*Off*>—enable or disable heuristic analysis during the scanning. **Default value:** `On`

3.2. Commands to manage updates and operation in Central protection mode

The following commands for managing updates and operation in central protection mode are available:

Command	Description
`update`	**Purpose:** Instructs to initiate the updating process of the anti-virus components (virus databases, anti-virus engine, etc. depending on the distribution) from Doctor Web's update servers

`esconnect`
 `<`*server*`>[:<`*port*`>]`
	Purpose: Connect Dr.Web for UNIX Internet Gateways to the specified central protection server (for example, Dr.Web Enterprise Server). For details, refer to Operation Modes.
	Arguments:
	• <*server*>—IP address or network name of the host on which the central protection server is operating. This argument is mandatory.
	• <*port*>—port number used by the central protection server. The argument is optional and should be specified only if the central protection server uses a non-standard port.
	Options:
	`--Key` <*path*>—path to the public key file of the central protection server to which connection is performed.
	`--Login` <*ID*>—login (workstation identifier) used for connection to the central protection server.
	`--Password` <*password*>—password for connection to the central protection server.
	`--Group` <*ID*>—identifier of the group to which the workstation is added on connection.
	`--Rate` <*ID*>—identifier of the tariff group applied to your workstation when it is included in one of the central protection server groups (can be specified only together with the `--Group` option).
	`--Compress` <*On*|*Off*>—enables (*On*) or disables (*Off*) forced compression of transmitted data. If not specified, usage of compression is determined by the server.
	`--Encrypt` <*On*|*Off*>—enables (*On*) or disables (*Off*) forced encryption of transmitted data. If not specified, usage of encryption is determined by the server.
	`--Newbie`—connect as a "newbie" (get a new account on the server).
	This command requires **drweb-ctl** to be started with *root* privileges. If necessary, use the **su** or **sudo** commands.

Command	Description
`esdisconnect`	**Purpose:** Disconnect Dr.Web for UNIX Internet Gateways from the central protection server and switch its operation to standalone mode. *The command has no effect if Dr.Web for UNIX Internet Gateways already operates in standalone mode.* **Arguments:** None. **Options:** None ! This command requires **drweb-ctl** to be started with *root* privileges. If necessary, use the **su** or **sudo** commands.

3.3. Configuration Management Commands

The following commands to manage configuration are available:

Command	Description
`cfset` *<section>*`.`*<parameter>* *<value>*	**Purpose:** to change the active value of the specified parameter in the current configuration. *Note that an equals sign is not allowed.* **Arguments:** • *<section>*—name of the configuration file's section where the parameter resides. This argument is mandatory. • *<parameter>*—name of the parameter. The argument is mandatory. • *<value>*—new value that is to be assigned to the parameter. The argument is mandatory. *The following format is always used to specify a parameter value: <section>.<parameter> <value>.* *Note that if you want to indicate several parameter values, you need to repeat the call of the command* `cfset` *as many times as the number of parameter values you want to ass. In addition, to ass a new value to the list of the parameter values, you need to use an option* `-a` *(see below). You cannot use the command option <parameter> value1, value2, because the string value1, value2 will be considered a unified parameter value.* *For description of the configuration file, refer to the section Appendix D. Configuration File, or to the documentation page displayed by* **man** `5 drweb.ini`.

Command	Description
	Options: `-a [--Add]`—do not substitute the current parameter value but add the specified value to the list (allowed only for parameters that can have several values, specified as a list). You should also use this option to when adding a new parameter group identified by a tag. `-e [--Erase]`—do not substitute the current parameter value but remove the specified value from the list (allowed only for parameters that can have several values, specified as a list). You can also use this option to delete the whole group of parameters with a tag. `-r [--Reset]`—reset the parameter value to the default. At that, *<value>* is not required in the command and is ignored if specified. Options are not mandatory. If they are not specified, then the current parameter value (the entire list of values, if the parameter currently holds several values) are substituted with the specified value. *If you use the* `-r option for sections that contain individualized parameter settings for different connection points to the` Dr.Web ClamD *monitor, parameter value in the individualized settings section will be changed to the value of its "parent" parameter having the same name and located in the general settings section of this component.* If it is necessary to add a new connection point *<point>* for Dr.Web ClamD, use the following command: `cfset ClamD.Endpoint.`*<point>* `-a`, for example: `cfset ClamD.Endpoint.point1 -a` This command requires **drweb-ctl** to be started with *root* privileges. If necessary, use the **su** or **sudo** commands.

`cfshow`
`[`*<section>*`]`
`[.`*<parameter>*`]`
	Purpose: to display parameter values in the current configuration. The parameters are output to the display as follows *<section>*.*<parameter>* = *<value>*. Sections and parameters of non-installed components are not displayed.
	Arguments:
	• *<section>*—name of the configuration file section parameters of which are to be displayed. The argument is optional. If not specified, parameters of all configuration file sections are displayed.
	• *<parameter>*—name of the displayed parameter. If not specified, all parameters of the section are displayed. Otherwise, only this parameter is displayed. If a parameter is specified without the section name, all parameters with this name from all of the configuration file sections are displayed.

Command	Description
	Options: `--Uncut`—display all configuration parameters (not only those used with the currently installed set of components). If the option is not specified, only parameters used for configuration of the installed components are displayed. `--Changed`—output only those parameters which have values different from the default ones. `--Ini`—display parameter values in the INI file format: at first, the section name is specified in square brackets, then the section parameters listed as <*parameter*> = <*value*> pairs (one pair per line). `--Value`—output only value of the specified parameter (the <parameter> argument is mandatory in this case).
`reload`	**Purpose:** to send the `SIGHUP` signal to the Dr.Web ConfigD configuration daemon. On receiving this signal, the Dr.Web ConfigD configuration daemon rereads the configuration and sends the required changes of it to Dr.Web for UNIX Internet Gateways components. Then the configuration daemon reopens the program log, restarts the components that use virus databases (including the anti-virus engine), and attempts to restart those components which were terminated abnormally. **Arguments:** None. **Options:** None

3.4. Commands to Manage Detected Threats and Quarantine

The following commands for managing threats and quarantine are available:

Command	Description
`threats [`<*action*> <*object*>`]`	**Purpose:** Apply the specified action to detected threats, selected by their identifiers. Type of the action is specified by the command’s option. If the action is not specified, displays information on detected but not neutralized threats. For each threat the following information is displayed: • Identifier assigned to the threat (its ordinal number) • The full path to the infected file • Information about the threat (name of the threat, threat type according to the classification used by the Doctor Web company) • Information about the file: size, the file owner’s user name, the time of last modification • History of operations applied to the threat: detection, applied actions etc.

Command	Description
	Arguments: None. **Options:** `-f [--Follow]`—wait for new messages about new threats and display them once they are received (CTRL+C interrupts the waiting). *If this option is applied along with any options mentioned below, it is ignored.* `--Cure` <*threat list*>—attempt to cure the listed threats (list threat identifiers separating them with commas). `--Quarantine` <*threat list*>—move the listed threats to quarantine (list threat identifiers separating them with commas). `--Delete` <*threat list*>—delete the listed threats (list threat identifiers separating them with commas). `--Ignore` <*threat list*>—ignore the listed threats (list threat identifiers separating them with commas). If it is required to apply the command to all detected threats, specify `All` instead of <*threat list*>. For example: `$ drweb-ctl threats --Quarantine All` moves all detected malicious objects to quarantine.
`quarantine [`<*action*> <*object*>`]`	**Purpose:** Apply an action to the specified object in quarantine. If an action is not specified, information on quarantined objects and their identifiers together with brief information on the original files moved to quarantine is displayed. For every isolated (quarantined) object the following information is displayed: • Identifier assigned to the quarantined object • The original path to the file, before it was moved to quarantine. • The date when the file was put in quarantine • Information about the file: size, the file owner's user name, the time of last modification • Information about the threat (name of the threat, threat type according to the classification used by the Doctor Web company) **Arguments:** None. **Options:** `-a [--Autonomous]`—start a separate instance of the Dr.Web File Checker component for checking files for performing the specified quarantine command and shut it down after the command is completed. *This option can be applied along with any options mentioned below.* `--Delete` <*object*>—delete the specified object from quarantine.

Command	Description
	Note that objects are deleted from quarantine permanently—this action is irreversible. `--Cure` <*object*>—try to cure the specified object in the quarantine. *Note that even if the object is successfully cured, it will remain in quarantine. To restore the cured object from quarantine, use the* `--Restore` *command.* `--Restore` <*object*>—restore the specified object from the quarantine to its original location. *Note that this command may require* ***drweb-ctl*** *to be started with superuser privileges. You can restore the file from quarantine even if it is infected.* `--TargetPath` <*path*>—restore an object from the quarantine to the specified location: either as a file with the name specified here (if the <*path*> is a path to a file), or just to the specified directory (if the <*path*> is a path to a directory). Can be used only in combination with the `--Restore` command. As an <*object*> specify the object identifier in quarantine. To apply the command to all quarantined objects, specify `All` instead of <*object*>. For example, `$ drweb-ctl quarantine --Restore All` restores all quarantined objects. *Note that for the* `--Restore All` *variant the additional option* `--TargetPath`*, if specified, must set a path to a directory, not a path to a file.*

3.5. Information Commands

The following information commands are available:

Command	Description
`appinfo`	**Purpose:** Output information on active Dr.Web for UNIX Internet Gateways components. The following information is displayed about each component that is currently running: • Internally-used name • Process identifier **GNU/Linux** (PID) • State (running, stopped etc.) • Error code, if the work of the component has been terminated because of an error • Additional information (optionally).

Command	Description
	For the configuration daemon Dr.Web ConfigD the following is displayed as additional information: • The list of installed components—*Installed* • The list of components which must be launched by the configuration daemon —*Should run*. **Arguments:** None. **Options:** `-f [--Follow]`—wait for new messages on component status change and output them once such a message is received (interrupt waiting by pressing CTRL+C).
`baseinfo`	**Purpose:** Display the information on the current version of the Virus-Finding Engine and status of virus databases. The following information is displayed: • Version of the anti-virus engine • Date and time when the virus databases that are currently used were issued. • The number of available virus records (in the virus databases) • The time of the last successful update of the virus databases and of the anti-virus engine • The time of the next scheduled automatic update **Arguments:** None. **Options:** None.
`certificate`	**Purpose:** Display the contents of the trusted certificate of Dr.Web used by Dr.Web for UNIX Internet Gateways. To save the certificate to a <*cert_name*>`.pem` file, you can use the following command: `$ drweb-ctl certificate > <cert_name>.pem` **Arguments:** None. **Options:** None
`idpass` <*identifier*>	**Purpose:** Display the password that has been generated by the scanning component of email messages Dr.Web MailD for the email message with the indicated identifier and used for the protection of enclosed archive with threats removed from the email message (i.e. if **`RepackPassword`** parameter is set in the component settings to `HMAC(`<*secret*>`)`). **Arguments:** • <*identifier*>—identifier of email messages.

Command	Description
	Options: `-s [--Secret]` *<secret>*—Secret word used for generation of an archive password. *If a secret word is not indicated when the command is called, the current secret word <secret> is used. It is indicated in the Dr.Web MailDsettings. And if* **`RepackPassword`** parameter is not available or is set to a value different from `HMAC(`*<secret>*`)`, *the command will return an error.*
`license`	**Purpose:** Show the information about the currently active license, or get a demo-version license, or get the key file for a license that has already been registered (for example, that has been registered on the company's website). If no options are specified, then the following information is displayed (if you are using a license for the standalone mode): • License number • Date and time when the license will expire If you are using a license provided to you by a central protection server (for the use of the product in the central protection mode or in the mobile mode), then the following information will be displayed: **Arguments:** None. **Options:** `--GetRegistered` *<serial number>*—get a license key file for the specified serial number, if the conditions for the provision of a new key file have not been breached (for example, breached by using the product not in the central protection mode, when the license is managed by a central protection server). *If the serial number is not the one provided for the demo period, you must first register it at the company's website.* For further information about the licensing of Dr.Web products, refer to the Licensing section. To register a serial number, an Internet connection is required.
`stat`	**Purpose:** Output statistics about the operation of components that process files (pressing CTRL+C or Q interrupts the statistics display) or about the operation of the network data scanning agent Dr.Web Network Checker. The statistics output includes: • Name of the component that initiated scanning • PID of the component • Average number of files processed per second during the last minute, 5 minutes, 15 minutes

Command	Description
	• Usage percentage of the scanned files cache. • Average number of scan errors per second. For the distributed scanning agent, the following information is output: • List of local components that initiated scanning • List of remote hosts that received files for scanning • List of remote hosts that sent files for scanning For local clients of the distributed scanning agent, their PID and name are specified; for remote clients—address and port of the host. For both clients—local and remote—the following information is output: • Average number of files scanned per second • Average number of sent and received bytes per second • Average number of errors per second **Arguments:** None. **Options:** `-n [--netcheck]`—Output statistics on operation of the network data scanning agent.

Usage Examples

Usage examples for Dr.Web Ctl (**drweb-ctl**):

1. Object scanning

1.1. Simple Scanning Commands

1. Perform scanning of the `/home` directory with default parameters:

```
$ drweb-ctl scan /home
```

2. Scan paths listed in the `daily_scan` file (one path per line):

```
$ drweb-ctl scan --stdin < daily_scan
```

3. Perform scanning of the boot record on the **sda**:

```
$ drweb-ctl bootscan /dev/sda
```

4. Perform scanning of the running processes:

```
$ drweb-ctl procscan
```

1.2. Scanning of Files Selected by Criteria

Examples for selection of files for scanning are listed below and use the result of the operation of the utility **find**. The obtained list of files is sent to the command **drweb-ctl** `scan` with the parameter `--stdin` or `--stdin0`.

1. Scan listed files returned by the utility **find** and separated with the NUL ('`\0`') character:

```
$ find -print0 | drweb-ctl scan --stdin0
```

2. Scan all files in all directories, starting from the root directory, on one partition of the file system:

```
$ find / -xdev -type f | drweb-ctl scan --stdin
```

3. Scan all files in all directories, starting from the root directory, with the exception of the `/var/log/messages` and `/var/log/syslog` files:

```
$ find / -type f ! -path /var/log/messages ! -path /var/log/syslog |
drweb-ctl scan –stdin
```

4. Scan all files of the *root* user in all directories, starting from the root directory:

```
$ find / -type f -user root | drweb-ctl scan --stdin
```

5. Scan files of the *root* and *admin* users in all directories, starting from the root directory:

```
$ find / -type f \( -user root -o -user admin \) | drweb-ctl scan --stdin
```

6. Scan files of users with UID in the range 1000–1005 in all directories, starting from the root directory:

```
$ find / -type f -uid +999 -uid -1006 | drweb-ctl scan --stdin
```

7. Scan files in all directories, starting from the root directory, with a nesting level not more than five:

```
$ find / -maxdepth 5 -type f | =drweb-ctl scan --stdin
```

8. Scan files in a root directory ignoring files in subdirectories:

```
$ find / -maxdepth 1 -type f | =drweb-ctl scan --stdin
```

9. Scan files in all directories, starting from the root directory, with following all symbolic links:

```
$ find -L / -type f | drweb-ctl scan --stdin
```

10. Scan files in all directories, starting from the root directory, without following symbolic links:

```
$ find -P / -type f | drweb-ctl scan --stdin
```

11. Scan files created not later than May 1, 2017 in all directories, starting with the root directory:

```
$ find / -type f -newermt 2017-05-01 | drweb-ctl scan --stdin
```

1.3. Scanning of Additional Objects

1. Scanning of objects located in the directory `/tmp` on the remote server *192.168.0.1* by connecting to it via SSH as a user *user* with the password *passw*:

   ```
   $ drweb-ctl remotescan 192.168.0.1 /tmp --Login user --Password passw
   ```

2. Configuration management

1. Display information on a current program package, including information about running processes:

   ```
   $ drweb-ctl appinfo
   ```

2. Output all parameters from the [Root] section of the active configuration:

   ```
   $ drweb-ctl cfshow Root
   ```

3. Set '`No`' as the value of the **`Start`** parameter in the `[ClamD]` section of the active configuration (this will disable the Dr.Web ClamD component):

   ```
   # drweb-ctl cfset ClamD.Start No
   ```

   Note that superuser privileges are required to perform this action. To elevate the privileges, you can use the **sudo** command, as shown in the following example:

   ```
   $ sudo drweb-ctl cfset ClamD.Start No
   ```

4. Forbid the update component to perform an update of files `123.vdb` and `567.dws`:

   ```
   # drweb-ctl cfset Update.ExcludedFiles -a 123.vdb
   # drweb-ctl cfset Update.ExcludedFiles -a 567.dws
   ```

   Note that in this case an option `-a` is used to add a new value to the already existing list of values of the parameter **`Update.ExcludedFiles`**.

5. Remove a file `123.vdb` from the list of files, the update of which is prohibited by the update component:

   ```
   # drweb-ctl cfset Update.ExcludedFiles -r 123.vdb
   ```

6. Reset the list of files, the update of which is prohibited for the update component, to the default value of:

   ```
   # drweb-ctl cfset Update.ExcludedFiles -e
   ```

7. Perform force update of anti-virus components of the product:

   ```
   $ drweb-ctl update
   ```

8. Restart the configuration of components of the installed Dr.Web program package:

```
# drweb-ctl reload
```

   Note that superuser privileges are required to perform this action. To elevate the privileges, you can use the **sudo** command, as shown in the following example:

```
$ sudo drweb-ctl reload
```

9. Connect the product to the server of central protection, operating on the *192.168.0.1* host under the condition that a public key of the server is located in file :cskey.pub

Configuration Parameters

The Dr.Web Ctl tool for managing the product from the command line does not have its own section with its parameters in the integrated configuration file of Dr.Web for UNIX Internet Gateways. It uses the parameters specified in the [Root] section of the configuration file.

Dr.Web Web Management Interface

The web interface of Dr.Web for UNIX Internet Gateways allows you to:

1. View the current state of the program’s components, start or stop some of the components.
2. View the status of updates and start an updating process manually, if required.
3. View the status of the product’s license and load a license key, if required.
4. View the list of detected threats and manage quarantined objects (threats detected in local file system via the Dr.Web File Checker component are displayed only).
5. Edit the settings of the components included in Dr.Web for UNIX Internet Gateways.
6. Connect the program to a central protection server or switch the program’s operation into standalone mode.
7. Start an on-demand scanning of local files (including a capability to do it by dragging and dropping files onto the page opened in your browser).

System Requirements of the Web Interface

Correct operation of the web interface is guaranteed for the following web browsers:

- **Internet Explorer**—version 11 and later.
- **Mozilla Firefox**—version 25 and later.
- **Google Chrome**—version 30 and later.

Accessing the Web Interface

To access the web interface, in the browser’s address bar type in an address that looks like:

```
https://<host_with_drweb>:<port>/
```

where *<host_with_drweb>* is the IP address or the name of the host where the product containing the web-interface-server component—Dr.Web HTTPD—is running, and *<port>* is the port (on this host) which Dr.Web HTTPD is listening on. To access a product component which operates on the local host, use IP address `127.0.0.1` or the name `localhost`. By default, the *<port>* is `4443`.

Thus, to access the web interface on the local host by default, enter the following URL in the browser’s address bar:

```
https://127.0.0.1:4443/
```

After connection to the managing server is established, a startup page opens and displays an authentication form. To access management functions, fill in the authentication form by specifying the login and password of a user who has administrative privileges on the host where the product operates.

Main Menu

In the left pane of the web interface, which appears once you have successfully passed authentication, there is a main menu, the items in which allow you to do the following:

- **Main**—opens the main page which displays the full list of installed components of Dr.Web for UNIX Internet Gateways and their status.
- **Threats**—opens a page which displays all the threats detected on the server. In this section, you can manage these detected threats (for example, move infected objects to quarantine, rescan, cure or delete detected malicious objects).
- **Settings**—opens a page with the component settings of Dr.Web for UNIX Internet Gateways installed on the server.
- **Information**—opens a page that shows brief information about the version of this web interface and about the state of virus databases.
- **Help**—opens a new browser tab with help information about the installed components of the product.
- **Scan file**—displays a panel for quick file scanning, which will stay available on top of any opened page of the web interface until you close this panel.
- **Sign out**—ends the current web interface session.

Managing the Components

You can view the list of components included in Dr.Web for UNIX Internet Gateways and manage their operation on the **Main** page.

The listed components of the product are divided into two groups: main components, which monitor threats, and service components, which are responsible for the overall correct operation of the product. The list of main components is displayed as a table in the upper part of the page (the list of components depends on the scope of supply for your product). For each component the following information is specified:

1. **Name of the component**. Click the name to open the settings page containing the settings for this component;
2. **State of the component**. The state of a component is indicated by a switch icon and by a note about the component's current state. To start a component or to suspend its operation, you only need to click its switch. The possible states of the switch are:

⬜	—the component is disabled and is not used;
🟩	—the component is enabled and works correctly;
▨	—the component is enabled but is not working because of an error.

If an error occurred in the operation of a component, instead of a note about the component's state an error message is displayed. If you click the (!) icon, a window will pop up with detailed information about the error that occurred and with recommendations for resolving this error.

3. **Average load**. The average numbers of files processed by the component per second within the last minute, 5 minutes, 15 minutes respectively are specified (displayed as three numbers separated with forward slashes "/").
4. **Errors**. The average numbers of errors encountered by the component per second within the last minute, 5 minutes, 15 minutes respectively are specified (displayed as three numbers separated with forward slashes "/").

To display a tooltip, place the cursor over the ? icon.

Below the table, which provides information about main components, you will find service components (such as the the scanning engine, the file scanning component, etc.) listed as a set of tiles. For each service component, its state and operational statistics are also displayed. To open the settings page of any of these components, click the name of a required component. As a rule, these components are started and stopped automatically when needed. If any of them may be started and stopped manually by the user, then, besides the name and the operational statistics, a switch for starting and stopping the component will be displayed on the tile of any such service component.

The bottom of the page displays whether the virus databases are up to date and license information. To force a virus database update, click **Update**. By clicking on the **Renew** button (or on the **Activate** button, depending on the current state of your license) you can renew or activate a license by uploading a valid key file that is appropriate for your product to the licensing server.

Threats Management

You can view the list of detected threats and manage the reaction to them on the **Threats** page.

This page contains the full list of threats detected by the components of Dr.Web for UNIX Internet Gateways that monitor and scan the file system. In the upper part of the page, you can see a menu which allows filtering the threats by their category:

- **All**—show all detected threats (including both active and quarantined threats).
- **Active**—show only active threats; i.e. detected but not neutralized yet.
- **Blocked**—show all blocked threats, that is, threats that were not neutralized, but for which the infected objects containing them were blocked.
- **Quarantined**—show threats that were moved to quarantine.
- **Errors**—show threats that were not processed because of an error.

Just next to each name of a threat category (to its right) in the upper menu, the quantity of detected threats that fall into this category is displayed. The currently selected category, for which the threats belonging to it are currently displayed, is emphasized in a darker font. To display threats of a required category, click the name of the category in the menu.

Threats detected by components that scan network traffic (SpIDer Gate, Dr.Web ICAPD), and also by Dr.Web ClamD are not displayed on the **Threats page. To trace the threats detected by these components, you can control threat counters and trace notifications available via SNMP (**Dr.Web SNMPD gives access to threat counters and notifications according to the MIB Dr.Web structure).

For each threat, the following information is listed:

- **File**—name of the file that contains a malicious object (file path is not specified).
- **Owner**—name of the user who owns the infected file.
- **Component**—name of the component of Dr.Web for UNIX Internet Gateways that detected the threat.
- **Threat**—name of the threat that was detected in the file (according to the classification used by the Doctor Web company).

For any object selected in the list, the following information is displayed:

- Name of the threat (displayed as a link that opens a page of the Dr.Web virus information library with the threat's description).
- File size, in bytes.
- Name of the component that detected the threat.
- Date and time when the threat was detected.
- Date and time when the threat was last modified.
- Name of the user who owns the infected file.
- Name of the group that includes the file owner.
- Identifier that was assigned to the quarantined file containing a threat (if the file was quarantined).
- Full path that points to the original location of the file (where the file was located at the moment of threat detection).

You can select any object in the list by clicking on it. To select multiple objects, set the checkboxes for the corresponding objects. To select all objects or cancel the selection, select the check box in the **File** field in the threat list's header.

To apply actions to objects selected in the list, click the corresponding button on the toolbar, which is located directly above the threat list. The toolbar contains the following buttons (note that some of them can be unavailable depending on the type of selected threats):

⊗	—instructs to remove (i.e. to permanently delete) selected files.
↺	—instructs to restore selected files from quarantine to their original location.

⋯	—instructs to apply an additional action to selected files (available actions are specified in the drop-down list): • **Quarantine**—instructs to put the selected files that contain threats to quarantine • **Cure**—will attempt to cure the threats • **Ignore**—instructs to ignore the threats detected in selected files and to remove the threats from the list

You can also filter displayed threats based on a search query. To filter unnecessary threats out and display only those that correspond to the query, use the search box. The box is displayed on the right side of the toolbar and is marked with 🔍. To filter the threat list, enter a word in the search box. All threats that do not have the entered word in their name or description, will be hidden (this filtering is not case-sensitive). To clear search results and display the unfiltered list, click ⊗ in the search box or erase the word.

Managing the Settings

You can view and change current configuration parameters of the components included in Dr.Web for UNIX Internet Gateways and listed on the main page. For that, open the **Settings** page. On this page you will also be able to switch the program into the *central protection* mode or into the *standalone* mode (for further information about these modes please refer to Operation Modes).

On the left side of the page, a menu is displayed, which contains the names of all the program’s components whose settings can be viewed and adjusted. To view and adjust the settings of any component, first click on the name of a desired in this menu. The name of the component whose settings you are currently viewing and editing will be highlighted in this menu on the left.

- The **Central protection** item in the menu will take you to the page for managing the central protection mode.
- The **Common settings** item in the menu corresponds to the settings of the Dr.Web ConfigD component, which is responsible for the overall functioning of the product.

If a component has sections with additional settings apart from the section with its main settings (for example, such sections are available for the Dr.Web ClamD component, which emulates the interface of the **ClamAV®** anti-virus and uses these additional sections to hold individual scanning parameters for different clients that use different connection addresses), then an icon indicating that you can expand/collapse additional sections is displayed to the left of the component’s name. If the icon looks like ▸, additional sections are hidden. If the icon looks like ▾, additional sections are displayed on the menu, one per line. To expand/collapse the list of additional sections, click this expand/collapse icon next to the name of the required component.

- The additional sections with settings are displayed as indented lines. To view or edit parameters of an additional section, click its name in the menu.
- To add an additional subordinate section with settings for a component, if it is allowed, click + to the right of the component’s name. In the window that will appear, specify a unique name

(tag) for the new subsection and click **OK**. To close the window without creating a subsection, click **Cancel**.

- To delete a subsection for a component, if allowed, click ✕to the right of the subsection's name (tag). In the window that will appear, confirm that you want to delete the subsection and click **OK**. To close the window without deleting the subsection, click **No**.

At the top of the settings page, you can see a menu that allows you to change the viewing mode. The following modes are available:

- **All**—show the table with all the component's configuration parameters that can be viewed and adjusted.
- **Changed**—show the table with the component's configuration parameters that have values different from the default ones.
- **Ini Editor**—show a text editor with this component's configuration parameters that have values different from the default ones. The displayed text has the same format as the configuration file (contains `parameter = value` pairs).

You can also filter displayed parameters based on a search query. To filter unnecessary parameters out and display only those that correspond to the query, use the search box. The box is displayed on the right side of the viewing mode menu and is marked with 🔍. To filter the parameter list, enter any word in the search box. All parameters that do not have the entered word in their description, will be hidden (this filtering is not case-sensitive). To clear the search results and display the unfiltered list, click ⓧ in the search box or erase the word in it.

Parameters can be filtered out only when they are displayed in tabular form (i.e. in the **All** and **Changed** viewing modes).

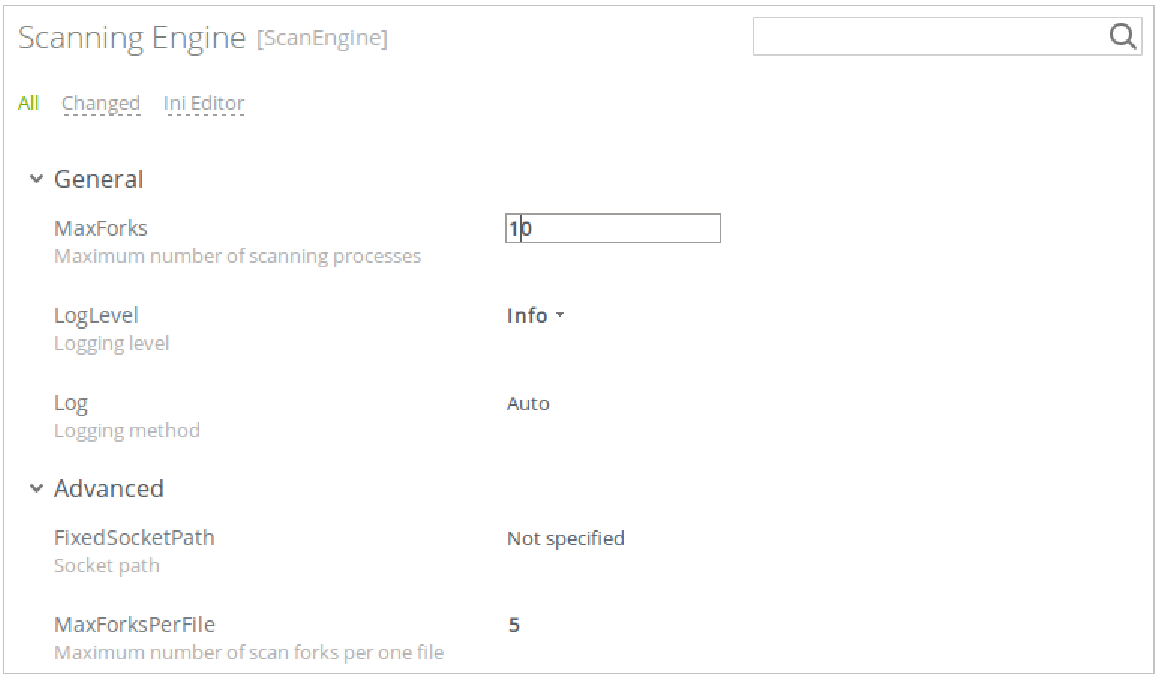
Viewing and Editing Component Settings in Tabular Form

When viewing parameters in tabular form (the **All** and **Changed** viewing modes), each table row contains a description of a parameter (on the left) and its current value (on the right). For Boolean parameters (those that have only two available values: "Yes" and "No"), a checkbox is displayed instead of a value (checked means "Yes", unchecked means "No").

> When you select to view all parameters (not only those that were changed), the modified (non-default) values are indicated in bold.

The complete parameter list is split into groups (such as **Main**, **Advanced**, etc.). To collapse or expand a group, click on its heading (its name). When a group is collapsed and its parameters are not displayed in the table, the following icon appears to the left of the group's name: ›. When a group is expanded and the parameters are displayed in the table, the following icon appears to the left of the group's name: ⌄.

To adjust a parameter, click its current value in the table (for a Boolean parameter—set or remove a check mark in the corresponding checkbox). If a parameter has a set of predefined values, they will all appear as a drop-down list after you click the current value. If a parameter has a numeric value, an editing box will appear after you click the current value. Specify a required value and press ENTER. The figure below shows examples of how to change parameter values (note that the set of components shown in the figure can differ from the one supplied to you). All changes made to parameter values are immediately applied to the configuration of the corresponding component.



Scanning Engine [ScanEngine]

All Changed Ini Editor

General

MaxForks
Maximum number of scanning processes
10

LogLevel
Logging level
Info

Log
Logging method
Auto

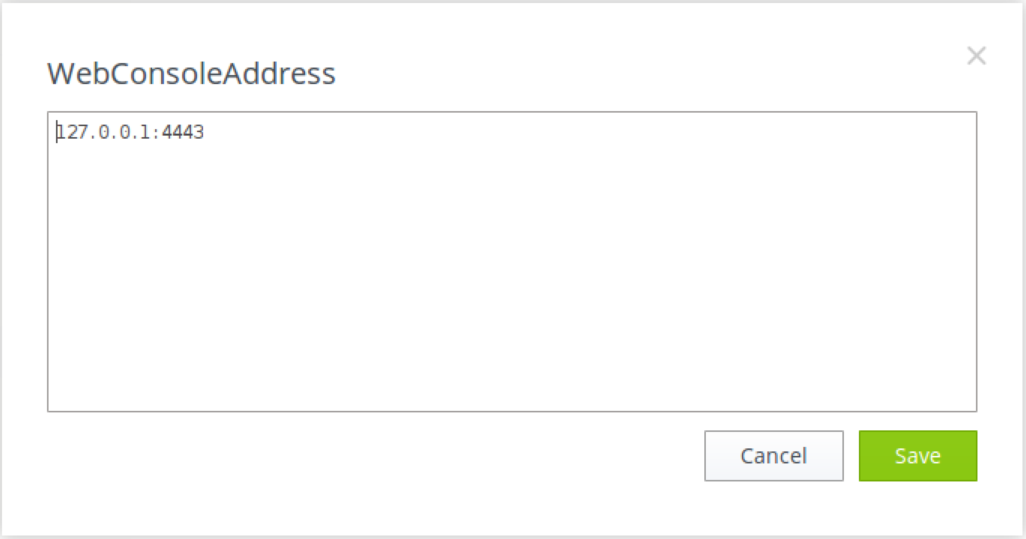
Advanced

FixedSocketPath
Socket path
Not specified

MaxForksPerFile
Maximum number of scan forks per one file
5

Figure 2. Component's settings in tabular form

If the parameter expects a string as its value or accepts a list of arbitrary values, a pop-up window will appear once you click on the parameter's current value to edit it. If the parameter accepts a list of values, they will be shown in a multi-line editing box (one value per line) as shown in the figure below. To edit the listed values, you need to change, delete or add any required lines in the editing box.



WebConsoleAddress

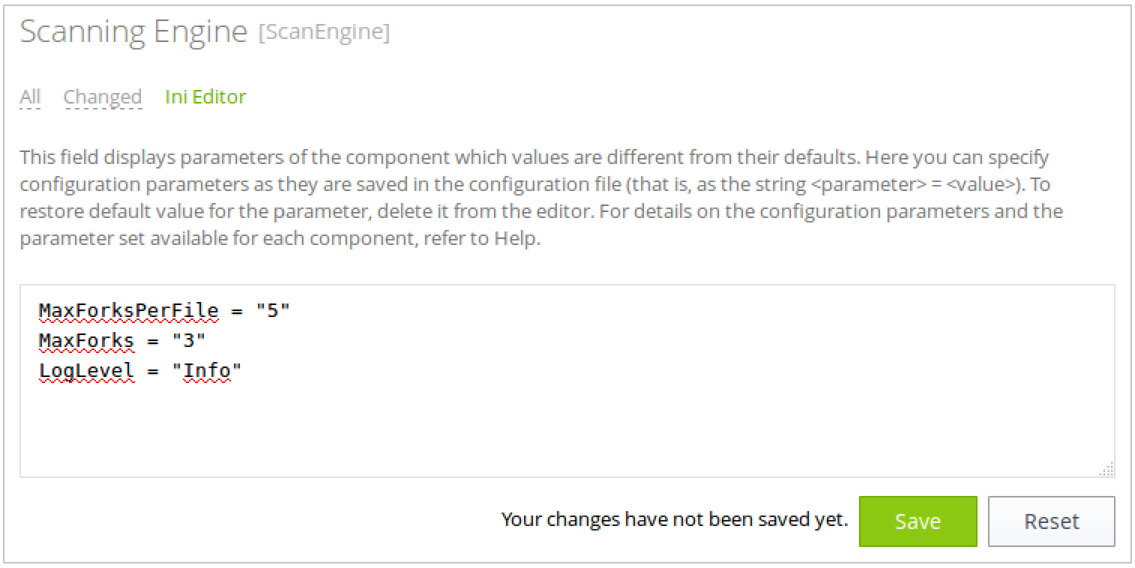
```
127.0.0.1:4443
```

Figure 3. Editing a list of values

After editing the value of a parameter, click **Save** to apply your changes and to close the window. To close the window without applying the changes click **Cancel** or click the × icon in the upper right corner of the pop-up window.

Viewing and Editing Components’ Settings in a Text Editor

When viewing parameters in the **Ini Editor** mode, they are displayed in the same format as in the configuration file of the product (as `parameter = value` pairs), where parameter is a parameter’s name that is written directly into the configuration file (into the settings section of the corresponding component). In this mode, only those parameters are displayed whose values differ from the default ones (that is, parameters whose values are emphasized in bold font in the **All** viewing mode). The figure below shows how parameters are displayed in this simple-view textual editor.



This field displays parameters of the component which values are different from their defaults. Here you can specify configuration parameters as they are saved in the configuration file (that is, as the string <parameter> = <value>). To restore default value for the parameter, delete it from the editor. For details on the configuration parameters and the parameter set available for each component, refer to Help.

```
MaxForksPerFile = "5"
MaxForks = "3"
LogLevel = "Info"
```

Your changes have not been saved yet.

Figure 4. Simple textual settings editor

To make any desired changes, edit the text in this text editor according to the same rules as described for editing the configuration file (this will modify only the section that contains the settings of the component highlighted on the left). If necessary, you can specify a new value for any parameter available for the component. In this case, the value of this parameter changes from its default setting to the value you enter in the editor. If you want to reset the parameter back to its default value, just erase the line containing this parameter in this text editor. If you do so, then, once you save the changes, the parameter will be restored to its default value.

Once you have finished editing parameters’ values, click **Save** to apply the changes or click **Cancel** to discard them.

If you click **Save**, the text is validated: the program checks whether all parameters are existent and their set values are valid. In case of an error, the appropriate message will be displayed.

For details on the configuration file, its, and its features that are important for specifying parameter values, refer to Appendix D. Configuration File section.

Additional Information

- Configuration parameters of Dr.Web ConfigD (Common settings).
- Configuration parameters of Dr.Web ICAPD.
- Configuration parameters of SpIDer Gate.
- Configuration parameters of Dr.Web Firewall for Linux.
- Configuration parameters of Dr.Web ES Agent.
- Configuration parameters of Dr.Web Updater.
- Configuration parameters of Dr.Web ClamD.
- Configuration parameters of Dr.Web File Checker.
- Configuration parameters of Dr.Web Scanning Engine.
- Configuration parameters of Dr.Web Network Checker.
- Configuration parameters of Dr.Web SNMPD.
- Configuration parameters of Dr.Web CloudD.
- Configuration parameters of Dr.Web LookupD.
- Managing the Central Protection.

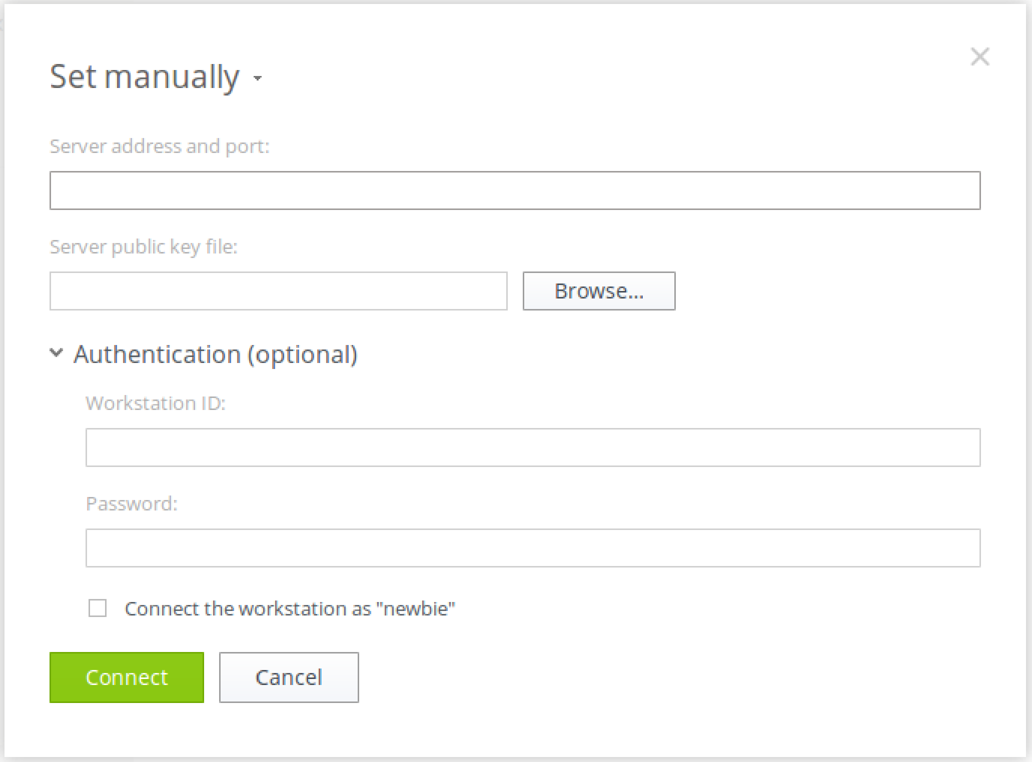
Managing the Central Protection

You can connect Dr.Web for UNIX Internet Gateways to a central protection server or switch back to the standalone mode, thereby disconnecting the product from the central protection server. To open the page where you can manage central protection, chose the item called **Central protection** from the settings menu on the **Settings** page.

To connect Dr.Web for UNIX Internet Gateways to a central protection server or to disconnect from it, use the corresponding checkbox on this page.

Connecting to an Anti-Virus Network

At an attempt to connect to a central protection server a pop-up window will appear on the screen; in this window you need to specify the parameters for connecting to the central protection server.



Set manually

Server address and port:

Server public key file:

Authentication (optional)

Workstation ID:

Password:

□ Connect the workstation as "newbie"

Figure 5. Connection to the central protection server

In the drop-down list located at the top of the window chose one of the methods for connecting to a central protection server. Three methods are available:

- *Load from file*
- *Set manually*
- *Detect automatically*

If you select the *Load from file* option, then in the corresponding field of this window you will also need to specify a path to a file that contains connection settings. The file is provided by the anti-virus network administrator. If you select the *Set manually* option, you will need to specify the address and the port of the central protection server. For the *Set manually* or *Detect automatically* options, you can also specify the path to the file containing the server's public key (provided by your network administrator or Internet service provider).

If these fields are filled in, then your connection to the central protection server will succeed only if a correct identifier/password pair was entered. If you leave these fields empty, then connection to the central protection server will succeed only if this connection is approved at the central protection server (either automatically or by the anti-virus network administrator, depending on that server's settings). Additionally, in the **Authentication** section you can specify your login (workstation identifier) and password for authentication on the central protection server, if you know them.

Moreover, you can use the **Connect as a "newbie"** option (to connect as a new user). In this case, if the Newbie mode is allowed on the central protection server for connections from workstations, then the central protection server, after approving this connection, automatically generates a unique identifier/password pair, which will be from this time on used for connecting your computer to the server. Note that in this mode the central protection server generates a new account for your workstation even if your workstation already has another account on the server.

> Connection parameters must be specified in strict accordance with the instructions provided by the administrator of your anti-virus network or service provider.

To connect to the server after having specified all the parameters, press the **Connect** button and wait until the connection procedure completes. To close the settings window without connecting to the server, press the **Cancel** button.

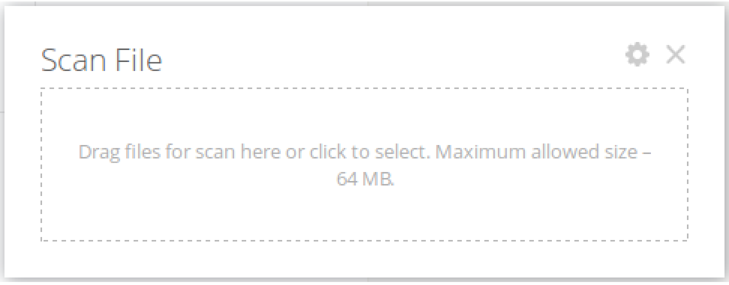
> Once you have connected Dr.Web for UNIX Internet Gateways to a central protection server, its operation will be managed by the central protection server, until you switch back to the standalone mode. Connection to the central protection server will be established automatically every time when Dr.Web for UNIX Internet Gateways is started.

Scanning local files

The web interface provides a capability to scan any files stored on your local computer (from which you are currently accessing the web interface) to determine whether the files have any malicious content, the scanning is done with the help of the scanning engine that is part of Dr.Web for UNIX Internet Gateways. The files selected for scanning will be uploaded (via the HTTP protocol) to your server on which Dr.Web for UNIX Internet Gateways is running, but after the scanning, even if any threats are found, the files will not be stored on the server, neither will they be moved to quarantine there. The user who sent the files to scan them will only be informed about the result of the scanning.

Opening a panel to scan local files and setting parameters for the scanning

You can select and upload the files that you want to scan via the scanning panel for local files which is displayed when you choose the **Scan file** item in the main menu of the web interface. The launched panel is displayed in the bottom right corner of the web interface. The figure below shows what the scanning panel for local files looks like.

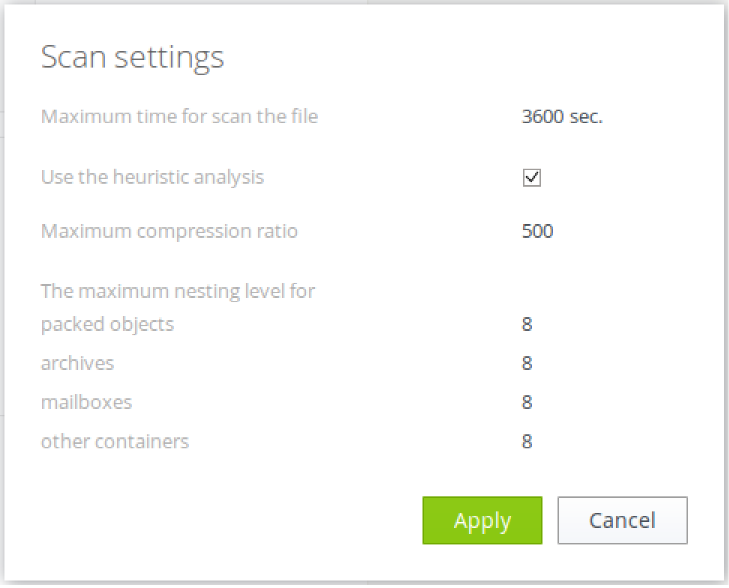


Scan File

Drag files for scan here or click to select. Maximum allowed size – 64 MB.

Figure 6. The scanning panel for local files

To close this panel, click the × icon on the panel’s top right-hand corner. By clicking the ⚙ icon you can display the settings for the scanning of local files: the maximum time to scan a file (which does not include the time it takes to upload the file to your server from your local computer), using the heuristic analysis during the scan, and also the maximum compression ratio for compressed objects and the maximum nesting level for objects packed into containers (such as archives).



Scan settings

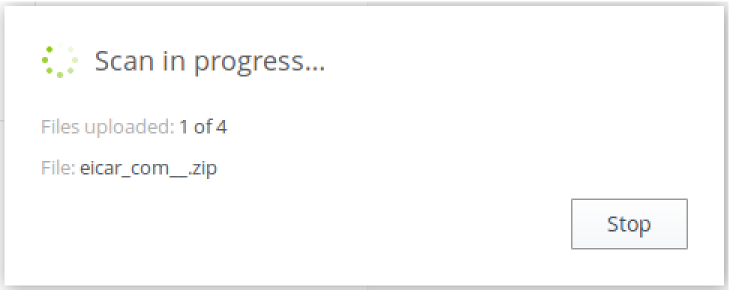
Maximum time for scan the file	3600 sec.
Use the heuristic analysis	☑
Maximum compression ratio	500
The maximum nesting level for	
packed objects	8
archives	8
mailboxes	8
other containers	8

Figure 7. Setting the parameters for the scanning of local files

To apply the changed settings and to return to the file selection mode where you can choose the files to scan, press the **Apply** button. To go back to file selection without applying your changes to the settings, press the **Cancel** button.

Launching the scanning of local files

To select files for scanning and to start their scanning, left-click on the target area that says **Drag files for scan here or click to select**. Upon your click there, a standard file selection window of your operation system’s file manager will open. You can choose multiple files at once for scanning. Please, note that you are not allowed to choose directories for scanning. You can also drag selected files with your mouse directly onto the target area of the file scanning panel from the file manager window. Once the files to be scanned have been specified, they will start being uploaded to your server where Dr.Web for UNIX Internet Gateways is installed; and once a file is uploaded, its scanning starts. During the uploading and scanning of the files the file scanning panel displays the overall progress of the scanning procedure.



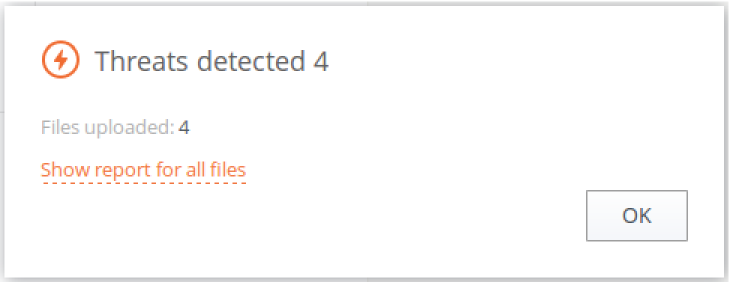
Scan in progress...

Files uploaded: 1 of 4

File: eicar_com__.zip

Figure 8. Current progress for the scanning of local files

If necessary, you can abort the scanning by pressing the **Stop** button. Once the scanning is completed, a report about the scanning of the uploaded files will be displayed on the file scanning panel.

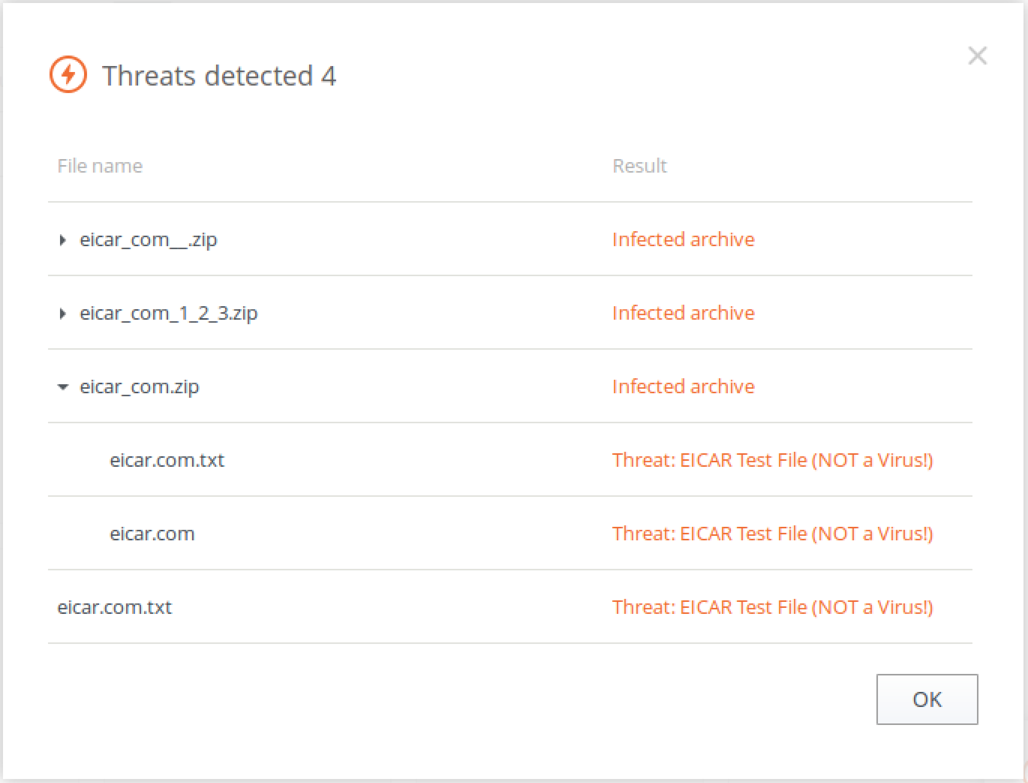


Threats detected 4

Files uploaded: 4

Figure 9. Results for the scanned local files

If multiple files were uploaded, an extended report about the scanning will be available. To see the extended report, click the link that says **Show report for all files**.



Threats detected 4

File name	Result
eicar_com__.zip	Infected archive
eicar_com_1_2_3.zip	Infected archive
eicar_com.zip	Infected archive
eicar.com.txt	Threat: EICAR Test File (NOT a Virus!)
eicar.com	Threat: EICAR Test File (NOT a Virus!)
eicar.com.txt	Threat: EICAR Test File (NOT a Virus!)

Figure 10. Extended report about the scanned local files

To close the report and to return to the state where the panel allows selecting new files for scanning, press **OK**.

> It is possible to start scanning local files (using the current settings for the scanning) even when the file scanning panel is closed. To start uploading and scanning local files, just drag and drop them from the file manager window onto a page of the web interface opened in your browser.

Dr.Web ICAPD

The Dr.Web ICAPD component connects to an HTTP proxy server (such as **Squid**) via the ICAP protocol. Typically, an HTTP proxy server is installed on a server (gateway) that is used to provide Internet access to LAN users. The proxy server uses Dr.Web ICAPD as an external filter. Thus, Dr.Web ICAPD analyzes user requests and server responses to these requests. If user access to any resource located on the external network must be forbidden, or transmitted data (a user request or a server response) contains a threat or cannot be scanned because of an error, Dr.Web ICAPD instructs the proxy server to return a special HTML page to the user, which is generated by Dr.Web ICAPD from a template.

In case of high intensity of the scanning of files transferred via the HTTP protocol, there is a possibility of having problems with scanning due to depletion of the number of available file descriptors by the Dr.Web Network Checker component.. In this case, it is necessary to increase the limit of the number of file descriptors available to Dr.Web for UNIX Internet Gateways.

Operating Principles

The Dr.Web ICAPD component uses the ICAP protocol (the *Internet Content Adaptation Protocol* described in RFC 3507) to interact with a proxy server, which is external with respect to Dr.Web for UNIX Internet Gateways and which handles HTTP/HTTPS connections from LAN hosts to web servers.

From the proxy server, the component receives tasks to check (“to adapt”, in ICAP terms) the requests sent from local hosts to servers, and responses, received from the servers. If a user request contains a URL that is included into the black list or belongs to any of the unwanted categories of web resources, Dr.Web ICAPD instructs the proxy server to break the connection with the web server and to return to the client an HTML page generated by Dr.Web ICAPD using a template which is supplied together with the component. The page contains a message informing the user that the access to requested resource is denied, and a description of the denial reason. A similar page is generated and then returned to a user by the proxy server if Dr.Web ICAPD detects a threat to be blocked in the web server’s response. A diagram showing the operation of this component is given in the figure below.

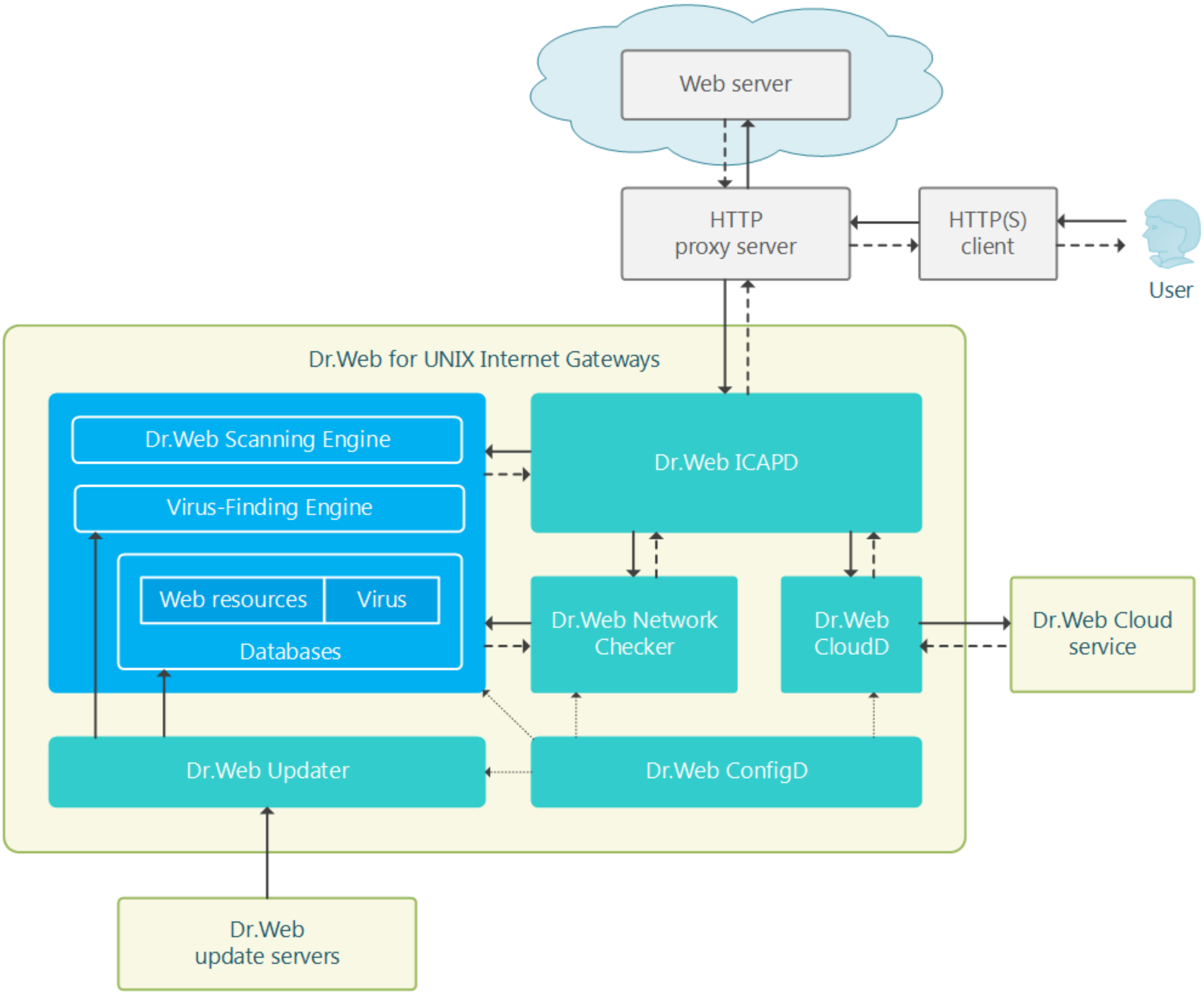


Figure 11. Diagram of the components' operation

To check whether any given URL belongs to any of the categories, the component not only uses the database of web resource categories, which is updated regularly from Doctor Web's update servers, but also refers to the Dr.Web Cloud service. Doctor Web keeps track of the following web resources categories:

- *InfectionSource*—websites containing malicious software ("infection sources").
- *NotRecommended*—fraudulent websites (that use "social engineering") visiting which is not recommended.
- *AdultContent*—websites that contain pornographic or erotic materials, dating sites, etc.
- *Violence*—websites that encourage violence or contain materials about various fatal accidents, etc.
- *Weapons*—websites that describe weapons and explosives or provide information on their manufacturing.
- *Gambling*—websites that provide access to online games of chance, casinos, auctions, including sites for placing bets, etc.
- *Drugs*—websites that promote use, production or distribution of drugs, etc.
- *ObsceneLanguage*—websites that contain the obscene language (in titles, articles, etc.).

- *Chats*—websites that offer a real-time transmission of text messages.
- *Terrorism*—websites that contain aggressive and propaganda materials or terroristic attacks descriptions, etc.
- *FreeEmail*—websites that offer the possibility of free registration of a web mailbox.
- *SocialNetworks*—different social networking services: general, professional, corporate, interest-based; thematic dating sites.
- *DueToCopyrightNotice*—websites that were specified by the holders of copyrights pertaining to content or works protected by copyright law (movies, music, etc.).

In the settings, the system administrator can specify the categories of web resources users' access to which is unwanted. It is also possible to configure one's own black lists to block the access to the necessary web resources, and white lists to allow access for users. The access to the web resources included into white lists will be allowed, even if they belong to the unwanted categories. If there is no information about a URL in the local black lists and the local database of web resource categories, the program refers to the Dr.Web Cloud service. It allows the program to check whether any information is available about the maliciousness of the URL. Such information is received from other Dr.Web's products on a real-time basis.

One and the same website can belong simultaneously to several categories. User access to such a website will be blocked if at least one category to which the website belongs has been set as unwanted by the administrator.

Even if the website is included into the white list by the administrator, the data (sent and downloaded from the website) is checked for threats.

Due to special aspects of the ICAP protocol, the scanning of large portion of data (`.iso` images, large archives, video files, etc.) can take a long time. It is recommended that you configure restrictions according to the MIME type of data to be scanned. In the HTTP proxy server settings, it is also recommended that you restrict the maximum size of data allowed to send for scanning via the ICAP protocol (see an example for the proxy Server **Squid**).

The Dr.Web Updater component is used to regularly and automatically update the databases of web resource categories from Doctor Web update servers. The same component is used to update virus databases for the Dr.Web Scanning Engine scanning engine. The Dr.Web CloudD component is used to refer to Dr.Web Cloud service (using of the cloud service is configured in Appendixes common settings and can be disabled, if necessary). To check transferred data, Dr.Web ICAPD uses the Dr.Web Network Checker component. The latter one initiates scanning via the Dr.Web Scanning Engine scanning engine.

Command-Line Arguments

To launch Dr.Web ICAPD from the command line of the operating system, the following command is used:

```
$ <opt_dir>/bin/drweb-icapd [<parameters>]
```

Dr.Web ICAPD can process the following parameters:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-icapd --help
```

This command outputs short help information on Dr.Web ICAPD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-icapd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[ICAPD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/bin/drweb-icapd` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-icapd` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-icapd`
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value**: `drweb`
`Start` *{Boolean}*	The component must be launched by the Dr.Web ConfigD configuration daemon. When you specify the `Yes` value for this parameter, it instructs the configuration daemon to start the component immediately; and when you specify the `No` value, it instructs the configuration daemon to terminate the component immediately. **Default value**: `No`
`DebugDumpIcap` *{Boolean}*	Instructs to include detailed ICAP messages into the log file on the debug level (i.e. when you set **`LogLevel`** `= DEBUG`). **Default value**: `No`
`ListenAddress`	Defines a network socket (IP address and port) on which Dr.Web ICAPD must listen for connections from HTTP

{network socket}	proxy servers. **Default value:** `127.0.0.1:1344`
`UsePreview` *{Boolean}*	Instructs Dr.Web ICAPD to use the *ICAP preview* mode. *It is recommended that you do not change the default value of this parameter, unless it is necessary.* **Default value**: `Yes`
`Use204` *{Boolean}*	Defines whether Dr.Web ICAPD is allowed to return the response code `204` not only in the *ICAP preview* mode. *It is recommended that you do not change the default value of this parameter, unless it is necessary.* **Default value**: `Yes`
`AllowEarlyResponse` *{Boolean}*	Defines whether Dr.Web ICAPD is allowed to use the ICAP's early response mode, i.e. is allowed to start sending an "early" response to the client before the entire request has been received from the HTTP proxy server. *It is recommended that you do not change the default value of this parameter, unless it is necessary.* **Default value**: `Yes`
`TemplatesDir` *{path to directory}*	Path to the directory that contains the templates for the HTML notifications sent upon blocking a web resource. **Default value:** `<var_dir>/templates/icapd` • For **Linux**, **Solaris**: `/var/opt/drweb.com/templates/icapd` • For **FreeBSD**: `/var/drweb.com/templates/icapd`
`Whitelist` *{domain list}*	List of domains that *can be used as the white list* (i.e. list of domains allowed for connection for users, even if these domains are included into blocked categories. In addition, user access will be allowed to all sub-domains of domains indicated in this list.) *The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).* **Example:** Add to the list of domains `example.com` and `example.net`. 1. Adding of values to the configuration file. • Two values in one string

	`[ICAPD]`
	` Whitelist = "example.com",`
	`"example.net"`
	• Two strings (one value per a string)
	`[ICAPD]`
	` Whitelist = example.com`
	` Whitelist = example.net`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# **drweb-ctl** cfset ICAPD.Whitelist -a`
	`example.com`
	`# **drweb-ctl** cfset ICAPD.Whitelist -a`
	`example.net`
	⚠ Actual usage of the domain list indicated in this parameter depends on the *method* of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.
	The list of default rules (see below) guarantees that access to domains (and their sub domains) from this list will be provided even if it contains domains from the list of blocked web source categories. Besides, this default set of rules guarantees that data downloaded from the white list domains *will be checked for threats*.
	Default value: *(not set)*

`Blacklist` *{domain list}*	List of domains that *can be used as the black list* (i.e. list of domains forbidden for connection for users, even if these domains are not included into blocked categories. In addition, user access will be forbidden to all sub-domains of domains indicated in this list.) *The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).* **Example:** Add to the list of domains `example.com` and `example.net`. 1. Adding of values to the configuration file. • Two values in one string

	`[ICAPD]`
	` Blacklist = "example.com",`
	`"example.net"`
	- Two strings (one value per a string)
	`[ICAPD]`
	` Blacklist = example.com`
	` Blacklist = example.net`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# `**`drweb-ctl`**` cfset ICAPD.Blacklist -a`
	`example.com`
	`# `**`drweb-ctl`**` cfset ICAPD.Blacklist -a`
	`example.net`
	Actual usage of the domain list indicated in this parameter depends on the *method* of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.
	The list of default rules (see below) guarantees that access to domains (and their sub-domains) from this list will be always forbidden. If this domain is simultaneously added to the lists `Whitelist` and `Blacklist`, the default rules guarantee that user access to it will be blocked.
	Default value: *(not set)*

`Adlist`
{list of strings}
	A list of regular expressions that describe advertisement URLs: URLs that match any of the regular expressions listed here are considered to be URLs of advertisements.
	The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add to the list the following expressions '`.*ads.+`' and '`.*/ad/.*\.gif$`'.
	1. Adding of values to the configuration file.
	- Two values in one string
	`[ICAPD]`
	` Adlist = ".*ads.+", ".*/ad/.*\.gif$"`

	- Two strings (one value per a string)
	`[ICAPD]`
	` Adlist = .*ads.+`
	` Adlist = .*/ad/.*\.gif$`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset ICAPD.Adlist -a '.*ads.+'`
	`# drweb-ctl cfset ICAPD.Adlist -a '.*/ad/.*\.gif$'`
	Regular expressions are specified using either the POSIX syntax (BRE, ERE) or the Perl syntax (PCRE, PCRE2).
	⚠ Actual usage of the expression list indicated in this parameter depends on the *method* of its usage in the management rules of access to web sources defined for Dr.Web ICAPD.
	The list of default rules (see below) guarantees that access to URL from this list will be always forbidden only if domains of these URLs are not in `Whitelist`.
	Default value: *(not set)*

`BlockInfectionSource` *{Boolean}*	Instructs to block attempted connections to websites containing malicious software (included into the *InfectionSource* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value:** `Yes`
`BlockNotRecommended` *{Boolean}*	Instructs to block attempts of connection to non-recommended websites (included into the *NotRecommended* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below):

	`url_category in "ICAPD.BlockCategory" :`
	`BLOCK as _match`
	Default value: `Yes`

`BlockAdultContent`
{Boolean}
	Instructs to block attempts of connection to websites containing adult content (included into the *AdultContent* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in "ICAPD.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockViolence`
{Boolean}
	Instructs to block attempts of connection to websites containing graphic violence (included into the *Violence* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in "ICAPD.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockWeapons`
{Boolean}
	Instructs to block attempts of connection to websites dedicated to weapons (included into the *Weapons* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in "ICAPD.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockGambling`
{Boolean}
	Instructs to block attempts of connection to gambling websites (included into the *Gambling* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in "ICAPD.BlockCategory" :`
	`BLOCK as _match`

	Default value: `No`
`BlockDrugs` *{Boolean}*	Instructs to block attempts of connection to websites dedicated to drugs (included into the *Drugs* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockObsceneLanguage` *{Boolean}*	Instructs to block attempts of connection to websites containing obscene language (included into the *ObsceneLanguage* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockChats` *{Boolean}*	Instructs to block attempts of connection to chat websites (included into the *Chats* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockTerrorism` *{Boolean}*	Instructs to block attempts of connection to websites dedicated to terrorism (included into the *Terrorism* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockFreeEmail` *{Boolean}*	Instructs to block attempts of connection to websites of free email services (included into the *FreeEmail* category).

	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockSocialNetworks` *{Boolean}*	Instructs to block attempts of connection to social networking websites (included into the *SocialNetworks* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value**: `No`
`BlockDueToCopyrightNotice` *{Boolean}*	Instructs to block attempts of connection to websites that were added according to copyright holder requests (included into the *DueToCopyrightNotice* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "ICAPD.BlockCategory" : BLOCK as _match` **Default value:** `Yes`
`ScanTimeout` *{time interval}*	Timeout for scanning one file initiated by Dr.Web ICAPD. *A value in the range from 1s to 1h can be specified* **Default value:** `30s`
`HeuristicAnalysis` *{On | Off}*	Indicates whether heuristic analysis is used for detection of unknown threats during data scanning initiated by Dr.Web ICAPD. The use of heuristic analysis raises the level of protection, but at the same time it increases the time spent on scanning. Action applied to threats detected by the heuristic analyzer is specified as the **`BlockSuspicious`** parameter value. **Allowed values:** • `On`—instructs to use heuristic analysis when scanning. • `Off`—instructs not to use heuristic analysis.

	Default value: `On`
`PackerMaxLevel` *{integer}*	Maximum nesting level when scanning packed objects. All objects at a deeper nesting level are skipped during data scanning initiated by Dr.Web ICAPD. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `8`
`ArchiveMaxLevel` *{integer}*	Maximum nesting level when scanning archives. All objects at a deeper nesting level are skipped during data scanning initiated by Dr.Web ICAPD. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `0`
`MailMaxLevel` *{integer}*	Maximum nesting level when scanning email messages and mailboxes. All objects at a deeper nesting level are skipped during data scanning initiated by Dr.Web ICAPD. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value :** `0`
`ContainerMaxLevel` *{integer}*	Maximum nesting level when scanning other containers (for example, HTML pages). All objects at a deeper nesting level are skipped during data scanning initiated by Dr.Web ICAPD. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `8`
`MaxCompressionRatio` *{integer}*	Maximum compression ratio of compressed/packed objects (ratio between the uncompressed size and the compressed size). If the ratio for an object exceeds the limit, this object is skipped during data scanning initiated by Dr.Web ICAPD. *The compression ratio must not be smaller than 2.* **Default value :** `500`
`BlockKnownVirus` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains any known threat. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below):

	`threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value**: `Yes`
`BlockSuspicious` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains any unknown threat (detected by the heuristic analyzer). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value**: `Yes`
`BlockAdware` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains adware. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value**: `Yes`
`BlockDialers` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains a dialer program. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value**: `Yes`
`BlockJokes` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains joke program. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value**: `No`

`BlockRiskware` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains riskware. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value:** `No`
`BlockHacktools` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains a hacktool. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "ICAPD.BlockThreat" : BLOCK as _match` **Default value:** `No`
`BlockUnchecked` *{Boolean}*	Instructs to block the receiving or the sending of data if it cannot be checked. **Default value:** `No`

Rules for Traffic Monitoring and Blocking of Access

In addition to the parameters listed above, section also contains seven *sets of rules* **`RuleSet*`** (**`RuleSet0`**, ..., **`RuleSet6`**) which control directly traffic scanning and blocking of access of the users to web resources and blocking downloading content from the Internet. For some values in conditions (for example, IP address ranges, lists of website categories, black and white lists of web sources, etc.), there is a substitution of values loaded from text files and also extracted from external data sources via LDAP (Dr.Web LookupD component is used). When configuring connections the whole list of rules is checked in the ascending order, until the rule containing the ultimate resolution is found. The gaps in the rule list are ignored.

The rules are described in detail in section Rules for Traffic Monitoring of Appendix D.

Viewing and editing of rules

For easy editing of the rules list gaps are left, i.e. **`RuleSet`**<*i*> sets that do not contain the rules. Note that you *cannot* add the items other than **`RuleSet0`**, ..., *RuleSet6*, but you can add and to remove any rule in any element of **`RuleSet`**<*i*>. Viewing and editing rules can be performed in any of the following ways:

- by viewing and editing the configuration file configuration file (in any text editor) (note that this file stores only those parameters which value is different from the default ones);

- via the web interface of the product management (if installed).
- via the command-line-based interface—Dr.Web Ctl (drweb-ctl `cfshow` **and drweb-ctl** `cfset` commands).

> If you edited the rules and made changes in the configuration file, in order to apply these changes, restart the program. To do that, use the **drweb-ctl** `reload` command.

Use of the command **drweb-ctl** `cfshow` to view rules.

To view the contents of the rules set **`ICAPD.RuleSet1`**, use the command

```
# drweb-ctl cfshow ICAPD.RuleSet1
```

The use of the **drweb-ctl** `cfset` command to edit the rules (hereinafter the <*rule*>—text of the rule).

- Replacing all the rules in a set **`ICAPD.RuleSet1`** with a new rule:

```
# drweb-ctl cfset ICAPD.RuleSet1 '<rule>'
```

- Adding a new rule to the rule set **`ICAPD.RuleSet1`**:

```
# drweb-ctl cfset -a ICAPD.RuleSet1 '<rule>'
```

- Removing a specific rule from the set **`ICAPD.RuleSet1`**:

```
# drweb-ctl cfset -e ICAPD.RuleSet1 '<rule>'
```

- Reset the rule set **`ICAPD.RuleSet1`** to the default state:

```
# drweb-ctl cfset -r ICAPD.RuleSet1
```

When you use the **drweb-ctl** tool to edit the list of rules, enclose the text of your added rule into single or double quotes, and use backward slashes ('\') as escape characters before any double quotes within the text of the rule—if the text of the rule itself happens to contain double quotes.

It is important to remember the following storage features of rules in **`RuleSet`**<*i*> variables of the configuration:

- The conditional part and colon can be omitted when adding unconditional rules. However, such rules are always stored in the list of rules as a string "` : `<*action*>";
- When adding rules that contain several actions (such as rules as '<*condition*>` : `<*action 1*>`, `<*action 2*>'), such rules will be modified into a chain of elementary rules '<*condition*>` : `<*action 1*>' and '<*condition*>` : `<*action 2*>'.
- The logging or rules does not allow for disjunction (logical "OR") of conditions in the conditional part, so, in order to implement the logical "OR", the chain of rules should be logged with each rule having a disjunct-condition in its condition.

To add an unconditional rule for skipping the connections (the *PASS* action) to the **ICAPD.RuleSet1** set, you only need to execute the following command:

```
# drweb-ctl cfset -a ICAPD.RuleSet1 'PASS'
```

However, to remove this rule from the specified rule set, it is required to execute the following command:

```
# drweb-ctl cfset -e ICAPD.RuleSet1 ' : PASS'
```

To add the **ICAPD.RuleSet1** rule to the rule set that changes a path to standard templates for connections from unresolved addresses and performs blocking, it is necessary to execute the following command:

```
# drweb-ctl cfset -a ICAPD.RuleSet1 'src_ip not in file("/etc/trusted_ip") :
set http_template_dir = "mytemplates", BLOCK'
```

However, this command will add *two rules* to the specified set, so, in order to remove them from the set of rules, you need to execute two following commands:

```
# drweb-ctl cfset -e ICAPD.RuleSet1 'src_ip not in file("/etc/trusted_ip") :
set http_template_dir = "mytemplates"'
# drweb-ctl cfset -e ICAPD.RuleSet1 'src_ip not in file("/etc/trusted_ip") :
BLOCK'
```

To add to the **ICAPD.RuleSet1** rule set such rule as “Block if a malicious object *KnownVirus* or URL from the category *Terrorism* are detected”, it is necessary to add the following two rules to this rule set:

```
# drweb-ctl cfset -a ICAPD.RuleSet1 'threat_category in (KnownVirus) : BLOCK
as _match'
# drweb-ctl cfset -a ICAPD.RuleSet1 'url_category in (Terrorism) : BLOCK as
_match'
```

To remove them from the set of rules, you also need to execute two commands, as it is shown in the example above.

Default set of rules

By default, the following sets of rules are specified:

```
RuleSet0 =
RuleSet1 = direction request, url_host in "ICAPD.Blacklist" : BLOCK as
BlackList
RuleSet1 = direction request, url_host not in "ICAPD.Whitelist", url match
"ICAPD.Adlist" : BLOCK as BlackList
RuleSet2 =
RuleSet3 = direction request, url_host not in "ICAPD.Whitelist",
url_category in "ICAPD.BlockCategory" : BLOCK as _match
RuleSet4 =
RuleSet5 = threat_category in "ICAPD.BlockThreat" : BLOCK as _match
RuleSet6 =
```

The first two rules (out of those preset by default) process outgoing HTTP connections: if a host (or a URL) to which a connection is attempted is included into the black list, the connection will be blocked on the basis of the black list. Other checks are not performed. If a host (a URL) is not included into the white list and belongs to any website category marked as unwanted for access, or matches any of the regular expressions that describe advertisement URLs, then the connection is blocked because the URL belongs to an unwanted category.

The rule specified in the **`RuleSet5`** checks whether the HTTP request or response contain any threats that belong to a threat category that must be blocked (according to the settings). If there are such threats, the connection will be blocked on the basis of detecting a threat. Note that because the `direction` condition is not specified, by default both client requests (*request*) and server responses (*response*) are checked.

Examples of Rules for Traffic Monitoring and Blocking of Access

1. Allow users with the following IP addresses *10.10.0.0 – 10.10.0.254* to access websites of all categories, except *Chats*:

   ```
   src_ip in (10.10.0.0/24), url_category not in (Chats) : PASS
   ```

   Note that if the rule

   ```
   url_host in "ICAPD.Blacklist" : BLOCK as BlackList
   ```

   is allocated in the list of rules above the indicated rule, then access to domains from the black list, i.e. domains listed in the parameter `ICAPD.Blacklist`, will also be blocked for users with the range of IP addresses *10.10.0.0 – 10.10.0.254*. And if this rule is allocated below, users with the range of IP addresses *10.10.0.0 – 10.10.0.254* will get access also to websites from the black list.

   Due to the fact that resolution `PASS` is terminal, no more rules are checked, therefore scanning of the downloaded data for viruses is not performed either. To grant users with the range of IP addresses *10.10.0.0 – 10.10.0.254* access to websites of all categories, except *Chats* if they are not in the black list, and to block download of threats at the same time, use the following rule:

   ```
   url_category not in (Chats), url_host not in "ICAPD.Blacklist",
   threat_category not in "ICAPD.BlockCategory" : PASS
   ```

2. Do not perform scanning of contents of video files (i.e. data with the type MIME "`video/*`", where `*` is any type of the MIME class `video`):

   ```
   content_type in ("video/*") : PASS
   ```

Integration with HTTP Squid Proxy Server

Integration of Dr.Web ICAPD with the HTTP proxy server **Squid** is described in section Integration with Squid Proxy Server.

SpIDer Gate

⚠ This component is included only in the product distributions for **GNU/Linux** OSes.

The component for monitoring network traffic and URLs SpIDer Gate is designed to check data (downloaded from the network to the local computer and to the network from the local host) for threats and to prevent connections with the network hosts, included to the unwanted categories of web resources and to the black lists defined by the administrator.

In the component settings there is an opportunity to indicate types of protocols for scanning.

To check whether an URL belongs to any of the categories (used for scanning of connections that utilize the HTTP/HTTPS protocol), the component not only uses the database of web resource categories, which is updated regularly from Doctor Web's update servers, but also refers to the Dr.Web Cloud service. Doctor Web keeps track of the following web resources categories:

- *InfectionSource*—websites containing malicious software ("infection sources").
- *NotRecommended*—fraudulent websites (that use "social engineering") visiting which is not recommended.
- *AdultContent*—websites that contain pornographic or erotic materials, dating sites, etc.
- *Violence*—websites that encourage violence or contain materials about various fatal accidents, etc.
- *Weapons*—websites that describe weapons and explosives or provide information on their manufacturing.
- *Gambling*—websites that provide access to online games of chance, casinos, auctions, including sites for placing bets, etc.
- *Drugs*—websites that promote use, production or distribution of drugs, etc.
- *ObsceneLanguage*—websites that contain the obscene language (in titles, articles, etc.).
- *Chats*—websites that offer a real-time transmission of text messages.
- *Terrorism*—websites that contain aggressive and propaganda materials or terroristic attacks descriptions, etc.
- *FreeEmail*—websites that offer the possibility of free registration of a web mailbox.
- *SocialNetworks*—different social networking services: general, professional, corporate, interest-based; thematic dating sites.
- *DueToCopyrightNotice*—websites that were specified by the holders of copyrights pertaining to content or works protected by copyright law (movies, music, etc.).

System administrator can specify the hosts accessing which is unwanted, based on the categories to which the hosts belong. Additionally, a user can configure one's own black lists to block the access to the necessary hosts, and white lists, to allow the access. The access to the hosts of white lists will be allowed, even if the hosts belong to the unwanted categories. If there is no information about URLs in the local black lists and database of web resources categories, the component can refer to Dr.Web Cloud service to check for the information whether these URLs are malicious, which is received from other Dr.Web products on a real-time basis.

One and the same website can belong simultaneously to several categories. Access to such website is blocked even if it belongs to any of the unwanted categories.

Even if the website is included to the white list, data (sent and downloaded from the website) is checked for threats.

In case of high intensity of the scanning of files transferred via the HTTP protocol, there is a possibility of having problems with scanning due to depletion of the number of available file descriptors by the Dr.Web Network Checker component. In this case, it is necessary to increase the limit of the number of file descriptors available to Dr.Web for UNIX Internet Gateways.

Within server products Dr.Web for UNIX Internet Gateways can be used in an organization to create a “wall” between the company’s server, for example, the web server with public access, and the Internet, an external network, because by default the Dr.Web ICAPD component that performs functions of controlling user access. This component operates together with the proxy server providing Internet access from the local network.

Operating Principles

The SpIDer Gate component monitors network connections established by user applications. The component checks whether the server which the client application is trying to connect to belongs to any of the web resources categories specified in the settings as unwanted. Moreover, the component can refer to Dr.Web Cloud service to check a URL. If the URL belongs to any of the unwanted categories (including that one which was returned by the request of Dr.Web Cloud service) or to a black list defined by the system administrator, the connection is interrupted, and the HTML page, containing the message that the access is not allowed, is shown (in case of HTTP/HTTPS connection). The HTML page is generated by SpIDer Gate according to the template supplied with the component. This page contains details upon the block. The similar page is displayed to the client if SpIDer Gate finds a threat that must be blocked in the contents of the server response. If the connection uses a protocol different from HTTP(S), the component scans only for permission to establish connection with this server.

Auxiliary component Dr.Web Firewall for Linux redirects connections with remote servers, which are established by the client applications. The component performs dynamic control of the **NetFilter** rules of **GNU/Linux** system component. The operation scheme for the component of monitoring network traffic and URLs is shown in the figure below.

Within Dr.Web for UNIX Internet Gateways server products a client application is a protected server resource of the company, (for example, a web server with public access), because by default the Dr.Web ICAPD component performs functions of managing access of the local network users user to the Internet. This component operates together with the proxy-server providing Internet access from the local network.

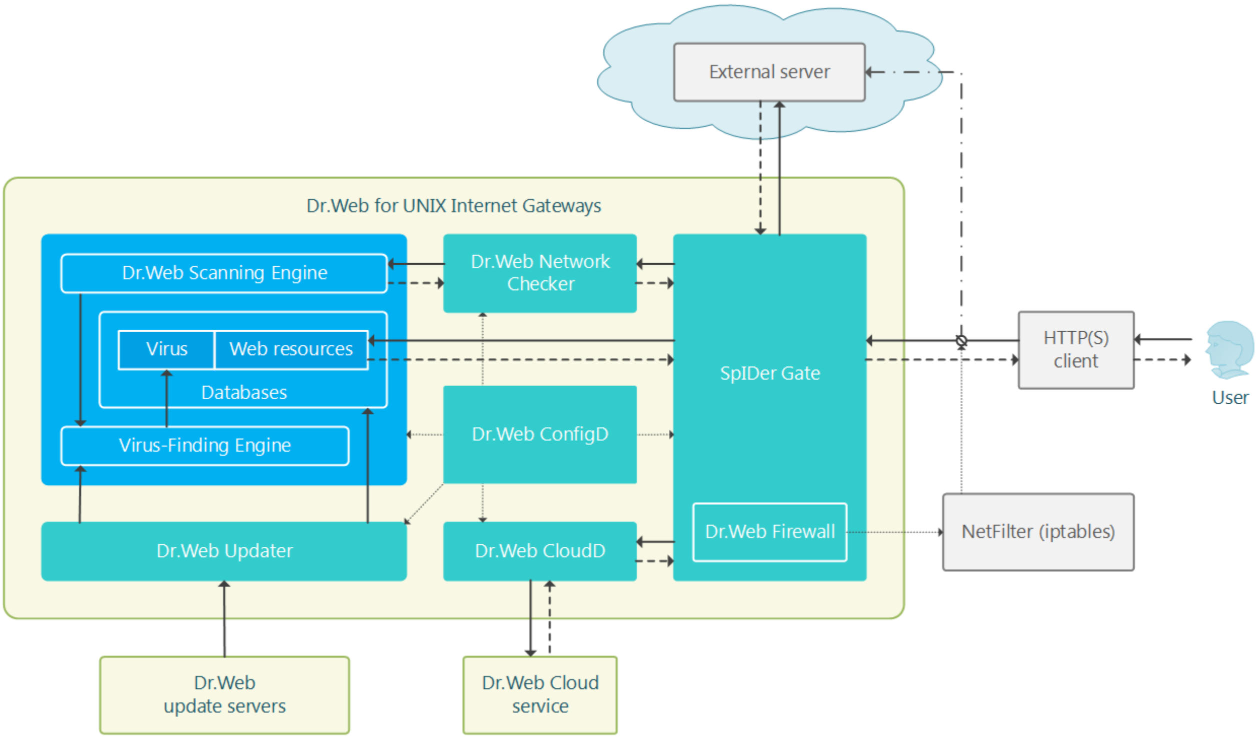


Figure 12. Diagram of the components' operation

The Dr.Web Updater component is used to regularly and automatically update the databases of web resource categories from Doctor Web update servers. The same component is used to update virus databases for the Dr.Web Scanning Engine scanning engine. The Dr.Web CloudD component is used to refer to Dr.Web Cloud service (using of the cloud service is configured in Appendixes common settings and can be disabled, if necessary). To check transferred data, SpIDer Gate uses the Dr.Web Network Checker component. The latter one initiates scanning via the Dr.Web Scanning Engine scanning engine.

Command-Line Arguments

To run SpIDer Gate, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-gated [<parameters>]
```

SpIDer Gate can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.

`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-gated --help
```

This command outputs short help information on SpIDer Gate.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

> To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-gated`

Configuration Parameters

The component uses configuration parameters which are specified in the `[GateD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-gated` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-gated` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-gated`

`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value:** `drweb`
`IdleTimeLimit` *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down. Minimum value—`10s`. **Default value:** `30s`
`TemplatesDir` *{path to directory}*	Path to the directory that contains the templates for the HTML notifications sent upon blocking a web resource. **Default value:** <*var_dir*>`/templates/gated` • For **Linux**, **Solaris**: `/var/opt/drweb.com/templates/gated` • For **FreeBSD**: `/var/drweb.com/templates/gated`
`CaPath` *{path}*	Path to the directory or file with system list of trusted root certificates. **Default value:** *Path to the list of trusted certificates. The path depends on your* **GNU/Linux** *distribution:* • *For* **Astra Linux**, **Debian**, **Linux Mint**, **SUSE Linux** *and* **Ubuntu**, *usually it is a path* `/etc/ssl/certs/`*;* • *For* **CentOS** *and* **Fedora***—a path* `/etc/pki/tls/certs/ca-bundle.crt`. • *For other distributions a path can be defined through results of execution of the command* **openssl** `version -d`. • *If a command is unavailable or an OS distribution could not be identified , the value* `/etc/ssl/certs/` *is used.*

Changes made to the settings of the connection scanning do not influence the scanning of connections that have already been established by the applications before making changes.

Specify more particular parameters of traffic monitoring in the settings of the auxiliary component Dr.Web Firewall for Linux.

Dr.Web Firewall for Linux

This component is included only in the distributions for **GNU/Linux** OS.

For the correct operation of the component, OS kernel must be built with inclusion of the following options:

- *CONFIG_NETLINK_DIAG, CONFIG_INET_TCP_DIAG;*
- *CONFIG_NF_CONNTRACK_IPV4, CONFIG_NF_CONNTRACK_IPV6, CONFIG_NF_CONNTRACK_EVENTS;*
- *CONFIG_NETFILTER_NETLINK_QUEUE, CONFIG_NETFILTER_NETLINK_QUEUE_CT, CONFIG_NETFILTER_XT_MARK.*

The set of required options from the specified list can depend on the used distribution kit of **GNU/Linux**.

Dr.Web Firewall for Linuxis an auxiliary component. It performs function of a connection manager for SpIDer Gate. Dr.Web Firewall for Linux ensures that the host connections go through SpIDer Gate so that the connection traffic is monitored.

Operating Principles

Dr.Web Firewall for Linux component ensures the correct SpIDer Gate operation. It analyzes the routing rules adjusted for **NetFilter** (**GNU/Linux** OS component) and modifies it so as the established connections are redirected to SpIDer Gate which performs a function of an intermediate (proxy) between a client application and a remote server.

Dr.Web Firewall for Linux can separately manage the rules of redirection of outgoing and incoming connections.

Command-Line Arguments

To run Dr.Web Firewall for Linux connection manager from the command line, type the following command:

```
$ <opt_dir>/bin/drweb-firewall [<parameters>]
```

Dr.Web Firewall for Linux can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h`

	Arguments: None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-filecheck --help
```

This command outputs short help information on Dr.Web Firewall for Linux.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

> ! To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-firewall`

Configuration Parameters

The component uses configuration parameters which are specified in the `[LinuxFirewall]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-firewall`

	• For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-firewall` • For **FreeBSD** : `/usr/local/libexec/drweb.com/bin/drweb-firewall`
`XtablesLockPath` *{path to file}*	Path to the **iptables** (**NetFilter**) table blocking file. If the parameter value is not specified, the `/run/xtables.lock` and `/var/run/xtables.lock` paths are checked. If the file is not found in the specified path or default paths, when launching the component, an error occurs. **Default value:** *(not set)*
`InspectHttp` *{On | Off}*	Instructs whether to check the data transferred over the HTTP protocol. *Real data scanning will be performed according to the indicated scanning rules (see below).* **Default value:** `On`
`InspectSmtp` *{On | Off}*	Parameter is not used. **Default value** : `Off`
`InspectPop3` *{On | Off}*	Parameter is not used. **Default value** : `Off`
`InspectImap` *{On | Off}*	Parameter is not used. **Default value** : `Off`

`InputDivert`
{Off | Auto(interface:<i_name> protected:<p_list>)}
	Defines the used method of diverting incoming connections (redirecting it to the SpIDer Gate checking component).
	Allowed values:
	• `Off`—redirecting of incoming connections is disabled.
	• `Auto(interface:`*<i_name>* `protected:`*<p_list>*`)`—redirection of incoming connections in automatic mode. Rules are controlled by Dr.Web Firewall for Linux. Connections that comes via the specified network interface *<i_name>* into the *<p_list>* port list are monitored. Port numbers in the *<p_list>* list are separated by commas. For example,

	`Auto(interface:eth0 protected:80,8080).` **Default value:** `Off`
`OutputDivert` *{Off | Auto}*	Defines the used method of diverting outgoing connections (redirecting it to the SpIDer Gate checking component). **Allowed values:** • `Off`—redirecting of outgoing connections is disabled. • `Auto`—redirection of outgoing connections in automatic mode. Dr.Web Firewall for Linux manages the rules. **Default value:** `Auto`

`ExcludedProc`
 	
{path to file}
	The list of processes *which can be used as the white list of processes*, i.e. list of the processes whose network activity must not be monitored.
	
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	
	Example: Add to the list of processes **wget** and **curl**.
	
	1. Adding of values to the configuration file.
	• Two values in one string
	`[LinuxFirewall]`
	` ExcludedProc = "/usr/bin/wget", "/usr/bin/curl"`
	• Two strings (one value per a string)
	`[LinuxFirewall]`
	` ExcludedProc = /usr/bin/wget`
	` ExcludedProc = /usr/bin/curl`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset LinuxFirewall.ExcludedProc -a /usr/bin/wget`
	`# drweb-ctl cfset LinuxFirewall.ExcludedProc -a /usr/bin/curl`

	⚠ Actual usage of the process list indicated in this parameter depends on the *method* of its usage in the scanning rules defined for Dr.Web Firewall for Linux. The list of default rules (see below) guarantees that traffic of all processes from the list is allowed *without any scanning*. **Default value:** *(not set)*
`SniCheckAddress` *{Boolean}*	Instructs to check the SNI host to which you are trying to connect to the SSL handshake stage, "check is listed in the black list or belongs to the blocked categories, is performed without unwrapping the SSL. In the current realization, the value of this variable does not influence the processing of protected traffic. To control such processing, it is necessary to create a rule containing the `sni_host in` and `sni_category in` conditions (see below). If you change the value of this parameter with the help of the cfset command of the **drweb-ctl** utility or with the help of the web interface, the affected dependent rules will adapt automatically. **Default value**: `No`
`UnwrapSsl` *{Boolean}*	Instructs to check encrypted traffic transferred via the SSL/TLS connections.

	In the current realization, the value if this variable does not influence processing of protected traffic. To control processing, it is necessary to create a rule containing the `SET Unwrap_SSL = true/false` action (see below). If you change the value of this parameter with the help of the cfset command of the **drweb-ctl** utility or with the help of the web interface, the affected dependent rules will adapt automatically. **Default value:** `No`
`HttpSafeSearch` *{Boolean}*	Instructs to use the “Safe search” option for searching engines that support this mode. **Default value:** `No`
`BlockInfectionSource` *{Boolean}*	Instructs to block attempted connections to websites containing malicious software (included into the *InfectionSource* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match` **Default value:** `Yes`
`BlockNotRecommended` *{Boolean}*	Instructs to block attempts of connection to non-recommended websites (included into the *NotRecommended* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match` **Default value:** `Yes`

`BlockAdultContent`
{Boolean}
	Instructs to block attempts of connection to websites containing adult content (included into the *AdultContent* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockViolence`
{Boolean}
	Instructs to block attempts of connection to websites containing graphic violence (included into the *Violence* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockWeapons`
{Boolean}
	Instructs to block attempts of connection to websites dedicated to weapons (included into the *Weapons* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockGambling`
{Boolean}
	Instructs to block attempts of connection to gambling websites (included into the *Gambling* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`

	Default value: `No`
`BlockDrugs` *{Boolean}*	Instructs to block attempts of connection to websites dedicated to drugs (included into the *Drugs* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match` **Default value:** `No`
`BlockObsceneLanguage` *{Boolean}*	Instructs to block attempts of connection to websites containing obscene language (included into the *ObsceneLanguage* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match` **Default value:** `No`
`BlockChats` *{Boolean}*	Instructs to block attempts of connection to chat websites (included into the *Chats* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match` **Default value:** `No`
`BlockTerrorism` *{Boolean}*	Instructs to block attempts of connection to websites dedicated to terrorism (included into the *Terrorism* category). *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match`

	Default value: `No`

`BlockFreeEmail`
{Boolean}
	Instructs to block attempts of connection to websites of free email services (included into the *FreeEmail* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockSocialNetworks`
{Boolean}
	Instructs to block attempts of connection to social networking websites (included into the *SocialNetworks* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`BlockDueToCopyrightNotice`
{Boolean}
	Instructs to block attempts of connection to websites that were added according to copyright holder requests (included into the *DueToCopyrightNotice* category).
	For the blocking to work, you should check that within the settings there is also a rule that looks like this (see the details below):
	`url_category in`
	`"LinuxFirewall.BlockCategory" :`
	`BLOCK as _match`
	Default value: `No`

`Whitelist` *{domain list}*	List of domains that *can be used as the white list* (i.e. list of domains allowed for connection for users, even if these domains are included into blocked categories. In addition, user access will be allowed to all sub-domains of domains indicated in this list.) *The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the*

section (in this case, all its values are combined into one list).

Example: Add to the list of domains `example.com` and `example.net`.

1. Adding of values to the configuration file.

   - Two values in one string

   ```
   [LinuxFirewall]
    Whitelist = "example.com",
   "example.net"
   ```

   - Two strings (one value per a string)

   ```
   [LinuxFirewall]
    Whitelist = example.com
    Whitelist = example.net
   ```

2. Adding values via the command **drweb-ctl** `cfset`.

   ```
   # drweb-ctl cfset
   LinuxFirewall.Whitelist -a
   example.com
   # drweb-ctl cfset
   LinuxFirewall.Whitelist -a
   example.net
   ```

<table>
<tr>
<td></td>
<td>

⚠ Actual usage of the domain list indicated in this parameter depends on the *method* of its usage in the scanning rules defined for Dr.Web Firewall for Linux.

The list of default rules (see below) guarantees that access to domains (and their sub domains) from this list will be provided even if it contains domains from the list of blocked web source categories but only in case of a request to a server via the HTTP protocol. Besides, this default set of rules guarantees that data downloaded from the white list domains *will be checked for threats* (due to the fact that data is returned in a response, and a variable `direction` has a value `response`).

Default value: *(not set)*

</td>
</tr>
<tr>
<td>

`Blacklist`

{domain list}

</td>
<td>

List of domains that *can be used as the black list* (i.e. list of domains forbidden for connection for users, even if these domains are not included into blocked categories. In addition, user access will be forbidden to all sub-domains of domains indicated in this list.)

The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).

Example: Add to the list of domains `example.com` and `example.net`.

1. Adding of values to the configuration file.

   - Two values in one string

```
[LinuxFirewall]
 Blacklist = "example.com",
"example.net"
```

   - Two strings (one value per a string)

</td>
</tr>
</table>

	`[LinuxFirewall]`
	` Blacklist = example.com`
	` Blacklist = example.net`
	
	2. Adding values via the command **drweb-ctl** `cfset`.
	
	`# `**`drweb-ctl`**` cfset LinuxFirewall.Blacklist -a example.com`
	`# `**`drweb-ctl`**` cfset LinuxFirewall.Blacklist -a example.net`
	
	⚠ Actual usage of the domain list indicated in this parameter depends on the *method* of its usage in the scanning rules defined for Dr.Web Firewall for Linux.
	
	The list of default rules (see below) guarantees that access to domains (and their sub-domains) from this list will be always forbidden over the HTTP protocol. If this domain is simultaneously added to the lists `Whitelist` and `Blacklist`, the default rules guarantee that user access to it will be blocked.
	
	Default value: *(not set)*

`ScanTimeout` *{time interval}*	Timeout for scanning one file initiated by SpIDer Gate. *A value in the range from 1s to 1h can be specified* **Default value:** `30s`
`HeuristicAnalysis` *{On | Off}*	Indicates whether heuristic analysis is used for detection of unknown threats during file scanning initiated by SpIDer Gate. Heuristic analysis provides higher detection reliability but, at the same time, it increases time of virus scanning. *Action applied to threats detected by the heuristic analyzer is specified as the* **`BlockSuspicious`** *parameter value.*

	Allowed values: - `On`—instructs to use heuristic analysis when scanning. - `Off`—instructs not to use heuristic analysis. **Default value:** `On`
`PackerMaxLevel` *{integer}*	Maximum nesting level when scanning packed objects. All objects at a deeper nesting level are skipped during file scanning initiated by SpIDer Guard. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `8`
`ArchiveMaxLevel` *{integer}*	Maximum nesting level when scanning archives. All objects at a deeper nesting level are skipped during file scanning initiated by SpIDer Gate. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value :** `8`
`MailMaxLevel` *{integer}*	Maximum nesting level when scanning email messages and mailboxes. All objects at a deeper nesting level are skipped during file scanning initiated by SpIDer Gate. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `8`
`ContainerMaxLevel` *{integer}*	Maximum nesting level when scanning other containers (for example, HTML pages). All objects at a deeper nesting level are skipped during file scanning initiated by SpIDer Gate. *A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.* **Default value:** `8`
`MaxCompressionRatio` *{integer}*	Maximum compression ratio of compressed/packed objects (ratio between the uncompressed size and the compressed size). If the ratio of an object exceeds the limit, this object will be skipped during file scanning procedures initiated by SpIDer Gate. *The compression ratio must not be smaller than 2.* **Default value :** `500`

`BlockKnownVirus` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains any known threat. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value:** `Yes`
`BlockSuspicious` *{Boolean}*	Instructs to block receiving or sending data if it contains any unknown threat detected by the heuristic analyzer. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value:** `Yes`
`BlockAdware` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains adware. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value:** `Yes`
`BlockDialers` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains a dialer program. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value:** `Yes`

`BlockJokes` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains joke program. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value**: `No`
`BlockRiskware` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains riskware. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value**: `No`
`BlockHacktools` *{Boolean}*	Instructs to block the receiving or the sending of data if it contains a hacktool. *For the blocking to work, you should check that within the settings there is also a rule that looks like this* (see the details below): `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match` **Default value**: `No`
`BlockUnchecked` *{Boolean}*	Instructs to block the receiving or the sending of data if it cannot be checked. The value of this parameter influences processing of the rules that are impossible to evaluate to true or false because of an error. If `No` is specified, the rule is skipped as the rule that has not been executed. If `Yes` is specified, the `BLOCK as BlackList` action is performed.

	Default value: `No`

Changes made to the settings of the connection scanning do not influence the scanning of connections that have already been established by the applications before making changes. If it is required to apply them to already running applications, it is necessary to force them to disconnect and then connect again, for example, by rebooting these applications.

Rules for Traffic Monitoring and Blocking of Access

In addition to the parameters listed above, section also contains eleven *sets of rules* **`RuleSet*`** (**`RuleSet0`**, ..., **`RuleSet10`**) which control directly traffic scanning and blocking of access of the users to web resources and blocking downloading content from the Internet. For some values in conditions (for example, IP address ranges, lists of website categories, black and white lists of web sources, etc.), there is a substitution of values loaded from text files and also extracted from external data sources via LDAP (Dr.Web LookupD component is used). When configuring connections the whole list of rules is checked in the ascending order, until the rule containing the ultimate resolution is found. The gaps in the rule list are ignored.

The rules are described in detail in section Rules for Traffic Monitoring of Appendix D.

Viewing and editing of rules

For easy editing of the rules list gaps are left, i.e. **`RuleSet`**<*i*> sets that do not contain the rules. Note that you *cannot* add the items other than **`RuleSet0`**, ..., *RuleSet6*, but you can add and to remove any rule in any element of **`RuleSet`**<*i*>. Viewing and editing rules can be performed in any of the following ways:

- by viewing and editing the configuration file configuration file (in any text editor) (note that this file stores only those parameters which value is different from the default ones);
- via the web interface of the product management (if installed).
- via the command-line-based interface—Dr.Web Ctl (drweb-ctl `cfshow` **and drweb-ctl** `cfset` commands).

If you edited the rules and made changes in the configuration file, in order to apply these changes, restart the program. To do that, use the **drweb-ctl** `reload` command.

Use of the command **drweb-ctl** `cfshow` to view rules.

To view the contents of the rules set **`LinuxFirewall.RuleSet1`**, use the command

```
# drweb-ctl cfshow LinuxFirewall.RuleSet1
```

The use of the **drweb-ctl** `cfset` command to edit the rules (hereinafter the <*rule*>—text of the rule).

- Replacing all the rules in a set **`LinuxFirewall.RuleSet1`** with a new rule:

```
# drweb-ctl cfset LinuxFirewall.RuleSet1 '<rule>'
```

- Adding a new rule to the rule set **`LinuxFirewall.RuleSet1`**:

```
# drweb-ctl cfset -a LinuxFirewall.RuleSet1 '<rule>'
```

- Removing a specific rule from the set **`LinuxFirewall.RuleSet1`**:

```
# drweb-ctl cfset -e LinuxFirewall.RuleSet1 '<rule>'
```

- Reset the rule set **`LinuxFirewall.RuleSet1`** to the default state:

```
# drweb-ctl cfset -r LinuxFirewall.RuleSet1
```

When you use the **drweb-ctl** tool to edit the list of rules, enclose the text of your added rule into single or double quotes, and use backward slashes ('\') as escape characters before any double quotes within the text of the rule—if the text of the rule itself happens to contain double quotes.

It is important to remember the following storage features of rules in **`RuleSet`**<*i*> variables of the configuration:

- The conditional part and colon can be omitted when adding unconditional rules. However, such rules are always stored in the list of rules as a string " `:` <*action*>";
- When adding rules that contain several actions (such rules as '<*condition*> `:` <*action 1*>`,` <*action 2*>'), such rules will be modified into a chain of elementary rules '<*condition*> `:` <*action 1*>' and '<*condition*> `:` <*action 2*>'.
- The logging or rules does not allow for disjunction (logical "OR") of conditions in the conditional part, so, in order to implement the logical "OR", the chain of rules should be logged with each rule having a disjunct-condition in its condition.

To add an unconditional rule for skipping the connections (the *PASS* action) to the **`LinuxFirewall.RuleSet1`** set, you only need to execute the following command:

```
# drweb-ctl cfset -a LinuxFirewall.RuleSet1 'PASS'
```

However, to remove this rule from the specified rule set, it is required to execute the following command:

```
# drweb-ctl cfset -e LinuxFirewall.RuleSet1 ' : PASS'
```

To add the **LinuxFirewall.RuleSet1** rule to the rule set that changes a path to standard templates for connections from unresolved addresses and performs blocking, it is necessary to execute the following command:

```
# drweb-ctl cfset -a LinuxFirewall.RuleSet1 'src_ip not in
file("/etc/trusted_ip") : set http_template_dir = "mytemplates", BLOCK'
```

However, this command will add *two rules* to the specified set, so, in order to remove them from the set of rules, you need to execute two following commands:

```
# drweb-ctl cfset -e LinuxFirewall.RuleSet1 'src_ip not in
file("/etc/trusted_ip") : set http_template_dir = "mytemplates"'
# drweb-ctl cfset -e LinuxFirewall.RuleSet1 'src_ip not in
file("/etc/trusted_ip") : BLOCK'
```

To add to the **LinuxFirewall.RuleSet1** rule set such rule as “Block if a malicious object *KnownVirus* or URL from the category *Terrorism* are detected”, it is necessary to add the following two rules to this rule set:

```
# drweb-ctl cfset -a LinuxFirewall.RuleSet1 'threat_category in (KnownVirus)
: BLOCK as _match'
# drweb-ctl cfset -a LinuxFirewall.RuleSet1 'url_category in (Terrorism) :
BLOCK as _match'
```

To remove them from the set of rules, you also need to execute two commands, as it is shown in the example above.

Default set of rules

By default, the following sets of rules are specified:

```
RuleSet0 =
RuleSet1 = divert output : set HttpTemplatesDir = "output"
RuleSet1 = divert output : set MailTemplatesDir = "firewall"
RuleSet1 = divert input : set HttpTemplatesDir = "input"
RuleSet1 = divert input : set MailTemplatesDir = "server"
RuleSet1 = proc in "LinuxFirewall.ExcludedProc" : PASS
RuleSet1 =  : set Unwrap_SSL = false
RuleSet2 =
RuleSet3 =
RuleSet4 =
RuleSet5 = protocol in (Http), direction request, url_host in
"LinuxFirewall.Blacklist" : BLOCK as BlackList
RuleSet5 = protocol in (Http), direction request, url_host in
"LinuxFirewall.Whitelist" : PASS
RuleSet6 =
RuleSet7 = protocol in (Http), direction request, url_category in
"LinuxFirewall.BlockCategory" : BLOCK as _match
RuleSet8 =
RuleSet9 = protocol in (Http), divert input, direction request,
threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match
RuleSet9 = protocol in (Http), direction response, threat_category in
"LinuxFirewall.BlockThreat" : BLOCK as _match
```

```
RuleSet9 = protocol in (Smtp), threat_category in
"LinuxFirewall.BlockThreat" : REJECT
RuleSet9 = protocol in (Smtp), url_category in "LinuxFirewall.BlockCategory"
: REJECT
RuleSet9 = protocol in (Smtp), total_spam_score gt 0.80 : REJECT
RuleSet9 = protocol in (Pop3, Imap), threat_category in
"LinuxFirewall.BlockThreat" : REPACK as _match
RuleSet9 = protocol in (Pop3, Imap), url_category in
"LinuxFirewall.BlockCategory" : REPACK as _match
RuleSet9 = protocol in (Pop3, Imap), total_spam_score gt 0.80 : REPACK as
_match
RuleSet10 =
```

The first rule indicates that if the connection is established by the process specified in the **ExcludedProc** parameter (see above), the connection is skipped without checking any other conditions. The next rule (is executed without any condition) blocks unwrapping of protected connections. This rule and all those that are situated below are considered only if a connection is not bound with the excluded process. Moreover, as all subsequent rules depend on the protocol, if unwrapping of protected connections is disabled, the rules are not executed because it is impossible to define whether the conditions evaluate to true.

The following rules are dedicated to the processing of the outgoing HTTP connections:

1. If a host with which a connection is established is included in a black list, the connection is blocked because the host is in the black list. Other checks are not performed.
2. If the host is included in a white list, the connection is skipped, and other check are not performed.
3. If the URL requested by the client is in the categories of web resources marked as unwanted for access, the connection is blocked due to the detection of a threat. Other checks are not performed.
4. If the response received from a remote host has threats via HTTP contains a threat belonging to the blocked categories, the connection is blocked because the threat was detected. Other checks are not performed.
5. If the data transferred from the local host to a remote host contains a threat belonging to the blocked categories, the connection is blocked because the threat was detected. Other checks are not performed.

These five rules will work only if `On` is specified in the **InspectHttp** parameter. Otherwise, none of these rules work.

The following six rules that are specified in the **RuleSet9** control the scanning of the data that is sent and received via email protocols; these rules are activated if it is detected that a transmitted email (over SMTP, POP3 or IMAP protocol) contains attachments or URLs belonging to the categories that should be blocked or qualified as spam (with the reliability rating not less than 0,8). If an email is transmitted over the SMTP protocol, the transmission (i.e. sending or receipt) of the email will be blocked, whereas for the IMAP and POP3 protocols the email will be processed to remove malicious content from its contents (“repackaging”).

> If the component for email message scanning for signs of spam Dr.Web ASE is unavailable, then email message scanning for signs of spam is not performed. In this case, rules that contain scanning of spam level (value total_spam_score) are unavailable.

Note that email processing rules are executed only if On is specified for the corresponding **Inspect**<*EmailProtocol*> parameters. Otherwise, none of these rules are executed. Moreover, the Dr.Web MailD component for email scanning should be installed for examination of a transmitted email for malware attachments. If the component is not installed, transmitted email will be blocked because of the error *"Unable to check"*. To allow transmitting messages that cannot be checked, set the **BlockUnchecked** = No parameter (see above). Moreover, if the email scanning component is not installed, it is recommended to specify No for the InspectSmtp**,** **InspectPop3, and InspectImap** parameters.

> Note that the set of default rules can change automatically if the values of the **SniCheckAddress** and **UnwrapSsl** parameters are changed.

Examples of Rules for Traffic Monitoring and Blocking of Access

1. Allow users with the following IP addresses *10.10.0.0 – 10.10.0.254* to access via HTTP websites of all categories, except *Chats*:

   ```
   protocol in (HTTP), src_ip in (10.10.0.0/24), url_category not in (Chats)
   : PASS
   ```

   Note that if the rule

   ```
   protocol in (HTTP), url_host in "LinuxFirewall.Blacklist" : BLOCK as
   BlackList
   ```

   is allocated in the list of rules above the indicated rule, then access to domains from the black list, i.e. domains listed in the parameter LinuxFirewall.Blacklist, will also be blocked for users with the range of IP addresses *10.10.0.0 – 10.10.0.254*. And if this rule is allocated below, users with the range of IP addresses *10.10.0.0 – 10.10.0.254* will get access also to websites from the black list.

   Due to the fact that resolution PASS is terminal, no more rules are checked, therefore scanning of the downloaded data for viruses is not performed either. To grant users with the range of IP addresses *10.10.0.0 – 10.10.0.254* access to websites of all categories, except *Chats* if they are not in the black list, and to block download of threats at the same time, use the following rule:

   ```
   protocol in (HTTP), url_category not in (Chats), url_host not in
   "LinuxFirewall.Blacklist", threat_category not in
   "LinuxFirewall.BlockCategory" : PASS
   ```

2. Do not perform scanning of contents of video files *downloaded from the Internet* (i.e. data with the type MIME "video/*", where * is any type of the MIME class video):

```
direction response, content_type in ("video/*") : PASS
```

Note that files loaded from the local computer (including those with the MIME type '`video/*`’) will be scanned because they are sent in *requests*, not in *responses*, i.e. for them a variable `direction` has a value `request`.

Dr.Web ClamD

The Dr.Web ClamD component performs emulation using the Dr.Web for UNIX Internet Gateways interface of the **clamd** anti-virus daemon, which is a core component of the anti-virus product **Clam AntiVirus** (**ClamAV** ®) from Sourcefire, Inc. This interface allows external applications that are able to interact with **ClamAV** ® to use Dr.Web for UNIX Internet Gateways for anti-virus scanning.

Operating Principles

The component is designed to check both the content of files in the local file system and the streams of data transmitted by an external application via a socket. Such checks are performed by the component at the request of an external application. Moreover, the component can check the content of those files for which an external application passed an open file descriptor via a socket.

File checks based on a passed file descriptor can be performed only if the descriptor was passed via a local UNIX socket.

If an external application has provided a path to a file in the local file system, the component sends the scanning task to the Dr.Web File Checker file checker component; otherwise, the component transmits data, received via the socket, to the Dr.Web Network Checker distributed scanning agent, as shown in the figure below.

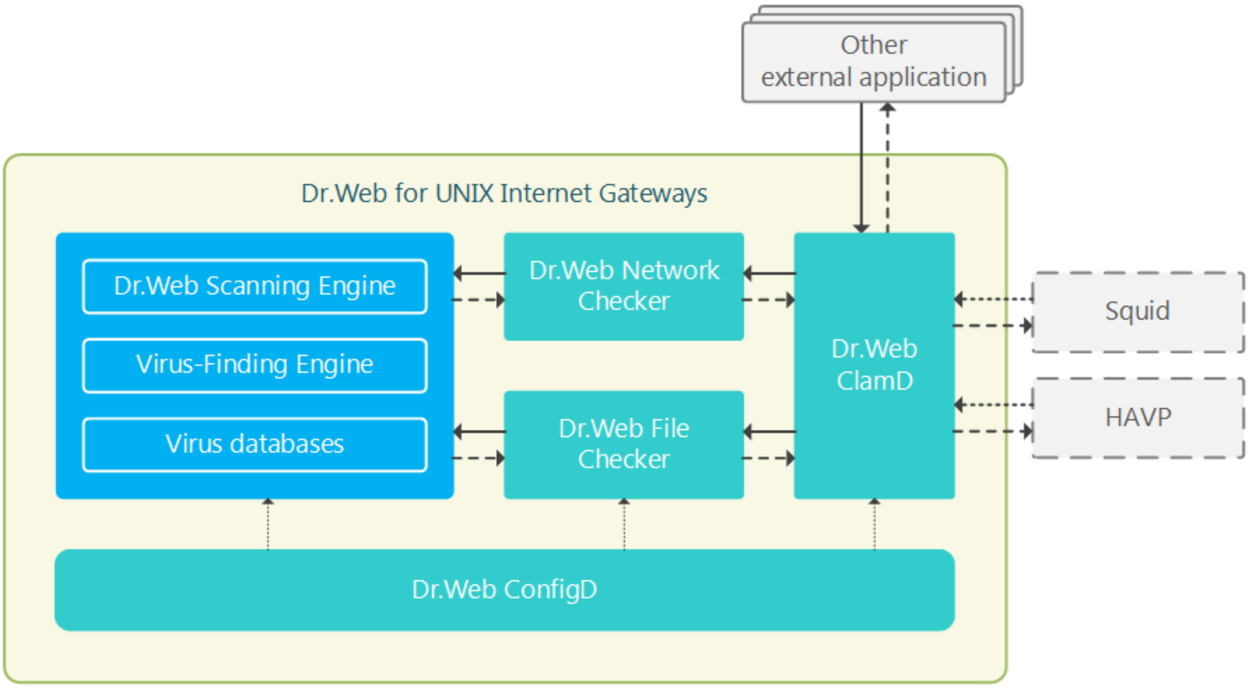


Figure 13. Diagram of the components' operation

By default, the component is not automatically launched upon the startup of Dr.Web for UNIX Internet Gateways. To enable starting of the component, it is necessary to set `the Yes` value for the Start parameter and to define at least one connection point for client applications. After that, the component starts waiting for external applications' requests to scan files or data streams. In the component's settings, you can configure several connection points for external applications and adjust different scanning settings for each of the points, if required.

The Figure above shows that external applications could be represented as HTTP proxy servers (such as **Squid** and **HAVP**), if they are equipped with the integration module with **clamd**. For details, see section Integration with External Applications.

Detected threats *cannot* be neutralized by Dr.Web for UNIX Internet Gateways; the external application receives only the results of the scanning. Thus, any detected threats should be neutralized by the external application.

Command-Line Arguments

To run Dr.Web ClamD, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-clamd [<parameters>]
```

Dr.Web ClamD can process the following parameters:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-clamd --help
```

This command outputs short help information on Dr.Web ClamD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed (as a rule, at the startup of the operating system). To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

> To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-clamd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[ClamD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-clamd` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-clamd` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-clamd`
`Start` *{Boolean}*	The component must be launched by the Dr.Web ConfigD configuration daemon. When you specify the `Yes` value for this parameter, it instructs the configuration daemon to start the component immediately; and when you specify the `No` value, it instructs the configuration daemon to terminate the component immediately. **Default value:** `No`

`Endpoint.`*<tag>*`.`**`ClamdSocket`**
{IP address | UNIX socket}
	Defines a new connection point naming it *<tag>* and allocates a socket (IPv4 address or address of a

	UNIX socket) for clients that need to check files for threats. *Only one socket can be specified for one <tag> point.* **Default value:** *(not specified)*
[Endpoint.<*tag*>.]**DetectSuspicious** *{Boolean}*	Inform about suspicious files detected by the heuristic analyzer. *If the* Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.* **Default value:** Yes
[Endpoint.<*tag*>.]**DetectAdware** *{Boolean}*	Inform about files containing adware. *If the* Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.* **Default value:** Yes
[Endpoint.<*tag*>.]**DetectDialers** *{Boolean}*	Inform about files containing dialers. *If the* Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.* **Default value:** Yes
[Endpoint.<*tag*>.]**DetectJokes** *{Boolean}*	Inform about files containing jokes. *If the* Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.* **Default value:** No
[Endpoint.<*tag*>.]**DetectRiskware** *{Boolean}*	Inform about files containing riskware. *If the* Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points*

	which do not have another value of this parameter specified for them. **Default value:** `No`

`[Endpoint.`*<tag>*`.]`**`DetectHacktools`**
{Boolean}
	Inform about files containing hacktools.
	If the `Endpoint.`*<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.*
	Default value: `No`

`[Endpoint.`*<tag>*`.]`**`ReadTimeout`**
{time interval}
	Sets the maximum time to wait for data from a client.
	If the `Endpoint.`*<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.*
	Default value: `5s`

`[Endpoint.`*<tag>*`.]`**`StreamMaxLength`**
{size}
	Sets the maximum size of data that can be received from a client (for transmitting data to scan as a stream of bytes).
	If the `Endpoint.`*<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.*
	Default value: `25mb`

`[Endpoint.`*<tag>*`.]`**`ScanTimeout`**
{time interval}
	Sets the maximum time to scan one file (or one portion of data) received from a client.
	A value in the range from 1s to 1h can be specified
	If the `Endpoint.`*<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.*
	Default value: `3m`

`[Endpoint.`*<tag>*`.]`**`HeuristicAnalysis`**
{On | Off}
	Indicates whether heuristic analysis is used for scanning.

	If the `Endpoint.`<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them. **Default value:** `On`

`[Endpoint.`*<tag>*`.]`**`PackerMaxLevel`**
{integer}
	Sets the maximum nesting level of packed objects that can be scanned.
	A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.
	If the `Endpoint.`<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.
	Default value: `8`

`[Endpoint.`*<tag>*`.]`**`ArchiveMaxLevel`**
{integer}
	Sets the maximum nesting level of archives that can be scanned.
	A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.
	If the `Endpoint.`<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.
	Default value: `8`

`[Endpoint.`*<tag>*`.]`**`MailMaxLevel`**
{integer}
	Sets the maximum nesting level of mail files that can be scanned.
	A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.
	If the `Endpoint.`<tag> prefix is specified, it means that the parameter's value is set only for the <tag> connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.
	Default value: `8`

`[Endpoint.`*<tag>*`.]`**`ContainerMaxLevel`**
{integer}
	Sets the maximum nesting level of objects in containers that can be scanned.
	A value in the range from 0 to 60 can be specified. If the value is set to 0, nested objects are not scanned.

	If the Endpoint.<*tag*> *prefix is specified, it means that the parameter's value is set only for the* <*tag*> *connection point; otherwise, it is set for all points which do not have another value of this parameter specified for them.* **Default value:** 8
[Endpoint.<*tag*>.]**MaxCompressionRatio** *{integer}*	Sets the maximum allowed compression ratio of compressed/packed objects (ratio between the uncompressed size and the compressed size). If the ratio of an object exceeds the limit, this object will be skipped during the scanning. *The compression ratio must not be smaller than 2.* **Default value:** 500

Special Aspects of Component Configuration

Parameters marked with an optional Endpoint.<*tag*> prefix can be grouped. Each group defines a unique connection *point* (*endpoint*) that can be used by clients to connect to the component and has a unique <*tag*> identifier assigned to it. All the scanning parameters belonging to the same group define the settings that are applicable only when data is scanned for the clients connected to the corresponding connection point. If a parameter is specified without an Endpoint.<*tag*>, prefix, this sets the value for all connection points. If you delete some parameter from some connection point, then instead of reverting to the program's hard-coded default value for this parameter, the program will use the current value of the corresponding "parent" parameter of the same name (set without the Endpoint.<*tag*> prefix).

The **ClamdSocket** parameter must always be specified with an Endpoint.<*tag*> prefix, as it defines both a listening socket and a group (connection point) to which this socket corresponds.

Example:

Let us assume that we need to set up two connection points for two groups of external applications (servers) — let the groups be called *servers1* and *servers2*. And the servers from the *servers1* group can connect through a UNIX socket, whereas the servers form the *servers2* group can connect via a network connection. Moreover, let us assume that heuristic analysis must be disabled by default, but must be used for servers from the *servers2* group. The following example shows how to configure this:

1) In the configuration file:

```
[ClamD]
HeuristicAnalysis = Off

[ClamD.Endpoint.servers1]
ClamdSocket = /tmp/srv1.socket
```

```
[ClamD.Endpoint.servers2]
ClamdSocket = 127.0.0.1:1234
HeuristicAnalysis = On
```

2) For command-line-based management tool Dr.Web Ctl:

```
# drweb-ctl cfset ClamD.HeuristicAnalysis Off
# drweb-ctl cfset ClamD.Endpoint -a servers1
# drweb-ctl cfset ClamD.Endpoint -a servers2
# drweb-ctl cfset ClamD.Endpoint.servers1.ClamdSocket /tmp/srv1.socket
# drweb-ctl cfset ClamD.Endpoint.servers2.ClamdSocket 127.0.0.1:1234
# drweb-ctl cfset ClamD.Endpoint.servers2.HeuristicAnalysis On
```

Both ways have an equal effect but if you edit the configuration file, you will also need to apply the changed settings by sending a `SIGHUP` signal to the **drweb-configd** component (to do that, you can issue the drweb-ctl `reload` command).

Integration with External Applications

The interface—that emulates the one of the **clamd** anti-virus daemon (included in **ClamAV**)—allows Dr.Web ClamD to communicate with any external application that is capable of connecting to this anti-virus daemon.

The table below shows examples of applications that can use **clamd** for anti-virus scans:

Product	Integration
HTTP services	
HTTP proxy server **Squid**	**Use of clamd:** Scanning of files received from the Internet. **Integration requirements:** Using **squidclamav** or **HAVP** as an intermediate component. **Links to documentation:** **Squid** documentation: http://www.squid-cache.org/Doc/ Description and source code files of **squidclamav**: http://squidclamav.darold.net/
HTTP proxy server which can perform anti-virus scans **HAVP**	**Use of clamd:** Scanning of files received from the Internet. **Integration requirements:** Configuring **HAVP** to use **clamd** for anti-virus scanning configuration:

Product	Integration
	`ENABLECLAMD true` `CLAMDSOCKET` *<path_to_clamd_UNIX_socket>* or (if TCP connection is used instead of a UNIX socket): `ENABLECLAMD true` `CLAMDSERVER` *<IP>* `CLAMDPORT` *<port>* where *<path_to_clamd_UNIX_socket>* or the *<IP>:<port>* pair corresponds to the socket of a connection point (*endpoint*) configured in Dr.Web ClamD. **Links to documentation:** **HAVP** documentation: http://www.server-side.de/documentation.htm

In the settings of the external software component that communicates directly with Dr.Web ClamD as with the **clamd** anti-virus daemon, specify an address for connecting to **clamd** as a path to a UNIX socket or as a TCP socket listened to by Dr.Web ClamD at one of its connection points (*endpoint*) set up in its configuration.

Example of how to connect **HAVP** to Dr.Web ClamD:

1. Configuring Dr.Web ClamD:

```
[ClamD]
Start = yes

[ClamD.Endpoint.proxy]
ClamdSocket = /var/run/drweb.clamd
```

2. Configuring **HAVP**:

```
ENABLECLAMD true
CLAMDSOCKET /var/run/drweb.clamd
```

Settings that configure connections to any other anti-virus products (`ENABLE*` parameters) must be set to `false`.

Dr.Web File Checker

The file checking component—Dr.Web File Checker—is designed for checking files and directories in the file system. It is used by other components of Dr.Web for UNIX Internet Gateways to check file system objects. Moreover, this component also functions as a quarantine manager, as it manages the contents of the directories where isolated (quarantined) files are kept.

Operating Principles

This component is used to access any file system objects (files, directories, boot records). It is started with superuser (*root*) privileges.

It indexes all checked files and directories and saves all the data about the objects that have been checked to a special cache to avoid repeated checking of objects that have been already checked and have not been modified since that (in this case, if a request to check such an object is received, the previous check result, retrieved from cache, is returned). A diagram showing how the component works is given in the figure below.

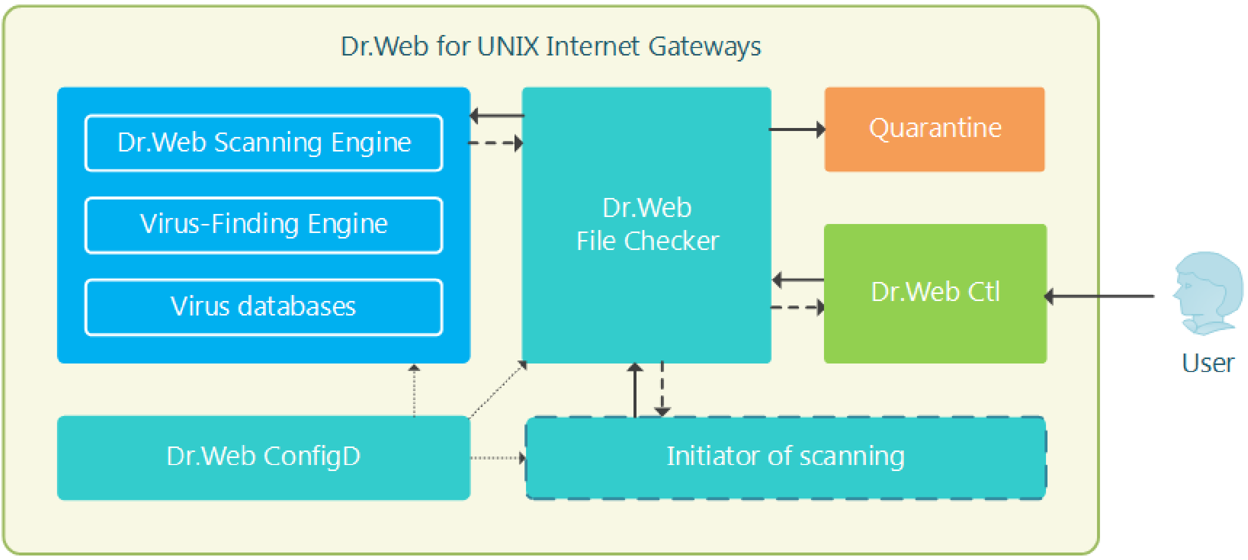


Figure 14. Diagram of the components' operation

When a request to check a file system object is received from Dr.Web for UNIX Internet Gateways's components, it checks whether this object requires scanning. If so, a scanning task is generated for Dr.Web Scanning Engine. If the scanned object contains a threat, Dr.Web File Checker neutralizes it (deletes or quarantines) if this action has been specified by the client component that initiated the scanning. Scanning can be initiated by various components of the product.

During the scanning, the file-checking component generates and sends to the client component a report detailing the results of the scanning and the applied actions, if any.

Apart from the standard scanning method, the following special methods are available for internal use:

- *The "flow" scanning method*. A client component that uses this scanning method initializes detection and neutralization parameters only once. These parameters will be applied to all future requests to check a file coming from this client component.
- *The "proxy" scanning method*. When this method is used, the file-checking component scans files without applying any actions to detected threats and without keeping any records about the detected threats to permit future action. Any necessary actions must be applied by the component that initiated the scanning process. This method is used by the Dr.Web ClamD component.

Files can be scanned with the *"flow"* and *"proxy"* scanning methods using the using the `flowscan and proxyscan` commands of the Command-Line Call Format utility (launched by the **drweb-ctl** command). However, for a normal on-demand scanning, it is recommended that you use the `scan` command.

The component collects statistics on scanned files averaging the number of files scanned per second in the last minute, 5 minutes, 15 minutes.

Command-Line Arguments

To launch Dr.Web File Checker, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-filecheck [<parameters>]
```

Dr.Web File Checker can process the following parameters:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-filecheck --help
```

This command outputs short help information on Dr.Web File Checker.

Startup Notes

The component cannot be launched directly from line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when receiving requests on file system scanning from other components of Dr.Web for UNIX Internet Gateways. To manage the operation of the component, as well as to scan files when needed, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is launched by using the drweb-ctl command).

To scan an arbitrary file or directory using Dr.Web File Checker you can use `scan` command of Dr.Web Ctl:

```
$ drweb-ctl scan <path to file or directory>
```

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-filecheck`

Configuration Parameters

The component uses configuration parameters which are specified in the `[FileCheck]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

This section stores the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-filecheck` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-filecheck` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-filecheck`
`DebugClientIpc` *{Boolean}*	Indicates whether detailed IPC messages should be included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). **Default value:** `No`

`DebugScan` *{Boolean}*	Indicates whether detailed messages received during file scanning should be included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). **Default value:** `No`
`DebugFlowScan` *{Boolean}*	Indicates whether detailed messages about file scanning by the *“flow”* method should be included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). **Default value:** `No`
`DebugProxyScan` *{Boolean}*	Indicates whether detailed messages about file scanning by the *“proxy”* method should be included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). Normally this scanning method is used by the Dr.Web ClamD component. **Default value:** `No`
`DebugCache` *{Boolean}*	Indicates whether detailed messages about the cached results of scanning should be included into the log file on the debug level (i.e. when **`LogLevel`** `= DEBUG`). **Default value:** `No`
`MaxCacheSize` *{size}*	Maximum allowed size of cache to store data about scanned files. *If 0 is specified, caching is disabled.* **Default value:** `50mb`
`RescanInterval` *{time interval}*	Period of time during which a file will not be rescanned if the results of its previous scan are available in the cache (the period during which the stored information is considered up-to-date). *The parameter can have a value from* `0s` *to* `1m` *(inclusive). If the set interval is less than* `1s`*—there will be no delay, the file will be scanned upon any request.* **Default value:** `1s`
`IdleTimeLimit` *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down. *The parameter can have a value from* `10s` *to* `30d` *(inclusive).* **Default value:** `30s`

Dr.Web Network Checker

Network checker agent Dr.Web Network Checker is designed for scanning the data received through the network in the checking engine, as well as distributed file scanning for threats. The component allows to arrange a connection between network hosts with installed Dr.Web for UNIX Internet Gateways for receiving and transmitting data (for example, file content) via the network hosts to perform its scanning. The component organizes automatic distribution of scanning tasks (by transmitting and receiving them over the network) to all available network hosts to which it is configured. The component balances the load between the hosts caused by scanning tasks. If there are no configured connections with remote hosts, the component transmits all the data to the local Dr.Web Scanning Engine only.

Note that the component is always used to scan the data received via network connections. Thus, if the component is missing or unavailable, the performance of the components that transmit data for scanning via the network connection will be incorrect (Dr.Web ICAPD, Dr.Web ClamD).

In case of high intensity of scanning of data transferred via the network, there is a possibility of having problems with scanning due to depletion of the number of available file descriptors. In this case, it is necessary to increase the limit of the number of file descriptors available to Dr.Web for UNIX Internet Gateways.

During scanning, data can be shared either over an open channel or over a protected one, applying SSL/TLS. To use a secure HTTPS connection it is required to provide an appropriate SSL server certificate and private key for hosts that share files. If you need to generate SSL keys and certificates, you can use the **openssl** utility. An example of how to use the **openssl** utility to generate a certificate and a private key is given in the section Appendix E. Generating SSL certificates.

Operating Principles

The Dr.Web Network Checker component allows to arrange connection between Dr.Web for UNIX Internet Gateways and a set of other nodews which have Dr.Web for UNIX Internet Gateways (or other Dr.Web for UNIX solution version 10.1 or above) installed on them. This will organize a distributed data scanning for threats (for example, file content). With the component, you can create and configure a *"scanning cluster"*, specifying the set of connections between cluster nodes (an instance of the distributed scanning agent Dr.Web Network Checkershould be launched at each node).

On each node within the cluster, Dr.Web Network Checker agent constitutes the automatic distribution of scanning jobs by transmitting data for scanning to all available nodes. At that, the agent sets up the load balancing on nodes, caused by file scanning, depending on resources available on remote nodes (the number of child scanning processes of Dr.Web Scanning Engine on each node acts as an indicator for the number of nodes available). The agent also considers the queue of files waiting for scanning on each host. Data received for scanning over the network is transmitted to the Dr.Web Scanning Engine scanning engine, as shown on the figure below.

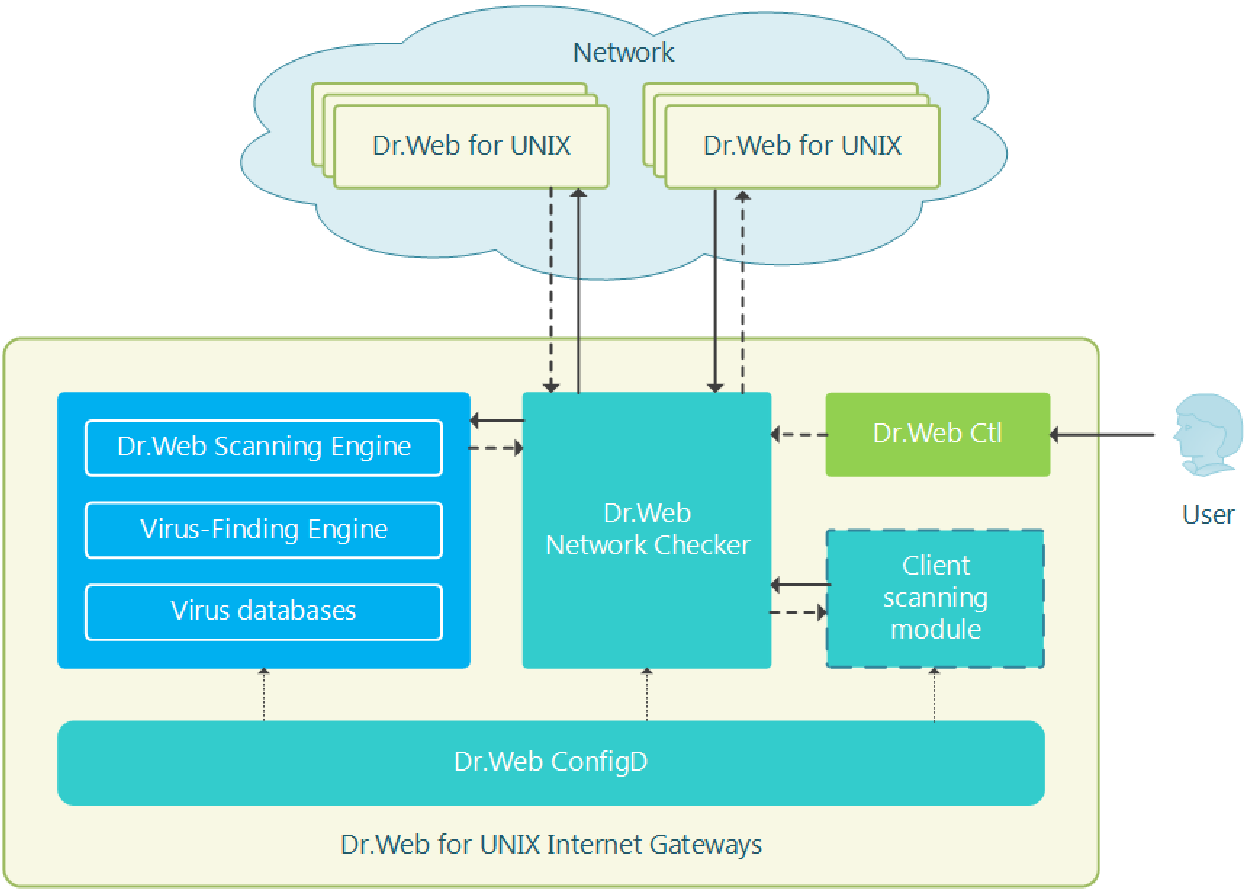


Figure 15. Diagram of the components' operation

In this case, any network node included in the scanning cluster can act as a scanning client that transmits data to a remote scan as well as a scanning server that receives data from the specified network nodes for verification. If necessary, the distributed scanning agent can be configured so that the node acts only as a scanning server or only as a scanning client.

On a local host, sending data for scanning via Dr.Web Network Checker can be started both at user's command specified via the Dr.Web Ctl command-line management tool and at requests received from some product components, for example, the Dr.Web ClamD component, which provides the interface of the **clamd** daemon included in **ClamAV®**. That is why the scheme contains an abstract "Client scanning module".

Note that components marked as "Client scanning module" always use the Dr.Web Network Checker for transmitting files to be scanned by Dr.Web Scanning Engine, even if Dr.Web Scanning Engine is located on the local host. Thus, if Dr.Web Network Checker is unavailable, these components will not work correctly.

It is possible to create your own component (external application) which will use Dr.Web Network Checker to check the files (including distributing the scanning jobs to the nodes of the scanning cluster). For this, the Dr.Web Network Checker component provides a custom API based on the **Google Protobuf** technology. The Dr.Web Network Checker API , as well as client application sample code that uses Dr.Web Network Checker, are supplied as part of `drweb-netcheck package`.

Command-Line Arguments

To run Dr.Web Network Checker, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-netcheck [<parameters>]
```

Dr.Web Network Checker can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-netcheck --help
```

This command outputs short help information on Dr.Web Network Checker.

Startup Notes

The component cannot be run directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is run automatically by the configuration daemon Dr.Web ConfigD configuration daemon when required (usually on operating system startup). If a value of the **`FixedSocketPath`** parameter in the configuration is specified, the agent is always running and available for clients via the specified UNIX socket. To start scanning via network, you can use the Dr.Web Ctl command-line tool for Dr.Web for UNIX Internet Gateways management (it is started by the command **drweb-ctl**). If there are no configured connections to remote hosts, the local scanning will be started.

To scan an arbitrary file or directory using Dr.Web Network Checker you can use `netscan` command of Dr.Web Ctl tool:

```
$ drweb-ctl netscan <path to file or directory>
```

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-netcheck`

Configuration Parameters

The component uses configuration parameters which are specified in the `[NetCheck]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/share/drweb-netcheck/linkchecker` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-netcheck` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-netcheck`
`FixedSocketPath` *{path to file}*	Path to the UNIX socket of the fixed Dr.Web Network Checker agent instance. If this parameter is specified, the Dr.Web ConfigD configuration daemon checks that there is always a running component copy of the distributed scanning agent that is available to the clients via this socket. **Default value:** *(not specified)*
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user’s number UID or as the user’s login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the “`name:`” prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value**: `drweb`
`IdleTimeLimit` *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down.

	Minimum value—`10s`. *If the* **`LoadBalanceAllowFrom`** *or* **`FixedSocketPath`** *parameter is set, this setting is ignored (the component does not finish its operation after the time interval expires).* **Default value:** `30s`
`LoadBalanceUseSsl` *{Boolean}*	The indicator which determines whether a secure SSL/TLS connection is used for connection to other hosts. **Allowed values:** • `Yes`—instructs to use SSL/TLS • `No`—instructs not to use SSL/TLS *If the parameter is set to* `Yes`*, a certificate and the corresponding private key should be specified for this host and for hosts with which it interacts (the parameters* **`LoadBalanceSslCertificate`** *and* **`LoadBalanceSslKey`***).* **Default value :** `No`
`LoadBalanceSslCertificate` *{path to file}*	Path to the SSL certificate used by Dr.Web Network Checker for communication with other hosts via a secure SSL/TLS connection. *Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.* **Default value:** *(not specified)*
`LoadBalanceSslKey` *{path to file}*	Path to the private key used by Dr.Web Network Checker for communication with other hosts via a secure SSL/TLS connection. *Please note that the certificate file and the private key file (which is specified by the mentioned parameter) must form a matching pair.* **Default value:** *(not specified)*
`LoadBalanceSslCa` *{path}*	The path to the directory or file that contains the list of root certificates that are trusted. Among these certificates, there must be a certificate that certifies the authenticity of the certificates used by agents within the scanning cluster when exchanging data over SSL/TLS protocols. *If the parameter value is empty, Dr.Web Network Checker working on this host does not authenticate certificates of interacting agents; however, depending on the settings, these agents can authenticate the certificate used by the agent operating on the host.* **Default value:** *(not specified)*
`LoadBalanceServerSocket` *{address}*	Network socket (IP address and port) which is listened on this host by Dr.Web Network Checker for receiving files sent by

	remote hosts for scanning (if it can operate as a scanning server). **Default value:** *(not specified)*

`LoadBalanceAllowFrom`
{IP address}
	IP address of a remote network host from which the Dr.Web Network Checker can receive files for scanning (as a scanning server).
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add to the list of host addresses 192.168.0.1 and 10.20.30.45.
	1. Adding of values to the configuration file.
	• Two values in one string
	`Section [NetCheck]`
	` LoadBalanceAllowFrom = "192.168.0.1", "10.20.30.45"`
	• Two strings (one value per a string)
	`[NetCheck]`
	` LoadBalanceAllowFrom = 192.168.0.1`
	` LoadBalanceAllowFrom = 10.20.30.45`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset NetCheck.LoadBalanceAllowFrom -a 192.168.0.1`
	`# drweb-ctl cfset NetCheck.LoadBalanceAllowFrom -a 10.20.30.45`
	If the parameter is empty, removed files cannot be received for scanning (the host does not operate as a scanning server).
	Default value: *(not specified)*

`LoadBalanceSourceAddress` *{IP address}*	IP address of a network interface used by Dr.Web Network Checker on the host for transferring files for their remote scanning (if the host operates as a scanning server and has several network interfaces). *If an empty value is specified, the network interface automatically selected by the OS kernel is used.* **Default value:** *(not specified)*
`LoadBalanceTo`	Socket (IP address or port) of a remote host to which Dr.Web Network Checker on the host can send files for their remote

{address}	scanning (as a network scanning client).
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add sockets 192.168.0.1:1234 and 10.20.30.45:5678 to the list.
	1. Adding of values to the configuration file.
	• Two values in one string
	`[NetCheck]`
	` LoadBalanceTo = "192.168.0.1:1234", "10.20.30.45:5678"`
	• Two strings (one value per a string)
	`[NetCheck]`
	` LoadBalanceTo = 192.168.0.1:1234`
	` LoadBalanceTo = 10.20.30.45:5678`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset NetCheck.LoadBalanceTo -a 192.168.0.1:1234`
	`# drweb-ctl cfset NetCheck.LoadBalanceTo -a 10.20.30.45:5678`
	If the parameter value is empty, local files cannot be transferred for a remote scanning (the host does not operate as a network scanning client).
	Default value: *(not specified)*

`LoadBalanceStatusInterval` *{time interval}*	Time interval considered by the host to send the next message containing information about its workload to all scanning clients (specified in the **`LoadBalanceAllowFrom`** parameter). **Default value:** `1s`
`SpoolDir` *{path to directory}*	Local file system directory used to store files sent over the network for scanning and received by Dr.Web Network Checker. **Default value:** `/tmp/netcheck`
`LocalScanPreference` *{fractional number}*	Relative weight (priority) of this host which is considered when a scanning server is selected to scan a file (a local file or a file received over the network). If the relative weight of the local station is greater than the weights of all hosts available as scanning servers, files are scanned locally. Minimum value—`1`.

	Default value: `1`

Dr.Web Scanning Engine

Dr.Web Scanning Enginescanning engine is designed to search for viruses and other malicious objects in files and boot records (*MBR – Master Boot Record*, *VBR – Volume Boot Record*) of disk devices. The component loads the anti-virus engine Dr.Web Virus-Finding Engine into memory and starts it as well as loads Dr.Web virus databases used by the engine for threat detection.

The scanning engine operates in the daemon mode, as a service which receives scanning requests from other Dr.Web for UNIX Internet Gateways components. *If the Dr.Web Scanning Engine and Dr.Web Virus-Finding Engine components are missing or unavailable, no anti-virus scanning is performed.*

Operating Principles

The component operates as a service which receives requests to scan file system objects (files and boot disk records) from Dr.Web for UNIX Internet Gateways components. It also queues scanning tasks and scans requested objects by using Dr.Web Virus-Finding Engine. If a threat is detected and the scanning task instructs to cure threats, the scanning engine attempts to cure it if this action can be applied to the scanned object. The figure below shows the operation scheme of Dr.Web Scanning Engine scanning engine.

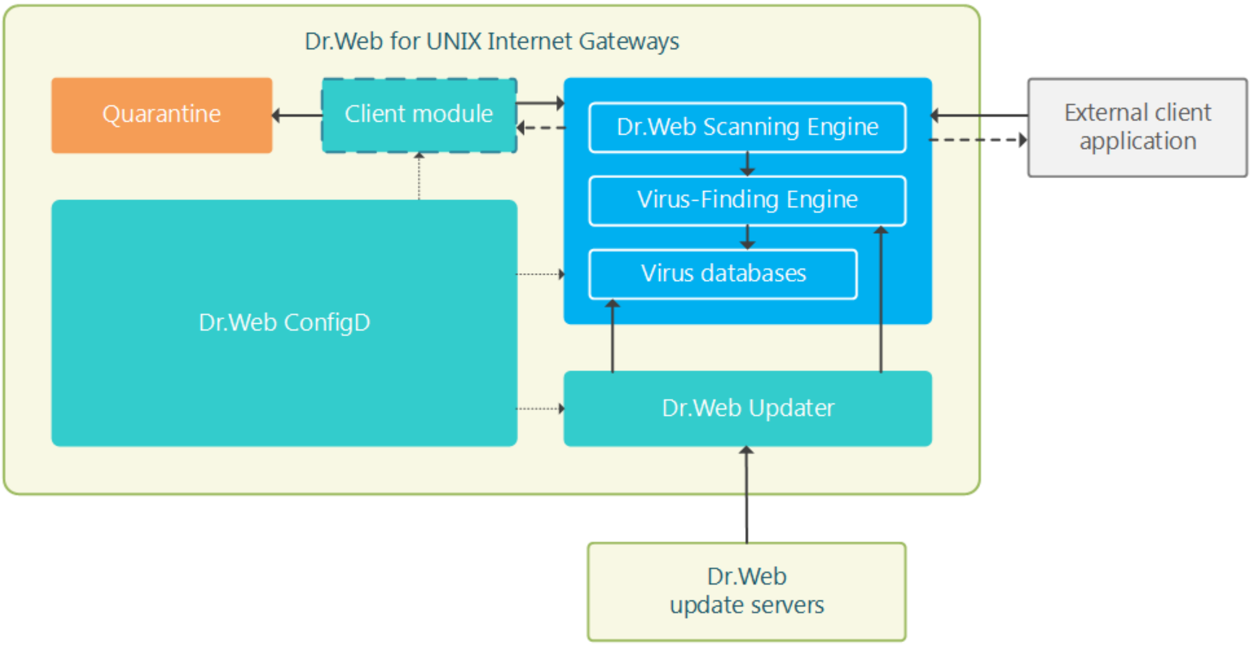


Figure 16. Diagram of the components' operation

The scanning engine, the anti-virus engine Dr.Web Virus-Finding Engine, and the virus databases form one unit and cannot be separated: the scanning engine downloads virus databases and provides the operation environment for the cross-platform anti-virus engine Dr.Web Virus-Finding Engine. The virus databases and the anti-virus engine are updated by the Dr.Web Updater update component that is included in the product, but this component is not a part of the scanning engine. The update component is run by the Dr.Web ConfigD configuration daemon periodically or forcefully, if the corresponding command is sent by the user. Moreover, if Dr.Web for UNIX Internet Gateways operates in central protection mode, updating of virus databases and anti-virus engine is performed by the Dr.Web ES Agent (it is not shown in the above-mentioned scheme). The latter component interacts with the central protection server and receives the updates.

The scanning engine can operate both under management of the configuration daemon Dr.Web ConfigD and in an autonomous mode. In the former case, the daemon runs the engine and ensures that virus databases are up to date. In the latter case, engine startup and updating of virus databases is performed by an external application that uses the engine. Dr.Web for UNIX Internet Gateways's components that issue requests to the scanning engine asking it to scan files for them (indicated as “Client modules” in the diagram) use the same interface as other external applications would.

> Users are provided with the opportunity to create own component (external application) using Dr.Web ASE for files checks. For this, Dr.Web Scanning Engine contains a special API, based on **Google Protobuf**. To obtain Dr.Web Scanning Engine API guide and examples of client application using Dr.Web Scanning Engine, contact Doctor Web partner care department (https://partners.drweb.com/).

Received tasks are automatically distributed into queues with different priorities: high, normal and low. Selection of the queue depends on the component that created a task: for example, tasks created by a file system monitor receive high priority as response time is important for monitoring. The scanning engine computes statistics of its operations, including the number of all tasks received for scanning and the queue length. As the average load rate, the scanning engine uses the average length of queues per second. This rate is averaged for the last minute, last 5 minutes and last 15 minutes.

Dr.Web Virus-Finding Engine supports signature analysis (signature-based threat detection) and other methods of heuristic and behavioral analysis designed for detection of potentially dangerous objects based on machine instructions and other attributes of executable code.

> Heuristic analysis cannot guarantee highly reliable results and may commit the following errors:
>
> - *Errors of the first type*. These errors occur when a safe object is detected as malicious (false positive detections).
> - *Errors of the second type*. These errors occur when a malicious object is detected as safe.
>
> Thus, objects detected by the heuristics analyzer are treated as *Suspicious*.

It is recommended that you choose to move suspicious objects to quarantine. After virus databases are updated, such files can be scanned using signature analysis. Keep the virus databases up to date in order to avoid errors of the second type.

Dr.Web Virus-Finding Engine allows to scan and cure both files and packed objects or objects in different containers (such as archives, email messages, etc.).

Command-Line Arguments

To run the scanning engine Dr.Web Scanning Engine from the command line, type the following command:

```
$ <opt_dir>/bin/drweb-se <socket> [<parameters>]
```

where the mandatory *<socket>* argument indicates the address of the socket used by Dr.Web Scanning Engine for processing requests of the client components. It can be set only as a file path (UNIX socket).

Dr.Web Scanning Engine can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.
Additional launch parameters (they are the same as configuration file parameters and substitute them when required):	
`--EnginePath`	**Function:** Specify a path to the library of Dr.Web Virus-Finding Engine anti-virus engine. **Short form:** No. **Arguments:** *<path to the file>*—a full path to the library that you want to use.
`--VirusBaseDir`	**Function:** Specify a path to the directory with virus database files. **Short form:** No. **Arguments:** *<path to the catalog>*—path to the virus database directory.
`--TempDir`	**Function:** Specify a path to the directory with temporary files. **Short form:** No. **Arguments:** *<path to the catalog>*—full path to the directory with temporary files.
`--Key`	**Function:** Specify a path to the key file. **Short form:** No. **Arguments:** *<path to the file>*—a full path to the key file that you want to use.

`--MaxForks`	**Function:** Determine the maximum allowed number of child processes which can be started by Dr.Web Scanning Engine during scanning. **Short form:** No. **Arguments:** <*number*>—the maximum allowed number of child processes.
`--MaxForksPerFile`	**Function:** Determine the maximum allowed number of child processes which can be used by Dr.Web Scanning Engine during scanning one compound file (an archive, a container, etc.). **Short form:** No. **Arguments:** <*number*>—the maximum allowed number of child processes. Cannot be less than 1 and greater than the `--MaxForks` parameter value.
`--WatchdogInterval`	**Description:** Determine frequency with which Dr.Web Scanning Engine checks whether child processes are operable and stops those processes that stopped responding. **Short form:** No. **Arguments:** <*time interval*>—frequency of checking child processes.
`--Shelltrace`	**Function:** turn on the shell tracing (log detailed information on file scanning performed by Dr.Web Virus-Finding Engine). **Short form:** No. **Arguments:** None.
`--LogLevel`	**Description**: Set the level of logging executed by Dr.Web Scanning Engine during the operation. **Short form:** No. **Arguments:** <*logging level*>. Allowed values: • DEBUG—the most detailed logging level. All messages and debug information are registered. • INFO—all messages are registered. • NOTICE—all error messages, warnings, and notifications are registered. • WARNING—all error messages and warnings are registered. • ERROR—only error messages are registered.
`--Log`	**Description**: Specify the method for logging component messages. **Short form:** No. **Arguments:** <*log type*>. Allowed values: • `Stderr[:ShowTimestamp]`—messages are output to a standard error stream *stderr*. Additional option `ShowTimestamp` instructs to add a time stamp to every message. • `Syslog[:`<*facility*>`]`—messages are transmitted to the system logging service **syslog**. Additional option <*facility*> is used to specify a level at which **syslog** registers messages. The following values are possible: o `DAEMON`—messages of daemons. o `USER`—messages of user processes.

	o MAIL—messages of mail programs.
	o LOCAL0—messages of local processes 0.
	…
	o LOCAL7—messages of local processes 7.
	• *<path>*—path to the file where all messages are registered.
	Examples:
	`--Log /var/opt/drweb.com/log/se.log`
	`--Log Stderr:ShowTimestamp`
	`--Log Syslog:DAEMON`

Example:

```
$ /opt/drweb.com/bin/drweb-se /tmp/drweb.ipc/.se --MaxForks=5
```

This command starts an instance of Dr.Web Scanning Engine scanning engine, instructs it to create the `/tmp/drweb.ipc/.se` UNIX socket for an interaction with the client components and to start no more than 5 child scanning processes while scanning a file.

Startup Notes

When necessary, any number of scanning engine Dr.Web Scanning Engine instances can be started. The instances provide the scanning service for client applications (not only for Dr.Web for UNIX Internet Gateways components). At that, if a value of the **`FixedSocketPath`** parameter is specified in the component's configuration, one instance of the scanning engine is always running by the Dr.Web ConfigD configuration daemon and is always available for the clients via this UNIX socket. The instances of the scanning engine started directly from the command line, will operate in an autonomous mode without establishing connection to the configuration daemon, even if it is running. To manage the operation of the component, as well as to scan files when needed, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is launched by using the **drweb-ctl** command).

To scan an arbitrary file or directory using Dr.Web Scanning Engine you can use `rawscan` command of Dr.Web Ctl tool:

```
$ drweb-ctl rawscan <path to file or directory>
```

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-se`

Configuration Parameters

The component uses configuration parameters which are specified in the `[ScanEngine]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

This section stores the following parameters:

LogLevel *{logging level}*	Logging level of the component. If the parameter value is not specified, the **DefaultLogLevel** parameter value from the [Root] section is used. **Default value:** `Notice`
Log *{log type}*	Logging method
ExePath *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-se` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-se` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-se`
FixedSocketPath *{path to file}*	Path to the UNIX socket of the fixed Dr.Web Scanning Engine scanning engine instance. If this parameter is specified, the Dr.Web ConfigD configuration daemon checks that there is always a running component copy of scanning engine that is available to the clients via this socket. **Default value:** *(not specified)*
IdleTimeLimit *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down. Minimum value—`10s`. *If the* **FixedSocketPath** *parameter is set, this setting is ignored (the component does not finish its operation after the time interval expires).* **Default value:** `30s`
MaxForks *{integer}*	Maximum allowed number of child processes run by Dr.Web Scanning Engine, which can be run simultaneously. **Default value: Automatically determined as twice the number of available CPU cores; or 4, if the resulting number is less than 4.**
MaxForksPerFile *{integer}*	Maximum allowed number of Dr.Web Scanning Engine child scanning processes, which can be used simultaneously for scanning container files (such as archive). Cannot be less than `1` and greater than the **MaxForks** parameter value. **Default value: Automatically determined as** `MaxForks/2`.
BufferedIo *{On | Off}*	Use buffered input/output (I/O) when checking files.

	Using buffered I/O in the **FreeBSD** *and* **Linux** *OSes can increase scanning speed of the files on slow disks.* **Default value:** `Off`
`WatchdogInterval` *{time interval}*	Rate at which Dr.Web Scanning Engine checks whether child processes are operable in order to detect processes that stopped responding (“watchdog”). **Default value:** `1.5s`

Dr.Web Updater

The update component Dr.Web Updater is designed for receiving all available updates for virus databases and anti-virus engine Dr.Web Virus-Finding Engine from Doctor Web update servers component.

If Dr.Web for UNIX Internet Gateways operates in central protection mode, the updates are received from the central protection server (for example, from Dr.Web Enterprise Server); at that, all updates are received from the server via Dr.Web ES Agent, and Dr.Web Updater is not used for downloading updates.

Operating Principles

The component is designed to establish connections to Doctor Web update servers to check for updates for virus databases and anti-virus engine Dr.Web Virus-Finding Engine , database of web resource categories. The lists of servers which constitute an available update zone are stored in a special file (the file is signed to prevent modification).

If the product is not connected to the central protection server or it is connected to the server in mobile mode, Dr.Web Updater is automatically started by the Dr.Web ConfigD configuration daemon. Startup is performed at periods specified in the settings. The component can be also started by the configuration daemon if the appropriate command is received from a user (unscheduled update). The component operation scheme is shown in the figure below.

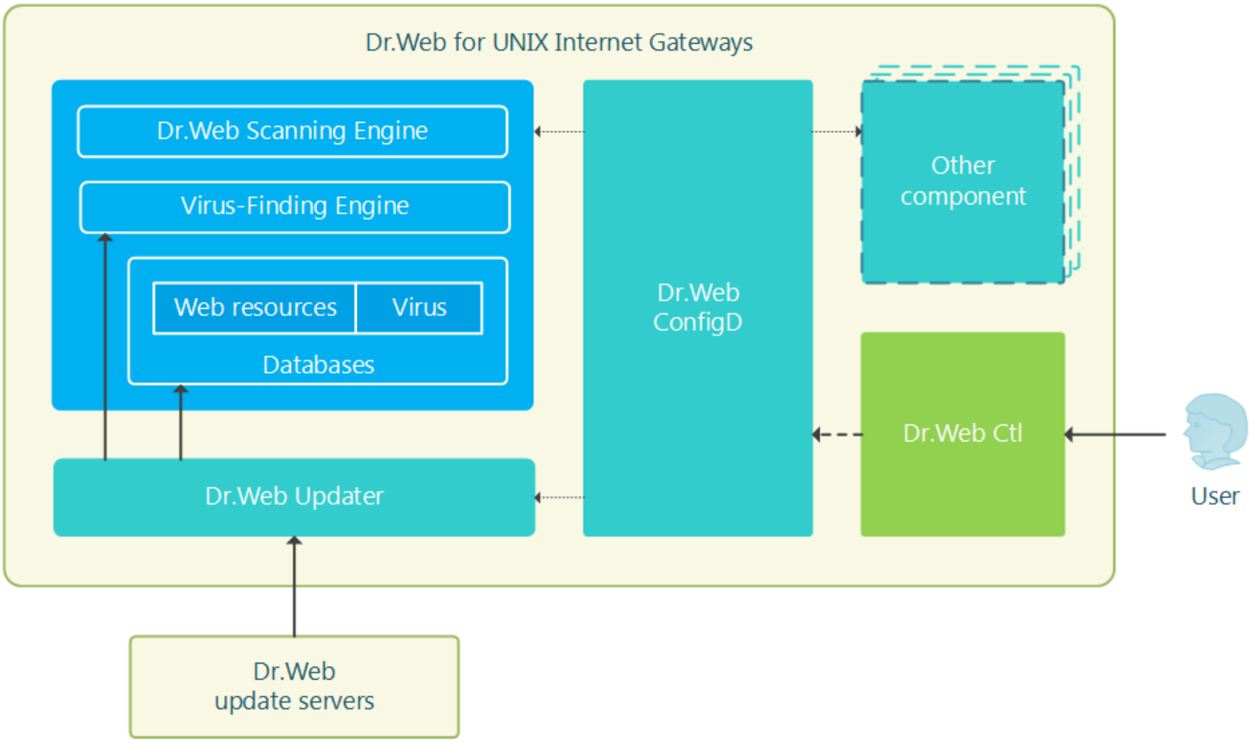


Figure 17. Diagram of the components' operation

When updates become available on the servers, they are downloaded to the <*var_dir*>/cache directory (for **Linux**—var/opt/drweb.com/cache), after that they are moved to the working directories of Dr.Web for UNIX Internet Gateways.

By default, all updates are performed from the updating zone which is common for all Dr.Web products. The list of the servers used by default, which are included to the updating zone, is specified in the files which are located in directories, defined in ***DrlDir** parameters, grouped by the update type: for virus databases and anti-virus engine, database of web resource categoriesUpon user request the special update zone can be created (for each update type), the server list which is specified in separate file (named custom.drl, by default), located in directory specified in ***CustomDrlDir** parameter. In this case, the update component will receive updates only from these servers, without using servers from the default zone.

If you do not want to use the special updating zone, clear the ***CustomDrlDir** value of the corresponding parameter in the component settings.

⚠ The content of the files with server lists is signed, so that the files cannot be modified. If you need to create a special list of update servers, contact technical support.

The component can back up the updated files for the next rollback of the updates, performed at user request. You can specify the location and the detail level of the backed up files in the settings. To roll back updates, use the command-line tool for Dr.Web for UNIX Internet Gateways for managing the solution from the Dr.Web Ctl command line (it is run by **drweb-ctl** command).

Command-Line Arguments

To run Dr.Web Updater, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-update [<parameters>]
```

Dr.Web Updater can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-update --help
```

This command outputs short help information on Dr.Web Updater.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component, as well as to update virus databases and scanning engine, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-update`

Configuration Parameters

The component uses configuration parameters which are specified in the `[Update]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`

`Log`
`{log type}`
	Logging method

`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/bin/drweb-update` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-update` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-update`
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value:** `drweb`
`UpdateInterval` *{time interval}*	The frequency to check for updates on Dr.Web update servers. This is a time period between a previous successful attempt to connect to the update servers (initiated automatically or manually) and the next attempt to perform an update. **Default value:** `30m`
`RetryInterval` *{time interval}*	Frequency of repeated attempts to perform an update using the update servers if the previous attempt failed. The parameter can have a value of `1m` to `30m` inclusively. **Default value:** `3m`
`MaxRetries` *{integer}*	Number of repeated attempts to perform an update using the update servers (at the rate specified in **`RetryInterval`**) if the previous attempt failed. *If the value is set to 0, repeated attempts are not made (the next update will be performed after the time period specified in* **`UpdateInterval`***).* **Default value:** `3`
`Proxy`	Stores the parameters for connecting to a proxy server that is used by the updater component (Dr.Web Updater) when it is connecting to

{connection string}	Dr.Web updates servers (for example, if direct connections to external servers are prohibited by your network's security policies).
	If the parameter value is not specified, the proxy server is not used.
	Allowed values:
	<connection string>—Proxy server connection string. The string has the following format (URL):
	`[`*<protocol>*`://][`*<user>*`:`*<password>*`@]`*<proxyhost>*`:`*<port>*
	where
	• *<protocol>*—Type of the used protocol (in the current version, only `http` is available).
	• *<user>*—Name of the user for connection to proxy.
	• *<password>*—Password for connection to proxy.
	• *<proxyhost>*—Address of the host where the proxy operates (IP address or domain name, i.e. FQDN).
	• *<port>*—Used port.
	The *<protocol>* and *<user>:<password>* parameters can be absent. The address of proxy *<proxyhost>:<port>* is obligatory.
	If the user name (<user>) or password (<password>) contains the following characters: '`@`',', '`%`' *or* '`:`', *these characters must be changed to the following codes:* "`%40`", "`%25`" *and* "`%3A`" *respectively.*
	Examples:
	1. In the configuration file:
	• Connection to a proxy on the host *proxyhost.company.org* using the port *123*:
	`Proxy` `= proxyhost.company.org:123`
	• Connection to the proxy on the node *10.26.127.0* using the port *3336* over HTTP protocol as user "*legaluser*" with the password "*passw*":
	`Proxy` `= http://legaluser:passw@10.26.127.0:3336`
	• Connection to the proxy on the node *10.26.127.0* using the port *3336* as a user "*user@company.com*" with the password ''*passw%123:*":
	`Proxy` `= user%40company.com:passw%25123%3A@10.26.127.0:3336`
	2. Specifying the same parameter value via command **drweb-ctl** `cfset`:
	`#` **`drweb-ctl`** `cfset Update.Proxy proxyhost.company.org:123`
	`#` **`drweb-ctl`** `cfset Update.Proxy http://legaluser:passw@10.26.127.0:3336`
	`#` **`drweb-ctl`** `cfset Update.Proxy user%40company.com:passw%25123%3A@10.26.127.0:3336`

	Default value: *(not specified)*

`ExcludedFiles`
{file name}
	Defines the name of the file that will not be updated by the Dr.Web Updater component.
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add to the list the following files: `123.vdb` and `456.dws`.
	1. Adding of values to the configuration file.
	- Two values in one string
	`[Update]`
	` ExcludedFiles = "123.vdb", "456.dws"`
	- Two strings (one value per a string)
	`[Update]`
	` ExcludedFiles = 123.vdb`
	` ExcludedFiles = 456.dws`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset Update.ExcludedFiles -a 123.vdb`
	`# drweb-ctl cfset Update.ExcludedFiles -a 456.dws`
	Default value: `drweb32.lst`

`NetworkTimeout` *{time interval}*	A time-out period imposed on the network-related operations of the updater component during the updating process. This parameter is used when a connection is temporarily lost: if the connection is established again before the timeout expires, the interrupted updating process will be continued. *Specifying the time out value larger than* `75s` *has no effect as the connection is closed by TCP timeout. The minimum allowed value is* `5s`. **Default value:** `60s`
`BaseDrlPath` *{path to file}*	Path to the signed file that contains the list of update servers of a standard update zone, which are used by the update component for updating virus databases and anti-virus engine. **Default value:** `<var_dir>/drl/bases/update.drl` - For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/bases/update.drl` - For **FreeBSD**: `/var/drweb.com/drl/bases/update.drl`

BaseCustomDrlPath *{path to file}*	Path to the signed file that contains the list of update servers of a special update zone, which are used by the update components for updating virus databases and anti-virus engine. *If the parameter is not empty, and the specified file exists, only servers are used for the update. The main file of the list (see above) is ignored. If the file identified by the parameter is empty, the update attempt will fail.* **Default value:** `<var_dir>`/drl/bases/update.drl • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/bases/custom.drl` • For **FreeBSD**: `/var/drweb.com/drl/bases/custom.drl`
BaseUpdateEnabled *{Boolean}*	Indicator that shows whether or not updating of virus databases and anti-virus engine is allowed. **Allowed values:** • `Yes`—updating is allowed and will be performed. • `No`—updating is not allowed and will not be performed. **Default value:** `Yes`
VersionDrlPath *{path to file}*	Defines a path to the signed file that contains the list of the servers of an update zone, which are used by the update component for updating Dr.Web for UNIX Internet Gateways components. **Default value:** <*var_dir*>`/drl/version/update.drl` • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/version/update.drl` • For **FreeBSD**: `/var/drweb.com/drl/version/update.drl`
VersionUpdateEnabled *{Boolean}*	Indicator that shows whether or not updating of Dr.Web for UNIX Internet Gateways component's version is allowed. **Allowed values:** • `Yes`—updating is allowed and will be performed. • `No`—updating is not allowed and will not be performed. **Default value:** `Yes`
DwsDrlPath *{path to file}*	Path to the signed file that contains the list of the servers of a standard update zone, which are used for updating database of web resource categories.. **Default value:** <*var_dir*>`/drl/dws/update.drl` • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/dws/update.drl` • For **FreeBSD**: `/var/drweb.com/drl/dws/update.drl`
DwsCustomDrlPath *{path to file}*	Path to the signed file that contains the list of the servers of a special update zone, which are used for updating database of web resource

	categories. *If the parameter is not empty, and the specified file exists, only servers are used for the update. The main file of the list (see above) is ignored. If the file identified by the parameter is empty, the update attempt will fail.* **Default value:** <*var_dir*>`/drl/dws/custom.drl` • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/dws/custom.drl` • For **FreeBSD**: `/var/drweb.com/drl/dws/custom.drl`
`DwsUpdateEnabled` *{Boolean}*	Indicator that shows whether or not updating of database of web resource categories is allowed. **Allowed values:** • `Yes`—updating is allowed and will be performed. • `No`—update is not allowed and will not be performed. **Default value:** `Yes`
`AntispamDrlPath` *{path to file}*	The parameter is not used. **Default value:** <*var_dir*>`/drl/antispam/update.drl` • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/antispam/update.drl` • For **FreeBSD**: `/var/drweb.com/drl/antispam/update.drl`
`AntispamCustomDrlPath` *{path to file}*	The parameter is not used. **Default value:** <*var_dir*>`/drl/antispam/custom.drl` • For **Linux**, **Solaris**: `/var/opt/drweb.com/drl/antispam/custom.drl` • For **FreeBSD**: `/var/drweb.com/drl/antispam/custom.drl`
`AntispamUpdateEnabled` *{Boolean}*	The parameter is not used. **Default value:** `No`
`BackupDir` *{path to directory}*	Path to the directory, where the previous versions of updated files are saved for possible rollback. Upon every update only updated files are backed up. **Default value:** `/tmp/update-backup`
`MaxBackups` *{integer}*	The maximum number of the previous versions of updated files, which are saved. If this number is exceeded the oldest copy is removed upon the next update. *If the parameter value is zero, the previous versions of the files are not saved.* **Default value:** `0`

Dr.Web ES Agent

Central anti-virus protection agent Dr.Web ES Agent is designed for connecting Dr.Web for UNIX Internet Gateways to the central protection server (for example, to Dr.Web Enterprise Server).

When Dr.Web for UNIX Internet Gateways is connected to the central protection server Dr.Web ES Agent, the license key file synchronized according to

Operating Principles

Dr.Web ES Agent establishes connection to the central protection server (for example, to Dr.Web Enterprise Server), which allows the network administrator to implement common security policy within the network, in particular, configure the same scanning settings and reaction on threat detection for all network stations and servers. Moreover, the central protection server also performs a role of an internal update server on the network, as it stores up-to-date virus databases, components (in this case, updating is performed via Dr.Web ES Agent, Dr.Web Updater is not used).

When connecting Dr.Web ES Agent to the central protection server, the agent ensures receipt of up-to-date settings for the program components and the license key file, which are then transmitted to the Dr.Web ConfigD configuration daemon for applying them to managed components. Moreover, the component also receives tasks to scan file system objects on the station (including scheduled tasks).

> Note that the current version of Dr.Web for UNIX Internet Gateways *does not* fully implement the central protection mode: the central protection server cannot manage operation settings of the program components.

Dr.Web ES Agent collects and sends the server statistics on detected threats and applied actions. The operation scheme is shown in the figure below.

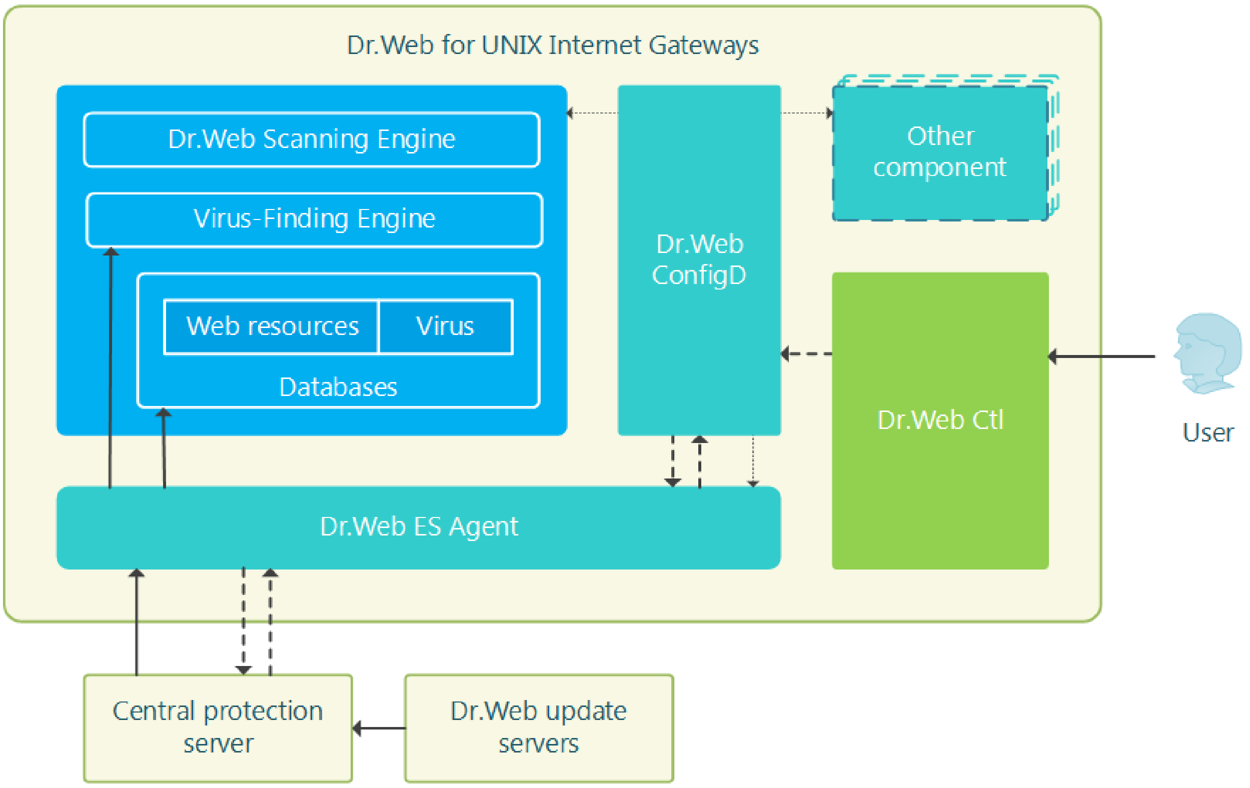


Figure 18. Diagram of the components' operation

To connect Dr.Web ES Agent to the central protection server, the password and identifier of the host ("station" in terms of the Central protection server) are required, as well as the public encryption key file, which is used by the server for authentication. Instead of the station identifier, you can specify the identifier of the main and tariff groups where the station is to be included. For required identifiers and public key file, contact the administrator of your anti-virus network.

Moreover, if this option is allowed on the central protection server, you can connect your host with the protected server ("workstation") as a "newbie". In this case, after the administrator confirms the request to connect, the central protection server automatically generates an identifier and a password, and sends them to the agent for future connections.

Command-Line Arguments

To run Dr.Web ES Agent, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-esagent [<parameters>]
```

Dr.Web ES Agent can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h`

	Arguments: None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-esagent --help
```

This command outputs short help information on Dr.Web ES Agent.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon at the startup of the operating system. To manage the operation of the component, as well as to connect Dr.Web for UNIX Internet Gateways to central protection server, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

> To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-esagent`

Configuration Parameters

The component uses configuration parameters which are specified in the `[ESAgent]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-esagent` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-esagent`

	• For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-esagent`
`DebugIpc` *{Boolean}*	Indicates whether detailed IPC messages are included in the log at debug level (if **`LogLevel`** `= DEBUG`) (interaction of Dr.Web ES Agent and the Dr.Web ConfigD configuration daemon). **Default value:** `No`
`MobileMode` *{On | Off | Auto}*	Indicates whether the program can operate in mobile mode when connected to central protection server. **Allowed values:** • `On`—instructs to use mobile mode if it is allowed by central protection server (that is, perform updates from update servers of Doctor Web via Dr.Web Updater). • `Off`—instructs not to use mobile mode and continue operation in central protection mode (updates are always received from the central protection server). • `Auto`—instructs to use mobile mode, if allowed by central protection server, and perform updates both from update servers of Doctor Web via Dr.Web Updater and from central protection server, depending on which connection is available and which connection quality is higher. *Note that behavior of this parameter depends on server permissions: if mobile mode is not allowed on the used server, this parameter has no effect.* **Default value:** `Auto`
`Discovery` *{On | Off}*	Indicates whether the agent is allowed to receive *discovery* requests from the network inspector built in the central protection server (*discovery* requests are used by the inspector to check the structure and state of the anti-virus network). **Allowed values:** • `On`—allow the agent to receive and process *discovery* requests. • `Off`—prohibit the agent to receive and process *discovery* requests. *Note that this parameter has higher priority than settings of the central protection server: if the parameter value is set to `Off`, the agent does not receive discovery requests even if this option is enabled on the server.* **Default value:** `On`
`UpdatePlatform` *{platform name}*	Indicates the agent to receive updates for the anti-virus engine from the central protection server. The anti-virus engine was developed for the indicated platform, where the *platform name* is a string, which contains the platform name. **Allowed values:** • **GNU/Linux:** `unix-linux-32`, `unix-linux-64`, `unix-linux-mips` • **FreeBSD:** `unix-freebsd-32`, `unix-freebsd-64`

	- **Solaris:** `unix-solaris-32`, `unix-solaris-64` - **Darwin:** `unix-darwin-32`, `unix-darwin-64` It is strongly recommended to change the parameter value only if you are sure it is required. **Default value:** *Depends on the currently used platform*

Dr.Web HTTPD

Dr.Web HTTPD provides infrastructure for local and remote interaction with Dr.Web for UNIX Internet Gateways via HTTP (for example, via a web browser). The component provides two interfaces: an interface to manage the product (in addition to installing the Dr.Web HTTPD, a separate package must also be installed containing the files of the web interface for managing the product via a web browser) and a service interface used by the Dr.Web Link Checker component, which is an extension for **Mozilla Firefox** and **Google Chrome** browsers (installed separately).

Besides managing Dr.Web for UNIX Internet Gateways through the product’s web interface, it is also possible to use the command interface (HTTP API) of Dr.Web HTTPD directly to interact with the components of Dr.Web for UNIX Internet Gateways via HTTPS. This capability allows you to create your own interface to manage Dr.Web for UNIX Internet Gateways.

To use a secure HTTPS connection it is required to provide an appropriate SSL server certificate and private key for Dr.Web HTTPD. By default, an SSL server certificate and an SSL private key are generated for Dr.Web HTTPD automatically during the installation procedure, but, if necessary, you can generate your own certificate and key. If you need to generate SSL keys and certificates, you can use the **openssl** utility. An example of how to use the **openssl** utility to generate a certificate and a private key is given in the section Appendix E. Generating SSL certificates.

Operating Principles

Dr.Web HTTPD is a web server for managing the operation of Dr.Web for UNIX Internet Gateways. With Dr.Web HTTPD, it is possible not to use external web servers (for example, **Apache HTTP Server** or **Nginx**) and management services like **Webmin**. Moreover, the component can function simultaneously with such servers and services on the same host and not impede their operation at that.

The Dr.Web HTTPD server processes requests received via HTTP and HTTPS protocols to the sockets specified in the settings. For this reason, the server does not have any conflicts with web servers when they operate on the same host. The operation scheme is shown in the figure below.

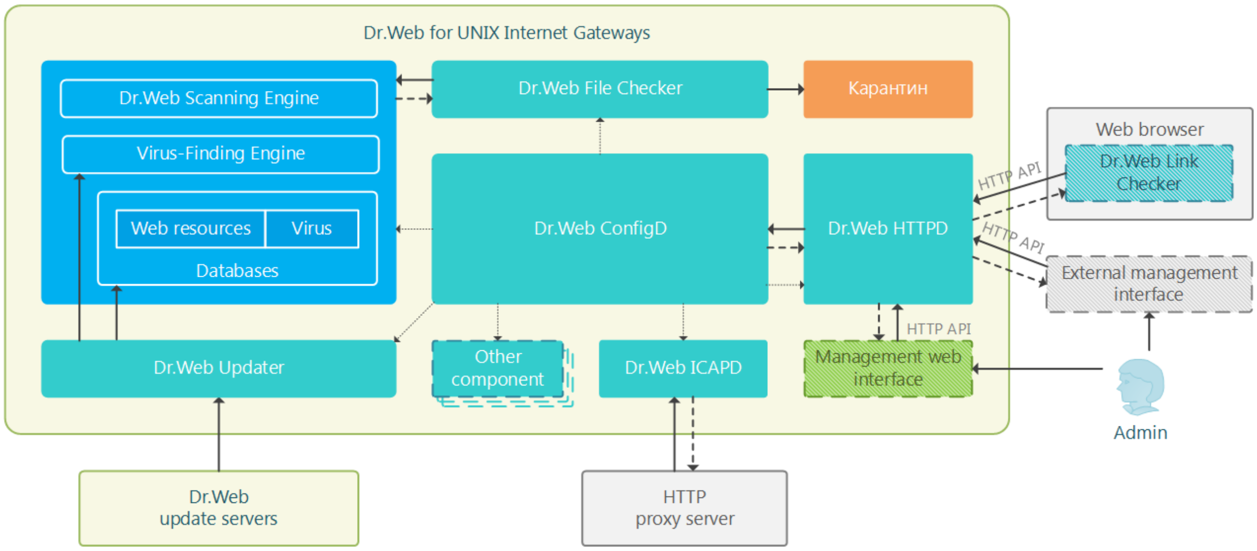


Figure 19. Diagram of the components' operation

HTTPS is used for product management; and HTTP, for processing requests from Dr.Web Link Checker—a web browser extension—(it is installed to the browser separately).

It is not mandatory to install Dr.Web management web interface and Dr.Web Link Checker extension for the proper functioning of the product. They can be missing. This is why the corresponding blocks are circled with a dashed line.

The Dr.Web HTTPD component issues commands to the Dr.Web for UNIX Internet Gateways Dr.Web ConfigD configuration daemon, as well as to the Dr.Web File Checker monitor. These commands are based on those that were received through the provided HTTP API (including those that were made via the management web interface or those made as requests from the Dr.Web Link Checker browser extension).

If the management web interface of Dr.Web for UNIX Internet Gateways, which uses Dr.Web HTTPD, is included in the product, it is described in the corresponding section.

If the Dr.Web's management web interface is not included in the product, it is possible to connect any external management interface to the product. For interaction with the product's components, this interface should use the HTTP API provided by Dr.Web HTTPD (not described in this user manual).

To obtain the HTTP API Dr.Web HTTPD manual, refer to Doctor Web partners care department (https://partners.drweb.com/).

Command-Line Arguments

To launch Dr.Web HTTPD from the command line of the operating system, the following command is used:

```
$ <opt_dir>/bin/drweb-httpd [<parameters>]
```

Dr.Web HTTPD can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-httpd --help
```

This command outputs short help information on Dr.Web HTTPD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when required (usually at the startup of the operating system). If the component is running and the web interface is installed, then to manage the components of Dr.Web for UNIX Internet Gateways, you can simply use any standard web-browser to access, via HTTPS, any of the addresses at which the web-interface is served. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-httpd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[HTTPD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** *<opt_dir>*`/bin/drweb-httpd` - For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-httpd` - For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-httpd`
`Start` *{Boolean}*	The component must be launched by the Dr.Web ConfigD configuration daemon. When you specify the `Yes` value for this parameter, it instructs the configuration daemon to start the component immediately; and when you specify the `No` value, it instructs the configuration daemon to terminate the component immediately. **It depends on whether product management interface is installed.**

`WebConsoleAddress`
{address, …}
	List of network sockets (every network socket specified as *<IP address>:<port>*) on which Dr.Web HTTPD is listening for connections to the web interface provided for managing the product, if the web interface is installed. Access to the web-interface at these Addresses is provided via HTTPS.
	If no value is specified, it is impossible to use the web interface.
	Default value: `127.0.0.1`

`LinkCheckerAddress`
{address, …}
	List of network sockets (every network socket consists of *<IP address>:<port>*) on which Dr.Web HTTPD is listening for connections (via HTTP) from the Dr.Web Link Checker browser extension that checks webpages loaded by the browser for malicious objects.
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The

	parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add sockets 192.168.0.1:1234 and 10.20.30.45:5678 to the list.
	1. Adding of values to the configuration file.
	- Two values in one string
	`[<%HTTPD_SECTION%>]`
	` LinkCheckerAddress = "192.168.0.1:1234", "10.20.30.45:5678"`
	- Two strings (one value per a string)
	`[<%HTTPD_SECTION%>]`
	` LinkCheckerAddress = 192.168.0.1:1234`
	` LinkCheckerAddress = 10.20.30.45:5678`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset <%HTTPD_SECTION%>.LinkCheckerAddress -a 192.168.0.1:1234`
	`# drweb-ctl cfset <%HTTPD_SECTION%>.LinkCheckerAddress -a 10.20.30.45:5678`
	If no value is specified, it is impossible to use the Dr.Web Link Checker browser extension. Note that at these addresses (sockets) you cannot access the web-interface provided for managing the product.
	Default value: *(not specified)*

`ServerSslCertificate` *{path to file}*	Path to the file with the server certificate used by the web interface server for communication with other hosts via HTTPS. *This file is generated automatically during the installation of the component.* *Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.* **Default value:** *<etc_dir>*`/certs/serv.crt` - For **Linux**, **Solaris**: `/etc/opt/drweb.com/certs/serv.crt` - For **FreeBSD**: `/usr/local/etc/drweb.com/certs/serv.crt`
`ServerSslKey` *{path to file}*	Path to the private key file used by the web interface server for communication with other hosts via HTTPS. *This file is generated automatically during the installation of the component.* *Please note that the certificate file (which is specified by the previous discussed parameter) and the private key file must form a matching pair.* **Default value:** *<etc_dir>*`/certs/serv.key`

	• For **Linux**, **Solaris**: `/etc/opt/drweb.com/certs/serv.key` • For **FreeBSD**: `/usr/local/etc/drweb.com/certs/serv.key`
`WebconsoleRoot` *{path to directory}*	Path to the directory which stores the files used by the management web interface if it is installed (similar to the `htdocs` directory of an **Apache HTTP Server**). **Default value:** <*opt_dir*>`/share/drweb-httpd/webconsole` • For **Linux**, **Solaris**: `/opt/drweb.com/share/drweb-httpd/webconsole` • For **FreeBSD**: `/usr/local/libexec/drweb.com/share/drweb-httpd/webconsole`
`LinkcheckerRoot` *{path to directory}*	Path to the directory which stores the files used by Dr.Web Link Checker web browser extension. **Default value:** <*opt_dir*>`/share/drweb-httpd/linkchecker` • For **Linux**, **Solaris**: `/opt/drweb.com/share/drweb-httpd/linkchecker` • For **FreeBSD**: `/usr/local/libexec/drweb.com/share/drweb-httpd/linkchecker`
`AccessLogPath` *{path to file}*	Path to the file where all HTTP/HTTPS requests from clients to the web interface server are registered. *If not specified, HTTP/HTTPS requests are not logged to a file.* **Default value:** *(not specified)*

Dr.Web SNMPD

Dr.Web SNMP agent (Dr.Web SNMPD) is designed for integration of Dr.Web for UNIX Internet Gateways suite with monitoring systems via SNMP. Such integration will allow to control operational status of Dr.Web for UNIX Internet Gateways as well as collect statistics on detected and neutralized threats. The agent provides the following information to monitoring systems and SNMP managers:

- State of a program component
- Number of detected threats of various types (according to the Dr.Web classification)

Moreover, the agent sends SNMP trap notifications upon detection of a threat and upon failures in neutralization of detected threats. The agent supports SNMP protocol of version 2c and 3.

Description of the information which can be sent by the agent is stored in a special section of MIB (*Management Information Base*) created by Doctor Web. In the MIB section, defined by Dr.Web for UNIX-like operating systems, the following information is specified:

1. Formats of SNMP trap notifications about detection and neutralizing of threats and about errors related to the program components.
2. Operation statistics of the program components:
3. Current state of the program components

For more details about information that can be obtained over the SNMP protocol, refer to the corresponding section.

Operating Principles

By default, the component is run automatically upon Dr.Web for UNIX Internet Gateways startup. When run, the component structures data according to the structure described in MIB Dr.Web and waits for requests to receive data from external SNMP managers. The component receives information on the status of the program components and notifications on detected threats directly from the Dr.Web ConfigD configuration daemon, as shown in the figure below.

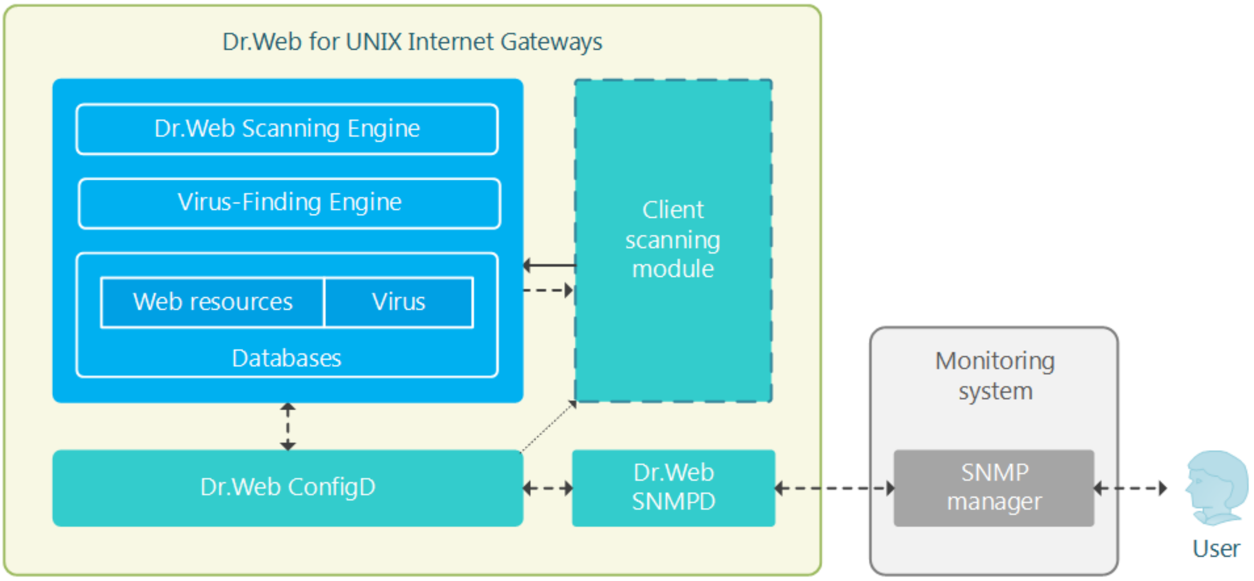


Figure 20. Diagram of the components' operation

Threats can be detected by the scanning engine during scanning initiated by Dr.Web for UNIX Internet Gateways components; thus, the scheme contains an abstract “client scanning module”. On threat detection, the appropriate count (of this threat type) is incremented by one and all SNMP managers that can receive notifications get an SNMP trap notifying on the detected threat.

> Collected values of counters (for example, counters of detected threats) are not saved between launches of Dr.Web SNMPD. Thus, once Dr.Web SNMPD is relaunched for any reason (including general restart of Dr.Web for UNIX Internet Gateways), the collected values of counters are reset to 0.

Integration with the System SNMP Agent

To enable correct operation of Dr.Web SNMP agent if the main system SNMP agent **snmpd** (**net-snmp**), already operates on the server, configure transmission of SNMP requests through the Dr.Web MIB branch from **snmpd** to Dr.Web SNMPD. For that purpose, edit the **snmpd** configuration file (usually for **Linux** the file is as follows: `/etc/snmp/snmpd.conf`), by adding the following line in it:

```
proxy -v <version> -c <community> <address>:<port> <MIB branch>
```

Where:

- *<version>* – SNMP version in use (`2c`, `3`).
- *<community>*—“community string” used by Dr.Web SNMPD.
- *<address>:<port>*—network socket which is listened by Dr.Web SNMPD.
- *<MIB branch>*—OID of the MIB branch containing descriptions of variables and SNMP notifications (*trap*) used by Dr.Web (the OID equals `.1.3.6.1.4.1.29690`).

If you are using the default settings of Dr.Web SNMP agent, then the added line should look like this:

```
proxy -v 2c -c public localhost:50000 .1.3.6.1.4.1.29690
```

Note that since port 161 in this case will be used by the system's standard **snmpd**, it is required to specify another port for Dr.Web SNMPD in the ListenAddress parameter (in this example, `50000`).

Command-Line Arguments

To launch Dr.Web SNMPD from the command line of the operating system, the following command is used:

```
$ <opt_dir>/bin/drweb-snmpd [<parameters>]
```

Dr.Web SNMPD can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-snmpd --help
```

This command outputs short help information on Dr.Web SNMPD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed (as a rule, at the startup of the operating system). To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-snmpd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[SNMPD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel` *{logging level}*	Logging level of the component. If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/bin/drweb-snmpd` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-snmpd` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-snmpd`
`Start` *{Boolean}*	The component must be launched by the Dr.Web ConfigD configuration daemon. When you specify the `Yes` value for this parameter, it instructs the configuration daemon to start the component immediately; and when you specify the `No` value, it instructs the configuration daemon to terminate the component immediately. **Default value**: `No`
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`.

	When a user name is not specified, the component operation terminates with an error after the startup. **Default value**: `drweb`
`ListenAddress` *{address}*	Address (IP address and port) listened by Dr.Web SNMPD, which is waiting for client connections (SNMP managers). *Note that interaction with* **snmpd** *requires a specified port, different from the standard port (161), and* **snmpd** *must be configured for proxying.* **Default value:** `127.0.0.1:161`
`SnmpVersion` *{V2c | V3}*	The used SNMP protocol version (*SNMPv2c* or *SNMPv3*). **Default value**: `V2c`
`V3EngineId` *{string}*	Identifier (string) of *Engine ID* for *SNMPv3* (according to RFC 3411). **Default value:** `800073FA044452574542`

`TrapReceiver`
{address list}
	List of addresses (IP address and port) where Dr.Web SNMPD sends *SNMP trap* notifications after Dr.Web for UNIX Internet Gateways components detected a threat.
	You can specify a list as the parameter value. The values in the list must be separated with commas (each value in the quotation marks). The parameter can be specified more than once in the section (in this case, all its values are combined into one list).
	Example: Add sockets 192.168.0.1:1234 and 10.20.30.45:5678 to the list.
	1. Adding of values to the configuration file.
	• Two values in one string
	`[SNMPD]`
	` TrapReceiver = "192.168.0.1:1234",`
	`"10.20.30.45:5678"`
	• Two strings (one value per a string)
	`[SNMPD]`
	` TrapReceiver = 192.168.0.1:1234`
	` TrapReceiver = 10.20.30.45:5678`
	2. Adding values via the command **drweb-ctl** `cfset`.
	`# drweb-ctl cfset SNMPD.TrapReceiver -`
	`a 192.168.0.1:1234`
	`# drweb-ctl cfset SNMPD.TrapReceiver -`
	`a 10.20.30.45:5678`

	Default value: *(not set)*
V2cCommunity *{string}*	The string “SNMP read community” for authentication of SNMP managers (*SNMPv2c* protocol) when Dr.Web MIB variables are accessed for reading. *The parameter is used if* **SnmpVersion** = `V2c`. **Default value:** `public`
V3UserName *{string}*	The user name for authentication of SNMP managers (*SNMPv3* protocol) when Dr.Web MIB variables are accessed for reading. *The parameter is used if* **SnmpVersion** = `V3`. **Default value :** `noAuthUser`
V3Auth *{SHA(<pwd>) | MD5(<pwd>) | None}*	Method to authenticate SNMP managers (*SNMPv3* protocol) when Dr.Web MIB variables are accessed for reading. **Allowed values:** • `SHA(`*<PWD>*`)`—SHA hash of the password is used (<*PWD*> strings). • `MD5(`*<PWD>*`)`—MD5 hash of the password is used (<*PWD*> strings). • `None`—authentication is disabled. where <*PWD*> is a plain text password. When specifying the parameter value from the command line, you may need to escape the brackets by using the slash mark `\` in some shells. **Examples:** 1. Parameter value in the configuration file: **`V3Auth`** `= MD5(123456)` 2. Specifying the same parameter value from the command line via command **drweb-ctl** `cfset`: **`drweb-ctl`** `cfset SNMPD.V3Auth MD5\(123456\)` *The parameter is used if* **SnmpVersion** = `V3`. **Default value :** `None`
V3Privacy *{DES(<secret>) | AES128(<secret>) | None}*	Method to encrypt SNMP messages (*SNMPv3* protocol). **Allowed values:** • `DES(`*<secret>*`)`—DES encryption algorithm is used.

	• `AES128(`<*secret*>`)`—AES128 encryption algorithm is used. • `None`—SNMP-messages are not encrypted. where <*secret*> is a secret key shared by the manager and the agent (*plain text*). When specifying the parameter value from the command line, you may need to escape the brackets by using the slash mark `\` in some shells. **Examples:** 1. Parameter value in the configuration file: **`V3Privacy`** `= AES128(supersecret)` 2. Specifying the same parameter value from the command line via command **drweb-ctl** `cfset`: **`drweb-ctl`** `cfset SNMPD.V3Privacy AES128\(supersecret\)` *The parameter is used if* **`SnmpVersion`** `= V3`. **Default value:** `None`

Integration with SNMP Monitoring Systems

Dr.Web SNMP agent can perform functions of a data provider for any monitoring system that uses SNMP protocol version *2c* or *3*. The list of available data and their structure are provided in a `Dr.Web MIB description file called DrWeb-Snmpd.mib`, supplied with the product. This file is located in the <*opt_dir*>`/share/drweb-snmpd/mibs` directory.

For easy configuration, the component is supplied with templates of settings for popular monitoring systems:

- Munin
- Nagios
- Zabbix

Customization templates for monitoring systems are located in the <*opt_dir*>`/share/drweb-snmpd/connectors` directory.

Integration with Munin Monitoring System

The **Munin** monitoring system includes the central server (master) **munin**, which collects statistics from clients **munin-node** residing locally on the monitored hosts. At request of the server, each monitoring client collects data about monitored host operation by starting *plug-ins* that provide data transferred to the server.

To enable connection between Dr.Web SNMPD and the **Munin** monitoring system, a ready-to-use plug-in Dr.Web used by **munin-node** is supplied. The plug-in resides in the *<opt_dir>*`/share/drweb-snmpd/connectors/munin/plugins` directory. This plug-in collects data required for construction of the following two graphs:

- Number of detected threats
- File scan statistics
- Email message scanning statistics.

These plug-ins support SNMP protocols versions 1, 2c and 3. Based on these template plug-ins, you can create any other plug-ins to poll the status of Dr.Web for UNIX Internet Gateways components via Dr.Web SNMPD.

In the *<opt_dir>*`/share/drweb-snmpd/connectors/munin` directory, the following files are located.

File	Description
`plugins/snmp__drweb_malware`	The **munin-node** plug-in for polling Dr.Web SNMPD via SNMP to gather the number of threats detected by Dr.Web on the host.
`plugins/snmp__drweb_filecheck`	The **munin-node** plug-in for polling Dr.Web SNMPD via SNMP to gather the statistics of files scanned by Dr.Web on the host.
`plugins/snmp__drweb_maild_multi`	The **munin-node** plug-in used for polling Dr.Web SNMPD via SNMP to gather the statistics of email messages scanned by Dr.Web on the host. Note that this plug-in uses the *multigraph*, a feature available in **Munin** version above 1.4.
`plugin-conf.d/drweb.cfg`	An example of the **munin-node** configuration for the environment variables of the Dr.Web plug-ins.

Connecting a host to Munin

In the present instruction, it is assumed that the **Munin** monitoring system is already deployed on the monitoring server and the monitored host features an installed and functioning Dr.Web SNMPD (it is possible for the component to function in proxy mode together with **snmpd**) and **munin-node**.

1. Monitored host configuration
   - Copy the `snmp__drweb_*` files to the directory with plug-in libraries **munin-node** (the directory depends on the OS). For example, in **Debian/Ubuntu** operating systems, the path is `/usr/share/munin/plugins`.

- Configure **munin-node** by connecting to it the supplied Dr.Web plug-ins. To do this, use the **munin-node-configure** utility that is distributed with **munin-node**.

  For example, the following command

  ```
  $ munin-node-configure --shell --snmp localhost
  ```

  will display on a terminal screen a list of commands for creation of required symbolic links to plug-ins. Copy and execute them in the command line. Note that the specified command presumes that:

  1) **munin-node** is installed at the same host where Dr.Web SNMPD is installed. If it is not the case, please specify the correct FQDN or an IP address of the monitored host instead of a `localhost` value;
  2) Dr.Web SNMPD uses SNMP version 2c. If it is not the case, specify the correct SNMP version in **munin-node-configure** command. The command has several arguments for flexible configuration of plug-ins, e.g., you can specify the SNMP protocol version, port that is listened by SNMP agent at the monitored host, an actual value of the *community string*, and so on. If required, refer to the manual on **munin-node-configure** command.

- If necessary, define (or redefine) parameter values of the environment, where installed Dr.Web plug-ins for **munin-node** must be executed. As the environment parameters, the value *community string* is used. It is the port utilized by the SNMP agent, etc. These parameters must be defined in the file `/etc/munin/plugin-conf.d/drweb` (create it if required). As an example of this file, use the supplied file `drweb.cfg`.

- In the **munin-node** configuration file (`munin-node.conf`), specify a regular expression to include all IP addresses of hosts that are allowed to connect **munin** servers (masters) to **munin-node** for receiving the values of the monitored parameters, for example:

  ```
  allow ^10\.20\.30\.40$
  ```

  In this case, only the IP address `10.20.30.40` is allowed to receive host parameters.

- Restart **munin-node**, for example, by using the following command:

  ```
  # service munin-node restart
  ```

2. **Munin** server (master) configuration

   Add the address and identifier of the monitored host to the **Munin** configuration file `munin.conf`, which is located, by default, in `/etc` directory (in **Debian/Ubuntu** operating systems it is `/etc/munin/munin.conf`):

   ```
   [<ID>;<hostname>.<domain>]
   address <host IP address>
   use_node_name yes
   ```

   where *<ID>* is the displayed host's identifier, *<hostname>* is the name of the host, *<domain>* is the name of the domain, *<host IP address>* is the IP address of the host.

For official documentation on configuration of the **Munin** monitoring system, refer to http://munin.readthedocs.io.

Integration with Zabbix Monitoring System

File templates, required for establishing connection between Dr.Web SNMPD and the **Zabbix** monitoring system, are located in the <*opt_dir*>`/share/drweb-snmpd/connectors/zabbix` directory.

File	Description
`zbx_drweb.xml`	Template for description of the monitored host that features installed Dr.Web.
`snmptt.drweb.zabbix.conf`	Configuring the **snmptt** utility—which is an *SNMP trap*handler

Template for description of the monitored host features:

- Description of counters ("*items*", according to the terminology of **Zabbix**). By default, the template is set to be used with SNMP v2.
- The set of predefined graphs: number of scanned files and distribution of detected threats by their type.

Connecting a host to Zabbix

In the present instruction, it is assumed that the **Zabbix** monitoring system is already deployed on the monitoring server and the monitored host features an installed and functioning Dr.Web SNMPD (it is possible for the component to function in proxy mode together with **snmpd**). Moreover, if you want to receive *SNMP trap* notifications from the monitored host (including notification on threats detected by Dr.Web for UNIX Internet Gateways on a protected server), install the `net-snmp` package on the monitoring server (standard tools **snmptt** and **snmptrapd** are used).

1. In the **Zabbix** web interface, on the **Configuration** –> **Templates** tab import the template of the monitored host from the <*opt_dir*>`/share/drweb-snmpd/connectors/zabbix/zbx_drweb.xml` file.
2. Add the monitored host to the appropriate list (at **Hosts** –> **Create host**). Specify correct parameters of the host and settings of the SNMP interface (they must match the settings of **drweb-snmpd** and **snmpd** on the host):
   - The **Host** tab:

     Host name: *drweb-host*

     Visible name: *DRWEB_HOST*

     Groups: select *Linux servers*

     Snmp interfaces: Click **add** specify the IP address and port are used by Dr.Web SNMPD (it is considered that Dr.Web SNMPD operates on the local host, so the address *127.0.0.1* and the port *161* are specified by default).

- The **Templates** tab:

  Click **Add**, check *DRWEB*, click **select**.

- The **Macros** tab:

  Macro: *{$SNMP_COMMUNITY}*

  Value: specify “read community” for SNMP V2c (by default, *public*).

  Click **Save**.

  Note: The *{$SNMP_COMMUNITY}* macro can be specified directly in the host template.

> By default, the imported *DRWEB* template is configured for SNMP v2. If you need to use another version of SNMP, edit the template accordingly on the appropriate page.

3. After the template is bound to the monitored host, if SNMP settings are specified correctly, the **Zabbix** monitoring system will start to collect data for counters (*items*) of the template; the collected data will be displayed on the **Monitoring** –> **Latest Data** and **Monitoring** –> **Graphs tabs**.

4. A special *item drweb-traps* is used for collecting *SNMP trap* notifications from Dr.Web SNMPD. The log pf received *SNMP trap* notifications is available on the **Monitoring** –> **Latest Data** –> **drweb-traps** –> **history** page. To collect notifications, **Zabbix** uses standard tools **snmptt** and **snmptrapd** from the `net-snmp` package. For details on how to configure the tools for receiving *SNMP trap* notifications from Dr.Web SNMPD, see below.

5. If necessary, you can configure a trigger that will change its state upon receiving an *SNMP trap* notification from Dr.Web SNMPD. Changing of its state can be used as an event source for generation appropriate notifications. The example below shows an expression for configuration of a trigger; the expression is specified in the **trigger expression** field:

   - For **Zabbix** *2.x*:

   ```
   ({TRIGGER.VALUE}=0 &
   {DRWEB:snmptrap[.*\.1\.3\.6\.1\.4\.1\.29690\..*].nodata(60)}=1 )|
   ({TRIGGER.VALUE}=1 &
   {DRWEB:snmptrap[.*\.1\.3\.6\.1\.4\.1\.29690\..*].nodata(60)}=0)
   ```

   - For **Zabbix** *3.x*:

   ```
   ({TRIGGER.VALUE}=0 and {drweb-host:snmptrap[".29690."].nodata(60)}=1 ) or
   ({TRIGGER.VALUE}=1 and {drweb-host:snmptrap[".29690."].nodata(60)}=0 )
   ```

   An event is triggered (the value is set to 1) if the log of *SNMP trap* notifications from Dr.Web SNMPD was updated within a minute. If the log was not updated within the next minute, the value of the trigger is set to 0 again.

   It is recommended to set in the **Severity** field for this trigger a notification type that is differ from *Not classified* value, for example, *Warning*.

Configuring Receipt of SNMP trap notifications for Zabbix

1. On the monitored host, in Dr.Web SNMPD settings (the `TrapReceiver` parameter), you should specify an address to be listened by **snmptrapd** on the host where **Zabbix** operates, for example:

   ```
   SNMPD.TrapReceiver = 10.20.30.40:162
   ```

2. In the configuration file of **snmptrapd** (`snmptrapd.conf`), specify the same address and an application for processing received *SNMP trap* notifications (in this example, **snmptthandler**, **snmptt** component):

   ```
   snmpTrapdAddr 10.20.30.40:162
   traphandle default /usr/sbin/snmptthandler
   ```

   Add the following string to the file, so that **snmptt** does not discard *SNMP trap* sent by Dr.Web SNMPD as unknown:

   ```
   outputOption n
   ```

3. The **snmptthandler** component saves received *SNMP trap* notifications to the file on the disk in accordance with the specified format, which corresponds to the regular expression set in the host template for **Zabbix** (*drweb-traps* item). The format of the saved notification is specified in the *<opt_dir>*`/share/drweb-snmpd/connectors/zabbix/snmptt.drweb.zabbix.conf`. file. The file must be copied to `/etc/snmp`.
4. Moreover, the path to the format files must be specified in the `snmptt.ini`:

   ```
   [TrapFiles]
   # A list of snmptt.conf files (this is NOT the snmptrapd.conf file).
   # The COMPLETE path and filename. Ex: '/etc/snmp/snmptt.conf'
   snmptt_conf_files = <<END
   /etc/snmp/snmptt.conf
   /etc/snmp/snmptt.drweb.zabbix.conf
   END
   ```

   After that, restart **snmptt** if it was started in daemon mode.

5. In the configuration file of the **Zabbix** server (`zabbix-server.conf`), specify (or check if they are already specified) the following settings:

   ```
   SNMPTrapperFile=/var/log/snmptt/snmptt.log
   StartSNMPTrapper=1
   ```

   where `/var/log/snmptt/snmptt.log` is a log file used by **snmptt** to register information on received SNMP trap notifications.

For official documentation on **Zabbix**, refer to https://www.zabbix.com/documentation/.

Integration with Nagios Monitoring System

Files with configuration examples, required for establishing connection between Dr.Web SNMPD and the **Nagios** monitoring system, are located in the *<opt_dir>*`/share/drweb-snmpd/connectors/nagios` directory.

File	Description
`nagiosgraph/rrdopts.conf-sample`	Example of the RRD configuration file
`objects/drweb.cfg`	Configuration file describing *drweb* objects
`objects/nagiosgraph.cfg`	The configuration file of the component for graph plotting used by **Nagiosgraph** used by **Nagios**
`plugins/check_drweb`	The script for collecting data from the host on which Dr.Web is installed
`plugins/eventhandlers/submit_check_result`	The script for handling *SNMP trap* notifications
`snmp/snmptt.drweb.nagios.conf`	Configuring the **snmptt** utility—which is an *SNMP trap*handler

Connecting a host to Nagios

In the present instruction, it is assumed that the **Nagios** monitoring system is already deployed on the monitoring server, including configuration of the web server and the graphical tool **Nagiosgraph**, and the monitored host features an installed and functioning Dr.Web SNMPD (it is possible for the component to function in proxy mode together with **snmpd**). Moreover, if you want to receive *SNMP trap* notifications from the monitored host (including notification on threats detected by Dr.Web for UNIX Internet Gateways on a protected server), install the `net-snmp` package on the monitoring server (standard tools **snmptt** and **snmptrapd** are used).

In the current manual, the following path conventions are used (real paths depend on the operating system and **Nagios** installation):

- *<NAGIOS_PLUGINS_DIR>*—directory with **Nagios**plug-ins, for example: `/usr/lib64/nagios/plugins`.
- *<NAGIOS_ETC_DIR>*—directory with **Nagios** settings, for example: `/etc/nagios`.
- *<NAGIOS_OBJECTS_DIR>*—directory with **Nagios** objects, for example: `/etc/nagios/objects`.
- *<NAGIOSGRAPH_DIR>*—**Nagiosgraph** directory, for example: `/usr/local/nagiosgraph`.
- *<NAGIOS_PERFDATA_LOG>*—file where **Nagios** records results of service check (must be the same as the `perflog` file from *<NAGIOSGRAPH_DIR>*`/etc/nagiosgraph.conf`). Records

from this file are read by the <*NAGIOSGRAPH_DIR*>/bin/insert.pl script and are recorded to the corresponding RRA archives **RRD Tool**.

Configuring **Nagios**:

1. Copy the `check_drweb` file to the <*NAGIOS_PLUGINS_DIR*> directory and the `drweb.cfg` file to the <*NAGIOS_OBJECTS_DIR*> directory.
2. Add hosts with Dr.Web that are to be monitored to the *drweb* group. On the hosts Dr.Web SNMPD must be running. By default, only *localhost* is added to this group.
3. If required, edit the `check_drweb` command which contains instruction to contact Dr.Web SNMPD on *drweb* hosts via the **snmplwalk** tool:

   ```
   snmpwalk -c public -v 2c $HOSTADDRESS$:161
   ```

   specify the correct version of SNMP protocol and parameters (such as “*community string*” or authentication parameters) as well as the port. The `$HOSTADDRESS$` variable must be included in the command (as this variable is later automatically substituted by **Nagios** to the correct host address when the command is invoked). OID is not required in the command. It is also recommended that you specify the command together with the full path to the executable file (usually `/usr/local/bin/snmpwalk`).
4. Connect *DrWeb* objects in the <*NAGIOS_ETC_DIR*>/nagios.cfg configuration file by adding the following string to the file:

   ```
   cfg_file= <NAGIOS_OBJECTS_DIR>/drweb.cfg
   ```

5. Add **RRD Tool** settings for *DrWeb* graphics from the `rrdopts.conf-sample` file to the <*NAGIOSGRAPH_DIR*>/etc/rrdopts.conf file.
6. If **Nagiosgraph** is yet to be configured, do the following steps for its configuration:
   - Copy the `nagiosgraph.cfg` file to the <*NAGIOS_OBJECTS_DIR*> directory and edit the path to the `insert.pl` script in the **process-service-perfdata-for-nagiosgraph** command; for example, as follows:

   ```
   $ awk '$1 == "command_line" { $2 = "<NAGIOSGRAPH_DIR>/bin/insert.pl" }
   { print }' ./objects/nagiosgraph.cfg > <NAGIOS_OBJECTS_DIR>/nagiosgraph.cfg
   ```

   - Connect this file in the <*NAGIOS_ETC_DIR*>/nagios.cfg configuration file by adding the following line to it:

   ```
   cfg_file=<NAGIOS_OBJECTS_DIR>/nagiosgraph.cfg
   ```

7. Check values of **Nagios** parameters in the <*NAGIOS_ETC_DIR*>/nagios.cfg configuration file:

```
check_external_commands=1
execute_host_checks=1
accept_passive_host_checks=1
enable_notifications=1
enable_event_handlers=1

process_performance_data=1
service_perfdata_file=/usr/nagiosgraph/var/rrd/perfdata.log
service_perfdata_file_template=$LASTSERVICECHECK$||$HOSTNAME$||$SERVICEDE
SC$||$SERVICEOUTPUT$||$SERVICEPERFDATA$
service_perfdata_file_mode=a
service_perfdata_file_processing_interval=30
service_perfdata_file_processing_command=process-service-perfdata-for-
nagiosgraph

check_service_freshness=1
enable_flap_detection=1
enable_embedded_perl=1
enable_environment_macros=1
```

Configuring Receipt of SNMP trap notifications for Nagios

1. On the monitored host in Dr.Web SNMPD settings (the **TrapReceiver** parameter), specify an address to be listened by **snmptrapd** on the host where **Nagios** operates, for example:

   ```
   SNMPD.TrapReceiver = 10.20.30.40:162
   ```

2. Check for existing the <*NAGIOS_PLUGINS_DIR*>/eventhandlers/submit_check_result script which will be invoked when *SNMP trap* is received. If the script is missing, copy the submit_check_result file to this location from the <*opt_dir*>/share/drweb-snmpd/connectors/nagios/plugins/eventhandlers/ directory. In this file, change the path specified in the CommandFile parameter. It must have the same value as the command_file parameter in the <*NAGIOS_ETC_DIR*>/nagios.cfg file.

3. Copy the snmptt.drweb.nagios.conf file to the /etc/snmp/snmp/ directory. In this file, change the path to the submit_check_result—for example, by using the following command:

   ```
   $ awk '$1 == "EXEC" { $2 =
   <NAGIOS_PLUGINS_DIR>/eventhandlers/submit_check_result }{ print}'
   ./snmp/snmptt.drweb.nagios.conf > /etc/snmp/snmp/snmptt.drweb.nagios.conf
   ```

4. Add the " /etc/snmp/snmptt.drweb.nagios.conf" string to the /etc/snmp/snmptt.drweb.nagios.conf file. After that, restart **snmptt** if it was started in daemon mode.

After all required configuration files of **Nagios** are added and edited, run **Nagios** in debug mode by using the following command:

```
# nagios -v <NAGIOS_ETC_DIR>/nagios.cfg
```

Upon receipt of this command, **Nagios** will check for configuration errors. If no error is found, **Nagios** can be restarted as usual (for example, by using the **`service`** `nagios restart` command).

For official documentation on **Nagios**, refer to http://www.nagios.org/documentation/.

Dr.Web SNMP MIB

The list of operating parameters of Dr.Web for UNIX Internet Gateways that can be fetched by external monitoring systems over the SNMP protocol is provided in the table below.

Parameter name	**OID of the parameter**	**Type and description of the parameter**
Common prefix for all names: `.iso.org.dod.internet.private.enterprises.drweb.drwebSnmpd` Common prefix for all OIDs: `.1.3.6.1.4.1.29690.2`		
alert	**Asynchronous notifications about events (SNMP traps)**	
threatAlert	`.1.1`	Notification about detecting a threat
threatAlertFile	`.1.1.1`	Name of the infected file (string)
threatAlertType	`.1.1.2`	Threat type (integer *)
threatAlertName	`.1.1.3`	Name of the threat (string)
threatAlertOrigin	`.1.1.4`	Identifier of the component that detected the threat (integer***)
threatActionErrorAlert	`.1.2`	Notification about an error occurred when trying to neutralize the threat
threatActionErrorAlertFile	`.1.2.1`	Name of the infected file (string)
threatActionErrorAlertType	`.1.2.2`	Threat type (integer *)
threatActionErrorAlertName	`.1.2.3`	Name of the threat (string)
threatActionErrorAlertOrigin	`.1.2.4`	Identifier of the component that detected the threat (integer***)
threatActionErrorAlertError	`.1.2.5`	Description of an error (string)
threatActionErrorAlertErrorCode	`.1.2.6`	Error code (integer corresponding to code from error catalogue)

Parameter name	**OID of the parameter**	**Type and description of the parameter**
threatActionErrorAlertAction	`.1.2.7`	Failed action (1—cure; 2—move to quarantine; 3—delete; 4—report; 5—ignore)
componentFailureAlert	`.1.3`	Notification about a component failure
componentFailureAlertName	`.1.3.1`	Component identifier (integer***)
componentFailureAlertExitCodeDescription	`.1.3.2`	Component exit code description (string)
componentFailureAlertExitCode	`.1.3.3`	Error code (integer corresponding to code from error catalogue)
infectedUrlAlert	`.1.4`	Notification about blocking a malicious URL (for HTTP/HTTPS connections)
infectedUrlAlertUrl	`.1.4.1`	The URL that was blocked (string)
infectedUrlAlertDirection	`.1.4.2`	HTTP message direction (integer: 1—request, 2—response)
infectedUrlAlertType	`.1.4.3`	Threat type (integer *)
infectedUrlAlertName	`.1.4.4`	Name of the threat (string)
infectedUrlAlertOrigin	`.1.4.5`	Identifier of the component that detected the threat (integer***)
infectedUrlAlertSrcIp	`.1.4.6`	IP address of connection source (string)
infectedUrlAlertSrcPort	`.1.4.7`	Port of connection source (integer)
infectedUrlAlertDstIp	`.1.4.8`	IP address of connection destination point (string)
infectedUrlAlertDstPort	`.1.4.9`	Port of connection destination point (integer)
infectedUrlAlertSniHost	`.1.4.10`	SNI of connection destination point (for SSL connections) (string)

Parameter name	OID of the parameter	Type and description of the parameter
infectedUrlAlertExePath	`.1.4.11`	Executable path of the program that establish the connection (string)
infectedUrlAlertUserName	`.1.4.12`	Name of the user with whose privileges is executing the program that establish the connection (string)
infectedAttachmentAlert	`.1.5`	Notification about detecting an infected email attachment
infectedAttachmentAlertType	`.1.5.1`	Threat type (integer *)
infectedAttachmentAlertName	`.1.5.2`	Name of the threat (string)
infectedAttachmentAlertOrigin	`.1.5.3`	Identifier of the component that detected the threat (integer***)
infectedEmailAttachmentAlertSocket	`.1.5.4`	IP address of the source of the email message (string)
infectedEmailAttachmentAlertMailFrom	`.1.5.5`	Sender of the email message (string)
infectedEmailAttachmentAlertRcptTo	`.1.5.6`	Recipients of the email message (string)
infectedEmailAttachmentAlertMessageId	`.1.5.7`	Value of `Message-ID` header of the email message (string)
infectedEmailAttachmentAlertAction	`.1.5.8`	Action that was applied to the whole email message or infected attachment (integer: 1—repack; 2—reject; 3—discard; 4—cure; 5—move to quarantine; 6—delete)
infectedEmailAttachmentAlertDivert	`.1.5.9`	Direction of the email message (integer: 1—incoming; 2—outgoing)
infectedEmailAttachmentAlertSrcIp	`.1.5.10`	IP address of connection source (string)
infectedEmailAttachmentAlertSrcPort	`.1.5.11`	Port of connection source (integer)

Parameter name	OID of the parameter	Type and description of the parameter
infectedEmailAttachmentAlertDstIp	`.1.5.12`	IP address of connection destination point (string)
infectedEmailAttachmentAlertDstPort	`.1.5.13`	Port of connection destination point (integer)
infectedEmailAttachmentAlertSniHost	`.1.5.14`	SNI of connection destination point (for SSL connections) (string)
infectedEmailAttachmentAlertProtocol	`.1.5.15`	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
infectedEmailAttachmentAlertExePath	`.1.5.16`	Executable path of the program that establish the connection (string)
infectedEmailAttachmentAlertUserName	`.1.5.17`	Name of the user with whose privileges is executing the program that establish the connection (string)
categoryUrlAlert	`.1.6`	Notification about blocking a URL belonging to the unwanted category
categoryUrlAlertUrl	`.1.6.1`	The URL that was blocked (string)
categoryUrlAlertCategory	`.1.6.2`	Web resource category to which the URL belongs (integer**)
categoryUrlAlertOrigin	`.1.6.3`	Identifier of the component that detected the threat (integer***)
categoryUrlAlertSrcIp	`.1.6.4`	IP address of connection source (string)
categoryUrlAlertSrcPort	`.1.6.5`	Port of connection source (integer)
categoryUrlAlertDstIp	`.1.6.6`	IP address of connection destination point (string)
categoryUrlAlertDstPort	`.1.6.7`	Port of connection destination point (integer)
categoryUrlAlertSniHost	`.1.6.8`	SNI of connection destination point (for SSL connections)

Parameter name	OID of the parameter	Type and description of the parameter
		(string)
categoryUrlAlertExePath	`.1.6.9`	Executable path of the program that establish the connection (string)
categoryUrlAlertUserName	`.1.6.10`	Name of the user with whose privileges is executing the program that establish the connection (string)
categoryUrlEmailAttachmentAlert	`.1.7`	Notification about detecting an unwanted URL in the email message
categoryUrlEmailAttachmentAlertType	`.1.7.1`	Web resource category to which the URL belongs (integer**)
categoryUrlEmailAttachmentAlertOrigin	`.1.7.2`	Identifier of the component that detected the threat (integer***)
categoryUrlEmailAttachmentAlertSocket	`.1.7.3`	IP address of the source of the email message (string)
categoryUrlEmailAttachmentAlertMailFrom	`.1.7.4`	Sender of the email message (string)
categoryUrlEmailAttachmentAlertRcptTo	`.1.7.5`	Recipients of the email message (string)
categoryUrlEmailAttachmentAlertMessageId	`.1.7.6`	Value of `Message-ID` header of the email message (string)
categoryUrlEmailAttachmentAlertAction	`.1.7.7`	Action that was applied to the whole email message or an attachment (integer: 1—repack; 2—reject; 3—discard; 4—cure; 5—move to quarantine; 6—delete)
categoryUrlEmailAttachmentAlertDivert	`.1.7.8`	Direction of the email message (integer: 1—incoming; 2—outgoing)
categoryUrlEmailAttachmentAlertSrcIp	`.1.7.9`	IP address of connection source (string)
categoryUrlEmailAttachmentAlertSrcPort	`.1.7.10`	Port of connection source (integer)

Parameter name	OID of the parameter	Type and description of the parameter
categoryUrlEmailAttachmentAlertDstIp	`.1.7.11`	IP address of connection destination point (string)
categoryUrlEmailAttachmentAlertDstPort	`.1.7.12`	Port of connection destination point (integer)
categoryUrlEmailAttachmentAlertSniHost	`.1.7.13`	SNI of connection destination point (for SSL connections) (string)
categoryUrlEmailAttachmentAlertProtocol	`.1.7.14`	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
categoryUrlEmailAttachmentAlertExePath	`.1.7.15`	Executable path of the program that establish the connection (string)
categoryUrlEmailAttachmentAlertUserName	`.1.7.16`	Name of the user with whose privileges is executing the program that establish the connection (string)
spamEmailAlert	`.1.8`	Notification about recognizing an email message as spam
spamEmailAlertOrigin	`.1.8.1`	Identifier of the component that detected the threat (integer***)
spamEmailAlertSocket	`.1.8.2`	IP address of the source of the email message (string)
spamEmailAlertMailFrom	`.1.8.3`	Sender of the email message (string)
spamEmailAlertRcptTo	`.1.8.4`	Recipients of the email message (string)
spamEmailAlertMessageId	`.1.8.5`	Value of `Message-ID` header of the email message (string)
spamEmailAlertAction	`.1.8.6`	Action that was applied to the whole email message or an attachment (integer: 1—repack; 2—reject; 3—discard; 4—cure; 5—move to quarantine; 6—delete)
spamEmailAlertDivert	`.1.8.7`	Direction of the email message (integer: 1—incoming; 2—

Parameter name	OID of the parameter	Type and description of the parameter
		outgoing)
spamEmailAlertSrcIp	`.1.8.8`	IP address of connection source (string)
spamEmailAlertSrcPort	`.1.8.9`	Port of connection source (integer)
spamEmailAlertDstIp	`.1.8.10`	IP address of connection destination point (string)
spamEmailAlertDstPort	`.1.8.11`	Port of connection destination point (integer)
spamEmailAlertSniHost	`.1.8.12`	SNI of connection destination point (for SSL connections) (string)
spamEmailAlertProtocol	`.1.8.13`	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
spamEmailAlertExePath	`.1.8.14`	Executable path of the program that establish the connection (string)
spamEmailAlertUserName	`.1.8.15`	Name of the user with whose privileges is executing the program that establish the connection (string)
blockedConnectionAlert	`.1.9`	Notification about blocking a network connection
blockedConnectionAlertOrigin	`.1.9.1`	Identifier of the component that detected the threat (integer***)
blockedConnectionAlertDivert	`.1.9.2`	Direction of the connection (integer: 1—incoming; 2—outgoing)
blockedConnectionAlertSrcIp	`.1.9.3`	IP address of connection source (string)
blockedConnectionAlertSrcPort	`.1.9.4`	Port of connection source (integer)
blockedConnectionAlertDstIp	`.1.9.5`	IP address of connection destination point (string)

Parameter name	OID of the parameter	Type and description of the parameter
blockedConnectionAlertDstPort	`.1.9.6`	Port of connection destination point (integer)
blockedConnectionAlertSniHost	`.1.9.7`	SNI of connection destination point (for SSL connections) (string)
blockedConnectionAlertProtocol	`.1.9.8`	Protocol type (integer: 1—SMTP; 2—POP3; 3—IMAP; 4—HTTP)
blockedConnectionAlertExePath	`.1.9.9`	Executable path of the program that establish the connection (string)
blockedConnectionAlertUserName	`.1.9.10`	Name of the user with whose privileges is executing the program that establish the connection (string)
stat	**Statistics on the operation of the software product**	
threatCounters	`.2.1`	Counters of detected threats
knownVirus	`.2.1.1`	Number of detected known viruses (counter; integer)
suspicious	`.2.1.2`	Number of detected suspicious objects (counter; integer)
adware	`.2.1.3`	Number of detected adware (counter; integer)
dialers	`.2.1.4`	Number of detected dialers (counter; integer)
joke	`.2.1.5`	Number of detected joke programs (counter; integer)
riskware	`.2.1.6`	Number of detected riskware (counter; integer)
hacktool	`.2.1.7`	Number of detected hacktools (counter; integer)
scanErrors	`.2.2`	Counters of the errors that occurred while files were scanned

Parameter name	OID of the parameter	Type and description of the parameter
sePathNotAbsolute	`.2.2.1`	Number of occurrences of the “Path is not absolute” error (counter, integer)
seFileNotFound	`.2.2.2`	Number of occurrences of the “File not found” error (counter, integer)
seFileNotRegular	`.2.2.3`	Number of occurrences of the “File is not a regular file” error (counter, integer)
seFileNotBlockDevice	`.2.2.4`	Number of occurrences of the “File is not a block device” error (counter, integer)
seNameTooLong	`.2.2.5`	Number of occurrences of the “Path or file name is too long” error (counter, integer)
seNoAccess	`.2.2.6`	Number of occurrences of the “Permission denied” error (counter, integer)
seReadError	`.2.2.7`	Number of occurrences of the “Read error” (counter, integer)
seWriteError	`.2.2.8`	Number of occurrences of the “Write error” (counter, integer)
seFileTooLarge	`.2.2.9`	Number of occurrences of the “File size too big” error (counter, integer)
seFileBusy	`.2.2.10`	Number of occurrences of the “File is busy” error (counter, integer)
seUnpackingError	`.2.2.20`	Number of occurrences of the “Unpacking error” (counter, integer)
sePasswordProtecetd	`.2.2.21`	Number of occurrences of the “Password protected” error (counter, integer)
seArchCrcError	`.2.2.22`	Number of occurrences of the “Archive Cyclic Redundancy

Parameter name	OID of the parameter	Type and description of the parameter
		Check error” (counter, integer)
seArchInvalidHeader	`.2.2.23`	Number of occurrences of the “Invalid archive header” error (counter, integer)
seArchNoMemory	`.2.2.24`	Number of occurrences of the “Not enough memory to process the archive” error (counter, integer)
seArchIncomplete	`.2.2.25`	Number of occurrences of the “Incomplete archive” error (counter, integer)
seCanNotBeCured	`.2.2.26`	Number of occurrences of the “Object cannot be cured” error (counter, integer)
sePackerLevelLimit	`.2.2.30`	Number of occurrences of the error that states that the maximum nesting level of packed objects was exceeded (counter, integer)
seArchiveLevelLimit	`.2.2.31`	Number of occurrences of the error that states that the maximum nesting level of archives was exceeded (counter, integer)
seMailLevelLimit	`.2.2.32`	Number of occurrences of the error that states that the maximum nesting level of email files was exceeded (counter, integer)
seContainerLevelLimit	`.2.2.33`	Number of occurrences of the error that states that the maximum nesting level of container files was exceeded (counter, integer)
seCompressionLimit	`.2.2.34`	Number of occurrences of the “Exceeded the maximum compression ratio” error (counter, integer)

Parameter name	OID of the parameter	Type and description of the parameter
seReportSizeLimit	`.2.2.35`	Number of occurrences of the “Exceeded the maximum size of the scanning results report” error (counter, integer)
seScanTimeout	`.2.2.40`	Number of occurrences of the “Scan timeout expired” error (counter, integer)
seEngineCrash	`.2.2.41`	Number of occurrences of the “Scanning Engine crash was detected” error (counter, integer)
seEngineHangup	`.2.2.42`	Number of occurrences of the “Scanning Engine stopped responding” error (counter, integer)
seEngineError	`.2.2.44`	Number of occurrences of the “Internal error of the Scanning Engine” (counter, integer)
seNoLicense	`.2.2.45`	Number of occurrences of the “No valid license found” error (counter, integer)
seNonSupportedDisk	`.2.2.50`	Number of Occurrences of the “Unsupported disk” error (counter, integer)
seUnexpectedError	`.2.2.100`	Number of occurrences of the “Unexpected error” (counter, integer)
scanLoadAverage	`.2.3`	Metrics of the file scanning load
filesScannedTable	`.2.3.1`	Average numbers of files scanned at the request of other components
filesScannedEntry	`.2.3.1.1`	Component of the product (entire table row, record)
filesScannedIndex	`.2.3.1.1.1`	Index of the component (identifier, integer***)
filesScannedOrigin	`.2.3.1.1.2`	Name of the component

Parameter name	OID of the parameter	Type and description of the parameter
filesScanned1min	`.2.3.1.1.3`	The average (averaged over one minute) number of files checked per second (string)
filesScanned5min	`.2.3.1.1.4`	The average (averaged over 5 minutes) number of files checked per second (string)
filesScanned15min	`.2.3.1.1.5`	The average (averaged over 15 minutes) number of files checked per second (string)
bytesScannedTable	`.2.3.2`	Average speed (in bytes per second) of scanning performed at the request of other components
bytesScannedEntry	`.2.3.2.1`	Component of the product (entire table row, record)
bytesScannedIndex	`.2.3.2.1.1`	Index of the component (identifier, integer***)
bytesScannedOrigin	`.2.3.2.1.2`	Name of the component
bytesScanned1min	`.2.3.2.1.3`	The average (averaged over one minute) number of bytes scanned per second (string)
bytesScanned5min	`.2.3.2.1.4`	The average (averaged over 5 minutes) number of bytes scanned per second (string)
bytesScanned15min	`.2.3.2.1.5`	The average (averaged over 15 minutes) number of bytes scanned per second (string)
cacheHitFilesTable	`.2.3.3`	Average numbers of scanning reports retrieved from the cache at the request of the components
cacheHitFilesEntry	`.2.3.3.1`	Component of the product (entire table row, record)
cacheHitFilesIndex	`.2.3.3.1.1`	Index of the component (identifier, integer***)
cacheHitFilesOrigin	`.2.3.3.1.2`	Name of the component

Parameter name	OID of the parameter	Type and description of the parameter
cacheHitFiles1min	`.2.3.3.1.3`	The average (averaged over one minute) number of reports retrieved from the cache per second (string)
cacheHitFiles5min	`.2.3.3.1.4`	The average (averaged over 5 minutes) number of reports retrieved from the cache per second (string)
cacheHitFiles15min	`.2.3.3.1.5`	The average (averaged over 15 minutes) number of reports retrieved from the cache per second (string)
errorsTable	`.2.3.4`	Average numbers of errors during the scanning that was performed at the request of the components
errorsEntry	`.2.3.4.1`	Component of the product (entire table row, record)
errorsIndex	`.2.3.4.1.1`	Index of the component (identifier, integer***)
errorsOrigin	`.2.3.4.1.2`	Name of the component
errors1min	`.2.3.4.1.3`	The average (averaged over one minute) number of scanning errors per second (string)
errors5min	`.2.3.4.1.4`	The average (averaged over 5 minutes) number of scanning errors per second (string)
errors15min	`.2.3.4.1.5`	The average (averaged over 15 minutes) number of scanning errors per second (string)
net	`.2.4`	Statistics on network activity
markedAsSpam	`.2.4.1`	Number of email messages marked as spam (counter, integer)
blockedInfectionSource	`.2.4.101`	Number of blocked URLs belonging to the “Infection

Parameter name	OID of the parameter	Type and description of the parameter
		Source" category (counter, integer)
blockedNotRecommended	`.2.4.102`	Number of blocked URLs belonging to the "Not Recommended" category (counter, integer)
blockedAdultContent	`.2.4.103`	Number of blocked URLs belonging to the "Adult Content" category (counter, integer)
blockedViolence	`.2.4.104`	Number of blocked URLs belonging to the "Violence" category (counter, integer)
blockedWeapons	`.2.4.105`	Number of blocked URLs belonging to the "Weapons" category (counter, integer)
blockedGambling	`.2.4.106`	Number of blocked URLs belonging to the "Gambling" category (counter, integer)
blockedDrugs	`.2.4.107`	Number of blocked URLs belonging to the "Drugs" category (counter, integer)
blockedObsceneLanguage	`.2.4.108`	Number of blocked URLs belonging to the "Obscene Language" category (counter, integer)
blockedChats	`.2.4.109`	Number of blocked URLs belonging to the "Chats" category (counter, integer)
blockedTerrorism	`.2.4.110`	Number of blocked URLs belonging to the "Terrorism" category (counter, integer)
blockedFreeEmail	`.2.4.111`	Number of blocked URLs belonging to the "Free Email Services" category (counter, integer)
blockedSocialNetworks	`.2.4.112`	Number of blocked URLs belonging to the "Social

Parameter name	OID of the parameter	Type and description of the parameter
		Networks” category (counter, integer)
blockedOwnersNotice	`.2.4.113`	Number of blocked URLs belonging to the “Copyright Owner`s Notice” category (counter, integer)
blockedBlackList	`.2.4.120`	Number of blocked URLs from the user’s black list (counter, integer)
info	**Information about the current state of the program**	
components	`.3.1`	Current state of the program’s components
configd	`.3.1.1`	drweb-configd component data
configdState	`.3.1.1.1`	Current state of the component (integer****)
configdExitCode	`.3.1.1.2`	Last exit code (integer corresponding to code from error catalogue)
configdExitTime	`.3.1.1.3`	Time of the last termination (*UNIX time*)
configdInstalledApps	`.3.1.1.101`	List of installed components
configdAppEntry	`.3.1.1.101.1`	Information about the installed component (entire table row, record)
configdAppIndex	`.3.1.1.101.1.1`	Index (ordinal number) of the installed component (integer)
configdAppName	`.3.1.1.101.1.2`	Name of the installed component (string)
configdAppExePath	`.3.1.1.101.1.3`	Path to the executable file of the component (string)
configdAppInstallTime	`.3.1.1.101.1.4`	Time when the component was installed (*UNIX time*)

Parameter name	OID of the parameter	Type and description of the parameter
configdAppIniSection	`.3.1.1.101.1.5`	Name of the section with the component’s parameters in the configuration file
scanEngine	`.3.1.2`	drweb-se component data
scanEngineState	`.3.1.2.1`	Current state of the component (integer****)
scanEngineExitCode	`.3.1.2.2`	Last exit code (integer corresponding to code from error catalogue)
scanEngineExitTime	`.3.1.2.3`	Time of the last termination (*UNIX time*)
scanEngineStatus	`.3.1.2.101`	Current state of the Dr.Web Virus-Finding Engine (integer)
scanEngineVersion	`.3.1.2.102`	Version of the Dr.Web Virus-Finding Engine (string)
scanEngineVirusRecords	`.3.1.2.103`	Number of virus records (integer)
scanEngineMaxForks	`.3.1.2.104`	Maximum number of child processes for scanning (integer)
scanEngineQueues	`.3.1.2.105`	Scan task queues
scanEngineQueuesLow	`.3.1.2.105.1`	The queue of low-priority tasks
scanEngineQueueLowOut	`.3.1.2.105.1.1`	Number of low-priority tasks popped from the queue and transferred to processing (counter, integer)
scanEngineQueueLowSize	`.3.1.2.105.1.2`	Number of low-priority tasks in the queue waiting to be processed (counter, integer)
scanEngineQueuesMedium	`.3.1.2.105.2`	The queue of normal-priority tasks
scanEngineQueueMediumOut	`.3.1.2.105.2.1`	The number of normal-priority tasks popped from the queue and transferred to processing (counter, integer)

Parameter name	OID of the parameter	Type and description of the parameter
scanEngineQueueMediumSize	`.3.1.2.105.2.2`	Number of normal-priority tasks in the queue waiting to be processed (counter, integer)
scanEngineQueuesHigh	`.3.1.2.105.3`	The queue of high-priority tasks
scanEngineQueueHighOut	`.3.1.2.105.3.1`	The number of high-priority tasks popped from the queue and transferred to processing (counter, integer)
scanEngineQueueHighSize	`.3.1.2.105.3.2`	Number of high-priority tasks in the queue waiting to be processed (counter, integer)
scanEngineVirusBasesTable	`.3.1.2.106`	The list of virus databases.
scanEngineVirusBasesEntry	`.3.1.2.106.1`	Information about the virus database (entire table row; record)
scanEngineVirusBaseIndex	`.3.1.2.106.1.1`	Index of the virus database (integer)
scanEngineVirusBasePath	`.3.1.2.106.1.2`	Path to the virus database file (string)
scanEngineVirusBaseRecords	`.3.1.2.106.1.3`	Number of records in the virus database (integer)
scanEngineVirusBaseVersion	`.3.1.2.106.1.4`	Version of the virus database (integer)
scanEngineVirusBaseTimestamp	`.3.1.2.106.1.5`	Timestamp of the virus database (*UNIX time*)
scanEngineVirusBaseMD5	`.3.1.2.106.1.6`	MD5 checksum (string)
scanEngineVirusBaseLoadResult	`.3.1.2.106.1.7`	Result of the downloading of this virus database (string)
scanEngineQueuesTab	`.3.1.2.107`	The list of scan task queues
scanEngineQueueEntry	`.3.1.2.107.1`	Information about the queue (entire table row, record)
scanEngineQueueIndex	`.3.1.2.107.1.1`	Index (ordinal number) of the queue (integer)

Parameter name	OID of the parameter	Type and description of the parameter
scanEngineQueueName	.3.1.2.107.1.2	Name of the queue (string)
scanEngineQueueOut	.3.1.2.107.1.3	The number of tasks popped from the queue and transferred to processing (counter, integer)
scanEngineQueueSize	.3.1.2.107.1.4	Number of tasks in the queue waiting to be processed (counter, integer)
fileCheck	.3.1.3	**drweb-filecheck** component data
fileCheckState	.3.1.3.1	Current state of the component (integer****)
fileCheckExitCode	.3.1.3.2	Last exit code (integer corresponding to code from error catalogue)
fileCheckExitTime	.3.1.3.3	Time of the last termination (*UNIX time*)
fileCheckScannedFiles	.3.1.3.101	Number of scanned files (counter, integer)
fileCheckScannedBytes	.3.1.3.102	Number of scanned bytes (counter, integer)
fileCheckCacheHitFiles	.3.1.3.103	Number of scan reports retrieved from the cache (counter, integer)
fileCheckScanErrors	.3.1.3.104	Number of error occurrences in the Scanning Engine (counter, integer)
fileCheckScanStat	.3.1.3.105	List of clients
fileCheckClientEntry	.3.1.3.105.1	Information about the client (entire table row; record)
fileCheckClientIndex	.3.1.3.105.1.1	Index (ordinal number) of the client (integer)
fileCheckClientName	.3.1.3.105.1.2	Name of the client component (string)

Parameter name	OID of the parameter	Type and description of the parameter
fileCheckClientScannedFiles	`.3.1.3.105.1.3`	The number of files scanned for this client (counter, integer)
fileCheckClientScannedBytes	`.3.1.3.105.1.4`	The number of bytes scanned for this client (counter, integer)
fileCheckClientCacheHitFiles	`.3.1.3.105.1.5`	The number of scan reports retrieved from the cache for this client (counter, integer)
fileCheckClientScanErrors	`.3.1.3.105.1.6`	Number of error occurrences in the Scanning Engine when working for this client (counter, integer)
update	`.3.1.4`	drweb-update component data
updateState	`.3.1.4.1`	Current state of the component (integer****)
updateExitCode	`.3.1.4.2`	Last exit code (integer corresponding to code from error catalogue)
updateExitTime	`.3.1.4.3`	Time of the last termination (*UNIX time*)
updateBytesSent	`.3.1.4.101`	Number of bytes sent (counter, integer)
updateBytesReceived	`.3.1.4.102`	Number of bytes received (counter, integer)
esagent	`.3.1.5`	drweb-esagent component data
esagentState	`.3.1.5.1`	Current state of the component (integer****)
esagentExitCode	`.3.1.5.2`	Last exit code (integer corresponding to code from error catalogue)
esagentExitTime	`.3.1.5.3`	Time of the last termination (*UNIX time*)
esagentWorkStatus	`.3.1.5.101`	Component’s current mode of operation (integer: 1—standalone mode, 2—is connecting, 3—is

Parameter name	OID of the parameter	Type and description of the parameter
		awaiting connection, 4—connection has been approved)
esagentIsConnected	`.3.1.5.102`	Is connected to the server (integer, 0—no, 1—yes)
esagentServer	`.3.1.5.103`	Address of the central protection server that is used (string)
netcheck	`.3.1.6`	drweb-netcheck component data
netcheckState	`.3.1.6.1`	Current state of the component (integer****)
netcheckExitCode	`.3.1.6.2`	Last exit code (integer corresponding to code from error catalogue)
netcheckExitTime	`.3.1.6.3`	Time of the last termination (*UNIX time*)
netcheckLocalSeForks	`.3.1.6.101`	The number of Scanning Engine processes available locally (integer)
netcheckRemoteSeForks	`.3.1.6.102`	Number of Scanning Engine processes available remotely (integer)
netcheckLocalFilesScanned	`.3.1.6.103`	The number of files that have been scanned locally (counter, integer)
netcheckNetworkFilesScanned	`.3.1.6.104`	The number of files that have been scanned via remote scanning (counter, integer)
netcheckLocalBytesScanned	`.3.1.6.105`	The number of bytes that have been scanned locally (counter, integer)
netcheckNetworkBytesScanned	`.3.1.6.106`	The number of bytes that have been scanned via remote scanning (counter, integer)
netcheckLocalBytesIn	`.3.1.6.107`	The number of bytes received from local clients (counter, integer)

Parameter name	OID of the parameter	Type and description of the parameter
netcheckLocalBytesOut	`.3.1.6.108`	The number of bytes sent back to local clients (counter, integer)
netcheckNetworkBytesIn	`.3.1.6.109`	The number of bytes received from remote hosts (counter, integer)
netcheckNetworkBytesOut	`.3.1.6.110`	The number of bytes sent to remote hosts (counter, integer)
netcheckLocalScanErrors	`.3.1.6.111`	Number of error occurrences in local Scanning Engine processes (counter, integer)
netcheckNetworkScanErrors	`.3.1.6.112`	Number of error occurrences in remote Scanning Engine processes (counter, integer)
httpd	`.3.1.7`	drweb-httpd component data
httpdState	`.3.1.7.1`	Current state of the component (integer****)
httpdExitCode	`.3.1.7.2`	Last exit code (integer corresponding to code from error catalogue)
httpdExitTime	`.3.1.7.3`	Time of the last termination (*UNIX time*)
snmpd	`.3.1.8`	drweb-snmpd component data
snmpdState	`.3.1.8.1`	Current state of the component (integer****)
snmpdExitCode	`.3.1.8.2`	Last exit code (integer corresponding to code from error catalogue)
snmpdExitTime	`.3.1.8.3`	Time of the last termination (*UNIX time*)
clamd	`.3.1.20`	drweb-clamd component data
clamdState	`.3.1.20.1`	Current state of the component (integer****)

Parameter name	OID of the parameter	Type and description of the parameter
clamdExitCode	`.3.1.20.2`	Last exit code (integer corresponding to code from error catalogue)
clamdExitTime	`.3.1.20.3`	Time of the last termination (*UNIX time*)
icapd	`.3.1.21`	drweb-icapd component data
icapdState	`.3.1.21.1`	Current state of the component (integer****)
icapdExitCode	`.3.1.21.2`	Last exit code (integer corresponding to code from error catalogue)
icapdExitTime	`.3.1.21.3`	Time of the last termination (*UNIX time*)
icapdConnectionsIn	`.3.1.21.101`	Number of accepted incoming connections (counter, integer)
icapdConnectionsCount	`.3.1.21.102`	Number of currently opened connections (counter, integer)
icapdOptions	`.3.1.21.103`	Number of *OPTIONS* requests (counter, integer)
icapdReqmod	`.3.1.21.104`	Number of *REQMOD* requests (counter, integer)
icapdRespmod	`.3.1.21.105`	Number of *RESPMOD* requests (counter, integer)
icapdBad	`.3.1.21.106`	Number of invalid requests (counter, integer)
smbspider	`.3.1.40`	**drweb-smbspider-daemon** component data
smbspiderState	`.3.1.40.1`	Current state of the component (integer****)
smbspiderExitCode	`.3.1.40.2`	Last exit code (integer corresponding to code from error catalogue)

Parameter name	OID of the parameter	Type and description of the parameter
smbspiderExitTime	`.3.1.40.3`	Time of the last termination (*UNIX time*)
smbspiderConnectionsIn	`.3.1.40.101`	Total number of opened connections (counter, integer)
smbspiderConnectionsCount	`.3.1.40.102`	Number of currently opened connections (counter, integer)
smbspiderShareTable	`.3.1.40.103`	Statistics on the protected **Samba** shared resources
smbspiderShareEntry	`.3.1.40.103.1`	Information about the protected **Samba** shared resource (entire table row; record)
smbspiderShareIndex	`.3.1.40.103.1.1`	Index (ordinal number) of the protected **Samba** shared resource (integer)
smbspiderSharePath	`.3.1.40.103.1.2`	Path to the protected **Samba** shared resource (string)
smbspiderShareConnectionsIn	`.3.1.40.103.1.3`	Total number of opened connections (counter, integer)
smbspiderShareConnectionsCount	`.3.1.40.103.1.4`	Number of currently opened connections (counter, integer)
gated	`.3.1.41`	drweb-gated component data
gatedState	`.3.1.41.1`	Current state of the component (integer****)
gatedExitCode	`.3.1.41.2`	Last exit code (integer corresponding to code from error catalogue)
gatedExitTime	`.3.1.41.3`	Time of the last termination (*UNIX time*)
gatedInterceptedIn	`.3.1.41.101`	Number of intercepted connections (counter, integer)
gatedInterceptedCount	`.3.1.41.102`	Number of currently monitored connections (counter, integer)
maild	`.3.1.42`	drweb-maild component data

Parameter name	OID of the parameter	Type and description of the parameter
maildState	`.3.1.42.1`	Current state of the component (integer****)
maildExitCode	`.3.1.42.2`	Last exit code (integer corresponding to code from error catalogue)
maildExitTime	`.3.1.42.3`	Time of the last termination (*UNIX time*)
maildStat	`.3.1.42.4`	Statistics of the **drweb-maild** component operation
maildStatNative	`.3.1.42.4.1`	Email scanning statistics via the component’s internal interface **drweb-maild** (messages received by SpIDer Gate during the scan of intersepted SMTP, POP3, IMAP connections)
maildStatNativePassed	`.3.1.42.4.1.1`	Number of missed messages (counter, integer)
maildStatNativeRepacked	`.3.1.42.4.1.2`	Number of repackaged messages (counter, integer)
maildStatNativeRejected	`.3.1.42.4.1.3`	Number of rejected messages (counter, integer)
maildStatNativeFailed	`.3.1.42.4.1.4`	Number of message scanning errors (counter, integer)
maildStatNativeQueueSize	`.3.1.42.4.1.5`	The queue line, that is the number of files waiting to be scanned via the interface (integer)
maildStatMilter	`.3.1.42.4.2`	Email scanning statistics via the component’s interface *Milter* of the **drweb-maild** component
maildStatMilterPassed	`.3.1.42.4.2.1`	Number of missed messages (counter, integer)
maildStatMilterRepacked	`.3.1.42.4.2.2`	Number of repackaged messages (counter, integer)
maildStatMilterRejected	`.3.1.42.4.2.3`	Number of rejected messages (counter, integer)

Parameter name	OID of the parameter	Type and description of the parameter
maildStatMilterFailed	`.3.1.42.4.2.4`	Number of message scanning errors (counter, integer)
maildStatMilterQueueSize	`.3.1.42.4.2.5`	The queue line, that is the number of files waiting to be scanned via the interface (integer)
maildStatSpamc	`.3.1.42.4.3`	Email scanning statistics via the component's interface *Spamd* of the **drweb-maild** component
maildStatSpamcPassed	`.3.1.42.4.3.1`	Number of missed messages (counter, integer)
maildStatSpamcRepacked	`.3.1.42.4.3.2`	Number of repackaged messages (counter, integer)
maildStatSpamcRejected	`.3.1.42.4.3.3`	Number of rejected messages (counter, integer)
maildStatSpamcFailed	`.3.1.42.4.3.4`	Number of message scanning errors (counter, integer)
maildStatSpamcQueueSize	`.3.1.42.4.3.5`	The queue line, that is the number of files waiting to be scanned via the interface (integer)
maildStatRspamc	`.3.1.42.4.4`	Email scanning statistics via the component's interface *Rspamd* of the **drweb-maild** component
maildStatRspamcPassed	`.3.1.42.4.4.1`	Number of missed messages (counter, integer)
maildStatRspamcRepacked	`.3.1.42.4.4.2`	Number of repackaged messages (counter, integer)
maildStatRspamcRejected	`.3.1.42.4.4.3`	Number of rejected messages (counter, integer)
maildStatRspamcFailed	`.3.1.42.4.4.4`	Number of message scanning errors (counter, integer)
maildStatRspamcQueueSize	`.3.1.42.4.4.5`	The queue line, that is the number of files waiting to be scanned via the interface (integer)
lookupd	`.3.1.43`	drweb-lookupd component data

Parameter name	OID of the parameter	Type and description of the parameter
lookupdState	`.3.1.43.1`	Current state of the component (integer****)
lookupdExitCode	`.3.1.43.2`	Last exit code (integer corresponding to code from error catalogue)
lookupdExitTime	`.3.1.43.3`	Time of the last termination (*UNIX time*)
cloudd	`.3.1.50`	drweb-cloudd component data
clouddState	`.3.1.50.1`	Current state of the component (integer****)
clouddExitCode	`.3.1.50.2`	Last exit code (integer corresponding to code from error catalogue)
clouddExitTime	`.3.1.50.3`	Time of the last termination (*UNIX time*)
vpnd	`.3.1.51`	drweb-vpnd component data
vpndState	`.3.1.51.1`	Current state of the component (integer****)
vpndExitCode	`.3.1.51.2`	Last exit code (integer corresponding to code from error catalogue)
vpndExitTime	`.3.1.51.3`	Time of the last termination (*UNIX time*)
vpndWorkStatus	`.3.1.51.101`	Component’s current mode of operation (integer: 0—turned off, 1—server, 2—client)
vpndConnectionState	`.3.1.51.102`	Status of the established connection (integer: 0—status not set, 1—connecting, 2—connected, 3—error, 4—setting up NAT, 5—creating a protected tunnel)
vpndNetworkName	`.3.1.51.103`	Name of the created personal network (string)

Parameter name	OID of the parameter	Type and description of the parameter
meshd	`.3.1.52`	drweb-meshd component data
meshdState	`.3.1.52.1`	Current state of the component (integer****)
meshdExitCode	`.3.1.52.2`	Last exit code (integer corresponding to code from error catalogue)
meshdExitTime	`.3.1.52.3`	Time of the last termination (*UNIX time*)
lotus	`.3.1.60`	**drweb-lotus** component data
lotusState	`.3.1.60.1`	Current state of the component (integer****)
lotusExitCode	`.3.1.60.2`	Last exit code (integer corresponding to code from error catalogue)
lotusExitTime	`.3.1.60.3`	Time of the last termination (*UNIX time*)
macgui	`.3.1.100`	**drweb-gui** (for **macOS**) component data
macguiState	`.3.1.100.1`	Current state of the component (integer****)
macguiExitCode	`.3.1.100.2`	Last exit code (integer corresponding to code from error catalogue)
macguiExitTime	`.3.1.100.3`	Time of the last termination (*UNIX time*)
macspider	`.3.1.102`	**drweb-spider** (for **macOS**) component data
macspiderState	`.3.1.102.1`	Current state of the component (integer****)
macspiderExitCode	`.3.1.102.2`	Last exit code (integer corresponding to code from error catalogue)

Parameter name	OID of the parameter	Type and description of the parameter
macspiderExitTime	`.3.1.102.3`	Time of the last termination (*UNIX time*)
macspiderWorkStatus	`.3.1.102.101`	Component’s current mode of operation (integer: 0—not set, 1—loading, 2—is running)
macfirewall	`.3.1.103`	**drweb-firewall** (for **macOS**) component data
macfirewallState	`.3.1.103.1`	Current state of the component (integer****)
macfirewallExitCode	`.3.1.103.2`	Last exit code (integer corresponding to code from error catalogue)
macfirewallExitTime	`.3.1.103.3`	Time of the last termination (*UNIX time*)
linuxgui	`.3.1.200`	**drweb-gui** (for **Linux**) component data
linuxguiState	`.3.1.200.1`	Current state of the component (integer****)
linuxguiExitCode	`.3.1.200.2`	Last exit code (integer corresponding to code from error catalogue)
linuxguiExitTime	`.3.1.200.3`	Time of the last termination (*UNIX time*)
linuxspider	`.3.1.201`	**drweb-spider** (for **Linux**) component data
linuxspiderState	`.3.1.201.1`	Current state of the component (integer****)
linuxspiderExitCode	`.3.1.201.2`	Last exit code (integer corresponding to code from error catalogue)
linuxspiderExitTime	`.3.1.201.3`	Time of the last termination (*UNIX time*)
linuxspiderWorkStatus	`.3.1.201.101`	Component’s current mode of operation (integer: 0—not set, 1

Parameter name	OID of the parameter	Type and description of the parameter
		—loading, 2—running via **fanotify**, 3—running via LKM)
linuxnss	`.3.1.202`	**drweb-nss** (for **Linux**) component data
linuxnssState	`.3.1.202.1`	Current state of the component (integer****)
linuxnssExitCode	`.3.1.202.2`	Last exit code (integer corresponding to code from error catalogue)
linuxnssExitTime	`.3.1.202.3`	Time of the last termination (*UNIX time*)
linuxnssScannedFiles	`.3.1.202.101`	Number of scanned files (counter, integer)
linuxnssScannedBytes	`.3.1.202.102`	Number of scanned bytes (counter, integer)
linuxnssScanErrors	`.3.1.202.103`	Number of scanning error occurrences (counter, integer)
linuxfirewall	`.3.1.203`	**drweb-firewall** (for **Linux**) component data
linuxfirewallState	`.3.1.203.1`	Current state of the component (integer****)
linuxfirewallExitCode	`.3.1.203.2`	Last exit code (integer corresponding to code from error catalogue)
linuxfirewallExitTime	`.3.1.203.3`	Time of the last termination (*UNIX time*)
ctl	`.3.1.300`	drweb-ctl component data
ctlState	`.3.1.300.1`	Current state of the component (integer****)
ctlExitCode	`.3.1.300.2`	Last exit code (integer corresponding to code from error catalogue)

Parameter name	OID of the parameter	Type and description of the parameter
ctlExitTime	`.3.1.300.3`	Time of the last termination (*UNIX time*)
license	`.3.2`	License status
licenseEsMode	`.3.2.1`	The license has been granted by the central protection server (integer: 0—no, 1—yes)
licenseNumber	`.3.2.2`	License number (integer)
licenseOwner	`.3.2.3`	License owner (string)
licenseActivated	`.3.2.4`	License activation date (*UNIX time*)
licenseExpires	`.3.2.5`	License expiration date (*UNIX time*)

*) Threat types:

Code	Threat Type
1	Known virus
2	Suspicious object
3	Adware
4	Dialer
5	Joke program
6	Riskware
7	Hacktool

**) Categories of URL:

Code	Threat Type
1	Infection source
2	Not recommended
3	Adult content

Code	Threat Type
4	Violence
5	Weapons
6	Gambling
7	Drugs
8	Obscene language
9	Chats
10	Terrorism
11	Free email
12	Social networks
13	URL added due to a notice from copyright owner
14	Added to black list

***) Codes of Dr.Web components:

Code	Component
1	Dr.Web ConfigD (**drweb-configd**)
2	Dr.Web Scanning Engine (**drweb-se**)
3	Dr.Web File Checker (**drweb-filecheck**)
4	Dr.Web Updater (**drweb-update**)
5	Dr.Web ES Agent (**drweb-esagent**)
6	Dr.Web Network Checker (**drweb-netcheck**)
7	Dr.Web HTTPD (**drweb-httpd**)
8	Dr.Web SNMPD (**drweb-snmpd**)
20	Dr.Web ClamD (**drweb-clamd**)
21	Dr.Web ICAPD (**drweb-icapd**)
40	SpIDer Guard for SMB (**drweb-smbspider-daemon**)
41	SpIDer Gate (**drweb-gated**)

Code	Component
42	Dr.Web MailD (**drweb-maild**)
43	Dr.Web LookupD (**drweb-lookupd**)
50	Dr.Web CloudD (**drweb-cloudd**)
51	Dr.Web VPND (**drweb-vpnd**)
52	Dr.Web MeshD (**drweb-meshd**)
60	Dr.Web for Lotus
100	**drweb-gui** for **macOS**
102	SpIDer Guard for **macOC**
103	Dr.Web Firewall for Linux for **macOS**
200	**drweb-gui** for **Linux**
201	SpIDer Guard (**drweb-spider**)
202	SpIDer Guard for NSS (**drweb-nss**)
203	Dr.Web Firewall for Linux (**drweb-firewall**) for**Linux**
300	Dr.Web Ctl (**drweb-ctl**)
400	Enterprise scanner (this is not a real component of the product)

****) Possible states of the components:

Code	Status
0	Not installed
1	Installed but not started
2	Is starting
3	Is running
4	Is exiting

To get the values of the variables directly, you can use the **snmpwalk** utility:

```
$ snmpwalk -Os -c <community> -v <SNMP version> <host address> <OID>
```

For example, to get statistics about the threats detected on the local machine, use the following command (if the settings of Dr.Web SNMPD are set to their default values):

```
$ snmpwalk -Os -c public -v 2c 127.0.0.1 .1.3.6.1.4.1.29690.2.2.1
```

Dr.Web CloudD

The Dr.Web CloudD component refers to Dr.Web Cloud (a cloud service of Doctor Web). Dr.Web Cloud service collects up-to-date information from all Dr.Web anti-virus solutions about detected threats to prevent users from visiting unwanted websites and to protect operating systems from infected files containing brand-new threats that do not have any description in Dr.Web virus databases. Moreover, the use of Dr.Web Cloud service reduces the probability of false positives of the Dr.Web Scanning Engine scanning engine and of the components monitoring the access to the Internet.

Operating Principles

The component is designed to refer to the Dr.Web Cloud service to scan contents of the specified file for threats unknown to the local Dr.Web Scanning Engine, and to check whether the specified URL belongs to any of Doctor Web's predefined categories of web resources.

Dr.Web CloudD is automatically run by the configuration daemon. The component is run upon receiving a command from the user or one of the Dr.Web for UNIX Internet Gateways components. The operation scheme is shown in the figure below.

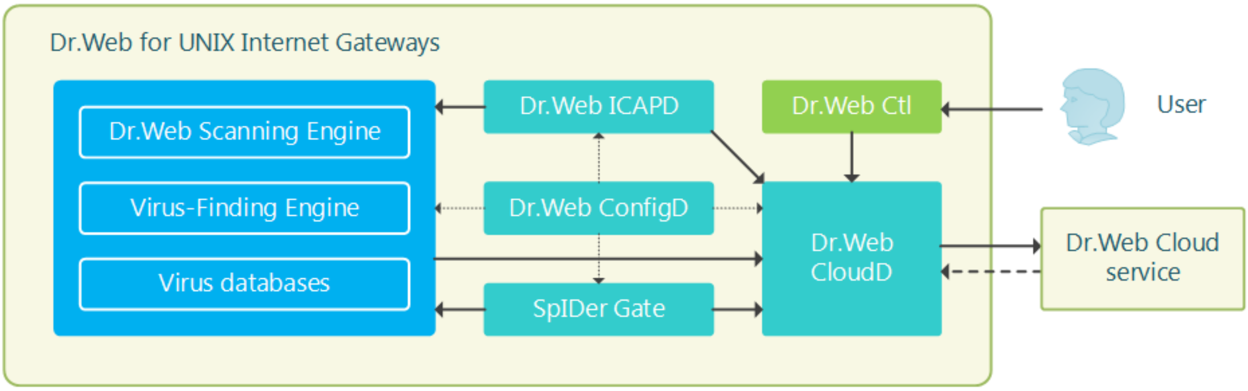


Figure 21. Diagram of the components' operation

This component is used for the requests to the Dr.Web Cloud service for scanning of the user requested URL by the scanning component for the network traffic and URL SpIDer Gate and the Dr.Web ICAPD.

Besides that, the component is used during the scanning of files on the command from the Dr.Web for UNIX Internet Gateways product management utility from the command line Dr.Web Ctl (it is started by the **drweb-ctl** command): upon detection of threats, the Dr.Web Scanning Engine scanning engine sends a report about the file to Dr.Web Cloud.

Command-Line Arguments

To run Dr.Web CloudD, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-cloudd [<parameters>]
```

Dr.Web CloudD can process the following options:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-clamd --help
```

This command outputs short help information on Dr.Web CloudD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the **drweb-ctl** command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-cloudd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[CloudD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel`	Logging level of the component.

{logging level}	If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/bin/drweb-cloudd` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-cloudd` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-cloudd`
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value:** `drweb`
`IdleTimeLimit` *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down. Minimum value—`10s`. **Default value:** `30s`
`PersistentCache` *{On | Off}*	Enable or disable saving of the cache of Dr.Web Cloud replies to the disk. **Default value:** `Off`
`DebugSdk` *{Boolean}*	Indicates whether detailed Dr.Web Cloud messages should be included into the log file on the debug level (**`LogLevel`** `= DEBUG`). **Default value:** `No`

Dr.Web LookupD

The Dr.Web LookupD component is designed to refer to external sources (directory services, such as **Active Directory**) to retrieve data by using the LDAP protocol (RFC 4511). The received data is used in rules according to which network connections are checked (for example, to check the user's authorization). This data is also used to block access to URLs if certain criteria are met.

In the component's settings, you can specify parameters for connection to several data sources. Dr.Web LookupD connects to the required data source only upon receiving a data request from any of the Dr.Web for UNIX Internet Gateways's components.

Sharing of data via the LDAP protocol can be performed either over an open channel or over a protected one, applying SSL/TLS. To use a secure connection, it is required to provide Dr.Web LookupD with an appropriate SSL certificate and key. If you need to generate SSL keys and certificates, you can use the **openssl** utility. An example of how to use the **openssl** utility to generate a certificate and a private key is given in the Appendix E. Generating SSL certificates section.

Operating Principles

The component is designed to request data from directory services (like **Active Directory**) and other data storages that support the LDAP protocol. Upon request, the received data (for example, users' identifiers and rights) is transferred to Dr.Web for UNIX Internet Gateways's components to be used in different rules for checks (for example, to allow a user to access a requested URL and etc.).

> This manual does not describe the operating principle of **LDAP** and **Active Directory**. If necessary, refer to the corresponding reference materials, for example, RFC 4511.

The Dr.Web LookupD component is launched automatically by the Dr.Web ConfigD configuration daemon when required (i.e. upon receiving a request for data). The diagram of the component's operation is shown in the figure below.

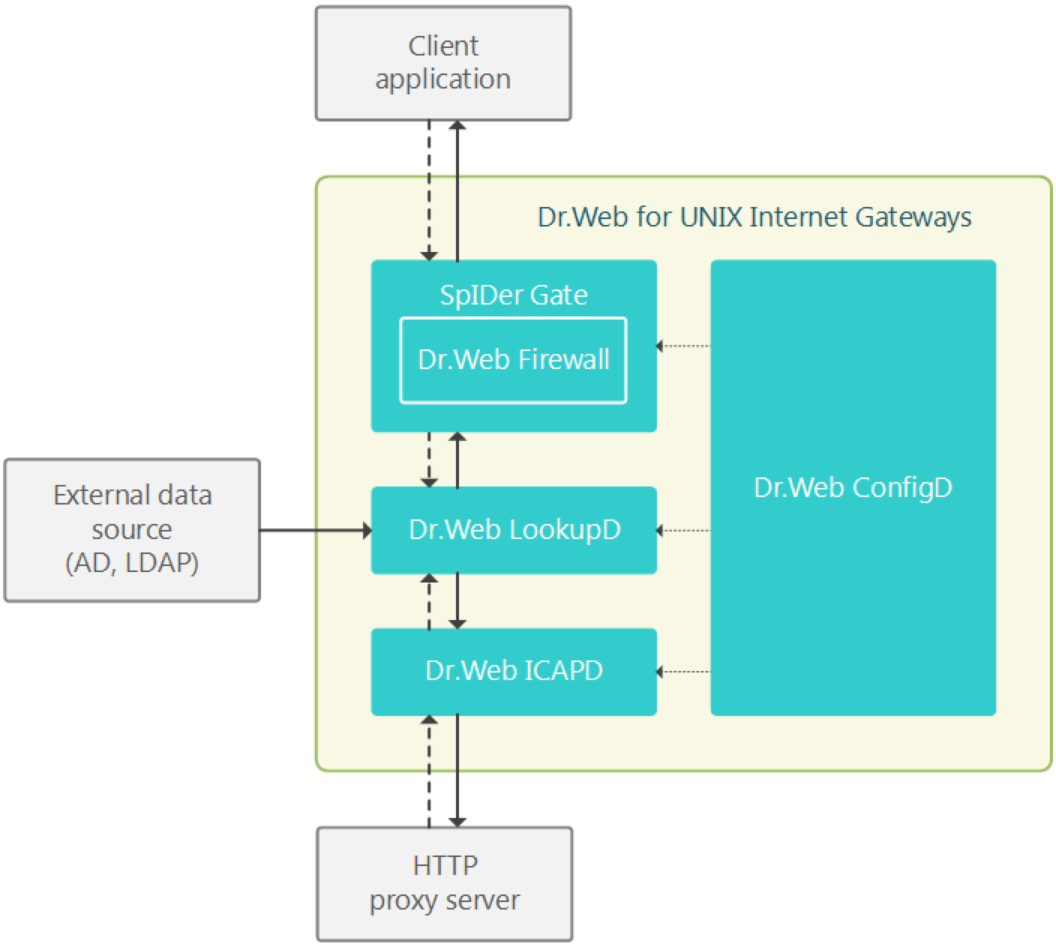


Figure 22. Diagram of the components' operation

Upon receiving a data request from any of the components (, for example, Dr.Web Firewall for Linux or Dr.Web ICAPD), the Dr.Web ConfigD configuration daemon starts Dr.Web LookupD (if it has not been started yet). Then the component makes a request to the required data source and returns a reply (usually a list of strings which meet the search criterion). In Dr.Web LookupD's settings you can specify an unlimited number of data sources. When forming a request for data retrieval, the client component must specify the source for data. Once Dr.Web LookupD is started, it will operate for some time waiting for new requests. If there are no more requests, after a waiting period Dr.Web LookupD shuts down automatically.

The basic way in which other components of the product use Dr.Web LookupD is for retrieving some data needed to check the validity of some conditions specified in the operation rules for these components. When checking the applicability of rules and the validity of conditions, data requests to Dr.Web LookupD are performed automatically.

Command-Line Arguments

To run Dr.Web LookupD, type the following command in the command line:

```
$ <opt_dir>/bin/drweb-lookupd [<parameters>]
```

Dr.Web LookupD can process the following parameters:

Parameter	Description
`--help`	**Function:** Instructs to output short help information about command-line parameters to the console or to the terminal emulator and to exit upon completion. **Short form:** `-h` **Arguments:** None.
`--version`	**Function:** Instructs to output information about the version of this component to the console or to the terminal emulator and to exit after completion. **Short form:** `-v` **Arguments:** None.

Example:

```
$ /opt/drweb.com/bin/drweb-lookupd --help
```

This command outputs short help information on Dr.Web LookupD.

Startup Notes

The component cannot be launched directly from the command line of the operating system in an autonomous mode (autonomously from other components). It is launched automatically by the Dr.Web ConfigD configuration daemon when needed. To manage the operation of the component, you can use the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (it is called by using the drweb-ctl command).

To request documentation about this component of the product from the command line, use the following command **man** `1 drweb-lookupd`

Configuration Parameters

The component uses configuration parameters which are specified in the `[LookupD]` section of the integrated configuration file of Dr.Web for UNIX Internet Gateways.

The section contains the following parameters:

`LogLevel`	Logging level of the component.

{logging level}	If the parameter value is not specified, the **`DefaultLogLevel`** parameter value from the [Root] section is used. **Default value:** `Notice`
`Log` *{log type}*	Logging method
`ExePath` *{path to file}*	Path to the executable file of the component. **Default value:** <*opt_dir*>`/bin/drweb-lookupd` • For **Linux**, **Solaris**: `/opt/drweb.com/bin/drweb-lookupd` • For **FreeBSD**: `/usr/local/libexec/drweb.com/bin/drweb-lookupd`
`RunAsUser` *{UID | user name}*	The parameter determines under which user name the component should be run. The user name can be specified either as the user's number UID or as the user's login. If the user name consists of numbers (i.e. similar to number UID), it is specified with the "`name:`" prefix, for example: **`RunAsUser`** `= name:123456`. *When a user name is not specified, the component operation terminates with an error after the startup.* **Default value :** `drweb`
`IdleTimeLimit` *{time interval}*	Maximum time that the component can remain idle. If the specified value is exceeded, the component shuts down. Minimum value—`10s`. **Default value:** `30s`
`DebugLibldap` *{Boolean}*	Indicates whether debug messages of the **`libldap`** library are also included into the log file on the debug level (i.e. when **LogLevel** `= DEBUG`). **Default value:** `No`
`LdapCheckCertificate` *{No | Allow | Try | Yes}*	The mode of certificate verification for LDAP connections via SSL/TLS. **Allowed values:** • `No`—do not request the server's certificate; • `Allow`—request the server's certificate. If the certificate is not provided, the session will continue in its normal way. If the server's certificate is provided but cannot be checked (it is impossible to find the corresponding root certificate), the certificate will be ignored and the session keeps running in its normal way. • `Try`—request the server's certificate. If the certificate is not provided, the will continue in its normal way. If the server's certificate is provided but cannot be checked (it is impossible to find the corresponding root certificate), the session will be terminated.

	- `Yes`—request the server's certificate. If the certificate is not provided or cannot be checked (it is impossible to find the corresponding root certificate), the session is terminated. *For `LDAP` data sources this certificate verification mode influences the way the URL is processed when the `ldaps://` scheme or the StartTLS extension is used; and for `AD` data sources it will influence connections to the server, if **`UseSSL`**`=Yes` has been specified in the corresponding section (see below).* **Default value:** `Yes`
`LdapCertificatePath` *{path to file}*	Path to the SSL certificate used for connection to the LDAP servers (**Active Directory**) via a secure SSL/TLS connection. *Please note that the certificate file and the private key file (which is specified by a parameter described below) must form a matching pair.* **Default value:** *(not specified)*
`LdapKeyPath` *{path to file}*	Path to the private key used for connection to the LDAP servers (**Active Directory**) via a secure SSL/TLS connection. *Please note that the certificate file and the private key file (which is specified by the mentioned parameter) must form a matching pair.* **Default value:** *(not specified)*
`LdapCaPath` *{path}*	Path to the directory or file with system list of trusted root certificates which are trusted for sharing data through the LDAP protocol via SSL/TLS. **Path to the list of trusted certificates. The path depends on your GNU/Linux** *distribution:* - *For* **Astra Linux**, **Debian**, **Linux Mint**, **SUSE Linux** *and* **Ubuntu**, *usually it is a path* `/etc/ssl/certs/`; - *For* **CentOS** *and* **Fedora**—*a path* `/etc/pki/tls/certs/ca-bundle.crt`. - *For other distributions a path can be defined through results of execution of the command* **openssl** `version -d`. - *If a command is unavailable or an OS distribution could not be identified , the value* `/etc/ssl/certs/` *is used.*

Data Source Sections

In addition to the general section `[LookupD]`, the configuration file should also contain individual sections that describe connections to the **LDAP** and **Active Directory** servers that you use (one section for each connection). These sections are named using the following scheme: `[LookupD.`*<type>*`.`*<name>*`]`, where

- *<type>*—is the connection type: `LDAP` (for **LDAP**) or `AD` (for **Active Directory**).
- *<name>*—is a unique identifier (tag) for the connection, by which the connection can be referred to from the rules.

For example: `[LookupD.LDAP.auth1]`. The set of parameters that are included inside the section of a data source depends on the type of connection. There is no restriction on the number of data source sections.

1. Parameters used in sections of LDAP type

`Url` *{string}*	URL that defines the used LDAP server and extracted data. According to RFC 4516, URL is built on the basis of the following scheme: *<scheme>*`://`*<host>*`[:`*<port>*`]/`*<dn>*`[?`*<attrs>*`[?`*<scope>*`[?`*<filter>*`[?`*<extensions>*`]]]]` Where: *<scheme>*—method of connection to the server (the following schemes are allowed: `ldap`, `ldaps` and `ldapi`); *<host>[:<port>]*—LDAP server address that receives a request; *<dn>*—distinguished name of an object. Information on this object has been sent; *<attrs>*—names of the record attributes, the values of which must be received in the request; *<scope>*—search scope (`base`, `one`, `sub`); *<filter>*—filtering condition for values of extracted attributes. *<extensions>*—list of LDAP extensions used in the request. **Features:** • In the list of attributes *<attrs>*, it is possible to use special characters of choice '`*`', '`+`' and '`1.1`'. • The following automatically resolved placeholders can be used in the *<dn>* and *<filter>* parts of the URL: ▫ `$u` is automatically replaced with the user name, sent by the client component. ▫ `$d` is automatically replaced with the domain, sent by the client component. ▫ `$D`—chain *<subdomain>*.*<domain>*, modified into `dc=`*<subdomain>*`,dc=`*<domain>*. ▫ `$$`—an `'$'` character. • If the condition *<filter>* requires usage of special characters (for example: '`*`', '`(`', '`)`', '`\`', character with code 0) as usual ones, they should be written as `\XX`. Besides, special characters in URL LDAP are encoded using sequences `%XX`. For example, when using URL according to the scheme `ldapi` of the character '`/`' as a part of the path to the local LDAP server socket, this character is encoded as `%2f`. • As allowed extensions in *<extensions>*, only `StartTLS` and `1.3.6.1.4.1.1466.20037` are supported, they include usage of the TLS mechanism (i.e. establishment of the protected connection with the LDAP server, even if it does not explicitly indicate usage of

	the protected scheme `ldaps`) If the name of the used extension is preceded by the character `'!'`, then usage of TLS *is required*, i.e. in case the establishment of the secure connection is impossible, the request *will not* be handled. Otherwise, the request will be handled even if the secure connection is not established.
	Indicated extensions could not be used with the protected `ldaps` scheme. For more information refer to RFC 4516 or **man** `ldap_search_ext_s`.
	Examples:
	`"ldaps://ds.example.com:990/$D?givenName,sn,cn?sub?(uid=$u)"`
	`"ldap://ldap.local/o=org,dc=nodomain?ipNetworkNumber?sub?(objectClass=ipNetwork)?!StartTLS"`
	Default value: *(not specified)*

`BindDn` *{string}*	An object in the LDAP directory to which the user is bound to get authorization. **Example: "cn=admin,dc=nodomain"** **Default value:** *(not specified)*
`BindPassword` *{string}*	The user's password for authentication on the LDAP server **Default value:** *(not specified)*
`ChaseReferrals` *{Boolean}*	Instructs the component to follow references to other LDAP servers, if the current LDAP server returns them as a reply to the request. **Default value:** `No`

2. Parameters used in sections of AD type

`Host` *{string}*	The domain name (FQDN) or the IP address of the host on which the server of the **Active Directory** service that you would like to connect to is running. **Example: "win2012.win.local"** **Default value:** *(not specified)*
`Port` *{integer}*	Port on the host which is listened to by the server of the **Active Directory** service. **Default value:** `389`
`Dn` *{string}*	*DN* of an object in the **Active Directory**; it is similar to the `dn` part of an LDAP URL. **Example: "dc=win,dc=local"**

	Default value: *(not specified)*
`User` *{string}*	The full identifier of a user on the server, to be used for identification. **Example: "Administrator@WIN.LOCAL"** **Default value:** *(not specified)*
`Password` *{string}*	Password of the user for authentication on the **Active Directory** server. **Default value:** *(not specified)*
`ChaseReferrals` *{Boolean}*	Instructs the component to follow references to other LDAP servers, if the current **Active Directory** server returns them as a reply to the request. **Default value:** `No`
`UseSSL` *{Boolean}*	Instructs to use SSL/TLS for connecting to the **Active Directory**. **Default value:** `No`

Adding sections for new data sources

To add a new section for a new data source of a supported type with a <*name*> tag with the help of the Dr.Web Ctl command-line-based management tool for Dr.Web for UNIX Internet Gateways (accessed via the drweb-ctl command), it is necessary to use the following command:

```
# drweb-ctl cfset LookupD.<type> -a <name>
```

Example:

```
# drweb-ctl cfset LookupD.AD -a WinAD1
# drweb-ctl cfset LookupD.AD.WinAD1.Host 192.168.0.20
```

The first command will add a section named `[LookupD.AD.WinAD1]` into the configuration file, and the second command will modify the value of the **`Host`** parameter within this section.

Alternatively, you can write the new section directly into the configuration file, for example, by adding it to the end of the file.

```
[LookupD.AD.WinAD1]
Host = 192.168.0.20
```

Both ways have an equal effect but if you edit the configuration file, you will also need to apply the changed settings by sending a `SIGHUP` signal to the **drweb-configd** component (to do that, you can issue the drweb-ctl `reload` command).

Appendices

Appendix A. Types of Computer Threats

Herein, the term *"threat"* is defined as any kind of software potentially or directly capable of inflicting damage to a computer or network and compromising the user's information or rights (that is, malicious and other unwanted software). In a wider sense, the term "threat" may be used to indicate any type of potential danger to the security of the computer or network (that is, vulnerabilities that can result in hacker attacks).

All of the program types stated below have the ability to endanger user data or confidentiality. Programs that do not conceal their presence in the system (e.g. spam distribution software and various traffic analyzers) are usually not considered as computer threats, although they can become threats under certain circumstances.

Computer Viruses

This type of computer threats is characterized by the ability to embed its code into other programs. Such implementation is called infection. In most cases, an infected file becomes a virus carrier and the embedded code does not necessarily match the original one. Most viruses are intended to damage or destroy data in the system.

In Doctor Web classification, viruses are divided by the type of objects they infect:

- *File viruses* infect files of the operating system (usually executable files and dynamic libraries) and are activated when the infected file is launched.
- *macro-viruses* are viruses that infect documents used by **Microsoft® Office** and some other applications supporting macro commands (for example, written in Visual Basic). *Macro commands* are a type of implemented programs (macros) written in a fully functional programming language. For instance, in **Microsoft® Word**, macros can be automatically Initiated upon opening (closing, saving, etc.) a document.
- *Script viruses* are created using script languages and usually infect other scripts (e.g. service files of an operating system). They are also able to infect other file formats that allow execution of scripts and thus take advantage of scripting vulnerabilities in web applications.
- *boot viruses* infect boot records of disks and partitions or master boot records of hard drives. They do not require much memory and remain ready to continue performing their tasks until a system roll-out, restart or shut-down is performed.

Most viruses have some kind of protection against detection. Protection methods are being constantly improved, and ways to overcome them are constantly being developed. All viruses may also be classified according to protection type they use:

- *Encrypted viruses* cipher their code upon every infection to hamper their detection in a file, boot sector or memory. All copies of such viruses contain only a small common code fragment (the decryption procedure) that can be used as a virus signature.
- *Polymorphic viruses* also encrypt there code, but besides that they also generate a special decryption procedure that is different in every copy of the virus. This means that such viruses do not have byte signatures.
- *Stealth viruses* perform certain actions to disguise their activity and thus conceal their presence in an infected object. Such viruses gather the characteristics of an object before infecting it and then plant these “dummy” characteristics that mislead the scanner searching for modified files.

Viruses can also be classified according to the programming language in which they are written (in most cases, it is Assembler, high-level programming languages, script languages, and others) or according to affected operating systems.

Computer Worms

Recently, malicious programs of the “computer worm” type have become much more common than viruses and other types of malware. Just like viruses, such programs can make copies of themselves, however they do not infect other objects. A worm gets into a computer from a network (most frequently as an attachment to an email or from the Internet) and sends the functioning copies of itself to other computers. To start their spread, worms can either rely on the computer user’s actions or can select and attack computers in an automatic mode.

Worms do not necessarily consist of only one file (the worm’s body). Many of them have an infectious part (the shellcode) that loads into the main memory (RAM) and then downloads the worm’s body as an executable file via the network. If only the shellcode is present in the system, the worm can be deleted by simply restarting the system (at which the RAM is erased and reset). However, if the worm’s body infiltrates the computer, then only an anti-virus program can cope with it.

Worms have the ability to cripple entire networks even if they do not bear any payload (i.e. do not cause any direct damage) due to their intensive distribution.

In Doctor Web classification, worms are divided by distribution method:

- *Net worms* distribute their copies via various network and file-sharing protocols.
- *Mail worms* spread themselves using email protocols (POP3, SMTP, etc.).
- *Chat worms* use protocols of popular messengers and chat programs (ICQ, IM, IRC, etc.).

Trojan Programs (Trojans)

This type of threats cannot reproduce itself. A Trojan substitutes a frequently-used program and performs its functions (or imitates its operation). Meanwhile, it performs some malicious actions in the system (damages or deletes data, sends confidential information, etc.) or makes it possible for hackers to access the computer without permission, for example, to harm the computer of a third party.

A Trojan’s masking and malicious facilities are similar to those of a virus. A Trojan may even be a component of a virus. However, most Trojans are distributed as separate executable files (through file exchange servers, removable data carriers or email attachments) that are launched by users or system tasks.

It is very hard to classify Trojans due to the fact that they are often distributed by viruses or worms and also because many malicious actions that can be performed by other types of threats are attributed to Trojans only. Here are some Trojan types which are distinguished as separate classes in Doctor Web:

- *Backdoors* are Trojans that make it possible for an intruder to log on into the system or obtain privileged functions bypassing any existing access and security measures. Backdoors do not infect files, but they write themselves into the registry modifying the registry keys.
- *rootkits* are used to intercept system functions of an operating system in order to conceal themselves. Besides, a rootkit can conceal processes of other programs (e.g. other threats), registry keys, folders and files. It can be distributed either as an independent program or as a component of another malicious program. There are two kinds of rootkits according to the mode of operation: *User Mode Rootkits (UMR)* that operate in user mode (intercept functions of the user mode libraries) and *Kernel Mode Rootkits (KMR)* that operate in kernel mode (intercept functions on the level of the system kernel, which makes it harder to detect).
- *Keyloggers* are used to log data that users enter by means of a keyboard. The aim of this is to steal personal information (i.e. network passwords, logins, credit card data, etc.).
- *Clickers* redirect hyperlinks to certain addresses (sometimes malicious) in order to increase traffic of websites or perform DDoS attacks.
- *Proxy Trojans* provide anonymous Internet access through a victim’s computer.

In addition, Trojans can also change the start page in a web browser or delete certain files. However, these actions can also be performed by other types of threats (viruses and worms).

Hacktools

Hacktools are programs designed to assist the intruder with hacking. The most common among them are port scanners that detect vulnerabilities in firewalls and other components of computer protection system. Besides hackers, such tools are used by administrators to check security of their networks. Occasionally, common software that can be used for hacking and various programs that use social engineering techniques are designated as among hacktools as well.

Adware

Usually, this term refers to a program code implemented into freeware programs that force display of advertisements to users. However, sometimes such codes can be distributed via other malicious programs and show advertisements in web browsers. Many adware programs operate with data collected by spyware.

Jokes

Like adware, this type of minor threats can not be used to inflict any direct damage to the system. Joke programs usually just generate messages about errors that never occurred and threaten to perform actions that will lead to data loss. Their purpose is to frighten or annoy users.

Dialers

These are special programs that are designed to scan a range of telephone numbers and find those where a modem answers. These numbers are then used to mark up the price of telephoning facilities or to connect the user to expensive telephone services.

Riskware

These software applications were not created for malicious purposes, but due to their characteristics can pose a threat to the computer’s security. Riskware programs can not only damage or delete data, but they are also used by crackers (i.e. malevolent hackers) or by some malicious programs to harm the system. Among such programs, there are various remote chat and administrative tools, FTP-servers, etc.

Suspicious objects

These are possible computer threats detected by the heuristic analyzer. Such objects can potentially be any type of threat (even unknown to IT security specialists) or turn out to be safe in case of false detection. It is recommended that you choose to move the files containing suspicious objects to the quarantine, they also should be sent to Doctor Web anti-virus laboratory for analysis.

Appendix B. Neutralizing Computer Threats

All Doctor Web anti-virus solutions use several malicious software detection methods simultaneously, and that allows them to perform thorough checks on suspicious files and control software behavior.

Detection Methods

Signature Analysis

Signature analysis is the first stage of detection procedure and is used to check file code segments for the presence of known virus signatures. A signature is a finite continuous sequence of bytes necessary and sufficient to identify a specific virus. To reduce the size of the signature dictionary, Dr.Web anti-virus solutions use signature checksums instead of complete signature sequences. Checksums uniquely identify signatures, which preserves correctness of virus detection and neutralization. The Dr.Web virus databases are composed so that some entries can be used to detect not just specific viruses, but whole classes of threats.

Origins Tracing™

On completion of signature analysis, Dr.Web anti-virus solutions use the unique Origins Tracing™ method to detect new and modified viruses which use the known infection mechanisms. Thus, Dr.Web users are protected against such threats as the notorious **Trojan.Encoder.18** ransomware (also known as **gpcode**). In addition to detection of new and modified viruses, the Origins Tracing™ mechanism allows to considerably reduce the number of false positives of the heuristics analyzer. Objects detected using the Origins Tracing™ algorithm are indicated with the `.Origin` extension added to their names.

Execution Emulation

The technology of program code emulation is used for detection of polymorphic and encrypted viruses when a search by checksums cannot be applied directly, or is very difficult to be performed (due to the impossibility of building secure signatures). The method implies simulating the execution of an analyzed code by an *emulator*—a programming model of the processor and runtime environment. An emulator operates with protected memory area (*emulation buffer*), in which execution of the analyzed program is modelled instruction by instruction. However, none of these instructions is actually executed by the CPU. When the emulator receives a file infected with a polymorphic virus, the result of the emulation is a decrypted virus code, which is then easily determined by searching against signature checksums.

Heuristic Analysis

The detection method used by the heuristics analyzer is based on certain knowledge (*heuristics*) about certain features (attributes) than might be typical for the virus code itself, and vice versa, that are extremely rare in viruses. Each attribute has a *weight* coefficient which determines the level of its severity and reliability. The weight coefficient can be positive if the corresponding attribute is indicative of a malicious code or negative if the attribute is uncharacteristic of a computer threat. Depending on the sum weight of a file, the heuristics analyzer calculates the probability of unknown virus infection. If the threshold is exceeded, the heuristic analyzer generates the conclusion that the analyzed object is probably infected with an unknown virus.

The heuristics analyzer also uses the FLY-CODE™ technology, which is a versatile algorithm to extract packed files. The technology allows making heuristic assumptions about the presence of malicious objects in files compressed not only by packers that Dr.Web is aware of, but by also new, previously unexplored programs. While checking packed objects, Dr.Web Anti-virus solutions also use structural entropy analysis. The technology detects threats by the characteristic way in which pieces of code are arranged inside a file; thus, one virus-database entry allows identification of a substantial portion of threats packed with the same polymorphous packer.

As any system of hypothesis testing under uncertainty, the heuristics analyzer may commit type I or type II errors (omit viruses or raise false positives). Thus, objects detected by the heuristics analyzer are treated as “suspicious”.

While performing any of the checks previously mentioned, Dr.Web anti-virus solutions use the most recent information about known malicious software. As soon as experts of Doctor Web anti-virus laboratory discover new threats, an update for virus signatures, behavior characteristics and attributes is issued. In some cases updates can be issued several times per hour. Therefore even if a brand new malicious program passes through the Dr.Web resident guards and penetrates the system, then after an update the malicious program is detected in the list of processes and neutralized.

Actions

To avert computer threats, Dr.Web products use a number of actions that can be applied to malicious objects. A user can leave the default settings, configure which actions to apply automatically, or choose actions manually upon every detection. Below, you can see a list of available actions:

- *Ignore*—instructs to skip the detected threat without performing any other action.
- *Report*—instructs to inform on the detected threat without performing any other action.
- *Cure*—instructs to cure the infected object by removing only malicious content from its body. Note that this action cannot be applied to all types of threats.
- *Quarantine (Move to Quarantine, Isolate)*—instructs to move the detected threat to a special directory and isolate it from the rest of the system.
- *Delete*—instructs to remove the infected object permanently.

If threat is detected in a file located in a container (an archive, email message, etc.), its removal is replaced with moving of a container to quarantine.

Appendix C. Contacting Technical Support

If you encounter any issues installing or using company products, before requesting for the assistance of the technical support, take advantage of the following options:

- Download and review the latest manuals and guides at https://download.drweb.com/doc/.
- Read the frequently asked questions at https://support.drweb.com/show_faq/.
- Browse the Dr.Web official forum at https://forum.drweb.com/.

If you have not found solution for the problem, you can request direct assistance from Doctor Web company technical support by one of the following ways:

- Fill in the web form in the corresponding section at https://support.drweb.com/.
- Call by phone in Moscow: +7 (495) 789-45-86.

Refer to the official website at https://company.drweb.com/contacts/offices/ for regional and international office information of Doctor Web company.

Appendix D. Configuration File

Configuration parameters of all Dr.Web for UNIX Internet Gateways components are managed by a special coordinating daemon Dr.Web ConfigD. These parameters are stored in the `drweb.ini` file, which default directory is *etc_dir* (for **GNU/Linux** `/etc/opt/drweb.com`).

> The text configuration file stores only those parameters which values differ from the defaults. If a parameter is absent in the configuration file, its default value is used.
>
> For details on conventions for <*opt_dir*>, <*etc_dir*>, and <*var_dir*>, refer to the Introduction.

You can view the list of all available parameters, including those that are absent in the configuration file and have default values, by using the following command:

```
$ drweb-ctl cfshow
```

You can change any parameter value in one of the two following ways:

1. Specify the parameter in the configuration file (by editing the file in any text editor) and send SIGHUP signal to the configuration daemon (the **drweb-configd** component) in order to apply the changes (to do that, you can issue the **drweb-ctl** `reload` command).
2. Type this command in the command line

```
# drweb-ctl cfset <section>.<parameter> <new value>
```

> Note, that this command can be executed only if the management tool Dr.Web Ctl is run with superuser privileges. To gain superuser privileges, use **su** or **sudo** command.

For further information about the `cfshow` and `cfset`command syntax of the command-line management tool Dr.Web Ctl (the **drweb-ctl** module), refer to the section Dr.Web Ctl.

File Structure

The configuration file has the following structure:

- File content is divided into named sections. Possible names of these sections are strictly predefined and cannot be changed. The section name is specified in square brackets and is similar to the component name Dr.Web for UNIX Internet Gateways, which uses the section parameters (except for [Root] section , which stores all parameters of the configuration daemon Dr.Web ConfigD).
- The '`;`' or '`#`' characters in the configuration file indicate the beginning of a comment—all text following the characters is skipped by components Dr.Web for UNIX Internet Gateways while reading configuration parameters.

- One line of the file can contain only one parameter value. The general format of specifying the value is as follows (white spaces before and after the character '=' are ignored):

  ```
  <Parameter name> = <Value>
  ```

- All parameter names are strictly predefined and cannot be changed.
- All section and parameter names are case-insensitive. Parameter values, except for names of directories and files in paths (for **UNIX**-like OS) are also case-insensitive.
- Order in which sections are specified in the file and order in which parameters are specified in the section are of no importance.
- Parameter values in the configuration file can be enclosed in quotation marks, and must be enclosed in quotation marks if they have white spaces
- Some parameters can have a list of values. If so, the values are either separated with commas or specified several times in different lines of the configuration file. In the former case, white spaces around a comma are ignored. If a white space character is a part of a parameter value, the character must be enclosed in quotation marks.

  Example of how to specify several values for one parameter:

  1) As a comma-separated list:

  ```
  Parameter = Value1, Value2,"Value 3"
  ```

  2) In different lines of the configuration file:

  ```
  Parameter = Value2
  Parameter = Value1
  Parameter = "Value 3"
  ```

  Note that the order in which parameter values are specified is of no importance.

  > ⚠ If parameter values are paths, then each listed parameter value must be put between quotation marks if the listing of components via commas is used. For example, if the parameter **ExcludedPaths** requires two paths `/etc/file1` and `/etc/file2`, then this parameter must be written into a configuration file either as a string
  >
  > ```
  > ExcludedPaths = "/etc/file1", "/etc/file2"
  > ```
  >
  > or as two strings
  >
  > ```
  > ExcludedPaths = /etc/file1
  > ExcludedPaths = /etc/file2
  > ```
  >
  > Otherwise, the component that uses this parameter can interpret the string '`/etc/file1, /etc/file2`' as one path.

- If a parameter can have more than one values, it is designated explicitly. So, if this is not explicitly designated in the current manual or within the comments in the configuration file, the parameter can have only one value.

For description of the configuration file sections, see description of Dr.Web for UNIX Internet Gateways components.

Parameter Types

Configuration parameters can be of the following types:

- *Address*—network connection address specified as *<IP address>:<port>* pair of values. In some cases, the port value can be omitted (if so, it is specified in the parameter description).
- *Boolean*—flag used as an indicator. Such parameters can have either `Yes` or `No` as values.
- *Integer*—parameter value can be a non-negative integer as a value.
- *Fractional number*—parameter value can be a non-negative number with a fractional part.
- *Time interval*—parameter value can be a time interval, consisting of a non-negative integer and a suffix (letter), which stands for a time unit. The following suffixes can be used:
  - `w` – weeks (`1w` = `7d`);
  - `d` – days (`1d` = `24h`);
  - `h` – hours (`1h` = `60m`);
  - `m` – minutes (`1m` = `60s`);
  - `s`—seconds.

  If the suffix is omitted, the interval is considered as in seconds. For the time interval, expressed in seconds, it is allowed to specify milliseconds after a point (but no more than three digits after the separator, for example, `0.5s` – 500 milliseconds). It is possible to specify several time intervals in different time units. In this case, the resulting interval is counted as a sum of intervals (in fact, a time interval is always converted to milliseconds before the value is written to configuration).

  In general terms, any time an interval can be represented as an expression of this form: N_1`w`N_2`d`N_3`h`N_4`m`N_5[.N_6]`s`, where N_1, ..., N_6 is a number of corresponding time unites included in this interval. For example, a year (365 days) can be represented as follows (all records are equal):`365d`, `52w1d`, `52w24h`, `51w7d24h`, `51w7d23h60m`, `8760h`, `525600m`, `31536000s`.

  The examples below show you how intervals of 30 minutes, 2 seconds, 500 milliseconds can be specified:

  1. In the configuration file:

  ```
  UpdateInterval = 30m2.5s
  ```

  2. Using the command **drweb-ctl** `cfset`:

  ```
  # drweb-ctl cfset Update.UpdateInterval 1802.5s
  ```

  3. Via a command-line parameter (for example, for the Command Line Arguments):

  ```
  $ drweb-se --WatchdogInterval 1802.5
  ```

- *Size*—parameter value can be the size of an object (file, buffer, cash, and so on), consisting of a non-negative integer and a suffix, which stands for a unit. The following suffixes can be used:
  - `mb`—megabytes (`1mb = 1024kb`);
  - `kb`–kilobytes (`1kb = 1024b`);
  - `b`–bytes.

  If the suffix is omitted, the size is considered as in bytes. It is possible to specify several sizes in different units. In this case, the resulting size is counted as their sum (in fact, a size value is always converted to bytes).
- *path to a directory (file)*—parameter value can be a string, which is a path to a directory (file). Note that the file path must be ended with the file name.

> In UNIX-like systems, names of catalogs and files are case sensitive. If it is not explicitly designated in a parameter description, paths cannot contain masks with special characters (`?`, `*`).

- *Logging level*—the level at which Dr.Web for UNIX Internet Gateways events are logged. The parameter of this type can have the following values:
  - `DEBUG`—the most detailed logging level. All messages and debug information are registered.
  - `INFO`—all messages are registered.
  - `NOTICE`—all error messages, warnings, and notifications are registered.
  - `WARNING`—all error messages and warnings are registered.
  - `ERROR`—only error messages are registered.
- *Log type*—parameter value defines how Dr.Web for UNIX Internet Gateways performs logging (its logging method). The parameter of this type can have the following values:
  - `Stderr[:ShowTimestamp]`—Messages are displayed in the *stderr*—standard error stream. This value can be used *only* in the settings of configuration daemon. At that, if it works in background mode ("*daemonized*"), i.e. it is launched with the parameter `-d` specified, this value *cannot* be used because components operating in the background mode cannot access I/O streams of the terminal). The additional parameter `ShowTimestamp` instructs to add a time stamp to every message.
  - `Auto`—messages for logging are sent to the configuration daemon Dr.Web ConfigD, which saves them to one location according to its configuration (the parameter **`Log`** in the `[Root]` section). This value is specified for all components *except for the configuration daemon* and is used as a default value.
  - `Syslog[:`<*facility*>`]`—messages are transmitted to the system logging service **syslog**.
  - Additional option <*facility*> is used to specify a level at which **syslog** registers messages. The following values are possible:
    - `DAEMON`—messages of daemons.
    - `USER`—messages of user processes.
    - `MAIL`—messages of mail programs.

        - `LOCAL0`—messages of local processes 0.

        …

        - `LOCAL7`—messages of local processes 7.

    - <*path*>—Messages are to be saved directly to the specified log.

    Example of how to specify the parameter value:

    1. In the configuration file:

    ```
    Log = Stderr:ShowTimestamp
    ```

    2. Using the command **drweb-ctl** `cfset`:

    ```
    # drweb-ctl cfset Root.Log /var/opt/drweb.com/log/general.log
    ```

    3. Via a command-line parameter (for example, for the Command Line Arguments):

    ```
    $ drweb-se --Log Syslog:DAEMON
    ```

- *action*—action performed by Dr.Web for UNIX Internet Gateways upon detection of certain threats or upon another event. The following values are possible:
    - `Report`—instructs only to notify on threat detection without performing any other action.
    - `Cure`—instructs to attempt to cure the threat (that is, remove only malicious content).
    - `Quarantine`—instructs to move the infected file to quarantine.
    - `Delete` — instructs to delete the infected file.

    > ⚠ Some of the actions can be applied only upon certain events (for example, a "scanning error" event cannot trigger the `Cure` action). Allowed actions are always listed in the parameter description of the *action* type.

Other parameter types and their possible values are specified in the description of these parameters.

Rules for Traffic Monitoring

The rules are represented by production rules such as IF <*conditional_part*> THEN <*action_part*>. At that, in the part <*conditional_part*> the following scanning types are specified: *"The variable value is (not) set"* or *"The variable value is (not) included in the specified set"*. The part <*action_part*> contains a set of (at least one) actions, and each of these actions is an *ultimate resolution* (skip or block a scanned object), or a*modifying action* which looks as *"Change features of the scanned object"*, *"Assign the set value to the specified variable"* or *"Add the set value to the array of values of the specified variable"*.

Part of the rule actions is executed only if the conditional part is true. If the conditional part evaluates to false, the actions specified in this rule are not performed, and the program jumps to the next rule. The rules are considered vertically down until an ultimate resolution is performed. After this, all undermentioned rules (if there are any) are ignored. When a rule is executed, it is important that actions in <*action_part*> are performed in order of their specification from left to right, and if there is an ultimate resolution in the chain of actions that interrupts the rule handling, the rest of the actions specified in <*action_part*> is not performed.

Rule Format

Format of the rule production

```
[<condition>[, <condition>[, …]]] : <action>[, <action>[, …]]
```

The conditional part of the rule (before `':'`) can be missing, in this case a part of the actions is executed without any condition. If the conditional part of the rule is missing, the `':'` separator can be omitted. The comma between conditions in the conditional part and actions in the action part performs a role of a logical conjunction (that is, “and”): the conditional part elevates to true, only if all its conditions are true, and all actions specified in the action part are performed in order of their specification from left to right until an ultimate resolution which interrupts the rule handling. In the rules the register is not important for the key words, names of variables and configuration parameters.

Conditions

The following types of conditions can be use in the conditional part of the rules:

Condition	Meaning of the Condition
<*variable*> <*value* >	The value of the specified variable coincides with the set value. *Can be used only for those variables that can contain a set of values simultaneously.*
<*variable*>`[not] in` <*set of values*>	The value of the specified variable is contained in the specified set of values (*for not—does not match any value from the specified set*).
<*variable*> `[not] match` <*set of values*>	The value of the specified variable matches any regular expression listed in the specified set (*for not—does not match any expression from the specified set*). Regular expressions are specified using either the *POSIX* syntax (*BRE*, *ERE*) or the *Perl* syntax (*PCRE*, *PCRE2*).

Condition	Meaning of the Condition
`<variable> [not] gt <value>`	The value of the specified variable is (not) greater than the set value. *Can be used only for those variables that can have a single value.*
`<variable> [not] lt <value>`	The value of the specified variable is (not) less than the set value. *Can be used only for those variables that can have a single value.*

*) An optional key word `not` means negation.

Part <*set of values*> to which a variable is compared can be specified in the following ways:

Syntax	Meaning
`(<value 1>[, <value 2>[, ...]])`	In the parentheses you directly list the set of values to check against (not less then one value). In case there is only one value and the `in` condition is used, you can omit the parentheses (and you will end up with a case <*variable*> <*value*>).
`"<section>.<parameter>"`	The set of values currently assigned to a certain configuration parameter; where between the quotation marks you should specify the name of a configuration parameter whose value (or set of values) must be checked (note that you also need to specify the name of the section to which the parameter belongs). The lists of the parameters that can be used as conditions depend on the component for which the rules are set. The lists are provided below.
`file("<file name>")`	List of values is read from the text file <*file name*> (one file string—one list element, leading and trailing spaces in strings are ignored). A path to the file must be absolute. If a <*file name*> contains quotes and apostrophes, they must be escaped: '\'.

Syntax	Meaning
	The file size must not exceed 64 MB. The file contents are read and inserted into the rules once, during the download of the configuration file. If there is no file or the file size is exceeded, an error `x102` appears during the download. In case the file contents are changed during the process, in order to apply all changes, you should reboot your computer after the changes are saved using the command **drweb-ctl** `reload`. A set of values from the file is not available for all variables. Whether you can use a variable to scan its value by using a set of values from the file is indicated below.
<*type_of_LOOKUP_request*>`@`<*tag*>`[@`<*value*>`]`	A set of values is requested via Dr.Web LookupD from an external data source (LDAP, ActiveDirectory), where <*LOOKUP_request_type*> is the type of the data source used (LDAP or AD); <*tag*> is a section name describing the connection that is used to sample the data, and <*value*> (optional) is a value that must be contained in the set of values retrieved from the data source. Values from Dr.Web LookupD are not available for all variables. Also, the condition <*scanning*> cannot be applied to all variables. Whether you can use a variable to scan its value by using Dr.Web LookupD is indicated below.

If a variable is multiple-valued, the condition <*variable*> `in` <*set of values*> is true, if intersection of the set of current values of the specified variable <*variable*> with the indicated set <*set of values*> is not empty. The condition `not in` is true in the opposite case. For example, suppose `X` is a variable, which the current value is a set with values *a*, *b*, *c*. Then

- `X in (a, b)` is true because values *a* and *b* are encountered in both sets;
- `X in (a, d, e)` is true because value *a* is encountered in both sets;
- `X in (d, e)` is false because there is no value of the variable (*a*, *b*, *c*) in the set (*d*, *e*).
- `X in ()` – false as array of variable values is not empty.
- `X not in ()` – true, the array of variable values is not empty.
- `X not in (d, e)` is true because there is no value of the variable (*a*, *b*, *c*) in the set (*d*, *e*).
- `X not in (a, d, e)` is false because value *a* is encountered in both sets;

In the description of the variables below, there is an indication for each variable whether it can adopt a set of values.

Actions

The actions can be divided into *ultimate resolutions*, defining whether the object is blocked or allowed and *actions that change the value of a variable*, which can be used to check the downward conditions.

Ultimate Resolutions

Resolution	Description (Meaning)
Common Resolutions	
`PASS`	Skip traffic (allow connection creation, send an object to a recipient). The downward rules (if there are any) are not used. For the rules of mail processing, there is merit in a command that allows a message to be transmitted to a recipient after all collected changes have been applied to it (i. e. all executed actions `REPACK`, `ADD_HEADER`, `CHANGE_HEADER`, see below).
`BLOCK as` <*reason*>	Block traffic (block connection creation, send an object to a recipient). The downward rules (if there are any) are not used. A blocking <*reason*> is recorded in the log. The same reason is used to define a browser notification to be shown to a user. Two standard reasons can be used as <*reason*> for `BLOCK`: • `BlackList`—the data is blocked because it is included in user's black list. • `_match`—the block happens because a web resource or file containing threat belongs to a category that triggers rule executing (for conditions `*_category in (...)`). The `_match` variable

Resolution	Description (Meaning)
	contains the list of blocked categories for which the correspondence has been executed.

Features of handling ultimate resolutions:

- `BLOCK as BlackList`, always processes as *"is included in a black list"* (without considering the condition specified in the rules with this resolution).
- `BLOCK as _match`, if `_match` is not empty, processes as *"belongs to the _match category"*.
- `BLOCK as _match`, if `_match` is empty, processes as *"is included in a black list"* (without considering the condition specified in the rules with this resolution).
- If all rules have been considered, and none of the rules with resolutions performs (or the rules do not have resolutions), this situation is the same as `PASS` action.

> For SpIDer Gate rules that process non-mail-related connections (i.e. when the previous condition says that the traffic must be HTTP traffic), the triggering of a mail-related resolution is treated as equivalent to the triggering of a `BLOCK as BlackList` resolution (additionally, a message about applying an unknown action is recorded into the log).
>
> In the rules for Dr.Web ICAPD, mail-related resolutions have no meaning and no effect: the rules that contain mail-related resolutions are ignored without any reaction to them.

Changing Value of a Variable

To change the variable value, the following instruction is used:

```
SET <variable> = ([<value 1>[, <value 2>[, ...]]])
```

If nothing is enclosed in brackets, the list of variable values is cleared. If there is only one value, the brackets should be omitted, that is, the following syntaxes should be used:

```
SET <variable> = <value >
```

Variables used in the rules

When indicating variables in the rules, the register of symbols is not considered. The variables with compound names could be saved using underscore for spacing or without it. Thus, records `variable_name`, `VariableName` and `variablename` represent the same variable. In this section, all variables are saved using underscore (i.e. `variable_name`).

Variable	Description	Can be used in	
		conditional part	action part (SET)

`protocol`	Network protocol type, used by the connection.
	The variable can simultaneously contain a set of values.
	Allowed values: `HTTP`, `SMTP`, `IMAP`, `POP3`.
	Usage Aspects:
	• The variable value can be defined only if SSL/TLS is not used or it was allowed to unwrap SSL.
	• It does not make sense to specify any other value except`HTTP` for the Dr.Web ICAPD rules: only HTTP can be specified for Dr.Web ICAPD.
	• A set of values for checking a variable value is available from the file.
	Examples:
	`protocol in (HTTP, SMTP)`
	`protocol in (POP3)`
	`protocol in file("/etc/file")`
		Yes	No

`sni_host`	SNI host (address), with which the connection is established via SSL/TLS.
	Usage Aspects:
	• If SSL is not used, the value of a variable is not defined, the condition evaluates to false.
	• It does not make sense to use it for the Dr.Web ICAPD rules (it does not process SSL, for that reason the condition always evaluates to false).
	• A set of values for checking a variable value is available from the file.
	Examples:
	`sni_host not in ('vk.com', 'ya.ru')`
	`sni_host in "LinuxFirewall.BlackList"`
		Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	`sni_host in`
	`file("/etc/file")`		

`sni_category`	The list of categories (*AdultContent*, etc.) which the host (that is identified from the SNI-header) belongs to (according to the databases of web resource categories), for hosts to which your computer is attempting to connect over SSL/TLS.
	The variable can simultaneously contain a set of values.
	Usage Aspects:
	• If SSL is not used, the value of a variable is not defined, the condition evaluates to false.
	• It does not make sense to use it for the Dr.Web ICAPD rules (it does not process SSL, for that reason the condition always evaluates to false).
	• For rules used by Dr.Web MailD and Dr.Web ICAPD, condition with `not in` will be *true*, even if according to the scanning results, the host does not belong to any of the predetermined categories ("safe" host). For rules of Dr.Web Firewall for Linux (SpIDer Gate), the condition in this case will be *false*.
	• If databases of web resource categories are not installed, the variable could not be used in rules (attempts to check if a condition in the rule is true will lead to the error x112).
	• A set of values for checking a variable value is available from the file.
	Examples:
	`sni_category not in`
	`(AdultContent, Chats)`
	`sni_category in`
	`"LinuxFirewall.BlockCategory"`
	`sni_category in (FreeEmail)`
	`sni_category not in`
	`file("/etc/file")`
		Yes	No

Variable	Description	Can be used in	
		conditional part	**action part (SET)**

`url`	URL requested by the client. Can be compared with the specified string or with a regular expression.
	Usage Aspects:
	• Can be used only in rules for Dr.Web ICAPD.
	• Dr.Web LookupD can be used to check the value of this variable.
	• A set of values for checking a variable value is available from the file.
	Examples:
	`url match ("drweb.com", "example\..*", "aaa\.ru/")`
	`url match "ICAPD.Adlist"`
	`url not match LDAP@BadURLs`
	`url match file("/etc/file")`
		Yes	No

`url_host`	URL/host with which the connection is established.
	Usage Aspects:
	• The variable value can be defined only if SSL/TLS is not used or it was allowed to unwrap SSL.
	• Dr.Web LookupD can be used to check the value of this variable.
	• A set of values for checking a variable value is available from the file.
	Examples:
	`url_host in ('vk.com', 'ya.ru')`
	`url_host not in "ICAPD.Whitelist"`
	`url_host in LDAP@hosts`
	`url_host not in file("/etc/file")`
		Yes	No

`url_category`	The list of categories to which the URL/host belongs. The information is based according to the database of categories or Dr.Web Cloud replies.	Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	The variable can simultaneously contain a set of values.
	Usage Aspects:
	• The variable value can be defined only if SSL/TLS is not used or it was allowed to unwrap SSL.
	• For rules used by Dr.Web MailD and Dr.Web ICAPD, condition with `not in` will be *true*, even if according to the scanning results, URL/host does not belong to any of the predetermined categories (“safe” URL/host). For rules of Dr.Web Firewall for Linux (SpIDer Gate), the condition in this case will be *false*.
	• If databases of web resource categories are not installed, the variable could not be used in rules (attempts to check if a condition in the rule is true will lead to the error x112).
	• A set of values for checking a variable value is available from the file.
	Examples:
	`url_category not in (AdultContent, Chats)`
	`url_category in "LinuxFirewall.BlockCategory"`
	`url_category in (FreeEmail)`
	`url_category in file("/etc/file")`		

`threat_category`	The list of categories to which the threat belongs, which is found in the transferred data (according to information from virus databases). *The variable can simultaneously contain a set of values.* **Usage Aspects:** • The variable value can be defined only if SSL/TLS is not used or it was allowed to unwrap SSL.	Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	- For rules used by Dr.Web MailD and Dr.Web ICAPD, condition with `not in` will be *true*, even if according to the scanning results, the object does not contain threats from any of the predetermined categories (“safe” object). For rules of Dr.Web Firewall for Linux (SpIDer Gate), the condition in this case will be *false*.
	- A set of values for checking a variable value is available from the file.
	Examples:
	`threat_category in "LinuxFirewall.BlockThreat"`
	`threat_category not in (Joke)`
	`threat_category in file("/etc/file")`		

`user`	The name of the user with whose privileges the process that is sending (or receiving) the traffic has been launched. **Usage Aspects:** - In the Dr.Web ICAPD rules, the name of that user is implied who has authenticated on the proxy server (if the proxy server supports authentication). If the proxy server does not support user authentication, the variable has an empty value. - Dr.Web LookupD can be used to check the value of this variable. - If you need to find out whether the user belongs to a certain user group, use an LDAP or an Active Directory data source that returns a list of groups and specify the name of the required group (for which you want to know whether the user is its member or not). Use the following format: <*type of the source for LookupD*>@<*source of groups*>@<*required group*>. Requests to Active Directory (AD@) return only lists of groups, therefore for these requests it	Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	is mandatory to use the @<*required group*> part.
	• A set of values for checking a variable value is available from the file.
	Examples:
	`user in ('user1', 'user2')`
	`user in AD@Winusergroups@Admins`
	`user in LDAP@AllowedUsers`
	`user not in file("/etc/file")`		

`src_ip`	The IP address of a host establishing the connection.
	Usage Aspects:
	• Dr.Web LookupD can be used to check the value of this variable.
	• A set of values for checking a variable value is available from the file.
	Examples:
	`src_ip not in (127.0.0.1, 10.20.30.41, 198.126.10.0/24)`
	`src_ip in LDAP@AllowedAddresses`
	`src_ip not in file("/etc/file")`
		Yes	No

`proc`	The process establishing the connection (the full path to the executable file).
	Usage Aspects:
	• It does not make any sense to use it for the Dr.Web ICAPD rules (the component does not contain information about processes, for that reason the condition always evaluates to false).
	• A set of values for checking a variable value is available from the file.
	Examples:
	`proc in ('/usr/bin/ls')`
	`proc not in ('/home/user/myapp', '/bin/bin1')`
	`proc in`
		Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	`"LinuxFirewall.ExcludedProc"`
	`proc in file("/etc/file")`		

`direction`	The type of traffic on the connection.
	Allowed values: `request` (client request), `response` (server reply).
	This variable cannot simultaneously contain a set of values; conditions of the `match` *and* `in` *type cannot be applied.*
	Examples:
	`direction request`
	`direction not response`
		Yes	No

`divert`	The direction of the connection.
	Allowed values: `input` (incoming—created/initiated from outside the local host), `output` (outgoing—created/initiated on the local host).
	This variable cannot simultaneously contain a set of values; conditions of the `match` *and* `in` *type cannot be applied.*
	Examples:
	`divert input`
	`divert not output`
		Yes	No

`content_type`	MIME type of data transferred during connection. **Usage Aspects:** • Can be defined if only SSL/TLS is not used or it was allowed to unwrap SSL. • The expression "`*/*`" matches data of any MIME type and HTTP replies without the header `Content-Type`. • Dr.Web LookupD can be used to check the value of this variable. • A set of values for checking a variable value is available from the file.	Yes	No

Variable	Description	Can be used in	
		conditional part	action part (SET)

	Examples:
	`content_type in ("multipart/byteranges", "application/octet-stream")`
	`content_type not in ("text/*", "image/*")`
	`content_type not in ("audio/*")`
	`content_type in ("*/*")`
	`content_type in LDAP@BlockedContent`
	`content_type not in file("/etc/file")`		

`unwrap_ssl`	Whether the traffic transferred via SSL/TLS is unwrapped.
	Allowed values: `true`, `false`.
	Usage Aspects:
	• The variable always has any value. The instruction `SET unwrap_ssl = ()` is impossible.
	• The variable cannot be used as a condition. It is necessary only to control SSL unwrapping (for example, to display a webpage containing notification about blocking triggered by our side).
	• It does not make sense to use it for the Dr.Web ICAPD rules (it does not process SSL, changing of the variable does not influence rule processing).
	Examples:
	`SET unwrap_ssl = TRUE`
	`set Unwrap_SSL = false`
		No	Yes

`http_templates_dir`	The path to the directory where the notification page template on blocking HTTP request is stored. If the path starts with a / (forward slash), it is an absolute path; if it starts with any other symbol, then it is a relative path. In the latter case it is given relative to the directory specified in the **`TemplatesDir`** parameter.	No	Yes

Variable	Description	Can be used in	
		conditional part	**action part (SET)**

	Usage Aspects:
	• It is useful only for the HTTP(S) protocol.
	Examples:
	`SET http_templates_dir = "/etc/mytemplates"`
	`set http_templates_dir = "templates_for_my_site"`		

Categories of unwanted websites and threats

1. Categories of unwanted websites (for the variables `sni_category`, `url_category`)

Convention	Website category
InfectionSource	Websites containing malicious software (“infection sources”).
NotRecommended	Fraudulent websites (that use “social engineering”) visiting which is not recommended.
AdultContent	Websites, containing porn or erotic materials, dating sites etc.
Violence	Websites that encourage violence or contain materials about various fatal accidents, etc.
Weapons	Websites that describe weapons and explosives or provide information on their manufacturing.
Gambling	Websites that provide access to online games of chance, casinos, auctions, including sites for placing bets, etc.
Drugs	Websites that promote use, production or distribution of drugs, etc.
ObsceneLanguage	Websites that contain the obscene language (in titles, articles, etc.).
Chats	Websites that offer a real-time transmission of text messages.
Terrorism	Websites that contain aggressive and propaganda materials or terroristic attacks descriptions, etc.
FreeEmail	Websites that offer the possibility of free registration of a web mailbox.
SocialNetworks	Different social networking services: general, professional, corporate, interest-based; thematic dating sites.
DueToCopyrightNotice	Wbsites, links to which are defined by the copyright holders of some copyrighted work (movies, music, etc.).

As values of the variables `sni_category` *and* `url_category`, *it is also possible to use names of the parameters that control blocking (see below).*

2. Threat categories (for the `threat_category` variable)

Convention	Threat categories
KnownVirus	Known threat (virus).
VirusModification	Modification of the known threat (virus).
UnknownVirus	Unknown threat, suspicious object.
Adware	Adware.
Dialer	Dialer.
Joke	Joke.
Riskware	Riskware.
Hacktool	Hacktool.

As a value of the variable `threat_category`, *it is also possible to use names of the parameters that control blocking (see below).*

Configuration parameters that can be used in rule conditions

Parameters used in the component rules of Dr.Web Firewall for Linux (indicated with the prefix `LinuxFirewall.`):

Parameter	Description and Usage Example

`Whitelist`	White list contains the list of domains, the access to which is allowed, even if these domains are included in the database of categories.
	Examples:
	`sni_host in "LinuxFirewall.Whitelist" : PASS`
	`url_host not in "LinuxFirewall.Whitelist" : BLOCK as _match`

`Blacklist`	Black list contains the list of domains, the access to which is blocked by the user (or the administrator).
	Examples:
	`sni_host in "LinuxFirewall.Blacklist" : SET Unwrap_SSL = FALSE`
	`url_host in "LinuxFirewall.Blacklist" : BLOCK as BlackList`

Parameter	Description and Usage Example

`BlockCategory`	“Meta-parameter”: its value is a list of names of those web resource categories (*Chats, AdultContent*, etc.) for which the corresponding **`Block*`** parameters in the `[LinuxFirewall]` section are set to `Yes`.
	Examples:
	`url_category in "LinuxFirewall.BlockCategory" : BLOCK as _match`
	`sni_category in "LinuxFirewall.BlockCategory" : BLOCK as BlackList`

`BlockThreat`	“Meta-parameter”: its value is a list of names of those threat types (*KnownVirus, Joke*, etc.) for which the corresponding **`Block*`** parameters in the `[LinuxFirewall]` section are set to `Yes`. **Examples:** `threat_category in "LinuxFirewall.BlockThreat" : BLOCK as _match`
`ExcludedProc`	The list of trusted processes, whose traffic must be skipped from checking. **Examples:** `proc in "LinuxFirewall.ExcludedProc" : PASS`

Parameters, used in the component rules of Dr.Web ICAPD (indicated with the prefix `ICAPD.`):

Parameter	Description and Usage Example
`Whitelist`	White list contains the list of domains, the access to which is allowed, even if these domains are included in the database of categories. **Examples:** `url_host not in "ICAPD.Whitelist" : BLOCK as BlackList`
`Blacklist`	Black list contains the list of domains, the access to which is blocked by the user (or the administrator). **Examples:** `url_host in "ICAPD.Blacklist" : BLOCK as BlackList`
`Adlist`	The Advertisements List. Stores a list of regular expressions that describe advertising sites. It is created by the user (or by the administrator). **Examples:** `url match "ICAPD.Adlist" : BLOCK as BlackList`
`BlockCategory`	“Meta-parameter”: its value is a list of names of those web resource categories (*Chats, AdultContent*, etc.) for which the corresponding **`Block*`** parameters in the

Parameter	Description and Usage Example
	[ICAPD] section are set to Yes. **Examples:** `url_category in "ICAPD.BlockCategory" : BLOCK as _match`
BlockThreat	"Meta-parameter": its value is a list of names of those threat types (*KnownVirus*, *Joke*, etc.) for which the corresponding **Block*** parameters in the [ICAPD] section are set to Yes. **Examples:** `threat_category in "ICAPD.BlockThreat" : BLOCK as _match`

Features of saving rules to the configuration file

- In the configuration file, in the settings sections of components that use rules, the rules are stored in such variables as **RuleSet**, each of them is a set (sequence) of unlimited number of rules. In addition, rules in each set are considered sequentially (vertically down) until the ultimate resolution is met.
- When writing an unconditional rule (rule that contains only actions without a conditional part) to the configuration file, an empty conditional part and a separator ':' will be added to it.

  For example, the following rule, which does not contain a conditional part and *consisting only of the action*:

  ```
  BLOCK as _match
  ```

  will be written to the configuration file as follows:

  ```
   : BLOCK as _match
  ```

- When writing a rule, which contains in the action part the set of *multiple* actions, to the configuration file, it will be written as a sequence of rules with the same conditional part and one action in the action part in the same order as the actions are listed.

  For example, the following rule that contains *two actions* in the action part:

  ```
  user in ('user1', 'user2') : SET http_templates_dir = "/etc/mytemplates",
  BLOCK as _match
  ```

  will be written to the configuration file as *sequences of two rules*:

  ```
  user in ('user1', 'user2') : SET http_templates_dir = "/etc/mytemplates"
  user in ('user1', 'user2') : BLOCK as _match
  ```

- The logging or rules does not allow for disjunction (logical "OR") of conditions in the conditional part, so, in order to implement the logical "OR", the chain of rules should be logged with each rule having an only disjunct-condition in its condition. For example, the following two rules are equal to the rule “Block if a malicious object *KnownVirus* or URL from the category *Terrorism* are detected”:

```
threat_category in (KnownVirus) : BLOCK as _match
url_category in (Terrorism) : BLOCK as _match
```

as the following records are equivalent: $(a \rightarrow x,\ b \rightarrow x)$; $((a \rightarrow x) \wedge (b \rightarrow x))$; $((a \vee b) \rightarrow x)$.

As for any configuration parameter, values of such parameters as **`RuleSet`** (i.e. rules) can be viewed and modified using the commands `cfshow` and `cfset` of the management tool Dr.Web Ctl (module **drweb-ctl**). For further information about the `cfshow` and `cfset`command syntax of the command-line management tool Dr.Web Ctl (the **drweb-ctl** module), refer to the section Dr.Web Ctl.

Appendix E. Generating SSL certificates

For the Dr.Web for UNIX Internet Gateways components that use a secure SSL/TLS data channel and application protocols, such as HTTPS, LDAPS, SMTPS, and so on, it is necessary to provide private SSL keys and the corresponding certificates. Keys and certificates for some components are generated automatically; and for others—they should be provided by the user. All the components use certificates in the PEN format.

To generate private keys and certificates used for connections via SSL/TLS, including verification certificates of Certification Authority (CA) and signed certificates, you can use the command-line utility **openssl** (included in an **OpenSSL** cryptographic package).

Consider sequence of actions required for generating a private key and the corresponding SSL certificate together with a SSL certificate signed by the CA verification certificate.

Generating a private SSL key and a certificate

The generation procedure consists of two steps:

1. Generating a private key (the RSA algorithm, the key's length is 2048 bits):

   ```
   $ openssl genrsa -out keyfile.key 2048
   ```

   If you want to password-protect the key, use the `-des3` option. The generated key is in the file `keyfile.key` located in the current directory. To view the key, use the command

   ```
   $ openssl rsa -noout -text -in keyfile.key
   ```

2. Generating a certificate for the specified time period, based on the existing private key (in this case, for 365 days)

   ```
   $ openssl req -new -x509 -days 365 -key keyfile.key -out certificate.crt
   ```

   Note that this command will request data (name, organization, and so on) that should identify the certifying object. The generated certificate will be located into the file `certificate.crt`.

   To check the contents of the generated certificate, use the command

   ```
   $ openssl x509 -noout -text -in certificate.crt
   ```

Registering a certificate as a trusted CA certificate

If you want to register a certificate in the system list of trusted CA certificates (for instance, such a certificate could be generated during the previous step), do the following:

1. Move or copy the certificate file to the system's trusted certificate directory (`/etc/ssl/certs/` in **Debian/Ubuntu**).

2. In the trusted certificate directory, create a symbolic link to the certificate, where the name of the link is the hash value of the certificate.
3. Reindex the contents of the system’s directory containing certificates.

The example given below performs all these three actions. This example assumes that the certificate that is registered as a trusted one is located in the file `/home/user/ca.crt`:

```
# cp /home/user/ca.crt ./
# ln -s ca.crt `openssl x509 -hash -noout -in ca.crt`.0
# c_rehash /etc/ssl/certs/
```

Creating a signed certificate

To create a signed certificate, do the following:

1. Generate a request for signing a certificate (*Certificate Signing Request – CSR*) based on the existing private key. If the key is absent, it should be generated. The request for signing is created with the following command:

   ```
   $ openssl req -new -key keyfile.key -out request.csr
   ```

   This command, as well as the command responsible for certificate creation, requests data that should identify the certified object. `keyfile.key` here is the existing file of the private key. The received request will be saved to the file `request.csr`.

   To check the result of request creation, use the command

   ```
   $ openssl req -noout -text -in request.csr
   ```

2. Create a signed certificated, based on the request and the existing CA certificate, by using the following command:

   ```
   $ openssl x509 -req -days 365 -CA ca.crt -CAkey ca.key -set_serial 01 -in
   request.csr -out sigcert.crt
   ```

   Note that, to create a signed certificate, you should have the following three files: the file of the root certificate `ca.crt` and its private key `ca.key` and the request for signing `request.csr`. The created signed certificate will be saved to the file `sigcert.crt`.

   Use the following command to check the result:

   ```
   $ openssl x509 -noout -text -in sigcert.crt
   ```

You may repeat these procedures as many times as unique certificates you want to create. For example, every agent for distributed file scanning rDr.Web Network Checker within a scanning cluster should has its own key and certificate.

Appendix F. Known Errors

If the occurred error is not present in this section, it is recommended that you contact technical support. Be ready to name the error code and describe steps to reproduce the issue.

To identify the error, we recommend you to configure logging to a separate file and enable output of extended information to the log. For that, execute the following commands:

```
# drweb-ctl cfset Root.Log <path to log file>
# drweb-ctl cfset Root.DefaultLogLevel DEBUG
```

To return to the default logging method and verbosity level, execute the following commands:

```
# drweb-ctl cfset Root.Log -r
# drweb-ctl cfset Root.DefaultLogLevel -r
```

Errors Determined by Code

If instead of receiving a textual error message or a numeric error code you received an internal error code that looks like `EC_XXX` (for instance, `EC_APP_TERMINATED`), then you can find out the numeric error code and the corresponding description of the error given in this section by using the table of the internal catalog of errors.

Error message: *Error on monitor channel.*

Error code: `x1`

Description: One of the components cannot connect with the Dr.Web ConfigD configuration daemon.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restart the configuration daemon by executing the command

   ```
   # service drweb-configd restart
   ```

2. Check whether the authentication mechanism for **PAM** is installed, configured and operates correctly. If not so, install and configure it (for details refer to administration guides and manuals for your OS distribution).
3. If **PAM** is configured correctly and restart of the configuration daemon does not help, restore program settings to the defaults.

To do it, clear the contents of the <*etc_dir*>/drweb.ini file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

```
# cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
# echo "" > /etc/opt/drweb.com/drweb.ini
```

Restart the configuration daemon after clearing the contents of the configuration file.

4. If it is not possible to start the configuration daemon, reinstall the `drweb-configd` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Operation is already in progress.*

Error code: `x2`

Description: Operation requested by the user is already in progress.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Wait until operation is finished. If necessary, repeat the required action after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Operation is in pending state.*

Error code: `x3`

Description: An operation requested by the user is in pending state (possibly, a network connection is currently establishing or one of the program components is loading or initializing, which takes a long time).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Wait for the operation to start. If necessary, repeat the required action after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Interrupted by user.*

Error code: `x4`

Description: The action is terminated by the user (possibly, it takes a long time).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the required action after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Operation canceled.*

Error code: `x5`

Description: The action is canceled (possibly, it takes a long time).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the required action again.

If the error persists, contact technical support and be ready to name the error code.

Error message: *IPC connection terminated.*

Error code: `x6`

Description: An inter-process communication (IPC) connection with one of the components is terminated (most likely, the component shuts down because of the user command or being idle).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. If the operation is not finished, start it again. Otherwise, the termination is not an error.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid IPC message size.*

Error code: `x7`

Description: A message of invalid size is received during component inter-process communication (IPC).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restart the program by executing the following command:

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid IPC message format.*

Error code: `x8`

Description: A message of invalid format is received during component inter-process communication (IPC).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restart the program by executing the following command:

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not ready.*

Error code: `x9`

Description: The required action cannot be performed because the necessary component or device is not initialized yet.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the required action after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Component is not installed.*

Error code: `x10`

Description: One of the components which is necessary to execute a function is not installed.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Install or reinstall the necessary component. If you do not know the component name, try to determine it reviewing the log file.

2. If installation or reinstallation of the necessary component does not help, reinstall the program.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unexpected IPC message.*

Error code: `x11`

Description: An unexpected message is received during component inter-process communication (IPC).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restart the program by executing the following command:

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *IPC protocol violation.*

Error code: `x12`

Description: Protocol violation happens during component inter-process communication (IPC).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restart the program by executing the following command:

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Subsystem state is unknown.*

Error code: `x13`

Description: It was discovered that the current state is not known for a certain subsystem that is part of this software and is needed for carrying out the requested operation.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the operation.
2. If the error persists, restart the program by executing the command

   ```
   # service drweb-configd restart
   ```

   and then repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Path must be absolute.*

Error code: `x20`

Description: Absolute path to file or directory is required (beginning with the root directory of the file system). Relative path is used now.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Change the path to the file or the directory so as to make the path absolute.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not enough memory.*

Error code: `x21`

Description: Not enough memory to complete the required operation (for example, an attempt to open a large file).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Increase size of available memory for program processes (for example, by changing the limits with the **ulimit** command), restart the program and repeat the operation.

   Note that in some cases the system service **systemd** can ignore the specified limit changes. In this case, edit (or create if it does not exist) a file `/etc/systemd/system/drweb-configd.service.d/limits.conf` and specify the changed limit value, for example:

   ```
   [Service]
   LimitDATA=32767
   ```

   The list of available limits of **systemd** can be viewed in the documentation **man** `systemd.exec`.

   Uninstall Dr.Web for UNIX Internet Gateways by entering the following command:

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *I/O error.*

Error code: `x22`

Description: An input/output (I/O) error occurs (for example, the drive is not initialized yet or the partition of the file system is not available anymore).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the required I/O device or the partition of the file system is available. If necessary, mount it and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *No such file or directory.*

Error code: `x23`

Description: A specified object of the file system (file or directory) is missing. Possibly, it is removed.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path. If necessary, change it and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Permission denied.*

Error code: `x24`

Description: There are not enough permissions to access the specified object of the file system (file or directory).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the path is correct and whether the component has required permissions. If it is necessary to access the object, change access permissions or elevate component’s permissions. Repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not a directory.*

Error code: `x25`

Description: A specified object of the file system is not a directory. Enter the path to the directory.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path. Change it and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Data file corrupted.*

Error code: `x26`

Description: Requested data is corrupted.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the operation.
2. If the error persists, restart the program by executing the command

   ```
   # service drweb-configd restart
   ```

   and then repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *File already exists.*

Error code: `x27`

Description: On attempt to create a file, another file with the same name is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path. Change it and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Read-only file system.*

Error code: `x28`

Description: On attempt to create or change an object of the file system (directory, file or socket), it is detected that the file system is read-only.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path. Change it so that the path indicates the writable partition of the file system and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Network error.*

Error code: `x29`

Description: A network error occurs (possibly, a remote node stops responding unexpectedly or the required connection fails).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the network is available and network settings are correct. If necessary, change network settings and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not a drive.*

Error code: `x30`

Description: An accessed input/output (I/O) device is not a drive.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the device name. Change the path so that it indicates to the drive and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unexpected EOF.*

Error code: `x31`

Description: During data reading, the end of the file is reached unexpectedly.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the name of the file. If necessary, change the path so that it indicates the correct file and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *File was changed.*

Error code: `x32`

Description: During scanning the file, it is detected that the file was changed.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat scanning.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not a regular file.*

Error code: `x33`

Description: During accessing an object of the file system. it is detected that it is not a regular file (that is, it is a directory, socket or other object of the file system).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the name of the file. If necessary, change the path so that it indicates the regular file and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Name already in use.*

Error code: `x34`

Description: On attempt to create an object of the file system (directory, file or socket), another object with the same name is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in

the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path. Change it and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Host is offline.*

Error code: `x35`

Description: A remote node is not available through the network.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the required node is available. If necessary, change the node address and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Resource limit reached.*

Error code: `x36`

Description: The limit defined for the use of a certain resource has been reached.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the availability of the required resource. If necessary, raise the limit on the use of this resource and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Different mount points.*

Error code: `x37`

Description: Attempt to restore a file which requires its movement between the file system directories, which belong to different mounting points.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Choose another path for the file restoration and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unpacking error.*

Error code: `x38`

Description: Archive unpacking unsuccessful (it is possibly password protected or corrupted).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Make sure that file is not corrupted. If the archive is protected with password, remove the protection by entering the correct password and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Virus database corrupted.*

Error code: `x40`

Description: Virus databases are corrupted.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the virus database directory. Change the path, if necessary (the **`VirusBaseDir`** parameter in the `[Root]` section of the configuration file).

   To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).

   You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Root.VirusBaseDir
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir -r
   ```

2. Update virus databases:

   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

```
$ drweb-ctl update
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Non-supported virus database version.*

Error code: `x41`

Description: Current virus databases are meant for earlier program version.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the virus database directory. Change the path, if necessary (the **`VirusBaseDir`** parameter in the `[Root]` section of the configuration file).

   To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).

   You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Root.VirusBaseDir
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir -r
   ```

2. Update virus databases:

   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Empty virus database.*

Error code: `x42`

Description: Virus databases are empty.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the virus database directory. Change the path, if necessary (the **VirusBaseDir** parameter in the [Root] section of the configuration file).
   - To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Root.VirusBaseDir
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir -r
   ```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Object cannot be cured.*

Error code: x43

Description: An attempt to apply the Cure action to an incurable object during threat neutralization.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Select an action that can be applied to the object and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Non-supported virus database combination.*

Error code: x44

Description: The current virus database combination cannot be supported.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Check the path to the virus database directory. Change the path, if necessary (the **VirusBaseDir** parameter in the [Root] section of the configuration file).
   - To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Root.VirusBaseDir
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Root.VirusBaseDir -r
   ```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Scan limit reached.*

Error code: x45

Description: When scanning an object, the specified limits have been reached (for example, the limit on the size of an unpacked file, on the nesting depth and others).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Change limits for scanning (in the component settings) by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** cfshow and **drweb-ctl** cfset commands.
2. After changing the settings, repeat the previously attempted operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Authentication failed.*

Error code: x47

Description: Invalid user credentials are used for authentication.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Enter valid credentials of the user with the necessary privileges. Try to complete authentication again.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Authorization failed.*

Error code: `x48`

Description: A user whose credentials are used for authorization does not have enough privileges.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Enter valid credentials of the user with the necessary privileges. Try to complete authentication again.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Access token is invalid.*

Error code: `x49`

Description: One of the program components provides invalid authorization token on attempt to access the operation, requiring elevated privileges.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Enter valid credentials of the user with the necessary privileges. Try to complete authentication again.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid argument.*

Error code: `x60`

Description: An invalid argument is used on attempt to run a command.

For details on the place and the reason of the error, refer to the program log (by default, it is located in

the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the required action again using valid argument.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid operation.*

Error code: `x61`

Description: An attempt to run an invalid command is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the required action again using valid command.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Superuser privileges required.*

Error code: `x62`

Description: Only a user with superuser privileges can perform this action.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Elevate you privileges to root privileges and repeat the required action. To elevate privileges, you can use the commands **su** and **sudo**.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Not allowed in central protection mode.*

Error code: `x63`

Description: The required action can be performed only if the program operates in standalone mode.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Change product’s operation mode to standalone mode and repeat the operation.

2. To change the mode

   - Clear the **Enable the central protection mode** check box on the **Central protection** of web interface management (if installed).
   - Or execute the command

   ```
   # drweb-ctl esdisconnect
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Non-supported OS.*

Error code: `x64`

Description: The program does not support operating system installed on the host.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Install the operating system from the list mentioned in system requirements.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Feature not implemented.*

Error code: `x65`

Description: Required features of one of the components are not implemented in the current version of the program.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Restore program settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

   Restart the program after clearing the contents of the configuration file by executing the command

   ```
   # service drweb-configd restart
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unknown option.*

Error code: `x66`

Description: The configuration file contains parameters unknown or non-supported in the current version of the program.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Open the <*etc_dir*>`/drweb.ini` file in any text editor, remove the line, containing invalid parameter. Save the file and restart the Dr.Web ConfigD configuration daemon by executing the command:

   ```
   # service drweb-configd restart
   ```

2. If it does not help, restore program's settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

   Restart the configuration daemon after clearing the contents of the configuration file.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unknown section.*

Error code: `x67`

Description: The configuration file contains sections unknown or non-supported in the current version of the program.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Open the <*etc_dir*>`/drweb.ini` file in any text editor, remove the unknown (non-supported) section. Save the file and restart the Dr.Web ConfigD configuration daemon by executing the command:

   ```
   # service drweb-configd restart
   ```

2. If it does not help, restore program's settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

Restart the configuration daemon after clearing the contents of the configuration file.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid option value.*

Error code: x68

Description: One of the parameters in the configuration file contains invalid value for the parameter.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Set the valid parameter value by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** cfshow and **drweb-ctl** cfset commands.

   If you do not know which value is valid for the parameter, refer to the help file of the component which uses this parameter. You may also restore parameter value to the default.
2. You may also directly edit the configuration file <*etc_dir*>/drweb.ini. To do this, open the configuration file in any text editor, find the line containing invalid parameter value, set valid value, then save the file and restart the Dr.Web ConfigD configuration daemon by executing the command

```
# service drweb-configd restart
```

3. If the previous steps do not help, restore program's settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>/drweb.ini file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

```
# cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
# echo "" > /etc/opt/drweb.com/drweb.ini
```

   Restart the configuration daemon after clearing the contents of the configuration file.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid state.*

Error code: x69

Description: The program or one of the components is in invalid state to complete the required operation.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Repeat the required action after some time.
2. If the error persists, restart the program by executing the command

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Only one value allowed.*

Error code: x70

Description: One of the parameters in the configuration file contains a list of values; while it is allowed to contain only a single value.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Set the valid parameter value by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands.

   If you do not know which value is valid for the parameter, refer to the help file of the component which uses this parameter. You may also restore parameter value to the default.
2. You may also directly edit the configuration file <*etc_dir*>`/drweb.ini`. To do this, open the configuration file in any text editor, find the line containing invalid parameter value, set valid value, then save the file and restart the Dr.Web ConfigD configuration daemon by executing the command

   ```
   # service drweb-configd restart
   ```
3. If the previous steps do not help, restore program’s settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

   Restart the configuration daemon after clearing the contents of the configuration file.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Tag value is invalid.*

Error code: x71

Description: One of the sections in the configuration file with a name containing a unique tag identifier has an invalid tag identifier.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. If the error occurs on attempt to create a section in the web interface or while using the command

   ```
   # drweb-ctl cfset <section>.<parameter> <new value>
   ```

   set valid value for the tag and save the section again.

2. If the section is saved directly in the configuration file *<etc_dir>*`/drweb.ini`, edit the file. To do this, open the configuration file in any text editor, find the section name containing invalid tag value and set valid value for the tag. Save the file and restart the Dr.Web ConfigD configuration daemon by executing the command

   ```
   # service drweb-configd restart
   ```

3. If the previous steps do not help, restore program’s settings to the defaults.

   To do it, clear the contents of the *<etc_dir>*`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

   Restart the configuration daemon after clearing the contents of the configuration file.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Record not found.*

Error code: `x80`

Description: On attempt to access a threat record, it is found out that the record is missing (possibly, another program component processed the threat).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update the threat list after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Record is in process now.*

Error code: `x81`

Description: On attempt to access a threat record, it is found out that another program component is processing the record now.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update the threat list after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *File has already been quarantined.*

Error code: `x82`

Description: On attempt to move the file with the detected threat to quarantine, it is found out that the file is already in quarantine (most likely, another program component processed the threat).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update the threat list after some time.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Cannot backup before update.*

Error code: `x89`

Description: Prior to downloading the updates from the updates server, an attempt to make a backup copy of the files to be updated failed.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the directory that stores backup copies of the files that are updated. Change the path, if necessary (the**`BackupDir`** parameter in the `[Update]` section of the configuration file).

   - To view and change the path, go to the **Updater** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Update.BackupDir
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Update.BackupDir <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Update.BackupDir -r
   ```

2. Update virus databases:

- Click **Update** on the **Main** page of the web interface, if it is installed.
- Or execute the command

```
$ drweb-ctl update
```

3. If the error persists, check whether the user under whose account the Dr.Web Updater component is running has a write permission to the directory specified in the **BackupDir**. The name of this user is specified in the **RunAsUser** parameter. If necessary, change the user specified in the **RunAsUser** parameter or grant the missing permissions in the directory's properties.
4. If the error persists, reinstall the `drweb-update` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid DRL file*

Error code: `x90`

Description: An integrity violation of one of the files with the list of update servers is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the file with the list of servers and change the path if necessary (parameters with ***DrlPath** in секции `[Update]` of configuration file).
   - To view and change the path, go to the **Updater** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view the current parameter value use the command (<**DrlPath*> needs to be substituted with a specified parameter name. If parameter name is unclear, refer to parameters values in section, skipping the command part in brackets):

```
$ drweb-ctl cfshow Update[.<*DrlPath>]
```

   To set new parameter value, execute the command (<**DrlPath*> needs to be substituted with a specified parameter name:

```
# drweb-ctl cfset Update.<*DrlPath> <new path>
```

   To restore parameter value to the default, execute the command (<**DrlPath*> needs to be substituted with a specified parameter name:

```
# drweb-ctl cfset Update.<*DrlPath> -r
```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

```
$ drweb-ctl update
```

3. If the error persists, reinstall the `drweb-update` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid LST file.*

Error code: `x91`

Description: An integrity violation of the file containing the list of updated virus databases is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update virus databases again after some time.
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

2. If the error persists, reinstall the `drweb-update` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid compressed file.*

Error code: `x92`

Description: An integrity violation of the downloaded file containing updates is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update virus databases again after some time.
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Proxy authentication error.*

Error code: `x93`

Description: The program fails to connect to update servers using the proxy server specified in the settings.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the parameters used to connect to a proxy server (they are set in the **`Proxy`** parameter in the `[Update]` section of the configuration file).
   - To view and set the connection parameters, go to the **Updater** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Update.Proxy
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Update.Proxy <new parameters>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Update.Proxy -r
   ```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *No update servers available.*

Error code: `x94`

Description: The program fails to connect to any of the update servers.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the network is available. Change network settings, if necessary.
2. If you can access the network only using a proxy server, set parameters to connect to the proxy server (you can set them in the **`Proxy`** parameter in the `[Update]`section of the configuration file).

- To view and set the connection parameters, go to the **Updater** page of the web interface (if it is installed).
- You also may use the commands of the command-line management tool.

To view current parameter value, execute the following command:

```
$ drweb-ctl cfshow Update.Proxy
```

To set a new parameter value, execute the command

```
# drweb-ctl cfset Update.Proxy <new parameters>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset Update.Proxy -r
```

3. If network connection parameters (including parameters of proxy server) are correct, but the error occurs, make sure you use the available list of update servers. The list of udpate servers used is displayed in parameters **`*DrlPath`** in `[Update]` section of configuration file. Note that, if parameters **`*CustomDrlPath`** indicate the existing correct file of servers list, the servers specified there will be used instead of the servers of the standard update zone (the value specified in the corresponding parameter**`*DrlPath`**, is ignored).

   - To view and set the connection parameters, go to the **Updater** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view the current parameter value use the command (<**DrlPath*> needs to be substituted with a specified parameter name. If parameter name is unclear, refer to parameters values in section, skipping the command part in brackets):

   ```
   $ drweb-ctl cfshow Update[.<*DrlPath>]
   ```

   To set new parameter value, execute the command (<**DrlPath*> needs to be substituted with a specified parameter name:

   ```
   # drweb-ctl cfset Update.<*DrlPath> <new path>
   ```

   To restore parameter value to the default, execute the command (<**DrlPath*> needs to be substituted with a specified parameter name:

   ```
   # drweb-ctl cfset Update.<*DrlPath> -r
   ```

4. Update virus databases:

   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid key file format.*

Error code: `x95`

Description: The key file format is violated.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether you have the key file and the path to it. You can specify the path to the key file in the **`KeyPath`** parameter in the `[Root]` section of the configuration file).
   - To view and set the path to the key file, go to the **Common Settings** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Root.KeyPath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Root.KeyPath <path to file>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Root.KeyPath -r
   ```
2. If you do not have the key file or the used key file is corrupted, purchase and install it. For more details on the key file, purchase and installation refer to section Licensing.
3. To install the key file, you may use the license activation form which is located at the bottom of the **Main** page of the web interface (if it is installed).
4. You can view current license options in user’s webpage **My Dr.Web** at https://support.drweb.com/get+cabinet+link/.

If the error persists, contact technical support and be ready to name the error code.

Error message: *License is already expired.*

Error code: `x96`

Description: The used license is expired.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Purchase a new license and install a key file that you will receive. For more details on ways to purchase the license and installation of the key file refer to section Licensing.
2. To install the received key file, you may use the license activation form which is located at the

bottom of the **Main** page of the web interface (if it is installed).

3. You can view current license options in user’s webpage **My Dr.Web** at https://support.drweb.com/get+cabinet+link/.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Network operation timed out.*

Error code: `x97`

Description: Network operation timed out (possibly, a remote node stops responding unexpectedly or the required connection fails).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check whether the network is available and network settings are correct. If necessary, change network settings and repeat the operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid checksum*

Error code: `x98`

Description: A checksum of the downloaded file containing updates is detected.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Update virus databases again after some time.
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

   ```
   $ drweb-ctl update
   ```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid demo key file.*

Error code: `x99`

Description: The used demo key file is invalid (for example, it was received from another computer).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Send a request for a new demo period for this computer or purchase a new license and install a key file that you will receive. For more details on ways to purchase the license and installation of the key file refer to section Licensing.
2. To install the received key file, you may use the license activation form which is located at the bottom of the **Main** page of the web interface (if it is installed).
3. You can view current license options in user's webpage **My Dr.Web** at https://support.drweb.com/get+cabinet+link/.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Key file is blocked.*

Error code: `x100`

Description: The used license is blocked (probably, the license agreement conditions on using the Dr.Web program are broken).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Purchase a new license and install a key file that you will receive. For more details on ways to purchase the license and installation of the key file refer to section Licensing.
2. To install the received key file, you may use the license activation form which is located at the bottom of the **Main** page of the web interface (if it is installed).
3. You can view current license options in user's webpage **My Dr.Web** at https://support.drweb.com/get+cabinet+link/.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid license.*

Error code: `x101`

Description: The used license is meant for other product or does not allow operation of the installed product components.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Purchase a new license and install a key file that you will receive. For more details on ways to purchase the license and installation of the key file refer to section Licensing.
2. To install the received key file, you may use the license activation form which is located at the bottom of the **Main** page of the web interface (if it is installed).
3. You can view current license options in user's webpage **My Dr.Web** at

https://support.drweb.com/get+cabinet+link/.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid configuration.*

Error code: `x102`

Description: One of the program components cannot be in operation because of incorrect configuration settings.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. If you do not know the name of the component which causes the error, try to determine it by reviewing the log file.
2. If this error is produced by Dr.Web Firewall for Linux, most likely that there is a conflict with another firewall. For example, it is known that Dr.Web Firewall for Linux conflicts with **FirewallD** in **Fedora**, **CentOS**, **Red Hat Enterprise Linux** (on every launch, **FirewallD** corrupts traffic routing rules indicated by Dr.Web Firewall for Linux). To resolve this error, restart the program by executing the command

```
# service drweb-configd restart
```

or

```
# drweb-ctl reload
```

> Note that if you allow **FirewallD** to operate, the noted Dr.Web Firewall for Linux error can repeatedly occur on every restart of **FirewallD**, including a restart of an OS. You can resolve this error by disabling **FirewallD** (refer to the manual of **FirewallD** included in the manual of your OS).

3. If the error is produced by another component, restore component settings to the defaults by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands.
   - Edit the configuration file manually (delete all parameters from the component section).
4. If the previous steps do not help, restore program's settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

```
# cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
# echo "" > /etc/opt/drweb.com/drweb.ini
```

Restart the program after clearing the contents of the configuration file by executing the command

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Invalid executable file*

Error code: `x104`

Description: On of the program components cannot run, because of the incorrect path or corrupted file contents.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. If you do not know the name of the component which causes the error, try to determine it reviewing the log file.
2. Check the path to the executable file of the component in the program configuration file (the **`ExePath`** parameter in the component section), by executing the following command (change <*component section*> for the name of the corresponding section of the configuration file)

   ```
   $ drweb-ctl cfshow <component section>.ExePath
   ```

3. Restore the path to the default by executing the following command (change <*component section*> for the name of the corresponding section of the configuration file):

   ```
   # drweb-ctl cfset <component section>.ExePath -r
   ```

4. If the previous steps do not help, reinstall the package of the corresponding component.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Virus-Finding Engine is not available.*

Error code: `x105`

Description: The file of Dr.Web Virus-Finding Engine is missing or unavailable (required for threat detection).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb32.dll** anti-virus engine file. Change the path, if necessary (the **`CoreEnginePath`** parameter in the [Root] section of the configuration file).
   - To view and change the path, go to the **Common Settings page of the** web interface (if it is

installed).

- You also may use the commands of the command-line management tool.

To view current parameter value, execute the command

```
$ drweb-ctl cfshow Root.CoreEnginePath
```

To set a new parameter value, execute the command

```
# drweb-ctl cfset Root.CoreEnginePath <new path>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset Root.CoreEnginePath -r
```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

```
$ drweb-ctl update
```

3. If the path is correct and the error persists after updating virus databases, reinstall the `drweb-bases` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *No virus databases.*

Error code: `x106`

Description: Virus databases are not found.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the virus database directory. Change the path, if necessary (the **`VirusBaseDir`** parameter in the `[Root]` section of the configuration file).
   - To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).
   - You also may use the commands of the command-line management tool.

   To view current parameter value, execute the command

```
$ drweb-ctl cfshow Root.VirusBaseDir
```

   To set a new parameter value, execute the command

```
# drweb-ctl cfset Root.VirusBaseDir <new path>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset Root.VirusBaseDir -r
```

2. Update virus databases:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

```
$ drweb-ctl update
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Process terminated by signal.*

Error code: x107

Description: A component shuts down (possibly, because of the user command or being idle).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. If the operation is not finished, start it again. Otherwise, the shutdown is not an error.
2. If a component shuts down constantly, restore its settings to the defaults by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** cfshow and **drweb-ctl** cfset commands.
   - Edit the configuration file manually (delete all parameters from the component section).
3. If it does not help, restore program’s settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>/drweb.ini file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

```
# cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
# echo "" > /etc/opt/drweb.com/drweb.ini
```

Restart the program after clearing the contents of the configuration file by executing the command

```
# service drweb-configd restart
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unexpected process termination.*

Error code: x108

Description: A component unexpectedly shuts down because of a failure.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Repeat the terminated operation.
2. If the component constantly shuts down abnormally, restore its settings to the defaults by any of the following methods:
   - On the page with the component settings on the web interface (if it is installed)
   - Use the **drweb-ctl** `cfshow` and **drweb-ctl** `cfset` commands.
   - Edit the configuration file manually (delete all parameters from the component section).
3. If it does not help, restore program’s settings to the defaults.

   To do it, clear the contents of the <*etc_dir*>`/drweb.ini` file (it is recommended that you make a backup of the configuration file), for example, by executing the following commands:

   ```
   # cp /etc/opt/drweb.com/drweb.ini /etc/opt/drweb.com/drweb.ini.save
   # echo "" > /etc/opt/drweb.com/drweb.ini
   ```

   Restart the program after clearing the contents of the configuration file by executing the command

   ```
   # service drweb-configd restart
   ```

4. If the error persists after restoring program settings, reinstall the component package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Incompatible software detected.*

Error code: `x109`

Description: A program component cannot be in operation because an incompatible software is detected. This software interrupts correct component operation.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. If this error is produced by SpIDer Gate, most likely that there is an incompatible software in operating system. This software generates rules for the **NetFilter** system firewall, which prevents SpIDer Gate from correct operation. Probably, you have **Shorewall** or **SuseFirewall2** installed in the system (in **SUSE Linux** OS). The application that configure the **NetFilter** system firewall sometimes check the integrity of the specified rule system and rewrite it. This is the main reason of SpIDer Gate conflict with such applications.

   Reconfigure incompatible software so as it does not interfere in SpIDer Gate operation. If it is not possible, disable the software so as it does not load at the operating system startup any more. You can try to configure the **SuseFirewall2** application (in **SUSE Linux** OS), following the steps:

1) Open the configuration file of **SuseFirewall2** (by default, this is the /etc/sysconfig/SuSEfirewall2 file).

2) Find the following text block:

```
# Type: yesno
#
# Install NOTRACK target for interface lo in the raw table. Doing so
# speeds up packet processing on the loopback interface. This breaks
# certain firewall setups that need to e.g. redirect outgoing
# packets via custom rules on the local machine.
#
# Defaults to "yes" if not set
#
FW_LO_NOTRACK=""
```

3) Set the parameter value to “no”:

```
FW_LO_NOTRACK="no"
```

4) Restart **SuseFirewall2**. To do it, use the following command:

```
# rcSuSEfirewall2 restart
```

⚠ Note that if **SuseFirewall2** does not have the FW_LO_NOTRACK option in its settings, to resolve the conflict, disable the application so that it does not load at the system startups any more (for example, it is necessary for OS **SUSE Linux Enterprise Server** 11).

After reconfiguring or disabling the conflict application, restart SpIDer Gate.

2. If the error is produced by another component, disable or reconfigure the incompatible software so as to prevent any interference with the Dr.Web for UNIX Internet Gateways operation.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Databases of web resource categories are missing*

Error code: x112

Description: Databases of web resource categories are missing.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Check the path to the database of web resource categories directory. Change the path, if necessary (the **DwsDir** parameter in the [Root] section of the configuration file).

- To view and change the path, go to the **Common Settings** page of the web interface (if it is installed).
- You also may use the commands of the command-line management tool.

To view current parameter value, execute the command

```
$ drweb-ctl cfshow Root.DwsDir
```

To set a new parameter value, execute the command

```
# drweb-ctl cfset Root.DwsDir <new path>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset Root.DwsDir -r
```

2. Update databases of web resource categories:
   - Click **Update** on the **Main** page of the web interface, if it is installed.
   - Or execute the command

```
$ drweb-ctl update
```

If the error persists, contact technical support and be ready to name the error code.

Error message: *LookupD is not available.*

Error code: `x115`

Description: Dr.Web LookupD component is missing (it is necessary to select data from external sources)

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-lookupd** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[LookupD]` section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

```
$ drweb-ctl cfshow LookupD.ExePath
```

To set a new parameter value, execute the command

```
# drweb-ctl cfset LookupD.ExePath <new path>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset LookupD.ExePath -r
```

2. If the configuration does not contain settings for Dr.Web LookupD component or if the error persists after entering the correct path, install or reinstall the `drweb-lookupd` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *SpIDer Gate is not available.*

Error code: `x117`

Description: SpIDer Gate component is missing (required for scanning network connections).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-gated** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[GateD]` section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow GateD.ExePath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset GateD.ExePath <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset GateD.ExePath -r
   ```

2. If the configuration does not contain settings for SpIDer Gate component or if the error persists after entering the correct path, install or reinstall the `drweb-gated` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Scanning Engine is not available.*

Error code: `x119`

Description: Dr.Web Scanning Engine component is missing or failed to start (required for threat detection).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-se** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[ScanEngine]` section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow ScanEngine.ExePath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset ScanEngine.ExePath <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset ScanEngine.ExePath -r
   ```

2. If the error persists after entering the correct path

   - Execute the command

   ```
   $ drweb-ctl rawscan /
   ```

   If the line `Error: No valid license provided,` is output, a valid key file is missing. Register the program and receive a license. After receiving the license, check whether the key file is available and install it, if necessary.

   - If you use 64-bit version of the operating system, make sure that 32-bit application support libraries are installed (see section System Requirements) and, if necessary, install them. After installing the libraries, restart Dr.Web for UNIX Internet Gateways by executing the command

   ```
   # service drweb-configd restart
   ```

   - If your operating system uses **SELinux**, configure the security policy for the **drweb-se** module (see section Configuring SELinux Security Policies).

3. If the configuration does not contain the Dr.Web Scanning Engine component settings or if the steps previously mentioned do not help, install or reinstall the `drweb-se` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *File Checker is not available.*

Error code: `x120`

Description: Dr.Web File Checker component is missing or failed to start (required for threat detection).

For details on the place and the reason of the error, refer to the program log (by default, it is located in

the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-filecheck** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[FileCheck]` section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow FileCheck.ExePath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset FileCheck.ExePath <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset FileCheck.ExePath -r
   ```

2. If the error persists after entering the correct path

   - If you use 64-bit version of the operating system, make sure that 32-bit application support libraries are installed (see section System Requirements) and, if necessary, install them. After installing the libraries, restart Dr.Web for UNIX Internet Gateways by executing the command

   ```
   # service drweb-configd restart
   ```

   - If your operating system uses **SELinux**, configure the security policy for the **drweb-filecheck** module (see section Configuring SELinux Security Policies).

3. If the configuration does not contain the Dr.Web File Checker component settings or if the steps previously mentioned do not help, install or reinstall the `drweb-filecheck` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *ES Agent is not available.*

Error code: `x121`

Description: Dr.Web ES Agent component is missing (it is necessary to connect to the central protection server).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-esagent** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[ESAgent]` section of the configuration file).

You may use the commands of the command-line management tool.

To view current parameter value, execute the following command:

```
$ drweb-ctl cfshow ESAgent.ExePath
```

To set a new parameter value, execute the command

```
# drweb-ctl cfset ESAgent.ExePath <new path>
```

To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset ESAgent.ExePath -r
```

2. If the configuration does not contain settings for the Dr.Web ES Agent component or if the error persists after entering the correct path, install or reinstall the `drweb-esagent` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Firewall for Linux is not available.*

Error code: `x122`

Description: Dr.Web Firewall for Linux component is missing (required for scanning network connections).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-firewall** executable file. Change the path, if necessary (the **`ExePath`** parameter in the `[LinuxFirewall]` section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

```
$ drweb-ctl cfshow LinuxFirewall.ExePath
```

   To set a new parameter value, execute the command

```
# drweb-ctl cfset LinuxFirewall.ExePath <new path>
```

   To restore the parameter value to the default, execute the command

```
# drweb-ctl cfset LinuxFirewall.ExePath -r
```

2. If the configuration does not contain settings for the Dr.Web Firewall for Linux component or if the error persists after entering the correct path, install or reinstall the `drweb-firewall` package.

For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Network Checker is not available.*

Error code: `x123`

Description: Dr.Web Network Checker component is missing (required for check of downloaded files).

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-netcheck** executable file. Change the path, if necessary (the **`ExePath`** parameter in the [Netcheck] section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow Netcheck.ExePath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset Netcheck.ExePath <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset Netcheck.ExePath -r
   ```

2. If the configuration does not contain settings for the Dr.Web Network Checker component or if the error persists after entering the correct path, install or reinstall the `drweb-netcheck` package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *CloudD is not available.*

Error code: `x124`

Description: Dr.Web CloudD (required for requests to the Dr.Web Cloud service) is missing.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the `/var/log/syslog` file or the `/var/log/messages` file, depending on OS).

Resolving the error:

1. Check the path to the **drweb-cloudd** executable file. Change the path, if necessary (the **ExePath** parameter in the [CloudD] section of the configuration file).

   You may use the commands of the command-line management tool.

   To view current parameter value, execute the following command:

   ```
   $ drweb-ctl cfshow CloudD.ExePath
   ```

   To set a new parameter value, execute the command

   ```
   # drweb-ctl cfset CloudD.ExePath <new path>
   ```

   To restore the parameter value to the default, execute the command

   ```
   # drweb-ctl cfset CloudD.ExePath -r
   ```

2. If the configuration does not contain settings for the Dr.Web CloudD component or if the error persists after entering the correct path, install or reinstall the drweb-cloudd package.

   For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support and be ready to name the error code.

Error message: *Unexpected error.*

Error code: x125

Description: Unexpected error occurs in operation of one of the components.

For details on the place and the reason of the error, refer to the program log (by default, it is located in the /var/log/syslog file or the /var/log/messages file, depending on OS).

Resolving the error:

1. Try restart the program by executing the command

   ```
   # service drweb-configd restart
   ```

If the error persists, contact technical support and be ready to name the error code.

Errors Without Codes

Symptoms:	Dr.Web MailD, SpIDer Gate, Dr.Web ICAPD (list of the indicated components depends on the installed product) do not scan messages; in the Dr.Web for UNIX Internet Gateways log there are messages Too many open files.

Description: Due to a large data scanning load, Dr.Web Network Checker has reached the ceiling of the number of available file descriptors.

Resolving the error:

1. Raise the limit of the number of open file descriptors available to the application via the command **ulimit** `-n` (default limit of the descriptor number for Dr.Web for UNIX Internet Gateways is 16384).

   Note that in some cases the system service **systemd** can ignore the specified limit changes. In this case, edit (or create if it does not exist) a file `/etc/systemd/system/drweb-configd.service.d/limits.conf` and specify the changed limit value:

   ```
   [Service]
   LimitNOFILE=16384
   ```

   The list of available limits of **systemd** can be viewed in the documentation **man** `systemd.exec`.
2. Once the limit is changed, restart Dr.Web for UNIX Internet Gateways by executing the following command:

   ```
   # service drweb-configd restart
   ```

If the error persists, contact technical support.

Symptoms: Web browser cannot establish connection to Dr.Web management web interface; components of Dr.Web anti-virus solutions are not in the list of running processes (**ps** `ax | `**grep** `drweb`); attempt to execute any **drweb-ctl** <*command*>, except for **drweb-ctl** `rawscan`, results in one of the following errors:

```
Error: connect: No such file or directory: "<path>/
.com.drweb.public"
```

or

```
Error: connect: Connection refused: "<path>/.com.drweb.public".
```

Description: Dr.Web for UNIX Internet Gateways cannot start as the configuration daemon Dr.Web ConfigD is not available.

Resolving the error:

1. Execute the command

   ```
   # service drweb-configd restart
   ```

   to restart Dr.Web ConfigD and Dr.Web for UNIX Internet Gateways.
2. If this command returns error message, or has no any effect, install `drweb-configd` component (package) separately.

   Also note that this may mean that **PAM** authentication is not used in the system. If so, please install and configure PAM (the product cannot operate correctly without **PAM**).
3. If the error persists, remove Dr.Web for UNIX Internet Gateways and then install it again.

For details on how to install and uninstall the product or product components, refer to sections Installing the Product and Uninstalling the Product.

If the error persists, contact technical support.

Symptoms:

1. After disabling SpIDer Gate, all network connections are broken (outgoing and, possibly, incoming via SSH and FTP protocols) and cannot be re-established;
2. Search through the **NetFilter** (**iptables**) rules using the following command

   ```
   # iptables-save | grep "comment --comment --comment"
   ```

   returns non-empty result.

Description:

This error is related to the incorrect **NetFilter** (**iptables**) operation, which version is earlier than 1.4.15. Because of this internal error, when SpIDer Gate adds the rules with a unique label (comment) to the list of rules, the rules are added incorrectly. As a result, on shutting down, SpIDer Gate cannot delete its rules of diverting connections.

Resolving the error:

1. Enable the SpIDer Gate monitor again
2. If you need SpIDer Gate disabled, remove the incorrect rules of **NetFilter** (**iptables**) by the following command

   ```
   # iptables-save | grep -v "comment --comment --comment" | iptables-
   restore
   ```

   Note that the **iptables-save** and **iptables-restore** commands require the superuser privileges. To elevate your privileges, you can use the **su** and **sudo** commands. Note also that this command removes all rules with the incorrect comments, for example, added by other applications that also perform routing traffic.

Additional information:

- To prevent this problem, it is recommended to upgrade your OS (or, at least, only **NetFilter** to version 1.4.15 or later one).
- You can also switch the diversion of connections towards SpIDer Gate into the Manual mode in the Dr.Web Firewall's settings if you want to manually divert connections towards SpIDer Gate by specifying the required rules with the help of the **iptables** utility (this way is not recommended).
- For details, refer to manuals **man**: `drweb-firewall(1)`, `drweb-gated((1)`, `iptables(8)`.

If the error persists, contact technical support.

The Internal Catalog of Errors

Error code	Symbolic representation	Internal message about the error	Description
0	`EC_OK`	*Success*	This is not an error
1	`EC_MONITOR_IPC_ERROR`	*Error on monitor channel*	Error x1
2	`EC_ALREADY_IN_PROGRESS`	*Operation is already in progress*	Error x2
3	`EC_IN_PENDING_STATE`	*Operation is in pending state*	Error x3
4	`EC_INTERRUPTED_BY_USER`	*Interrupted by user*	Error x4
5	`EC_CANCELED`	*Operation canceled*	Error x5
6	`EC_LINK_DISCONNECTED`	*Link disconnected*	Error x6
7	`EC_BAD_MESSAGE_SIZE`	*Invalid IPC message size*	Error x7
8	`EC_BAD_MESSAGE_FORMAT`	*Invalid IPC message format*	Error x8
9	`EC_NOT_READY`	*Not ready*	Error x9
10	`EC_NOT_INSTALLED`	*Component is not installed*	Error x10
11	`EC_UNEXPECTED_MESSAGE`	*Unexpected IPC message*	Error x11
12	`EC_PROTOCOL_VIOLATION`	*Protocol violation*	Error x12
13	`EC_UNKNOWN_STATE`	*Subsystem state is unknown*	Error x13
20	`EC_NOT_ABSOLUTE_PATH`	*Path must be absolute*	Error x20
21	`EC_NO_MEMORY`	*Not enough memory*	Error x21
22	`EC_IO_ERROR`	*IO error*	Error x22
23	`EC_NO_SUCH_ENTRY`	*No such file or directory*	Error x23
24	`EC_PERMISSION_DENIED`	*Permission denied*	Error x24
25	`EC_NOT_A_DIRECTORY`	*Not a directory*	Error x25
26	`EC_DATA_CORRUPTED`	*Data file corrupted*	Error x26
27	`EC_FILE_EXISTS`	*File already exists*	Error x27

Error code	Symbolic representation	Internal message about the error	Description
28	`EC_READ_ONLY_FS`	*Read-only file system*	Error x28
29	`EC_NETWORK_ERROR`	*Network error*	Error x29
30	`EC_NOT_A_DRIVE`	*Not a drive*	Error x30
31	`EC_UNEXPECTED_EOF`	*Unexpected EOF*	Error x31
32	`EC_FILE_WAS_CHANGED`	*File was changed*	Error x32
33	`EC_NOT_A_REGULAR_FILE`	*Not a regular file*	Error x33
34	`EC_NAME_ALREADY_IN_USE`	*Name already in use*	Error x34
35	`EC_HOST_OFFLINE`	*Host is offline*	Error x35
36	`EC_LIMIT_REACHED`	*Resource limit reached*	Error x36
37	`EC_CROSS_DEVICE_LINK`	*Mounting points are different*	Error x37
38	`EC_UNPACKING_ERROR`	*Unpacking error*	Error x38
40	`EC_BASE_CORRUPTED`	*Virus base corrupted*	Error x40
41	`EC_OLD_BASE_VERSION`	*Non-supported virus database version*	Error x41
42	`EC_EMPTY_BASE`	*Empty virus database*	Error x42
43	`EC_CAN_NOT_BE_CURED`	*Object cannot be cured*	Error x43
44	`EC_INVALID_BASE_SET`	*Non-supported virus database combination*	Error x44
45	`EC_SCAN_LIMIT_REACHED`	*Scan limit reached*	Error x45
47	`EC_AUTH_FAILED`	*Authentication failed*	Error x47
48	`EC_NOT_AUTHORIZED`	*Authorization failed*	Error x48
49	`EC_INVALID_TOKEN`	*Access token is invalid*	Error x49
60	`EC_INVALID_ARGUMENT`	*Invalid argument*	Error x60
61	`EC_INVALID_OPERATION`	*Invalid operation*	Error x61
62	`EC_ROOT_ONLY`	*Superuser privileges required*	Error x62
63	`EC_STANDALONE_MODE_ONLY`	*Not allowed in central protection mode*	Error x63

Error code	Symbolic representation	Internal message about the error	Description
64	`EC_NON_SUPPORTED_OS`	*Non-supported OS*	Error x64
65	`EC_NOT_IMPLEMENTED`	*Feature not implemented*	Error x65
66	`EC_UNKNOWN_OPTION`	*Unknown option*	Error x66
67	`EC_UNKNOWN_SECTION`	*Unknown section*	Error x67
68	`EC_INVALID_OPTION_VALUE`	*Invalid option value*	Error x68
69	`EC_INVALID_STATE`	*Invalid state*	Error x69
70	`EC_NOT_LIST_OPTION`	*Only one value allowed*	Error x70
71	`EC_INVALID_TAG`	*Tag value is invalid*	Error x71
80	`EC_RECORD_NOT_FOUND`	*Record not found*	Error x80
81	`EC_RECORD_BUSY`	*Record is in process now*	Error x81
82	`EC_QUARANTINED_FILE`	*File has already been quarantined*	Error x82
89	`EC_BACKUP_FAILED`	*Cannot backup before update*	Error x89
90	`EC_BAD_DRL_FILE`	*Invalid DRL file*	Error x90
91	`EC_BAD_LST_FILE`	*Invalid LST file*	Error x91
92	`EC_BAD_LZMA_FILE`	*Invalid compressed file*	Error x92
93	`EC_PROXY_AUTH_ERROR`	*Proxy authentication error*	Error x93
94	`EC_NO_UPDATE_SERVERS`	*No update servers available*	Error x94
95	`EC_BAD_KEY_FORMAT`	*Invalid key file format*	Error x95
96	`EC_EXPIRED_KEY`	*License is already expired*	Error x96
97	`EC_NETWORK_TIMEDOUT`	*Network operation timed out*	Error x97
98	`EC_BAD_CHECKSUM`	*Invalid checksum*	Error x98
99	`EC_BAD_TRIAL_KEY`	*Invalid trial license*	Error x99
100	`EC_BLOCKED_LICENSE`	*Blocked license key*	Error x100
101	`EC_BAD_LICENSE`	*Invalid license*	Error x101
102	`EC_BAD_CONFIG`	*Invalid configuration*	Error x102

Error code	Symbolic representation	Internal message about the error	Description
104	`EC_BAD_EXECUTABLE`	*Invalid executable file*	Error x104
105	`EC_NO_CORE_ENGINE`	*Core engine is not available*	Error x105
106	`EC_NO_VIRUS_BASES`	*No virus databases*	Error x106
107	`EC_APP_TERMINATED`	*Process terminated by signal*	Error x107
108	`EC_APP_CRASHED`	*Unexpected process termination*	Error x108
109	`EC_INCOMPATIBLE`	*Incompatible software detected*	Error x109
112	`EC_NO_DWS_BASES`	*No web resource databases*	Error x112
115	`EC_NO_LOOKUPD`	*LookupD is not available*	Error x115
117	`EC_NO_GATED`	*GateD is not available*	Error x117
119	`EC_NO_SCAN_ENGINE`	*ScanEngine is not available*	Error x119
120	`EC_NO_FILE_CHECK`	*FileCheck is not available*	Error x120
121	`EC_NO_ESAGENT`	*ESAgent is not available*	Error x121
122	`EC_NO_FIREWALL`	*Firewall is not available*	Error x122
123	`EC_NO_NET_CHECK`	*NetCheck is not available*	Error x123
124	`EC_NO_CLOUDD`	*CloudD is not available*	Error x124
125	`EC_UNEXPECTED_ERROR`	*Unexpected error*	Error x125

Appendix G. List of Abbreviations

In this manual the following terms will be used without explanation hereinafter:

Convention	Complete form
AD	Microsoft Active Directory
DN	(LDAP) Distinguished Name
EPM	ESP Package Manager (package manager)
FQDN	Fully Qualified Domain Name

Convention	Complete form
GID	Group ID (system user group identifier)
GNU	GNU project (GNU is Not Unix)
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure (via SSL/TLS)
ICAP	Internet Content Adaptation Protocol
ID	Identifier
IP	Internet Protocol
LDAP	Lightweight Directory Access Protocol
MBR	Master Boot Record
OID	(SNMP) Object ID
PID	Process ID (system process identifier)
PAM	Pluggable Authentication Modules
RPM	Red Hat Package Manager (package manager)
RRA	Round-Robin Archive
RRD	Round-Robin Database
SNI	Server Name Indication
SNMP	Simple Network Management Protocol
SP	Service Pack
SSH	Secure Shell
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UID	User ID (system user identifier)
URL	Unified Resource Locator

Convention	Complete form
VBR	Volume Boot Record
OS	Operating System
FS	File System

Index

Index

Index